HEURISTICS FOR USE CASE DESCRIPTIONS

KARL ALAN COX

A thesis submitted in partial fulfilment of the requirements of Bournemouth University
for the degree of Doctor of Philosophy

NOVEMBER 2002

Page Numbering as Bound

A true master is forever a student

- Long Jin Jang

Abstract

Use cases, as part of the Unified Modelling Language, have become an industry standard. The major focus has been on the use case diagram. It is only recently that any detailed attention has been paid to the use case description. The description should be written in such a way as to make it communicable to its reader. However, this does not always appear to be the case. This thesis presents the 7 C's of Communicability as quality features of use case descriptions that make them more comprehensible. The 7 C's are derived from software engineering best practice on use case descriptions and from theories of text comprehension. To help in writing descriptions, the CP Use Case Writing Rules are proposed, a small set of guidelines derived from the 7 C's. Going beyond requirements, software engineers often employ use case descriptions to help them build initial design models of the proposed system. Despite Jacobson's claim that "objects naturally fall out of use cases", finding design-oriented classes and objects in use case descriptions is shown not to be straightforward. This thesis proposes a Question Set which allows the engineer to interrogate the description for important elements of specification and design.

Experimentation shows that the CP Writing Rules furnish descriptions that are as comprehensible as those written by other guidelines proposed in the literature. It is also suggested that descriptions be written from the perspective of their intended audience. The limitations of conducting requirements engineering experiments using students are considered and it is suggested that experimenters should not expect large effects from the results.

An industrial case study shows that although the CP Rules could not be applied to all events in the use case descriptions, they were applied to most and at varying levels of abstraction. The case study showed that the 7 C's did identify problems with the written descriptions. The Question Set was well received by the case study stakeholders, but it was considered time consuming. One of the overriding findings from the case study was that project time constraints would not allow the company to use the techniques suggested, although they recognised the need to do so. Automation would make industrial application of the CP Rules and 7 C's more feasible.

Contents Overview

Table of Contents

List of Tables

List of Figures

Acknowledgements

I would like to thank my supervisors: Dr Keith Phalp and Professor Martin Shepperd for their friendship, guidance and support. I would also like to thank all at Company X for allowing me access to their company, especially my brother, John, and Chris Jones. I would also like to thank the many undergraduate and postgraduate students from Bournemouth University who have participated in this thesis. I'd also like to thank my Mum for reading various drafts of this work and for all the good advice.

Author's Declaration

Parts of this thesis have been described in the following publications,

Cox, K., 2000a. Fitting scenarios to the requirements process. 2nd International Workshop on the Requirements Engineering Process: Innovative Techniques, Models, Tools to support the RE Process, London, 6-8 September 2000. In: Tjoa, A., Wagner, R. and Al-Zobaidie, A. (editors), *Proceedings of DEXA'2000, 11th International Workshop on Database and Expert Systems Applications*, Los Alamitos, CA, IEEE Computer Society Press, pp995-999.

Cox, K., Phalp, K. and Shepperd, M., 2001. Comparing use case writing guidelines. In: Achour-Salinesi, C., Opdahl, A., Pohl, K. and Rossi, M. (editors), *7th International Workshop on Requirements Engineering: Foundation for Software Quality*, Interlaken, Switzerland, 4-5 June 2001, Essen, Essener Informatik Beitrage, pp101-112.

Phalp, K. and Cox, K., 2002. Supporting communicability with use case guidelines: an empirical study. *6th International Conference on Empirical Assessment in Software Engineering*, Keele, Keele University, 8-10 April 2002.

Chapter One

Introduction

1.1 Scenarios and Use Cases

Scenarios can be defined as stories or descriptions of things or events or behaviours that might or do take place between a system and a user or between a user and their environment. Scenarios have been used in different disciplines to describe situations at work and instances of system usage. Social scientists have used scenarios to discuss organisational behaviour (Jarke et al. 1998). Human Computer Interaction (HCI) specialists have used scenarios in the development of interfaces and prototypes (Kyng 1995). It is only relatively recently that scenarios have become popular in software engineering, primarily through use cases (Jacobson et al. 1992). For software engineering, Jarke and Kurki-Suonio (1998) state that scenario work is still "somewhat unconventional" (p.1035) and that scenarios are one of the recent success stories in requirements engineering, human computer interaction and strategic management but that almost nothing is known about them (Jarke 1998).

A technique that is often considered appropriate for requirements engineering is the use case (Jacobson et al. 1992, Booch et al. 1999) because, amongst other reasons, it is used as a means of communication between stakeholders when discussing requirements. The use case's industry-wide up-take is no doubt due to its apparent simplicity and integration in the Unified Modelling Language (Rational 1997), which has become an industry standard for modelling object-oriented systems (OMG 2001). The recent proliferation of books and articles on the use case shows its diagrammatic potential and possible problems (e.g. Jacobson 1994a, 1994b, Firesmith 1995, Cox and Phalp 1999, 2000a) but in comparison the use case description has received much less attention.

It appears the use case is becoming an accepted tool for software engineers. Indeed, Fowler (2000) states that he "cannot imagine a situation now where [he] would not use use cases," (p.46). Pooley and Stevens (1999), however, describe an example case study

of an electronic chess game (p.179). Just how, exactly, should one document all possible chess plays with use cases? Nevertheless, there has been limited research into the application of scenarios and use cases in software engineering (Jarke et al. 1998) and practitioners consider scenario development more a craft than engineering (Weidenhaupt et al. 1998). Indeed, Anda and Jorgensen (2000) state that the "lack of studies on Use Case Model understanding means that the guidelines and practices on how Use Cases should be described to ease their understandability... is highly subjective," (p.2).

1.2 Problems with Scenarios and Use Cases

Recent industrial surveys conducted by the CREWS[1] research project show that much of industry is using scenarios and use cases in one form or other but in an ad hoc manner. There are problems that need to be resolved regarding their usage and development (Weidenhaupt et al. 1998, Jarke et al. 1997). An identified problem is that of the form of the scenario or use case description. How should it be written? How is it to be managed and used? (Jarke et al. 1998). Jarke (1999) elaborates upon this. The UML "hardly satisfies the demand for an adequate communications medium between users, developers, and other stakeholders." The problem is fourfold:

- "The process by which use cases are selected, and by which scenarios are developed from them, is only vaguely defined.
- Systematic guidance [of which] specific scenarios to elaborate within a given use case is completely missing.
- There is no systematic procedure [of] how to validate the use cases and scenarios against the requirements, and how to feed the results back in order to expand the scenarios, and to refine or correct the requirements.
- Use cases and scenarios are hardly supported by present UML-oriented tools." (Jarke 1999)

---

[1] Co-operative Requirements Engineering With Scenarios, EU-funded ESPRIT project 21903

Maiden and Corrall (2000) point out that "engineers rarely know… what the content and structure of … scenarios should be," (p2/1) because they lack usage guidance. Alexander (2002a) confirms this and suggests two views: minimalist against full blown detail. These should compromise to deliver a concrete scenario that is neither too verbalised nor too abstract in its contextual description. Importantly, (and especially in consideration of chapter 8, the industrial case study) it depends upon the context the analyst finds himself in.

This work does not explore the scenario per se (though Appendix A does detail those that relate more closely to use cases and software engineering) because its diversity in form, style and usage across many fields would require a much larger body of work than one thesis alone. This work confines itself to the use case description because the UML has become the de facto object-oriented modelling language in recent years and the use case is an important part of the UML. There will be, though, occasional, inevitable, reference to the scenario because one can argue that good use case descriptions will beget good use case scenarios. The use case diagram is not explored in this thesis because the concepts of the diagram are generally understood and are documented in various texts (see, for example, Jacobson et al. 1992, Booch et al. 1999, Arlow and Neustadt 2002).

1.3 Ways to Solve the Use Case Problem

A fundamental purpose in writing a document is that it will be read sooner or later. Therefore, the document should be written in such a way that it is easily understood. The use case (and the scenario) should not escape this simple yet vital characteristic. Since the scenario and use case have an underlying principle that they should serve as a means of communication among stakeholders, their ability to be read and *understood* is paramount. When the writer writes a description for the first time, he will no doubt read and check for problems of understanding. However, the writer's understanding is not necessarily the same as the reader's. To enable better understandability, the use case ought to be written in such a manner that allows ease of comprehension and this in turn should help in recognition of the completeness and consistency of the description. This thesis explores

the use case description and attempts to solve part of the larger problems outlined above by proposing heuristics for writing and validating a well-structured and comprehensible description. The writing guidelines are compared against an alternative set to test their efficacy.

It has been stated that the use case description is meant to provide a simple means for locating objects. Though Jacobson (2001) claims that "objects naturally fall out of use cases," (p.xiii) this is not always the case. This thesis proposes some guidance, in the form of a question set, to allow the designer to interrogate the use case for elements of structural design that might be implicit or 'hidden' in the description. Hall et al. (2002) note that developers found more problems in requirements than other stakeholder groups; by proposing heuristics in writing, assessing and exploiting the use case description, this might help reduce some of those problems.

1.4 Aims of this Thesis

As suggested above, this thesis has two key aims:

1. How to make textual use case descriptions more understandable.

This will be done by exploring the literature to identify aspects that are considered important to this aim. From this, the suggestion is to present guidelines that can help the writer in creating comprehensible use case descriptions and then to test this through experimentation and a case study. The second aim continues the use case description theme,

2. To present guidelines that help the software engineer in extracting relevant specification and design information from use case descriptions.

As stated, these guidelines will be in the form of questions to enable an 'interrogation' of the use case description. These will be explored through experimentation and a case study to test their efficacy.

The focus on the use case description is at the micro level, in that it explores the individual events (sentences) in descriptions in terms of,

1. Grammar (syntactic structure) and style and, (aim 1)
2. The detail that can elicited from the descriptions (aim 2)

Though the use case description can be represented in a number of formats (OMG 2001), the thesis only considers its textual representation because this is its most common format in texts.

There is little consideration of templates (which can be viewed as being the macro level) since the aim is to be more fine-grained than the template addresses (chapter 2). There is no consideration of use case decomposition since this is does not address the grammar and style of individual events in aim 1.

1.5 Thesis Chapters

Chapter two describes a survey of the literature on use cases and explores what has been said about writing them. Exploration of theories in discourse processes suggests ways in which text can become more comprehensible. The chapter suggests qualities of good use case descriptions. It is suggested that use case writers take this into consideration by building them into a guideline set. The chapter also considers how use cases have been exploited to enable design and suggests that a questioning approach might be of benefit.

Chapter three outlines the methodological approach taken to the study, considers the strengths and weaknesses of the alternative approaches and justifies the approach taken in this thesis.

Chapter four proposes a set of writing rules (the CP Use Case Writing Rules) developed from the literature survey in chapter two. A grammar survey of use cases and scenarios reveals that writers have been liberal in their use of grammar structures.

Chapter five describes a pilot study that explores the application of writing guidelines to use cases. This compares the CREWS Use Case Authoring Guidelines (Achour 1998a, Achour et al. 1999) against those suggested by the author (CP Rules). The results indicate that both sets provide equally comprehensible descriptions with CP performing better than CREWS in one domain. The pilot also explores the comprehension and logic of the descriptions. There appears to be little difference in terms of comprehension.

Chapter six presents an experiment that replicates the first part of the pilot. The CP Writing Rules are again compared against the CREWS Guidelines. The results show that the CP Rules perform at least as well as CREWS, but that there is no overall significant difference.

Chapter seven describes an experiment that builds upon the second part of the pilot study. Namely, what information can be extracted from the use case description? Subjects were asked specific questions on the experimental material and generic questions that might be applicable to all use cases. The results indicate that the generic Question Set is a potentially useful tool in identifying elements of specification and design.

Chapter eight describes an industrial case study that applies the heuristics suggested in this thesis, with varying degrees of success. The study is in two parts: firstly, it discusses the issues in writing descriptions and, secondly, shows that the Question Set has potential but needs refinement. Feedback also shows that although the company recognises the importance of the requirements engineering and design techniques used by the author, project deadlines would not allow them to use these techniques.

Chapter nine comments upon the successes, and weaknesses, of the research. The chapter concludes by outlining where future work is required to provide a larger body of

experimental evidence and identifies the need for further case studies and tool support. The chapter also assesses threats to the validity of this thesis and discusses the contributions to software engineering knowledge made.

Chapter Two

Literature Survey

2.1 Introduction

2.1.1 Requirements Engineering

Requirements engineering is about describing the problem domain, determining what desired effects the client wants to exert upon that domain (the requirements) and specifying the external face of the proposed system to (a) enable those desired effects to occur and, (b) to give designers a specification to help them build the proposed system. The identification of requirements engineering as a key phase of the software development process is nothing new (Boehm 1981) but it is still often neglected in education as well as industry (Bray 2002). Many software projects fail due to poor or non-existent requirements processes (Glass 1998). Pressman (1997) states: "No matter how well designed or well coded, a poorly analysed and specified program will disappoint the user and bring grief to the developer," (p.286). The Standish Report (1995) "estimates that in 1995 American companies and government agencies will spend $81 billion for cancelled software projects" and $59 billion for those ('challenged' projects) that overrun in time, in budget and on average deliver only 61% of specified features and functions. Standish surveyed IT executive managers for factors on why projects succeed, are challenged and fail. Success primarily needs: user involvement, executive management support and a clear statement of requirements. Projects are challenged primarily due to: a lack of user input, incomplete requirements and specifications, and changing requirements and specifications. Projects fail primarily because of incomplete requirements and a lack of user involvement. There is no reason to believe the UK is particularly different. After a survey of twelve software companies, (although there are no financial figures) Hall et al. (2002) report that 48% of development problems are in the requirements phase. Of this, 63% is organisational; within this, communication among stakeholders is the biggest problem (24%). Hall et al. state "...immature companies are especially susceptible to problems in requirements. Given that 70% of

software companies are said to remain at CMM level 1, the scale of requirements problems across the industry could be very large. This is worrying considering the criticality of requirements to project success," (p.8).

2.1.2 Introduction to Use Cases

The use case approach is generally considered a tool for eliciting, describing and validating functional requirements and a guide for conducting design (Jacobson et al. 1992). The use case approach became accepted by object-oriented methodologists (e.g. Rumbaugh 1994, Booch 1994) and has achieved wide recognition as a key requirements tool with its inclusion in the Unified Modelling Language (UML). Some methods that avoided use cases adapted their notations to accommodate scenarios through object sequence diagrams (Coad et al. 1995). These types of scenario are (generally) system internal. (For discussion of the diagram see, for example, Christerson and Jacobson (1995), Booch et al. (1999), Cox (2000b). The Object Modelling Language (OML) (Graham et al. 1997) has different use case relationships (stereotypes) to UML. The principles described in the thesis are as applicable to the OML as they are to the UML since they concern descriptions rather than diagrams.)

Jacobson et al. (1995) define a use case thus: "A use case is a sequence of transactions in a system whose task is to yield a result of measurable value to an individual actor of the system," (p.105). Reenskaug et al. (1996) make this definition: "A use case is a set of interactions between the environment and the system, followed from beginning to end. A use case can be seen as a set whose members are actual sequences of interactions. A use case is thus more than a scenario; it is a set of all the possible scenarios that can result from the user stimulus of the system," (p.21). These definitions are quite general. Reenskaug's point is important: the use case contains scenarios, that is, the use case description can be defined as a general store for use case instances or scenarios.

The use case is also considered a style of scenario (Alexander 2000a) and as such some of this discussion will inevitably consider the scenario. Appendix A examines the

scenario further. However, some consideration of the scenario is useful in that it helps define the use case and helps shape the author's viewpoint and subsequent discussions regarding abstraction in descriptions. This is especially important in chapters 7 and 8. Thus, the early parts of this chapter explore and define the use case description.

The use case description is composed of many elements. These form what is termed a use case template. The typical elements are:

Use case name (including a reference number, and describes the actor's goal), actors (humans and things that interact directly with the system), context (reason for the use case), trigger (what starts the use case), pre- and post-conditions (system states before and after the use case is instantiated (performed)). The main flow of events describes a typical usage of the system to perform a complete transaction. There are also alternative and exceptional flows of events that should be placed after the main flow. Alternatives describe different, equally plausible events in the description and exceptions describe unusual events or threats to the success of the description and (hopefully) ways to avoid these potentially fatal events.

Templates are sometimes organised into tables to separate the elements. See Harwood (1997) and Wiegers (1999), or Sindre and Opdahl (2001) for a different perspective on this subject. Others have divided the main flow of events section into two columns, one describing actor actions and the other system responses (e.g. Wirfs-Brock 1995, Constantine and Lockwood 1999). The focus of this thesis is finer-grained, that is, it is the individual sentences and their relationship to other individual sentences *within* the flows of events that is considered. To bound text in boxes is fine, but these authors have still not addressed the structures of the individual sentences that make up the use case description.

Use cases are a recommended means of establishing a model of the problem domain and then as a driver to enable the first steps into the structural design of the system (Jacobson et al. 1992, Rosenberg 1999, Booch et al. 1999). Thus they are more than a tool to help

describe requirements. The notion of crossing from a problem domain-oriented view to a solution-oriented view of the system is not new to software engineering. An example is the object-oriented analysis view feeding the object-oriented design view (Coad and Yourdon 1991). "What is often not so clear is how to properly use use cases from a methodological point of view and within a precise production process," (Insfran et al. 2002, p.65). Hurlbut (1997) states that "detailed mappings from use cases to other modelling constructs that implement these use cases... is ambiguous," (p.2). It can be construed that there is a lack of detailed guidance about moving from a use case description to elements of design. The assumption here, as explained in section 2.3.1, is that use case descriptions are essentially a means for describing specification and if specification work has been done, there should already be a representation of the problem domain to work from.

The chapter begins by defining the use case description (section 2.2). Section 2.3 examines CREWS scenario definitions and considers how abstraction should be used in use cases. Section 2.4 examines software engineering approaches to writing use case descriptions, showing that there is little consideration of this important issue. Section 2.5 considers various models of comprehending text as an identification of the principles of understanding and writing more comprehensible text. Section 2.6 discusses metaphors applicable to these models, describing ways that might help provide for more comprehensible use cases. Section 2.7 establishes important qualities of good use case descriptions. Section 2.8 explores ways in which use cases and scenarios are used to feed design models and section 2.9 concludes that there needs to be guidance in the writing of use case descriptions and helping identify elements of design.

2.2 Use Cases and Scenarios

2.2.1 Defining the Use Case

Figure 2-1 defines the use case using an entity relationship diagram. The use case is a vehicle for describing scenarios – use cases describe general system use whilst scenarios

depict concrete instances of this use. A use case approach is a means of collating and organising scenarios, provides a template for scenario descriptions and gives an overview of possible user-machine scenarios via the use case diagram. There are also use case instances that become scenarios when they are instantiated (OMG 1999). This author considers a use case instance and a use case scenario to mean the same thing. Schneider and Winters (1998) introduce primary and secondary scenarios to use cases. The primary scenario describes a most typical success scenario without any exceptions or alternatives, referred to as a "happy day scenario" (p.31). Secondary scenarios are the exceptions and alternatives not addressed in the primary scenario. Taken together, the primary and secondary scenarios combine to make the complete use case.

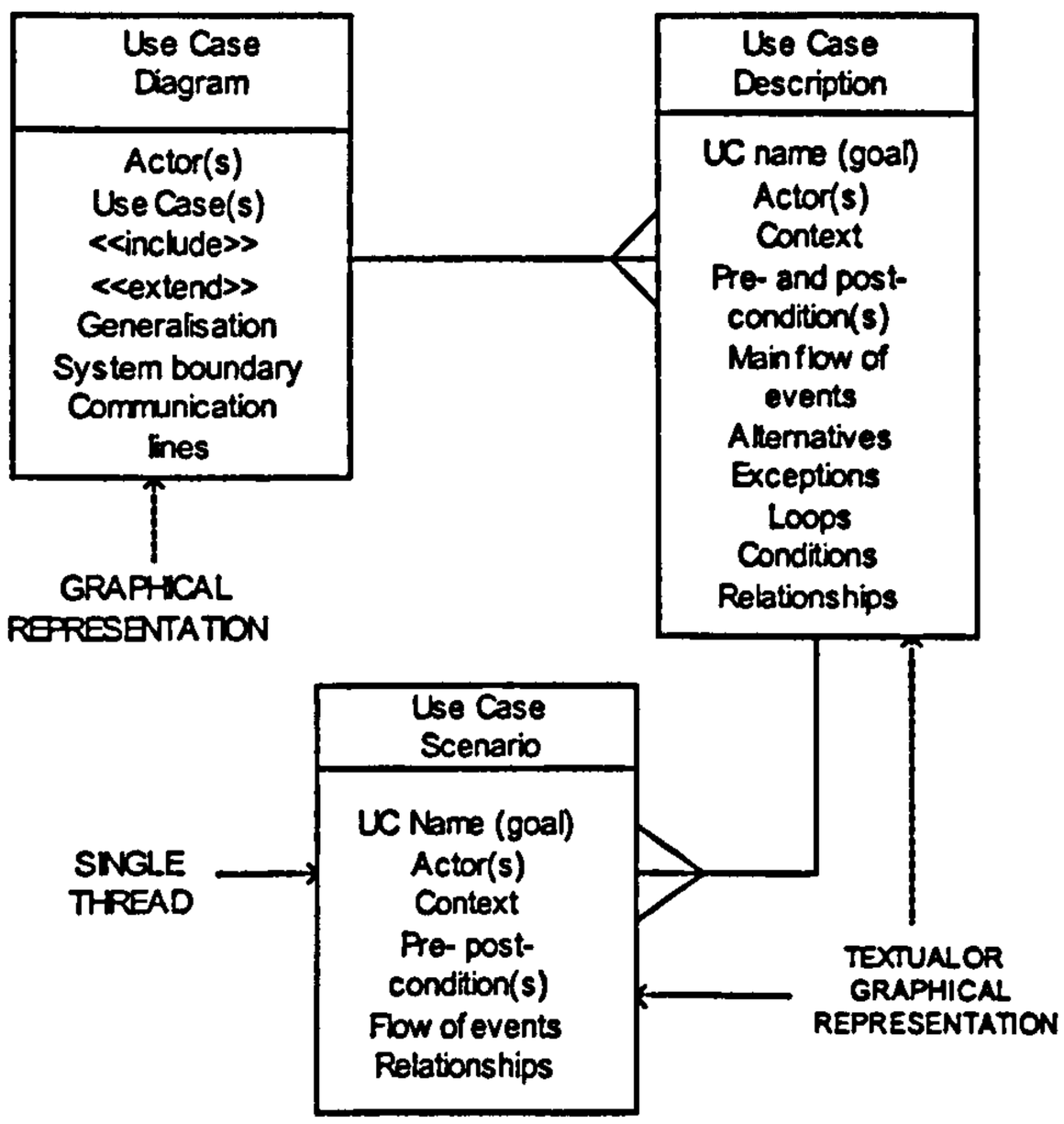


Figure 2-1. Use Case Entity Relationship Diagram

Fowler (2000) describes a use case as a collection of scenarios that share a "common user goal," (p.40). How is this done? Cockburn (2001) helps by defining abstraction levels of use cases based on goals: summary (overview), user (at the interface – to varying degrees) and subfunction (goals that carry out user goals). The summary level use cases allow the reader to go through other use cases at a lower level of abstraction or

granularity. Yet, these summary use cases are still stepped through from start to finish. This thus makes them scenarios since they are a single thread (figure 2-1). This thesis is concerned with the grammar and style of individual events of use cases, so does not consider use case decomposition. Grammar and style occur in each and every event in a description so it does not matter at which level of decomposition one is considering.

It is worthwhile defining what a scenario and a use case description should be. Alexander (2000a) provides the clearest scenario definition: "...a scenario means some kind of description of a set of activities, most commonly sequential," (p.1/1). For the use case, Jacobson's definition is generally accepted (in section 2.1.2), that a use case describes an actor's transactions with the system to achieve some discernible result or value once the use case is completed. Of course, Jacobson's perspective is qualified once the use cases are at a subsystem level. The actors there are other subsystems (see chapter 8).

It is quite easy to substitute scenario for use case and vice versa. The use case entity relationship diagram (figure 2-1) can be described as a class diagram (figure 2-2). This only provides the barest outline of the use case description. Figure 2-2 shows that for each use case description class there can be zero-to-many associated include (using another use case description to save repeating oneself) and extend (when some unusual behaviour interrupts the main flow) use cases. Each description can have zero-to-many exception and alternative paths which are part of, rather than associates to, the use case description, because they form an actual detailed part of that description and are considered very important to use case modelling (Alexander 2000c). The include and extend classes associate because they act as links from this use case description to others. The description can have one-to-many scenarios. There must be at least one scenario per description because each description can be considered a scenario in itself.

The structure of the use case description is rather similar to that of the scenario. Perhaps the major difference is that of contextual information. Use cases prefer to ignore parochial end user details in favour of generality. Even use case instances (or use case scenarios) lack the depth of contextual awareness that a scenario necessarily must convey. This is both a strength

and a weakness. A strength because generality breeds reusability. A weakness because the influence of the individuals who want to shape how the system is used will possibly be compromised.

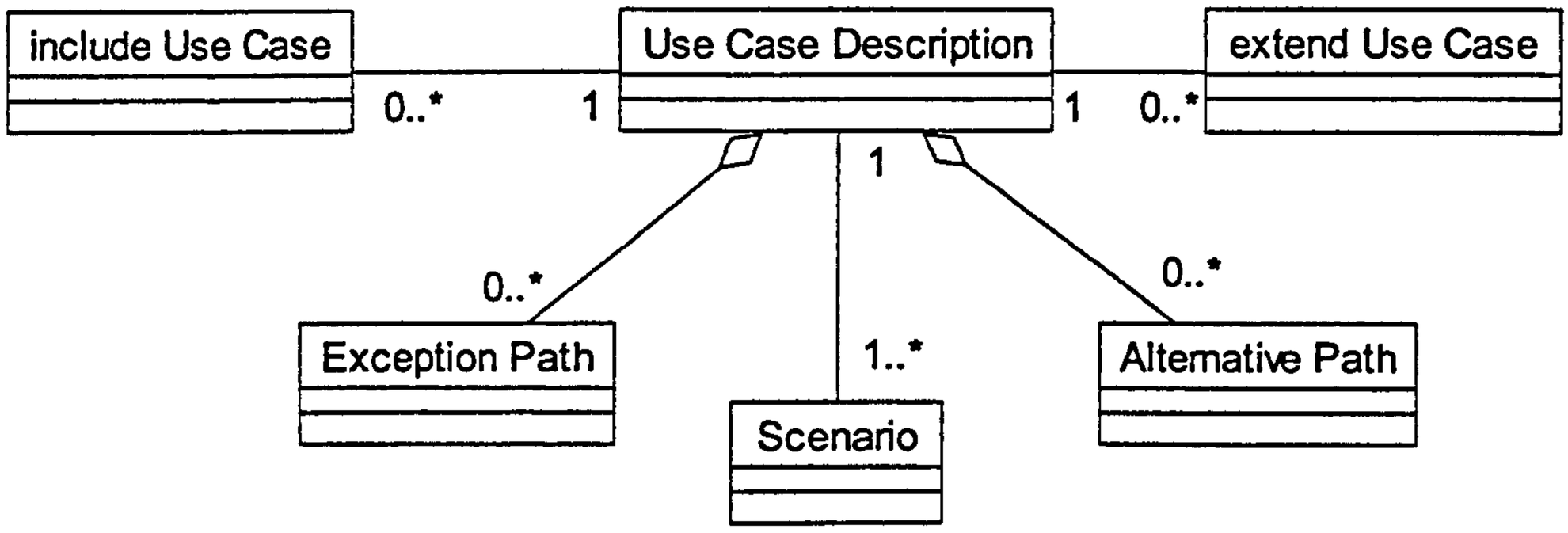


Figure 2-2. Use Case Description Class Diagram

2.2.2 Identified Problems with Textual Representation

In an attempt to gain greater understanding of scenarios, various classification frameworks have been suggested (e.g. Rolland et al. 1998a, Filippidou 1998, Alexander 2002a). For the purpose of this thesis, only one element of classification is considered. This is the Description Facet measure of the Form View (the Form View considers in what format the scenario is represented):

> "*Medium: SET (ENUM { text, graphics, image, video, software prototype })*
> *Notations: ENUM { formal, semiformal, informal }*" (Rolland et al. 1998a, p.30)

The Medium this thesis will discuss is text because this is the way use case descriptions are generally represented. Jarke et al. (1998) state that a semiformal representation is "probably the most important form" (p.167). A reason they give for this is to eliminate ambiguity in "representing the result of an agreement process" (p.167). However, they recognise that the desired semi-formality is constrained by a need for understandability without special training to enable a rapid feedback cycle between stakeholders. This

suggests that use case descriptions should be grounded by specific sentence formats (grammar structures), but that they should also be comprehensible to all stakeholders without special training. In 1992 Nardi wrote that the format of a scenario should be a one-to-two page narrative of "well-crafted prose" (p.14). A decade on, this, apparently, has not been achieved. Wieringa (2001) pleads that text still needs clarity and precision, that there should be "no room for vagueness" (p.134) and Hooks (2000) has not seen any "improvement in the quality of requirements being written today," (p.194).

2.2.2.1 A Problem

Since use case descriptions are a recognised tool for describing requirements, they ought to be well crafted. Use case descriptions should meet the demand for semi-formality to remove vagueness. There ought to be "systematic ways to create normal-case... scenarios" (Jarke et al. 1998, p.165). Importantly, use case descriptions ought to be understandable to readers without the need for special training. One eminent research project that has explored these issues is CREWS.

2.3 CREWS Scenario Definitions

From 1996 to 1999 the CREWS project (Co-operative Requirements Engineering With Scenarios) explored the usage of scenarios and use cases in systems development. CREWS was a Europe-wide research group. Their work is perhaps the most important contribution to scenario research in recent times. As part of their initial research they surveyed companies across Europe (Weidenhaupt et al. 1998, Jarke et al. 1997) to explore if and how industry use scenarios in their work. From this and academic work conducted, CREWS (Jarke 1999, Jarke et al. 1998) suggested there are three common scenario types (figure 2-3), thus providing three core scenario definitions: system internal scenario, interaction scenario and environmental scenario.

Scenarios can be represented at different levels of abstraction. Benner et al. (1993) suggest that a scenario can contain several levels: "It is crucial that scenarios be

expressible in concrete terms, yet at arbitrary levels of abstraction," (p.120). This suggests that differing levels of detail can be represented in one scenario and that it might be possible to trace a scenario from the problem environment (C) through interaction (B) to the system itself (A). This would be important in tracing the design of the system to the requirements. Indeed, Jarke et al. (1998) propose the possibility of a *white-box scenario* which would be a combination of an interaction scenario (type B in figure 2-3) and a system internal scenario (type A). This could be a sequence diagram (Jacobson et al. 1992, Booch et al. 1999). This indicates it is possible to trace a scenario from something that is environmental (scenario type C), which is fundamentally described as linking the work environment and system interaction (scenario type B), to scenario type A. How the scenario life cycle is managed is another matter (Weidenhaupt et al. 1998). For the purposes of requirements, the engineer should only concern himself with scenario types B and C.

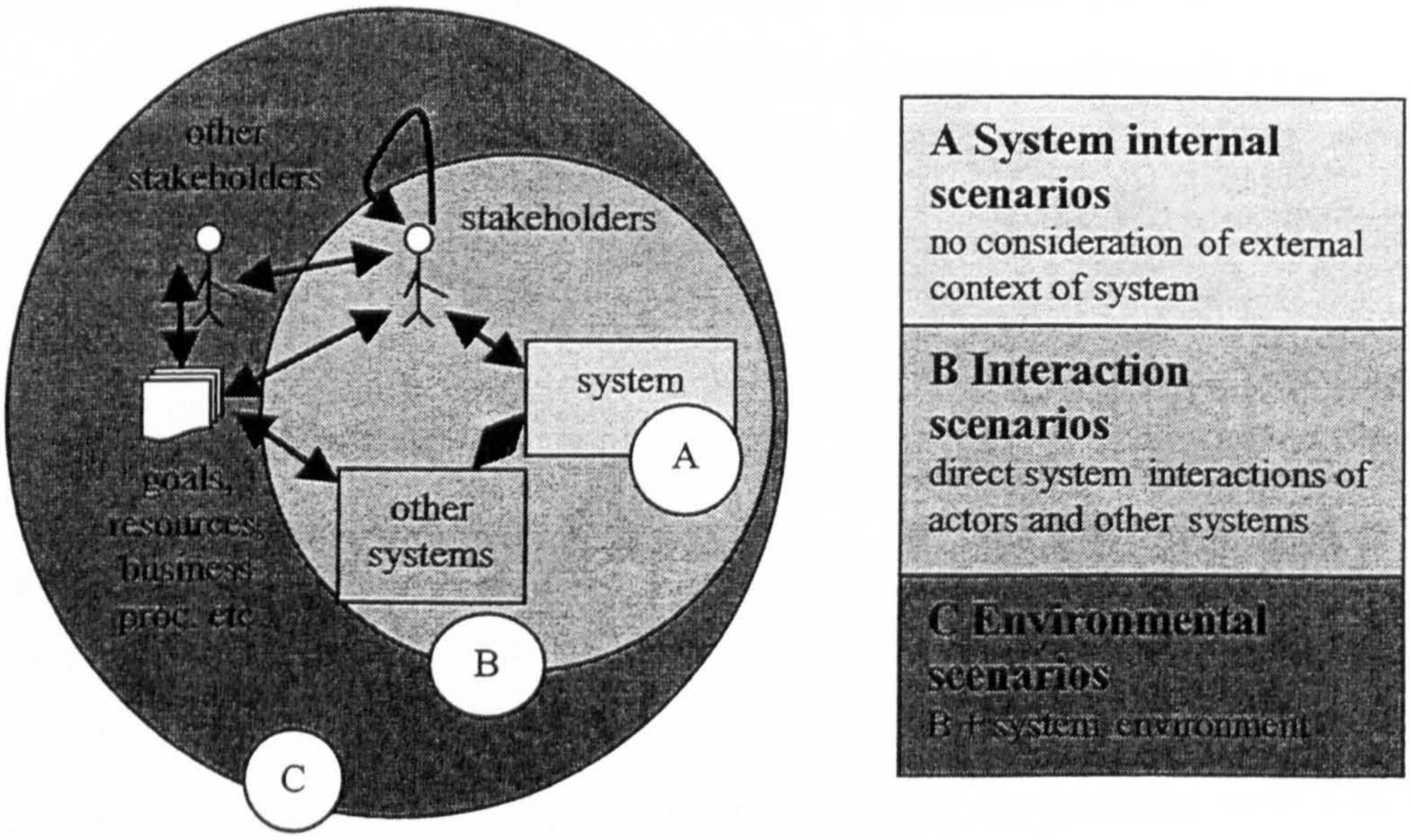


Figure 2-3 CREWS scenario-type model (Jarke et al. 1998, p.158)

(Note that the "stakeholders" in figure 2-3 do not necessarily equate to actors. If they directly interact with they system they can be considered actors (Leibundgut 2002),

otherwise they are defined as people or things that have an indirect interest in the system.)

2.3.1 Abstraction

Use cases are often used to describe requirements for systems to be designed and implemented in the object-oriented paradigm (Arlow 1998). However, there has been much debate as to where the use case is most effective. Jacobson et al. (1992) see the use case as useful for requirements, specification and design. The UML community takes a similar viewpoint, e.g. Booch et al. (1999). Jackson (1998, 2001) and Kovitz (1999) see use cases as a means for describing a specification, because use cases deal with interactions between a user (actor) and the machine (system). Rosenberg (1999) has a slightly different take on the matter. He sees use cases as ways of describing "units of behaviour", requirements as describing the "laws that govern that behaviour" and functions as "the individual actions that occur within that behaviour" (p.123). Jarke et al. (1998) ask a key research question, "[w]hat is the appropriate level of abstraction in a scenario, given a certain purpose?" (p.165). This question is equally applicable to use case descriptions. Scenarios can be represented at varying levels of abstraction from the problem domain to system design (Carroll 2000a, 2000b). Mixing abstraction representations in one scenario is considered possible (Carroll 2000b) and indeed this is commonly seen in scenarios (see Appendices A1 and A2) and use case descriptions (e.g. Regnell et al. 1995). Alexander (2002a) notes that authors should be careful not to mix abstraction levels. This author argues that such internal design representation (describing the solution system's internal structure) in the requirements and specification phase forces consideration of solution information when it is not required. Indeed, the Object Management Group (OMG 2001), who have standardised the UML for industry, state, "The use case construct is used to define the behaviour of a system or other semantic entity without revealing the entity's internal structure," (p.2-137). However, they then backtrack somewhat: "A use case describes the interactions between the users and the entity as well as the responses performed by the entity, as these responses are *perceived* from the outside of the entity" (author's italics, p.2-141). This could mean that only

external events are documented because the user has visual, aural or tactile awareness of the event. Or it could mean that the user (outside the entity) is somehow aware of what the entity or system is doing internally because he is curious to know how the system works. In light of this confusion, the author here presents his own view on the abstraction argument: A use case can be considered more cohesive (a good design quality (Budgen 1994)) if it does not combine different abstraction levels into one description.

Use case descriptions were originally designed to describe interactions at the machine interface (Jacobson et al. 1992) without consideration of internal design (type A, figure 2-3) or the problem environment (type C, figure 2-3). Use cases are tentatively labelled specification, and also (entirely) internal design descriptions. However, if one examines the texts describing use cases and scenarios in requirements and design, then their usage appears in almost all phases (see Cox (2000a) and Appendix A3). For example, Insfran et al. (2002) explicitly describe system internal responses in a three-column format. They employ the CREWS scenario definitions in this. Column 1 depicts general information that equates to environmental type scenario C in figure 2-3. Column 2 represents system interation (type B, figure 2-3) and column 3 represents system internal type A (figure 2-3). This underlying assumption that use cases are universally applicable is a dangerous one since the use case does not model what happens away from the machine interface where the requirements will have their effect. Jackson (2001) provides a clear example of this use case weakness (pp.5-6). External design (describing the external appearance and behaviour of the system), though, is another matter. Indeed, Rosenberg (1999) recommends that use cases are used to help develop prototypes and the graphical user interface (GUI). The question arises, then, whether internal design issues are necessary in requirements and specifications. For instance, in the example of Regnell et al. (1995) they consider events in a use case description such as *card validation* as something that will "have [an] effect on the users" (p.4) of ATMs. Interestingly, Regnell and Davidson (1997) state "A use case may be described either from an external (*black-box*) point of view suitable for requirements, or from an internal (*white-box*) point of view suitable for design," (p.1). They do not state that a use case can contain both black and white box events in the same description. An ATM card validation concerns how the ATM

internally checks that this is a valid card (and is white box). Is the card validation a shared phenomenon (Jackson 1995) between the problem domain (ATM user) and the machine domain (ATM banking system)? Events such as card validation are indeed important to the user but they are internal design. From the user's viewpoint, although he might be implicitly aware that some card validation occurs, he is (generally) visually unaware of it unless the card is rejected. Bray (2002) suggests that such validation checks (which could be design constraints) should be specified separately (p.249) although different viewpoints might require different information. For instance, a bank might need to know when a card is validated although the customer probably will not be concerned. Even so, in terms of the bank's viewpoint, when the card is validated is still a matter of internal design for the system itself (and obviously this should occur before the customer is allowed to withdraw money). Indeed, Mattingly and Rao (1998) note that "internal (white box) interactions, although important for design, should be separate from the use case. They do not describe the interface to the system, and confuse the simplicity of what use cases should strive to be," (p.78). A specification documents inputs and outputs to a system and the relationship between those inputs and outputs (Davis 1991). It is this author's contention that such 'relationships' between input and output should not form part of a use case description (although they might be valid as specification per se) and that use cases should take the viewpoint of the actor, not the system.

2.4 Software Engineering and Use Case Descriptions

> "Reading comprehension is one of the most complex and uniquely human of cognitive activities. During reading, the successful comprehender connects the various events, persons, and objects that he or she encounters so that the text appears to be a coherent whole rather than a random list of facts and events. Frequently, the relations between parts of a text are implicit and, therefore, must be inferred. If all goes well, the result of the inferential processing is a mental representation of the text that is relatively stable and that can be accessed at a later point in time to answer questions, retell the story, and so forth." (van den Broek et al. 1996, p.165).

The ability to read a description in a coherent way and to build a representational model that the writer has in mind is vital to the description's ease of understanding. Conversely, the writer must also realise the perspective of the reader to be able to present a description that matches the reader's mental model (Traxler and Gernsbacher 1995). Communicating the same understood message from the writer and reader's perspective is important. Therefore, the writer must construct their description in a coherent and straightforward manner. The next subsection reviews recent software engineering literature that has commented upon the way use case descriptions (and scenarios) are written.

2.4.1 On Writing Use Case Descriptions and Scenarios

Some authors have little to say on writing descriptions, others proffer examples and a few provide detailed guidance. Fowler (2000), for instance, does not give much information except offer a few alternatives, such as to number events discretely or write between one and three paragraphs of text. He suggests the writer describe primitive steps in the base scenarios and some alternative scenarios. One might expect *The UML User Guide* to provide guidance, but Booch et al. (1999) do not consider the use case description in any detail. They present a paragraph of text that applies formats such as *include/extend use case (Get Apples)*. The OMG (2001) states a "use case can be described in plain text, using operations and methods together with attributes, in activity graphs, by a state machine, or by other behaviour description techniques, such as preconditions and postconditions," (p.2-142).

From a scenario perspective, Carroll (1995a) suggests that "Scenarios need not be in the form of textual narrative... They can be storyboards of annotated cartoon panels, video mockups, scripted prototypes, or physical situations contrived to support certain user activities" (p.3). However, this thesis only explores textual representation since they are the most common format for use case descriptions.

Cockburn (1997) proposed that description sentences take this format:

"*<time or sequence factor>…<actor>…<action>…<constraints>*

Thus, typical sentences might look like:

At any time after the clerk gets the quote, he may cancel the sale," (p.59).

Cockburn (2001) has since published a book on writing use cases. He gives advice on what structure to use to write events in use cases:

"The sentence structure should be absurdly simple:

> Subject… verb… direct object … prepositional phrase.

For example:

> The system… deducts… the amount… from the balance account.

That's all there is to it. I mention this matter because many people accidentally leave off the first noun, making it no longer clear who is controlling the action" (p.90). It is interesting to note that the time dimension has been dropped. However, if required, it could be added as part of a prepositional phrase.

Cockburn shows two approaches to writing the use case:

> "Write the use case from a bird's eye view:
>
> The customer puts in the ATM card and PIN.
> The system deducts the amount from the account balance.

Some writers like to use another style that has the quality of a play, describing actors performing their parts.

Customer: Puts in the ATM card and PIN.

System: Deducts the amount from the account balance.

Note that the information is the same in both styles." (p.91).

There are ambiguity risks inherent in writing natural language (Gause and Weinberg 1989) and as such, paragraphs of eloquent prose and even single sentences are often open to interpretation. However, moving from natural language to more formal constructs for scenarios, as expressed, for example, by Hsia et al. (1994) makes customer validation difficult because they are unlikely to be familiar with predicate logic or the UML (Ham 1998). Indeed, there are proposed tools that are designed to formally verify natural language specifications (section 2.4.2). Because the use case is primarily a means of communication between user/client and engineer, the language of its expression should remain one that is comprehended by both parties.

Pooley and Stevens (1999) note that typically use case descriptions have a structure like this:

"...The system checks that the..."

They append a footnote to this:

"'The system checks...' rather than 'Check' or '...is checked by the system'." (p.29)

The authors thus address the grammar structure of individual events in the description and suggest that passive voice ('is checked by') and sentences without a subject ('Check') are avoided. No explanation is given as to why this is the case. There is no further discussion in the text and example descriptions presented consist of bullet-point lists. Ratjens (1997) similarly suggests that the writer should maximise active voice and minimise passive. An explanation of such avoidance is that the passive might introduce ambiguity through Chomsky's notion of 'deletion' (Chomsky 1975). Information about

actors or system interface actions might be filtered out and left to the reader to, at best, deduce or infer, and at worst, guess (Arlow and Neustadt 2002).

Bray (2002) does not describe a writing preference but documents that descriptions can be paragraphs of prose at one extreme; at the other end might be a stylised specification language, for example:

"select < race button
display > race list
select < race
display > race details..." (p.255)

The norm, according to Bray, is semi-formal text with separate events on numbered lines. His examples use passive and active grammar construction.

Graham (1996, 1998) describes task scripts and addresses the grammar of individual events. "The scripts can be written using a task action grammar if desired; which we call SVDPI form because all the sentences can be arranged in the form: Subject Verb Direct object Preposition Indirect object(s)" (1998, p.141). Graham continues by assessing granularity.

> "It is always possible to keep on decomposing tasks ad infinitum, merely by adding more detail or descending to a lower level of Physics. We stop when words that are not in the vocabulary of the normal user would be introduced at the next stage of the decomposition. For example, in order to capture the script 'the clerk moves the mouse to the Quantity field and enters...' is not atomic because the word mouse is not part of the ontology of ordering. We should have stopped at 'the clerk enters the quantity...' In other words an **atomic** script is arrived at by the decomposing task objects until:
>
> 1. the task script is a single sentence – ideally in the SVDPI format;

2. further decomposition would introduce terms that are not in the domain ontology." (p.141).

The SVDPI format is similar to Cockburn's one structure.

Ericksson and Penker (1998) give detailed guidance on use case construction that is useful in providing a template. However, they do not give much advice on the grammar of individual events. One example uses *if* statements, negatives, passive voice and multiple alternatives (p.322). Kulak and Guiney (2000), however, do provide more detail on writing:

> "Use cases are written in natural language, so the only syntax that guides you when writing them is the use of language. You can be sure that your work will be judged on the quality of the writing in the use cases as much as anything else.
>
> In general, use the active voice ("The actor did it") rather than the passive voice ("It was done by the actor"). Writing in the passive voice may be a clue that information is missing. If you're saying, "It was done," without specifying who did it, perhaps it is time to find out who did it." (p.95).

But what is "quality writing"? This depends on one's perspective, ability and also, importantly, the target audience. Kulak and Guiney also indicate that certain grammar structures should be avoided. For instance, "The use of any kind of constraint language in the use case text (such as Object Constraint Language)" (p.36) and that "Alternative paths listed separately from a basic course of events allow you to avoid IF-THEN-ELSE structures. Users usually do not understand programming structures of any kind: IF-THEN-ELSE, DO-WHILE, TRY-CATCH, and so on" (p.44). Alternative and exceptional flows are important. If there is no consideration of them then "you are planning for failure as soon as the exception arises" (Alexander 2002b). Kulak and Guiney would also remove any implementation specific terminology, such as the card slot in an ATM. To not consider existing or potential features for a product in a use case

description is considered by some to be dubious (Leibundgut 2002) and that such removal of machine names is not desirable, as the case study in this thesis describes (chapter 8).

In terms of a semi-formal structure to the description, Kulak and Guiney recommend, "a tiered numbering scheme to indicate repetition. For example:

1. Indicate customer's color choice from the set displayed.
- (Optional) Indicate as many as five alternative choices.
2. Next step" (p.93).

They also point out that, along with the main body of text, one should "Write exceptions so that they can be read naturally, and do not capitalise words such as *otherwise*" (p.94). What the authors mean by "read naturally" is not expanded upon.

Rosenberg (1999) has little to say. "Once you have some text [paragraphs of prose] in place for a use case, it's time to refine it by making sure the sentences are clear and discrete, the verbs are strong, and actors and potential domain objects are easily identifiable" (p.46). There is general agreement that strong verbs are recommended to provide meaningful description. Whether domain objects are easily identifiable is another matter (see section 2.8 and chapters 7 and 8).

Ham (1998) is similarly uninterested in syntactic representation. He states

> "The particular form that a use case should take is less important than the content. The only requirement is a consistent presentation of use case contents that provides clear understandability by subject matter experts. The use of formal notation languages, e.g. Unified Modeling Language and predicate logic, should be left out unless the user community is fully conversant in the notation presented."

However, the use of semi-formal flow diagrams as scenario representations has proven beneficial in requirements validation (Cox 2000c) showing that users are not homogeneous.

Schneider and Winters (1998) are generic in their approach to writing descriptions:

> "Scenarios can be written very formally or in a less formal style. Keep in mind who will be reading the scenarios and choose a style that is comfortable for the readers... A scenario could be written as informal text... Or it could be a numbered list of steps... Or it could be pseudocode... Our experience is that, in general, a numbered list is easier to understand across a wide range of audience types. It shows each step distinctly, steps can be referred to by number in discussions, and it is an easy style to read for most audiences." (pp.37-38).

However, Schneider and Winters do take a code-oriented approach to repetition in descriptions: "Use repetition when you need to repeat a step or a set of steps multiple times. Indicate clearly where the repetition starts and ends. Also indicate clearly how you will end the repetition... Typically we use a `for` or `while` loop to indicate repetition" (pp.27-28). This use of programming constructs, as seen, is something that Kulak and Guiney would avoid.

How to write use cases has been addressed by a few authors to varying degrees of detail. To this author's knowledge, only Anda et al. (2001) (section 2.4.1.2) have considered the ability to *comprehend* the use case description (within the context of the use case model (diagram and description)). Cockburn (2001) has also addressed this issue to an extent:

> "Ultimately a use case, or any specification, will be read by people. If it is not easily understood, it does not serve its core purpose. You can increase readability by sacrificing some amount of precision and even accuracy, making up for the lack with increased conversation. Once you sacrifice readability, however, your constituents won't read use cases." (p.217).

A goal of use cases is that of a communication mechanism throughout the project between the clients, users and engineers. Use cases are used to describe the functional requirements of the system and also its internal processes. As such, their readability is paramount. Cockburn (2001) provides more detail on constructing the use case:

1. "Keep matters short and to the point. Long use cases make for long requirements, which few people enjoy reading.
2. Start from the top and create a coherent story line. At the top will be a strategic use case. The user goal and eventual subfunction-level use cases branch off from here.
3. Name the use cases with short verb phrases that announce the goal to be achieved.
4. Start from the trigger [something that forces the use case to begin] and continue until the goal is delivered or abandoned and the system has done any bookkeeping it needs to with respect to the transaction.
5. Write full sentences with active verb phrases that describe the subgoals being completed.
6. Make sure the actor and the actor's intent are visible in each step.
7. Make the failure conditions stand out and their recovery actions are readable. Let it be clear what happens next, preferably without having to name step numbers.
8. Put alternative behaviours in the extensions rather than *if* statements in the main body.
9. Create extension use cases only under very selected circumstances." (p.206).

Cockburn does not say how one makes a coherent storyline.

In general, the literature shows that authors are more concerned with the information that is represented in a description than with ways of writing descriptions. There has, though, been some interest in how to construct the title of the use case, actor specification and the importance of contextual information and the first event in the description (see Appendix A4). The only full set of structural guidelines for individual sentences in the use case is provided by CREWS (Achour 1998a, 1998b).

2.4.1.1 CREWS Use Case Authoring Guidelines

Part of CREWS research was to devise Use Case Authoring Guidelines that help in the process of description construction. The Guidelines cover two areas of use case writing: style and content (grammar structure). The Guidelines are well-regarded and are a highly commendable step in the direction of providing detailed guidance in the construction of use case descriptions. The CREWS Style Guidelines have six parts. The examples of use are provided by this author (as used in the pilot study chapter 5).

2.4.1.1.1 Style Guidelines (Achour et al. 1999)

SG1: write the use case normal course as a list of discrete actions in the form: <action#> <action description>. Each action description should start on a new line. Since each action is atomic, avoid sentences with more than two clauses.

For example,

1. *The operator gives the tool to the mechanic.*
2. *The application prompts the user to select an option.*
3. *The dentist selects the medical file from the list, opens the file and writes up his notes on it.*

Action descriptions 1 and 2 are fine but description 3 contains three clauses and as such violates SG1.

SG2: use the sequential ordering of action descriptions (and hence their unique number identifiers) to indicate strict sequence between actions. Variations should be written in a separate section.

For example,

1. *The patient's record appears on the screen.*
2. *The doctor enters the patient's new address.*

Alternatives

2. *The doctor deletes the patient's record.*

The alternatives go below the main flow and the sentence numbers agree (2 and 2).

SG3: iterations and concurrent actions can be expressed in the same section of the use case, whereas alternative actions should be written in a different section.

Examples are given in the Content Guidelines CG6, CG7 and CG8 below.

SG4: use consistent agent, object and action names in all action descriptions in a use case. Avoid use of synonyms and homonyms, and anaphoric references such as he, she, them and it. Be consistent in your use of terminology.

For example,

1. *The User selects the Booking a Room option.*
2. *She types her debit card number into the Reserve a Room option.*
3. *The customer pays by banker's card.*

Sentence 2 uses a pronoun "she" but should say "User". The "Booking a Room" is replaced with "Reserve a Room", an inconsistency in terminology and there is a possible synonym problem with "debit card" in sentence 2 and "banker's card" in sentence 3.

SG5: use present tense and active voice when describing actions.

For example,

1. *The operator presses the button.*
2. *The amount is paid.*

Sentence 1 is correct (present simple active voice) but sentence 2, though being present tense, is in the passive voice and violates SG5.

SG6: avoid use of negations, adverbs and modal verbs in the description of an action

Some examples:

1. *The system does not display the record.* Negation.
2. *The athlete runs quickly.* Adverb (quickly).
3. *The application might ask for further information.* Modal verb (might).

We need to know what we can do, not what we cannot. Adverbs add clutter to the descriptions and modal verbs introduce doubt, which we don't want.

There is the possibility of some ambiguity in the first guideline (SG1) with regard atomic actions in each sentence, which can also contain up to 'two clauses'. This is fine when the second clause is subordinate. What happens, though, when both clauses are main clauses? Then there is a problem because essentially *two atomic actions* are described in one event and it is no longer discrete. A sentence can contain a conjunction (for example, and, or, if, when) and can indicate a new main clause. For example,

> The sun shone and everyone felt happy (Leech 1991, p.92).

'The sun shone' is a main clause, 'and' conjunction, and 'everyone felt happy' another main clause. An example from an ATM use case description could be,

> The User enters the PIN and the system displays a list of menu options.

As such, it might be wiser to restrict the use case writer to one clause per sentence, thus making sure each action is discrete.

SG4 refers to anaphoric references. Cox and Phalp (2000b) suggest that CREWS Guidelines only mean pronouns (since they only list pronouns as examples); there are various types of anaphora, which add complexity to sentence structure and meaning dependent upon placement (Jackendoff 2002).

2.4.1.1.2 Content Guidelines (Achour et al. 1999)

The aim of the Content Guidelines is to guide a use case author in constructing each event in the description. The Guidelines aim to act as a template so authors can substitute their vocabulary into the template and know that it is constructed to a standard format.

CG1: <agent> <'move' action> <object> from <source> to <destination>.

For example,

The clerk sends the report from the store to the office.

CG2: <source agent> <'put' action> <object> to <destination agent>.

For example,

The clerk gives the report to the manager.

CG3: <destination agent> <'takes' action> <object> from <source agent>.

For example,

The manager gets the report from the clerk.

CG4: <agent> <action> <agent>.

For example,

The clerk informs the customer.

CG5: <agent> <action> <object>.

For example,

The operator presses the button.

CG6: 'If' <alternative assumption> 'then' <action>.

For example,

'If' the record is blank 'then' search for customer ID number.

CG7: 'Loop' <repetition condition> 'do' <action>.

For example,

1. *While records available*
   2. *Fill in records.*

CG8: <action 1> 'meanwhile' <action 2>

For example,

Enter consultation notes 'meanwhile' search for X-ray record

The first thing of note regarding the Content Guidelines is their formal appearance. This is in part because of the desire for conformity with tool support (CREWS L'Ecritoire). However, the Guidelines can also be used as standalone rules. The formal appearance might require that certain stakeholders (perhaps end users) be trained to understand the Content Guidelines, thus complicating the problem as described by Jarke et al. (1998) that use cases should be understandable to all stakeholders. Although the author's examples are written in plain text, the Content Guidelines do not necessarily presume this.

The English language has been classified (in typological terms) as an SVO (Subject Verb Object) language (Graddol et al. 1994). Thus it might be considered that the terminology of the CREWS Guidelines seems curious in its usage of 'agent' instead of 'subject' in terms of English grammar. Swan (1980) explains,

"In a passive sentence, the agent is the expression that says who (or what) an action is done by. *This picture was probably painted by* ***a pupil of Rubens***" (pp. xv-xvi).

The use of agent, as from a grammar perspective, is passive voice related. CREWS base the Content Guidelines on Fillmore's case grammar (1968). This uses agent (typically used in liguistics) and the formal structure depicted in the Content Guidelines. This does give the Content Guidelines a pseudcode appearance (Anda et al. 2001).

Arlow and Neustadt (2002) suggest that events should state who or what is responsible for their occurrence. That is, the actor or the system should be named in the event so that the reader knows which of the two conducted the event. The CREWS Content Guidelines 6, 7 and 8 do not include an agent (an actor or the system). Thus the reader might be left uncertain who initiated, or is responsible for, that action or actions, which might lead to confusion and possibly the incorrect implementation of the product.

The next section briefly discusses experimental evaluations of the CREWS Guidelines.

2.4.1.2 Experimenting the CREWS Use Case Authoring Guidelines

To this author's knowledge, there has been no empirical evaluation of use case writing styles and structures except that conducted by CREWS, this author and by Anda et al. (2001), though there is some currently underway at the University of Essen, Germany and is as yet unpublished. The lack of empirical evidence is somewhat surprising considering the impact the use case has had over the last few years.

CREWS describe an experiment that compares elements of their Guidelines to each other (that is, Content against Style against Style and Content) and a control (Achour et al. 1999). Subjects had an hour to write a use case description of a supermarket checkout transaction. CREWS suggest that using their Guidelines produces better use cases than not using guidelines, showing there is more completeness and better structure in the written descriptions (Achour et al. 1999). They also state that there were perhaps too many Guidelines when presented as Style *and* Content (notably Content) and that the Guidelines should perhaps come with examples of their application. In replication, Cox and Phalp (2000c) suggested that the Content Guidelines were useful in completeness and the Style Guidelines in structure. Also many of the Content Guidelines were rarely used: CG1 was not used, CGs 3 and 8 once, and CG2 twice, suggesting the Content Guidelines might be more effective if reduced. Nothing could be generalised from the replication because subject numbers were small. It was noted that further experimentation was required to give a better assessment of the CREWS Guidelines.

Anda et al. (2001) described a variation on this single use case description experiment. They explored the understandability of the wider use case model (the diagram as well as the descriptions). They provided three sets of guidelines: minimal (similar to what general texts say on use cases), template (which provide a framework for the overall use case model) and style (a slightly reduced version of the CREWS Guidelines based on the findings of Cox and Phalp (2000c)). Subjects constructed use case diagrams and wrote descriptions based on the guidelines provided. These were then presented to other

subjects who acted as 'customers' for validation. All subjects were then asked to complete a questionnaire about the guidelines and the understandability of the completed use case models. The results were evaluted in terms of how understandable they were, using as a base the suggested alternative marking scheme of Cox and Phalp (2000c) that considered the overall comprehension of the description (in terms of plausibility, readability, consistent structure and alternative flows). The results showed that the template guidelines produced only slightly more understandable use case models than the style guidelines. The style guidelines were important (for the descriptions) in terms of level of detail (no internal design nor user interface details were allowed), realism (a logical and complete sequence) and consistency (correct use of terminology). The overriding result was that template guidelines and style guidelines were significantly better for constructing, and subsequently comprehending, use cases than the minimal (no) guidelines.

2.4.1.3 A Need for Further Work

In the context of the work presented in this thesis, the experiments described in section 2.4.1.2 provide a platform for a continued exploration of how guidelines improve the writing of use case descriptions in terms of their understandability. Anda et al. (2001) explored the whole use case model whereas Achour et al. (1999) and this author focus on the use case description. The overriding impact of these few experiments was that writing guidelines do seem to make a difference to the quality of use case descriptions. They seem to make them more complete, consistent and understandable. However, there are, to this author's knowledge, no other empirical studies exploring writing descriptions. Thus the limited work conducted thus far is not sufficient to state categorically what guidelines are effective and what are not.

Achour et al. (1999) and Anda et al. (2001) both suggest that guidelines can help. This thesis thus explores this claim further by building upon the work already conducted. The CREWS Use Case Guidelines (section 2.4.1.1) are the starting point for bringing more structure to use case descriptions. However, there are some potential problems with them,

as identified in sections 2.4.1.1.1 and 2.4.1.1.2. For instance, in terms of the Content Guidelines, their formal appearance might make their use problematic to the non-technical user. In terms of the Style Guidelines, for instance, there needs to be clarification of the term anaphoric reference. There also does not appear to be explicit consideration of how the writer makes a description more understandable other than to assume that by avoiding passive voice and anaphoric references, communicability can be achieved. Since this is an important aim of the use case description it needs to be addressed further. However, it is worth noting that since the CREWS Guidelines are relatively established through this experimentation and are well regarded, the set is taken as a baseline for experimentation in this thesis. Nonetheless, further work is required to develop writing guidelines that actually help construct a more understandable use case description and to present a more comprehensive evaluation scheme derived from the qualities of good use case descriptions.

Only a little work (when compared to that written about the use case diagram), then, has been done on how exactly one writes a use case description so that it can be better understood by the reader, that is, in how to produce a more communicable use case description. It is sensible, therefore, to look to a different field to explore the issues of text comprehension in detail since the software engineering community has not addressed this issue in detail. Elements of linguistics have been applied to natural language specifications and the next section explores this field with reference to use cases and scenarios.

2.4.2 Natural Language Specification

This section provides an outline of the work of the Natural Language (NL) community. A goal of this thesis is the exploration of use case writing guidelines and styles and as such this section focusses more on that area than other aspects of NL software engineering.

The NL community has taken elements of linguistics and applied these to software engineering. Exploiting NL for programming has been suggested, for instance, by

mapping problem domain nouns and verbs to abstract data types in Ada (Abbott 1983), and for understanding information in knowledge-based and expert systems (Abbott 1987). In terms of requirements and specification, focus appears to be on a formal validation of NL requirements (Vadera and Meziane 1994). Holt (1999) presents a tool for automating the translation of informal requirements into formal representations. Saeki et al. (1989) present a tool to automatically extract nouns and verbs from NL text with the aim of translating these to more formal models, such as state transition diagrams for a formal specification. Liang and Palmer (1994) examine NL sentences to try to extract formal processing structures such as events and transitions in finite state machines.

The problem domain is often described in NL. Approaches have been suggested to capture that NL in a tool or process and to extract conceptual domain properties. Goldin and Berry (1994) automate the identification of domain knowledge by scanning NL texts. Castell and Hernandez (2000) use linguistic structures to "reduce ambiguity and vagueness inherent in natural language" (p.91) to provide an automated conceptual model of the system under design. Rolland and Proix (1992) propose a specialisation of Fillmore's case grammar (1968) to help understand and map NL elements to a conceptual model. They define sets of patterns combining cases and classes of verbs to create structure patterns in the hope that CASE tools will provide graphical interfaces *and* text to allow the analyst more freedom to describe the conceptual domain. The notion of verb and noun structure patterns has been used in the CREWS-SAVRE tool to generate exception questions from use cases and scenarios, which allows domain dependent language and concepts as well as generic generation (Sutcliffe et al. 1998, Maiden 2002). The CREWS Use Case Authoring Guidelines were implemented into the L'Ecritoire prototype case tool (Rolland et al. 1998b, Rolland 2002). The CREWS Guidelines approach (section 2.4.1.1) employs two linguistic structures (Achour 1997, 1998b). It uses Fillmore's case grammar (1968) as its underlying model for the Content Guidelines combined with Grice's principles (1975), which has four maxims (quantity, quality, relevance and manner). These combine to provide structure to the L'Ecritoire scenario process (Rolland et al. 1998b).

Fliedl et al. (2000) state that although scenario approaches are reported in texts as user friendly, if used properly (that is, by exploring alternative and exception paths), they are no more understandable to the user than other approaches. This suggests that scenario management is a problem, as recognised by Weidenhaupt et al. (1998). To make use cases more usable, Fliedl et al. (2000) examine them for various elements (or notions) and then model them through event driven approaches. The notions are,

- Activity / Action (e.g. agentive (subject) verbs)
- Completion of activity (e.g. past participle of verbs)
- Property / state / behaviour potential (e.g. adjectives and adverbs)
- Event (e.g. "ergative verbs (something happens to a thing-type but the thing-type does not act")) (p.85)
- Restriction (e.g. temporal / causal adverbs).

These aspects are used to help shape pre-design by modelling through Petri-nets, for instance. Though linguistic analysis of the use case description has occurred, there is no notion of how to structure one.

Gelbukh et al. (2000) develop a 'scenario' dictionary of dialogues between users and machines, showing that indirect anaphora (which are more than simple pronouns) are a problem for relating clauses and maintaining coherence. They do not exploit this scenario dictionary for use case descriptions or scenario scripts.

Though there has been much interest in translating NL into formal specifications and also in identifying conceptual domain abstractions through application of elements of linguistics, there has been, to this author's knowledge, almost no consideration of writing use cases so they can be better understood. Use cases tend to be written in natural language but there appears to be little interest in how these are structured. Indeed, Melchisedech (1998) reports that NL is used in requirements and specification documents (along with other more formal techniques) but that a recognised problem with NL is the

lack of precision and ambiguity risk. He suggests that a semi-formal (or formal) approach might help reduce these problems.

As part of the linguistics community, discourse analysts have for many years researched text comprehension (discourse process), and this is explored next to try to identify what structures can make text more understandable.

2.5 The Discourse Processing Perspective

It is important to establish what one does when reading. Readers are "building a description of the shapes and position of things from images," (Marr 1982, p.32). Graesser and Britton (1996) suggest a more detailed explanation than Marr: "What is text understanding? ...Text understanding is the dynamic process of constructing coherent representations and inferences at multiple levels of text and context, with the bottleneck of a limited-capacity working memory," (p.350). Text comprehension is not straightforward. Reading (and hopefully understanding of that text) is a complex process. Research into reading comprehension has realised that focusing solely on either the product of reading (what a reader can remember, understand, infer from a text) or its process (how mental models of text are constructed, assimilated and comprehended) is not enough (van den Broek and Gustafson 1999, Goldman et al. 1999). They suggest focus on the *integration* of process and product. One produces the other and the other cannot occur without the first.

What are the key elements that enforce ease of reading and understanding? Certainly, there are recognised factors that make reading easier or harder dependent upon their presence or not in the text. "The construction of meaningful narrative representations is substantially facilitated when it is easy for the reader to establish local coherence from sentence to sentence in a text," (Goldman et al. 1999, p.5).

2.5.1 Coherence and Inference

Coherence is vital to text if it is to make any sense as a whole, and as individual elements within that text, to have a semantic relationship (Halliday and Hasan 1976). Trabasso et al. (1989) explain coherence: "In the representation of a text, relations between clauses play a major role. By connecting the ideas of one clause to another via inferences, the comprehender organizes and structures the discourse into a functional memory representation," (p.2).

Local coherence is the backward referencing that a clause makes to the previous clause in the text. There is also global coherence where a clause backward references a clause or object in the text prior to the last clause. McNamara and Kintsch (1996) report an experiment where "participants' average reading times per word were greater for the low-coherence than for the high-coherence text," (p.282), thus, a more coherent text enables faster reading and understanding of that text. Inferences do far more than shape a reference to the last clause, however. "Inferences during comprehension take the reader beyond what is explicitly stated in the text...Thus inferences are particularly important to the interpretation of the text and the construction of a more complete mental model of the situation described in the text... Inferences that are made rely on the generic concepts in the reader's knowledge base," (Goldman et al. 1999, p.6).

Magliano (1999) states the same thing as Goldman:

> "Text understanding involves more than the construction and retrieval of a mental representation of the explicit text. Rather, understanding is achieved when readers are able to activate and apply text-appropriate world knowledge in order to generate inferences regarding what the text is about. Furthermore, readers use their world knowledge in order to infer how elements of a text are related, and thus ensure that their mental representation of the text is coherent." (p.55).

There is a clear consensus among discourse processing researchers about the significance of world knowledge and perspective (Kaakinen et al. 2002) in understanding and achieving text coherence.

To enable a better understanding of the use case, readers of the description should be able to make some sense of it. The reader creates a mental model of what he / she is reading and ought to be able to infer what the next step in the description might be based upon the knowledge already assimilated and the mental model created. In simple spoken language if there is a question, one normally expects a response to that question. This is known as an adjacency pair. "The most common adjacency pair is the [Question -> Reply to question] sequence," (Graesser et al. 1996, p.2). For instance, a description of an ATM in operation might state:

1. The User inserts the card into the ATM.

The response that one might infer is the next logical step in the process:

2. The User enters the PIN.

If there was no such statement then the structure of the mental model created by the reader might be drawn into doubt. It could be that the writer of the description chooses to ignore sentence 2, assuming that the reader already has an accepted mental model of how the ATM process runs. However, Ozyurek and Trabasso (1997) state that since evaluation of sentences occur throughout a text, rather than at the end, if there is an event missing (a span problem (Jackson 1995)) or there is too much information (a scope problem (Jackson 1995)), this can upset the overall view of the piece.

Garnham and Oakhill (1996) describe the three ideas of text comprehension proposed by Bransford (Bransford and Franks 1971, Bransford et al. 1972). Bransford's first idea is that the "mental representation of a text does not correspond to any of its linguistic representations. We use syntactic and compositional semantic rules to work out the

meaning of the phrases and clauses of a text... However, we do not typically retain detailed syntactic or semantic representations of what we hear and read," (Garnham and Oakhill 1996, p.315). The point is that the structure of use cases should be simple because word order is not typically remembered. However, Gernsbacher's "Advantage of First Mention" (1997, p.273) principle shows that people recall the first of a pair of names far more readily than the second. For example, the sentence

Lisa and Maggie played a tennis match

In recall, Lisa is remembered more than Maggie (Gernsbacher 1997). The "Advantage of First Mention" should be employed when writing use case descriptions. It should be the case that the principle actor of the use case description be the first mentioned actor or subject in the description. This includes contextual details provided before the main flow of events.

Bransford's second idea considers comprehension to be an integrative process. "Syntactically based theories of comprehension inevitably focus on individual sentences. Indeed, such theories often seemed to suggest that a text representation is simply a concatenation of sentence representations," (Garnham and Oakhill 1996, p.315). This might suggest that a simple syntactic format is enough for use cases. In terms of raw grammar structure, it is. However, "Text comprehension is an incremental process. The mental model of a text constructed to a particular point forms (part of) the context for the interpretation of the next clause of the text," (Garnham and Oakhill 1996, p.316). The reader can infer what the next event is to be in the use case because she/he understands that there should be coherence because there is the adjacency pair structure (as described above as the "Question->Reply to Question" structure).

Bransford's third idea is that

> "understanding is a *constructive process*...in creating a mental representation of the content of a text, information that is explicit in the text (almost always) has to be

> combined with relevant knowledge about the world... Construction and integration are closely connected because, in many cases, background knowledge contributes crucially to the integration of information from different clauses of a text." (Garnham and Oakhill 1996, p.316).

Without relevant project specific knowledge, it is very difficult to comprehend the detailed contents of the use case. It could also be the case that writing a use case or scenario would be very difficult without sufficient world knowledge. The construction manner of text comprehension is "a wholistic description of the overall situation linguistically communicated," (Bransford et al. 1972, p.202). To understand the content of the description, the reader has to understand the domain that description fits within. Of equal importance, the writer of the description must have understanding of the domain the description (and the project as a whole) fits within. Understanding the problem domain is probably a necessary requirement for the use case writer and reader. Domain knowledge, though, is often only learned from practical experience.

2.5.2 Complex Sentences

It is generally the case that people store knowledge about sentences and clauses, and interpret the meaning of those phrases dependent upon their structure (Garnham and Oakhill 1996). So it can be considered that the structure of phrases is important. However, sentences can quite easily be misinterpreted although their meaning appears to be obvious. Here are some examples:

1. Following an air crash on the border between two countries, where should the survivors be buried? (Barton and Sanford 1993).

Why should survivors be buried? The final clause has been misread because the reader has already assumed what is meant – bury the "victims". The simpler the structure for the use case, the better, so long as the writer can express sufficiently the problem being described.

2. No head injury is too trivial to be ignored (Wason and Reich 1979).

This is generally taken to mean that one should *not* ignore head injuries. However, the literal meaning states the reverse. The key is that this structure is complex – with a negative, passive and adjective. Clark (1997) points out that if the literal meaning of the sentence is to be dogmatically adhered to, the reader misses what the message is meant to convey, that is, no head injury should go untreated. Readers do not follow the literal meaning but usually interpret such a complex sentence into one that can be understood *and* which makes sense in the given context.

2.5.3 Referential Continuity

Referential continuity is very important in use cases. "Referential continuity is one of the major factors that determines whether a text is easy to understand," (Garnham and Oakhill 1996, p.321). Here is an example:

1. The cup is in front of the forks.
   It is full of milk.

It is clear that the subject of the second sentence is the cup from the first sentence. Forks are not considered containers for milk. Also, the fork noun is plural. Now consider:

2. The cup is in front of the fork.
   It is full of milk.

Does the pronoun "It" refer to the fork or the cup? Grammatically, "It" refers to the very last noun. However, the mental model of a fork is not one of a container for milk. So it is rightly assumed that the "It" refers to the cup. (Garnham and Oakhill 1996, p.320-321).

If the example is less clear cut then there will be problems in understanding the text. For example:

Fred is standing next to George.

He puts the money in his pocket.

Who is "he" and whose pocket is "his"?

This would suggest that the use of the pronoun anaphoric device has the opportunity to mislead the reader. However, if there is only one actor (in a use case), and assuming the system has no gender, then why not use a pronoun? In any case, the system ought to be referred to as 'the system' or 'the machine' or whatever name it has. That is, call the system by its proper name instead of using "it". Otherwise, "it" (3rd person singular personal pronoun) might well be interpreted as a reference to the actor as much as to the system (since an actor can be a hardware device or a software application as well as human). The problem with allowing pronouns in use case descriptions are many and perhaps the problems outweigh their usefulness. For instance, although there will possibly only be one actor, the system, in responding to actor driven events, needs to state its own case. As such, the system name will become the *subject* of the structure. Differentiating between actors and systems is easy when proper nouns are used but when both become the subject of sentences, the use of pronouns can easily lead to misunderstanding. It is also the case that a human actor is playing a role in the use case description, or at least, should be interpreted as such. A role does not necessarily have a gender; a role, say, Manager, or Customer, is non-gender specific. So to give the actor a gender might be considered misleading. Thus, the role of the personal pronoun is greatly reduced, if not invalidated altogether. However, if one is describing a specific instance of a use case – a scenario – then one might introduce a name for the Customer or Manager to add further context.

2.5.4 Structure Foundations

It is much more difficult to comprehend text, that is, develop a foundation of mental representations "if the information does not lend itself to building cohesive mental

representations, for example, if sentences, paragraphs, or stories are self-embedded or scrambled," (Gernsbacher 1996, p.290).

So the idea of nesting sub-routines in a use case might be a bad one, primarily because it makes it difficult for people to follow a coherent path. For example,

1.

1.2. - difficult and actually there is the implicit notion of moving back to 1 after

1.2.1. completing 1.2.1. But this is implicit. Could the reader not move to 2?

2.

Gernsbacher (1996) describes a Structure Building Framework to model language comprehension that enforces the establishment of logical coherence.

> "According to the Structure Building Framework, the goal of comprehension is to build cohesive mental representations, or structures. The first process involved in building a structure is laying a foundation. The next process involves developing the structure by mapping on incoming information when that information coheres with the previous information. However, if the incoming information is less coherent, comprehenders employ a different process: They shift to initiate a new structure. Thus, most representations comprise several branching substructures." (p.289).

Graesser and Britton (1996) explain Gernsbacher's Structure Building Framework and shed light on an important point that should be incorporated into writing use case descriptions.

> "This comparatively time-consuming process provides an explanation of why it takes longer to read initial words in sentences than later words, and initial sentences in paragraphs than later sentences. The initial foundation is subsequently elaborated as additional information in a text is comprehended. If an incoming proposition is relevant to the foundation, coherence within the level is maintained, and the time to

comprehend the proposition is comparatively fast. Additional time processing is needed to handle coherence breaks or to shift focus from one structural foundation to another structural foundation." (Graesser and Britton 1996, p.343).

Use case descriptions ought to try to impose quicker reading, building upon such propositions in the description already read.

To achieve coherence in a text it is important to create a network of propositions. A text with few connecting propositions is considered difficult to understand (e.g. Mannes and St George 1996, Fletcher et al. 1996, van den Broek et al. 1996). The noun-phrase argument of propositions is critical "for connecting propositions and establishing text coherence; that is two propositions are connected if they share at least one noun argument," (Graesser and Britton 1996, p.345).

Turner et al. (1996) provide an explanation of propositional analysis:

"A predicate specifies a relation between arguments. For example, in the following sentence the predicate is *hit*.

(1) John hit the ball with the bat in the park at noon.

In Example 1, the predicate specifies the relations between John, the ball, the bat, the park and noon. In general, the predicate *hit* has a standard semantics which includes the notion of the hitter, the hittee, the instrument, and often the location and time. We can represent the semantics of the predicate *hit* in a general predicate frame with standard slots:

2) hitter, *hit*, hittee, instrument, location, time

In these frames, the slots (i.e. the *arguments* of the proposition of which *hit* is the predicate) are filled from Example 1 as follows:

(3) John, *hit*, ball, bat, park, noon." (p.34).

It is clear that coherence can be achieved by repetition of noun phrases in consecutive sentences. Thus it should be considered that using proper names instead of pronouns

enhances understanding of text. However, it is not recommended that use case descriptions be written in the style of Turner et al's third example above because this does not appear to be obviously comprehensible. As a means of examining the subsequent description they might be useful. Indeed, Robertson (1995) uses propositional analysis to identify objects that form part of the design of a software system. Structures should be kept very simple because the use case reader's working memory should be free of less important predicates.

The next section suggests five important metaphors identified in discourse processing and their possible usefulness in making use case descriptions more comprehensible.

2.6 Discourse Process Metaphors for Use Case Descriptions

From this examination of the models and processes suggested in text comprehension Graesser and Britton (1996, p.342) suggest five metaphors that capture the various models of text understanding that can be influential in writing use cases.

2.6.1 Understanding is the Assembly of a Multilevel Representation

Use case descriptions should continually build upon the foundations constructed (figure 2-4) at the start rather than add events that force new foundations to be laid.

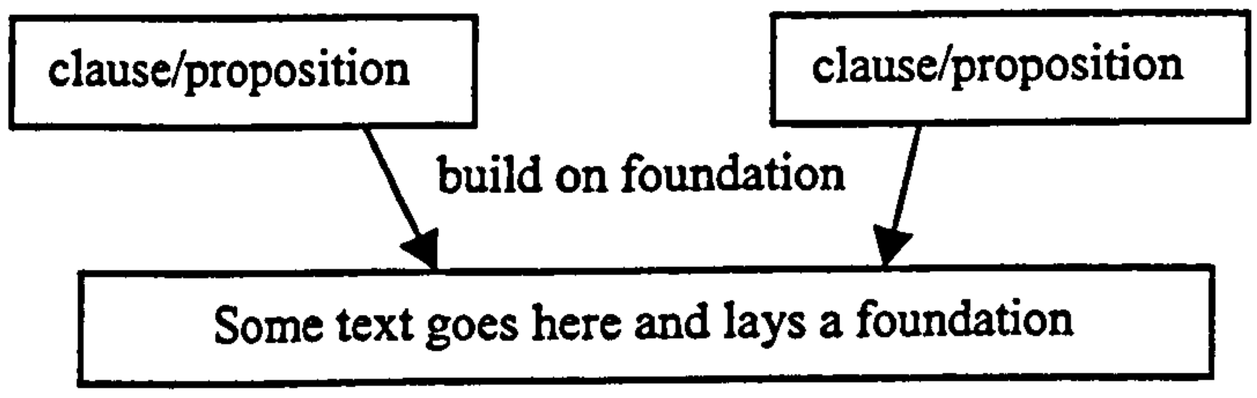


Figure 2-4. Building mental foundations for text understanding

2.6.2 Understanding is the Construction of a Coherent Representation

The goal is that the reader should be able to comprehend a description quickly. To facilitate this, it is recommended that the whole use case is written to achieve logical coherence between events (figure 2-5). This means that the use case writer has to make sure that only relevant information to the continuity of the use case is introduced.

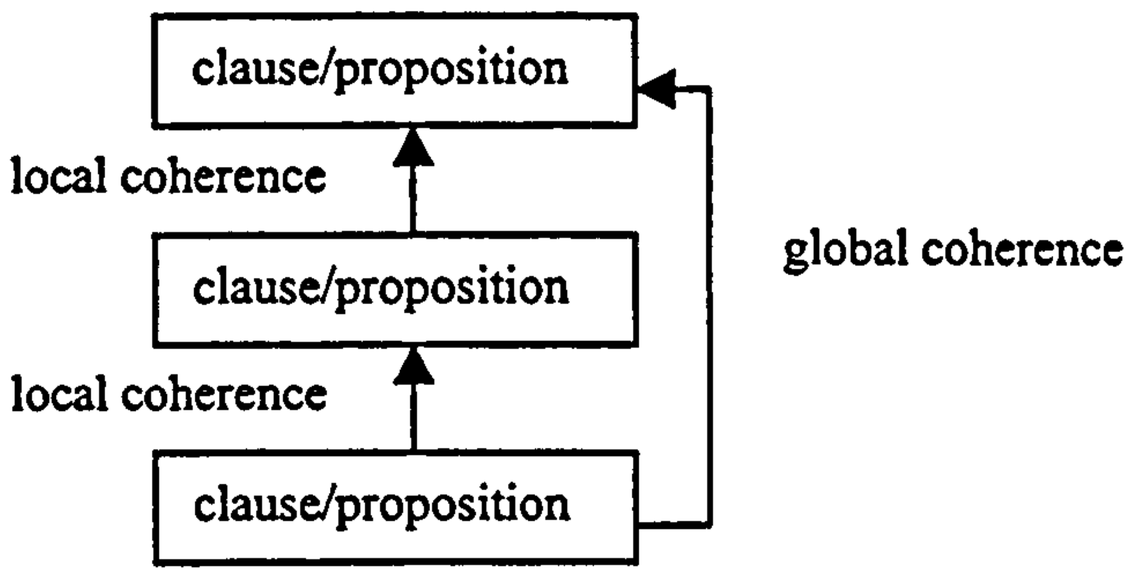


Figure 2-5. Logical coherence in text

> "A basic principle in writing is that ideas should flow easily, one from another. In this way, as each new idea is introduced, it is easily integrated with the information that immediately precedes it. That is, the text should always be locally coherent. Models of text comprehension agree that readers make use of the information in working memory when integrating each new sentence into the text representation." (O'Brien and Myers 1999, p.35).

Each successive sentence ('event' in use case parlance) in a description should logically cohere to a preceding sentence. For example:

The Customer selects Change PIN option.

The System asks for existing PIN.

The Customer enters existing PIN.

The System asks for new PIN...

Figure 2-6 shows logical coherence. There is local coherence to the previous line and also global coherence to events prior to the previous line.

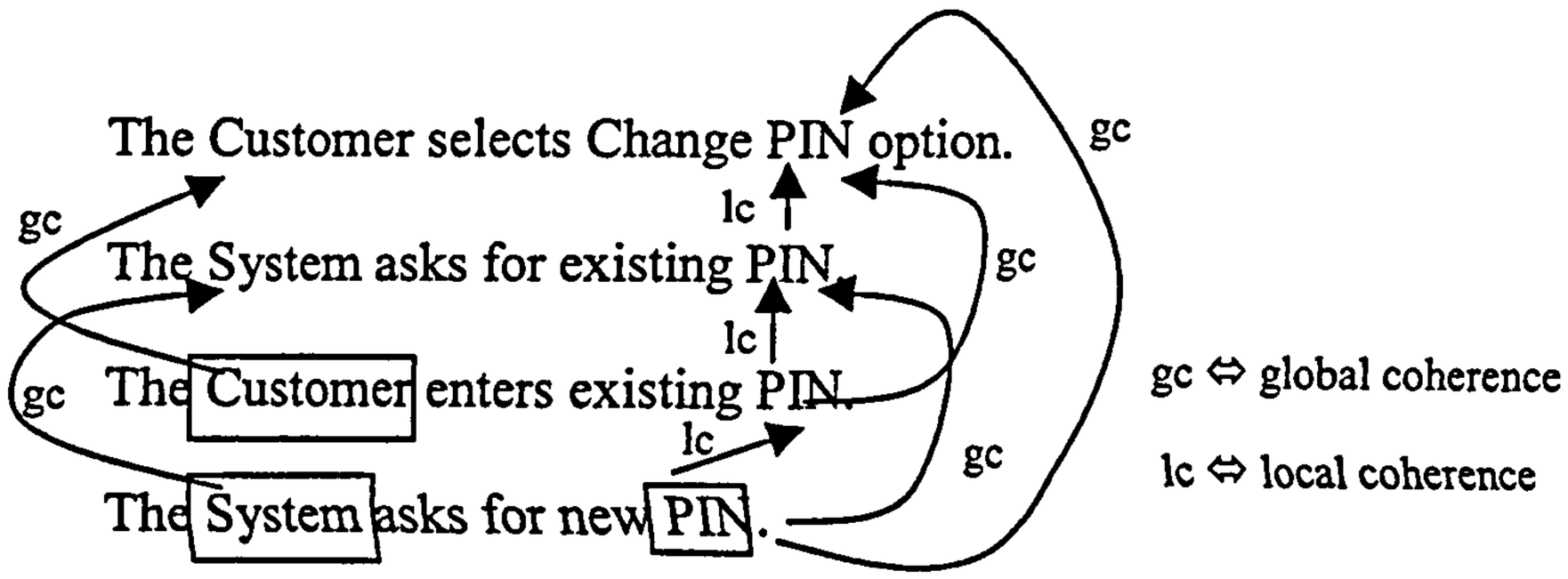


Figure 2-6. Logical Coherence example

2.6.3. Understanding is a Complex Dynamic System

Because understanding is complex and levels of complexity are, at least in part, governed by the syntactic nature of the text, the use case structure (syntax) should aim to be as simple as allowable so long as the use case writer can still express what is needed. Indeed, Perfetti (1997) states discourse theorists finally understood that, "Syntax had a purpose after all, and smoothly functioning discourse depended critically on syntax's formal power to flexibly arrange elements" (p.339). However, Perfetti also points out that syntax reduces the ability to understand propositions. But without some syntax, propositions become even more difficult to understand. Some trade off must be made. There has to be syntax but of a relatively simple structure.

For the use case structure, one should consider applying the adjacency pair rule. For example,

Input -> Response to input, or

Interaction -> Response to interaction – this focuses on describing specification.

This is unsurprising because engineers understand the input -> output idea. Indeed, it is specification: describe an input and then show the system's output. (If the audience is the

user, as opposed to engineer, the adjacency pair might be rephrased as action->response or question->response because users might not be aware of the input / output idea.) The output should have meaning to or describe a relationship with the input. What is required from use cases is the same: local coherence.

2.6.4. Understanding is a Process of Managing Working Memory

To help the reader organise their working memory, the use case description should not be complex; the structures should be straightforward and coherent. Again, the application of the adjacency pair rule helps this. Korn (2000) suggests that adjectives and adverbs introduce subjective possibilities of success into scenarios. Subjectivity can lead to misunderstanding. Removal of unnecessary adverbs, adjectives and confusing pronouns, for instance, helps reduce the difficulty in organising working memory by simplifying the use case description. This should then speed up understanding and reduce ambiguity.

2.6.5. Understanding is Inference Generation

The notion of building upon foundations of previous events in the use case is that a reader can expect something to happen next (figure 2-7). That is, the reader infers that an event will now take place based upon information already comprehended. Using the adjacency pair rule should help inference making become more accurate because if there is an input with a specific purpose (sub-goal), one expects a response to achieve that purpose (see figure 2-8).

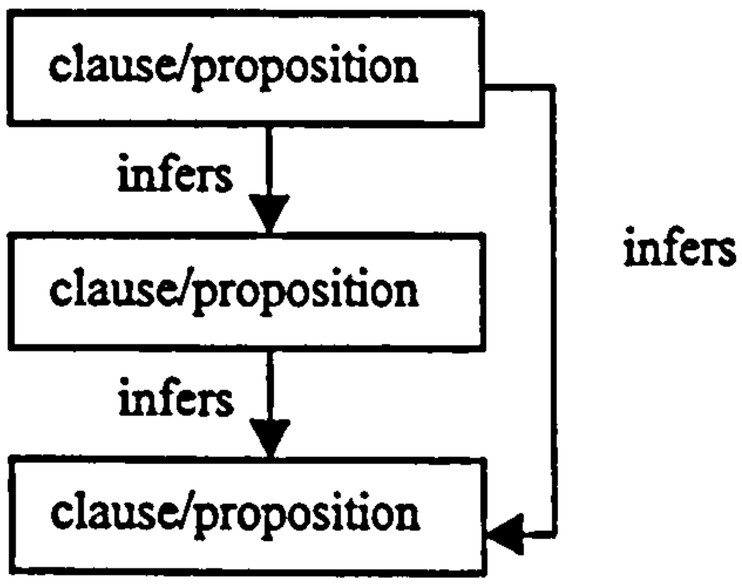


Figure 2-7. Inference generation

2.6.6 Use Case Description Meta-model

Figure 2-8 shows a meta-model for a use case description. The notion of such a model is not unusual. For instance, Pohl et al. (2001) use a scenario meta-model to maintain traceability to an evolving software architecture.

Figure 2-8 shows how local coherence is significant to the use case description. The use case description describes a functional/user requirement, which should describe how to obtain a business or user goal. The use case description must fulfil this goal. The description itself (which can be construed as the main flow of a use case description) is composed of three major elements: system, of which there is one, since this model does not consider distributed systems; event, of which there are two to many, since there should be at a minimum an input and a corresponding output, or vice versa; and actor, of which there is at least one, since without an actor there is no notion of some kind of interaction at the interface or between subsystems. Input and output are kinds of events. The actor makes inputs and receives outputs. The system receives these inputs and delivers outputs. There should be local coherence between input and output. It is also the case that other use cases are types of inputs and outputs. This corresponds to other use case names (UML would denote these as include and extend relationships) that are underlined as events in the description (Cockburn 2001). Each input and output has a purpose (a sub-goal). The sum of purpose 'part fulfils' should achieve the goal of the use case description. This allows for the notions of purpose and goal at different granularities, detailing the difference suggested by Kaindl (1998); this equates to the sub-goals needed to achieve the goal.

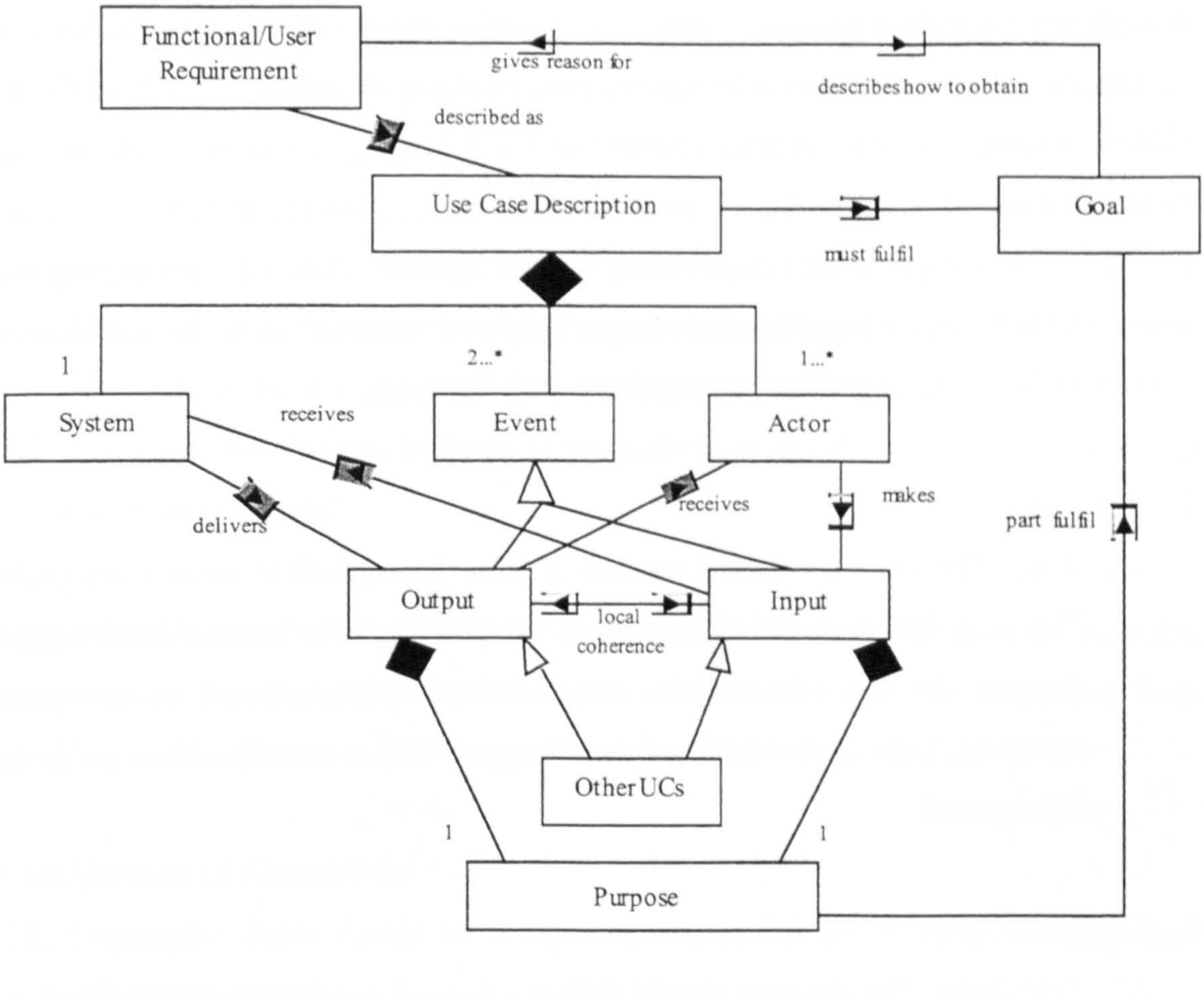


Figure 2-8. Use Case Description meta-model

Figure 2-8 employs local coherence between events in the use case description. The following section describes other desirable qualities of use cases as identified above.

2.7 The 7 C's of Communicability

This section organises the ideas, comments and suggestions of authors described earlier in this chapter into categories relevant to improving the communicability of use case descriptions. Thus there are no new concepts or ideas introduced in this section except the categorisation. Table 2-1 explicitly describes the categories and their sources.

The 7 C's are proposed as a set of heuristics that might be valuable in general validation when actually writing descriptions. It is hoped this will help requirements engineers, and

software engineers in general, in their task of writing use case descriptions. These do not provide any prescribed method to guarantee success. Note that this is guidance on some of the allegedly easier tasks of requirements, writing descriptions. But as Cockburn (2001), Wieringa (2001) and Bray (2002) point out, writing is a complicated task not to be underestimated. It is also hoped, in some small way, to answer Robertson's claim that the hard work of requirements engineering is often ignored (2001). By stimulating the use case writer with these heuristics, these might help as a 'catalyst' to further clarification of the task of determining use case descriptions. The 7 C's are:

1. Coverage.
   1.1. Span: The use case should contain all that is required to answer the problem. That is, is there enough information in the description or is some detail missing?
   1.2. Scope: The use case should contain detail only relevant to the problem statement. Extra unnecessary information provided is out of problem scope and is not required.

2. Cogent.
   2.1. Text Order: The use case should follow a logical path. Is this path logical or are events in the description in the wrong order?
   2.2. Dependencies: The use case should complete as an end-to-end transaction (which can include alternative / exceptional flows). Does the actor reach a state that stops the transaction from terminating as we expect?
   2.3. Rational Answer: The logic of the use case description should provide a plausible answer to the problem. Are there any events that appear out of place or you recognise as incorrect?

3. Coherent. The sentence you are writing now should *repeat a noun* in the last sentence or a previous sentence, if possible. The description is easier to read and quicker to understand if there is logical coherence throughout. Are there any events in the description that do not cohere to others?

4. Consistent Abstraction. The use case should be at a consistent level of abstraction throughout. Mixing abstraction levels (problem domain, interface specification, internal design mixes) will cause difficulty in understanding. Is abstraction consistent?

5. Consistent Structure.
   5.1. Variations: Alternative paths should be excluded from the main flow. Inclusion of alternative paths in the main flow reduces readability.
   5.2. Sequence: Numbering of events in the main flow should be consistent. Are there any inconsistencies?

6. Consistent Grammar. Simple present tense should be used throughout. Adverbs, adjectives, pronouns, synonyms and negatives should be avoided. Have they been used?

7. Consideration of Alternatives.
   7.1. Separation: There should be a separate section for any alternative/exceptional paths to the main flow.
   7.2. Viable: Alternatives should make sense. Are they viable?
   7.3. Numbering: Alternative numberings should exactly match the numbers in the main flow. Do they or is there inconsistency?

Table 2-1 shows how the 7 C's relate to the ideas described in this chapter.

The notion of *Coverage* introduces the attributes *Scope* (too much information) and *Span* (not enough information) (Jackson 1995). These can be viewed as finer-grained completeness, a suggested quality good use cases should portray (Achour et al. 1999).

7 C's of Communicability		Rationale
Coverage	Scope Span	Jackson (1995) – refined notions of completeness for requirements.
Cogent	Text Order	Gernsbacher (1996, 1997) – structure building framework Graesser et al. (1996) – inference building (Question -> Reply to Question)
	Dependencies	Jacobson et al. (1992), OMG (2001) – representing a complete transaction. e.g. Trabasso et al. (1989), Goldman et al. (1996) – local and global coherence. Garnham and Oakhill (1996) – referential continuity. Graesser et al. (1996) – inference building (Question -> Reply to Question)
	Rational Answer	Gernsbacher (1996, 1997) – structure building framework Grasser et al. (1996) – inference building (Question -> Reply to Question) Anda et al. (2001) – the realism of the use case
Coherent		e.g. Trabasso et al. (1989), Goldman et al. (1996) – local and global coherence.
Consistent Abstraction		Anda et al. (2001) and see section 2.3.1 for arguments on this.
Consistent Structure	Variations	Kulak and Guiney (2000), CREWS – keep variations to a separate section.
	Sequence	Schneider and Winters (1998), CREWS – consistent sequential numbering
Consistent Grammar		e.g. Pooley and Stevens (1999) – avoid passive voice; the consensus is there are many grammatical elements to avoid (section 2.4). Some structures might improve comprehension e.g. Graham (1998) SVDPI.
Consideration of Alternatives	Separation	Kulak and Guiney (2000) – keep variations to a separate section. Alexander (2002b) – failure to deal with exceptions leads to system failures.
	Viable	Alexander (2002b) – failure to deal with exceptions leads to system failures.
	Numbering	Cockburn (2001), CREWS – there should be consistency in numbering.

Table 2-1. Justifying the 7 C's of Communicability

The *Cogent* facet contains three elements: *Text Order*, *Dependencies* and *Rational Answer*. If a use case description follows a logical order, this ought to make the construction of a Structure Building Framework easier (Gernsbacher 1997). It is also the case that if the description's text is in the correct order then inference building is less difficult, for instance, by application of the Question -> Reply to Question sequence (Graesser et al. 1996). *Dependencies* can be defined as determining the completeness of the description; that is, that it describes a complete transaction (Jacobson et al. 1992). At

a finer-grained level, each event has a dependency on a previous event (logical coherence (e.g. Goldman et al. 1999)), this implies referential continuity (Garnham and Oakhill 1996) and inference building (Question -> Reply to Question) (Graesser et al. 1996). *Rational Answer* considers the notion of plausibility as first suggested as part of an independent means for assessing the quality of a use case description (Cox and Phalp 2000b) and further used by Anda et al. (2001) in use case assessment – they call it "realism" (p.409). The following example presents something that might not appear rational,

1. The Customer inserts card into ATM
2. The Customer selects Change PIN.
3. The Customer enters new PIN with a sledgehammer.

The prepositional phrase in event 3 'with a sledgehammer' is not considered rational and this will have an effect on any mental structure built, forcing the creation of a new structure (Gernsbacher 1996). Inference generation (Graesser et al. 1996) is also broken – the reader is unlikely to infer the use of a sledgehammer to work the keypad! Though the use of a sledgehammer is irrational, it also raises design and security issues for the ATM. For instance, is the ATM capable of withstanding a hit with a sledgehammer, are there ways in which the ATM can be broken or vandalised, what about the safety and security of valid customers? These are clearly exceptional events that need to be flagged and considered if the system under design will be able to cope with these potential problems (Alexander 2002b).

The *Coherent* facet implies good local coherence and global coherence for better understanding of the text (e.g. Trabasso et al. 1989).

Arguments surrounding the consistency of *Abstraction* in use cases and scenarios are stated in section 2.3.1 and so not repeated here, suffice to say this author would avoid mixing internal design and external design events in the same description. Anda et al. (2001) consider the level of detail in use case descriptions as important, though not only

would they remove any internal design detail (as would this author), they also disallow detail of the user interface. For an ATM, for instance, this might be problematic. Where does the Customer place his card? Leibundgut (2002) does not recommend that features of the interface be ignored in use case descriptions if this feature is typical of that kind of interface.

Consistent Structure asks that *Variations* be kept out of the main flow of events. There appears to be a consensus on this, (e.g. Kulak and Guiney 2000). *Sequence* asks that each event be correctly numbered, as recommended (e.g. Achour et al. 1999, Schneider and Winters 1998).

There is general consensus about the avoidance of certain *Grammar* constructs in descriptions, for instance, the passive voice (e.g. Pooley and Stevens 1999) and that certain grammar structures are recommended (e.g. Graham 1998, Achour et al. 1999, Cockburn 2001).

Consideration of Alternatives suggests there be *Separation* (a separate section) for alternative and exceptional flows (e.g. Kulak and Guiney 2000). The viability of these exceptions and alternatives is important. Failure to consider them could lead to system failure (Alexander 2002b). *Numbering* is a check to make sure that the alternative and exceptional paths listed correspond exactly to their equivalent in the main flow. For instance, if event 4 in the main flow has the Customer asking for a receipt and the alternative has the Customer doing something else, the numbers should correspond: 4 in the alternative to 4 in the main flow. Slight variations are also accepted, such as a4/4a (alternative) or e4/4e (exception) (Cockburn 2001). However, 4.1 as an alternative to 4 is not.

The following section discusses the second key aim of this thesis: how have use case descriptions been used in moving towards design? The section considers the guidance suggested in the literature for this.

2.8 Guidance for Use Case Descriptions in Analysis and Design

"The behaviour of a system should be exactly that which is required to provide the use case to the users of the system. In order to evaluate whether that use case has been provided, we should know *what* functionality was allocated to *which* class or classes for each use case." (Insfran et al. 2002, p.67).

Use cases are often used to find classes when moving from requirements analysis to design. How this is done, though, is not entirely obvious. Indeed, Berrisford (1998) states: "can anything be less object-oriented than a use case?" (p.6). Historically, though noun / verb searches were used to find classes and objects in the problem domain (as described in section 2.4.2 on natural language specification), they were considered unscalable to large systems (Rubin and Goldberg 1992) especially if there are large amounts of textual documentation (Lubars et al. 1993), and use cases produce large amounts of text. Approaches were offered to alleviate this problem. For instance, Jacobson et al. (1992) developed their object 'robustness' model by noun / verb searches to bring scalability. Rosenberg (1999) performs a typical noun search on descriptions to find domain objects in the robustness model and then a verb search to find object operations, as do Arlow and Neustadt (2002) who also employ the class responsibility collaboration approach (Beck and Cunningham 1989). Pais et al. (2001) expand on the robustness model by adding numerous further object types. Fernandes and Machado (2001) are more prescriptive. They allocate three objects (entity, interface and control) per description regardless of its length or complexity. Ying (2001) proposes an entity model between the description and class model using stakeholders as validators. Satzinger and Orvik (2001) suggest a noun search and brainstorming sessions. Jacobson (2001) later proposes that engineers do almost nothing because "objects naturally fall out of use cases," (p.xiii). Though improving robustness, applying prescriptive or typical noun/verb searches are very beneficial, the task of moving from use cases to design classes is neither obvious nor simple. Analysts still rely mainly on heuristics acquired through experience and "very few tools exist... in making the transition from... use cases... to class diagrams" (Overmyer et al. 2001, p.402).

The notion of questioning use cases and scenarios to find classes, objects and their components is not new. Rubin and Goldberg (1992) explore scripts (a form of use case) to find initiator and participants (behaviours) of objects, asking,

> "Which roles and responsibilities are needed to accomplish the required task? Furthermore, we must answer why a particular object exists, why it is linked to another object, why a particular service is provided by an object, and how the object participates in fulfilling the functional requirements." (p.49).

These are typical questions that engineers ask of their models, for instance, in terms of determining class responsibility at an analysis level and in terms of coupling and cohesion in design. van Lamsweerde (2000) uses why and how questions on goals to help locate domain objects and their attributes. Robertson (1995) uses propositional analysis on scenarios to examine their structure to find the 'obvious' classes and operations in the problem domain and then questions them with 'wh-' interogatives. This yields further details about the objects and starts a generalisation into classes. Robertson applies general questions focussed in the problem domain.

Sutcliffe (1998) uses a questioning approach to explore and validate user goals and system dependencies for each event in the scenario to compare against the specification. Sutcliffe builds on Potts's Inquiry Cycle model (Potts et al. 1994) which examines dependencies between scenario events to compare against the specification. Sutcliffe goes further by providing sets of questions. There is no explicit questioning to locate classes and other elements of design. Sutcliffe interrogates goals, inbound event dependencies, output requirements analysis, social impact analysis and stakeholder analysis. Examples are, respectively,

- "Does the user's goal require computerised support?" (p.56) If yes, then a function is needed in the specification; if no, this could be a non-functional requirement.
- "Is there a system function to deal with [each event]?" (p.56). If not, a functional requirement should be added or elaborated.

- "... for each component that produces output, is there a corresponding user requirement for information in the scenario?" (p.59) If there is not, checks should be made that the output is correct or needed.
- "Does system output trigger an activity that a human agent is responsible for? This indicates a direct command." (p.61)
- "Will the new system de-skill their job?" (p.61) Automation is often blamed for this.

This thesis does not concern itself with user goal, social impact or stakeholder analysis. However, dependency checking is important. Examination of dependencies between events that are system focussed is important. Stakeholders – actors – should be examined but only in the context that the necessary actors are identified and that they conduct the necessary tasks (and their impact upon dependencies).

Other texts provide heuristics or questions for structuring the use case model, such as how to find the right actors (e.g. Armour and Miller 2001, Dennis and Wixom 2000). This approach is very useful to ensure correctness of the use case model but limited in enabling design.

Insfran et al. (2002) propose that system internal responses be described to make a three column use case description format. For them, the system response "is the most important part of the use case specification because it is the basis for building the conceptual schema," (p.66). Since this author considers internal design to be an issue for design, and not requirements engineering, exploration of internal design should occur in the design phase. Thus, to help in the movement from requirements analysis to design there should be some guidance in doing this. A question-oriented approach appears to be a favourable one to take. The process of doing such work is described by Insfran et al. (2002), "The technique consists of walking through the use case specification... the designer has to focus its attention on only one use case and in particular on one step of the involved use case at a time," (p.67).

2.8.1 An Identified Problem

The application of use case descriptions to the design of software systems should be more obvious than it apparently is. There is the notion that questioning descriptions will yield relevant information. This has typically been the case in locating domain objects (Jacobson et al. 1992). The identification of design elements has been considered in terms of the way objects collaborate (e.g. UML's sequence and collaboration diagrams (Booch et al. 1999)). However, there is the underlying assumption that the objects are those suited to internal design and that they have been pre-determined. If this is not the case, then some further exploration of use case descriptions needs to occur. It seems that taking an interrogating approach to allocate each event to relevant classes or other elements should be necessary to assure completeness. It is therefore worth considering further what specification and design details can be elicited from use case descriptions to help develop a heuristic set to interrogate descriptions.

2.9 Discussion

This chapter has examined various elements of use cases. The opening sections (2.1 and 2.2) considered general aspects of the use case and scenario. Section 2.3 explored the CREWS notions of scenario types and discussed abstraction levels to further define use cases. The chapter then focussed on the details of use case writing from the perspective of the software engineering community (section 2.4). This identified that there has been relatively little work in this area despite the call for systematic guidance in scenario construction (Jarke et al. 1998). Section 2.5 explored the work of the discourse processes community to identify aspects that enable better text understanding. The notions of logical coherence and adjacency pair were considered to help in writing a more comprehensible description. Section 2.6 continued this discussion by identifying five metaphors of discourse processes that might be important to the way use cases are written:

- Mental foundations must be laid so that further events in the description can build upon those foundations.
- Nouns and actions in the description should refer to previous nouns and actions.
- Descriptions should not be overly complex syntactically and should not be overly rich in detail.
- Through the application of coherence and the adjacency pair, along with a relatively simple syntax, the description should be more comprehensible.
- There should be local and global coherence and inference in a description and sentences that do not cohere or infer should be kept to a minimum. This leads to a more communicable description.

Section 2.7 introduced the 7 C's of Communicability that describe qualities that make a good use case description and shows how the literature relates to this. Section 2.8 considered the role of the use case description in enabling design and finds that there is a detailed lack of guidance in moving towards design but that,

- A questioning approach appears to be very useful and,
- The use case description needs to be examined in depth, in other words, one step or event at a time.

2.9.1 Problems to be Solved

2.9.1.1 Writing

As suggested in this chapter, there is little guidance in writing comprehensible use case descriptions. This is an aspect of the use case model that appears to have been ignored to a large degree by many authors. However, this lack of guidance has also been identified as a problem that needs resolution. Jarke et al. (1998) state that there is a need to provide heuristics in writing semi-formal, yet understandable, use case descriptions. The CREWS Guidelines are a noble attempt to address this problem. However, there are some queries over their Guidelines (see sections 2.4.1.1.1 and 2.4.1.1.2) and there has only been a

limited amount of experimentation to test the CREWS Guidelines (section 2.4.1.2). This suggests some problems with them (Cox and Phalp 2000c), though Guidelines do seem to be of value (Anda et al. 2001). Thus further exploration is warranted. From the little guidance afforded in the literature on writing a comprehensible description, it is this author's suggestion that guidelines can be employed to write better use cases in an attempt to address some of the research questions as outlined by Jarke et al. (1998). The exploration of discourse processing indicates there are ways in which text can be structured to enable better comprehensibility. Section 2.7 identified general qualities that make a good description. These elements should be combined to produce writing guidelines to help construct a more coherent use case description.

2.9.1.2 Moving towards design

Section 2.8 shows there is an identified gap in guiding the designer in exploiting the use case description for elements of specification and design (section 2.8). The second aim of this thesis (section 1.4) is to provide guidance in helping the designer exploit use case descriptions for design. A means of closing that gap would be one that questions each step in a use case description about those elements, especially in the identification of elements of design classes. This thus draws on the suggestions that authors have made, for example, the consideration of dependencies (Sutcliffe 1998), by taking a questioning approach (Armour and Miller 2001) to each event in the description (Insfran et al. 2002).

The next chapter discusses the methodology this thesis takes in trying to solve the stated problems.

Chapter Three

Methodology

3.1 Motivation for Empirical Research

Though Fenton (2001) argues that "a truly 'empirical' basis for software engineering remains a distant dream" (p.195) he is concerned that software engineering methods are being adopted based solely upon who "shouted the loudest" (p.195). Without empirical evidence, the software community will continue to believe in and use methods and techniques, such as use cases, based on "common wisdom" (Pfleeger 1994, p.17), "blind faith" (Fenton 2001, p.196) or 'hype' alone (Glass 2000). Therefore, an empirical mode of research, such as experimentation, is not only valid but is important to explore the suggestions made in this thesis (Tichy 1998). Jarke et al. (1998) recognise that requirements engineering techniques must also have an empirical base to test assumptions or theories against, stating that "scenario [and use case] capture and generation must be grounded in empirical fact," (p.166). There is a lack of empirical evidence that reports the merits and weaknesses of techniques particularly used in the requirements phase, though Alexander (2000c) provides empirical evidence to show that use cases are a better means for locating exceptions than by evaluating requirements statements alone. Scenarios have also shown to be more effective than checklists in software inspection (Miller et al. 1998). Chapter 2 (sections 2.4.1.3 and 2.8) indicated that there are certain problems with use case descriptions, namely a lack of guidance and structure in their usage in writing and design. Thus a motivation for an empirical approach to assessing use case descriptions is to add to the body of knowledge that already exists and not to assess by hype alone.

Typical empirical research methods for software engineering are surveys, experiments and case studies (Fenton and Pfleeger 1996). Other approaches to conducting research are that of ethnography (Hughes 2000) and action research (Zuber-Skerritt 1996). The applicability of these research methods to this work is discussed in section 3.3 onwards through experiments, case studies (section 3.4) and surveys (section 3.5). Section 3.6

discusses the ethics of the research conducted and section 3.7 summarises the chapter. The following section (3.2) outlines some issues in conducting research.

3.2 Research issues

In a critique of research methods employed in information systems (IS), Galliers and Land (1987) suggested that the typical "transfer of research suited to the science laboratory to the study of IS is almost always doomed to fail" (p.901). This might also be considered the case in software engineering. Potts (1993) suggests two approaches to conducting software engineering research. The typical approach is "research-then-transfer" (p.19) and Potts sees this as a reason why much of software engineering research is not used in industry; that is, a large body of laboratory work might prove a theory but might not be of much use to practicing software engineers. This is re-iterated by Davis and Hickey (2002) who state that requirements engineers ought to acquaint themselves more closely with practitioners if they want their research to have any effect on practice. Of course, there has to be some amount of laboratory research to initially test theories, heuristics and hypotheses. Indeed, "the generalizability of a theory is often tested much more effectively and efficiently in a laboratory than in a lifelike setting" (Jarvenpaa 1988, p.1503).

Galliers and Land (1987) suggest that other approaches are of value to IS research as well as laboratory study. These being (among others) subjective / argumentative (based more on opinion and speculation than observation), descriptive / interpretive (which is a description of events from the researcher's viewpoint) and action research. These carry the risk of researcher bias. This, in some way, shapes the second, preferred research perspective of Potts (1993). He suggests that research be conducted as "industry-as-laboratory" (p.24), typically conducting case study-oriented research to assess the effectiveness of the proposed approach via an 'experiment – evaluate – modify' iteration. It might also be that much of the evaluation is based upon the researcher's interpretation, description and subjective opinion of how a theory, heuristic or tool, for instance, performs in a real-life environment, and how that tool might be altered to better suit that

particular environment. Thus within Potts's two views of research, there are various types of research method that describe and reflect on a study. This is the case for this thesis, as is discussed below.

3.3 Experiments

Experimentation can be viewed as 'research in the small' in that experiments are conducted in often artificial settings (a laboratory or a classroom, for instance) and are highly constrained, for instance by time limits and availability of subjects. However, they have the advantage in that they are planned and controlled which allows the experimenter to predict the outcome of the experiment (through hypothesis generation). Two risks of conducting requirements engineering-oriented experiments, as indicated in section 2.5.1, are, first, the lack of problem domain knowledge from experimental subjects, and second, that the experiment is conducted out of the context of the problem environment. These potentially introduce external variability. They are controlled to a degree by providing subjects with familiar tasks.

Since this thesis proposes heuristics, it is necessary to test them through experimentation to provide an empirical base, and to determine whether it is then worth applying them to a case study. However, it is wise to test the proposed experimental structure and the suggested heuristics in a pilot study (Glass 1995).

Chapter 5 describes a pilot study that explores the feasibility of conducting experiments for writing heuristics and for their comprehension. The pilot showed that the experimental structure is acceptable and so is expanded upon by two larger experiments in chapter 6, in terms of writing guidelines and chapter 7, which explores the other issue from the pilot on how use case descriptions are comprehended and what kind of information can be retrieved from them.

Figure 3-1 shows the choice and order of research methods used to test and asssess the heuristics proposed in this thesis through the pilot and subsequent experiments.

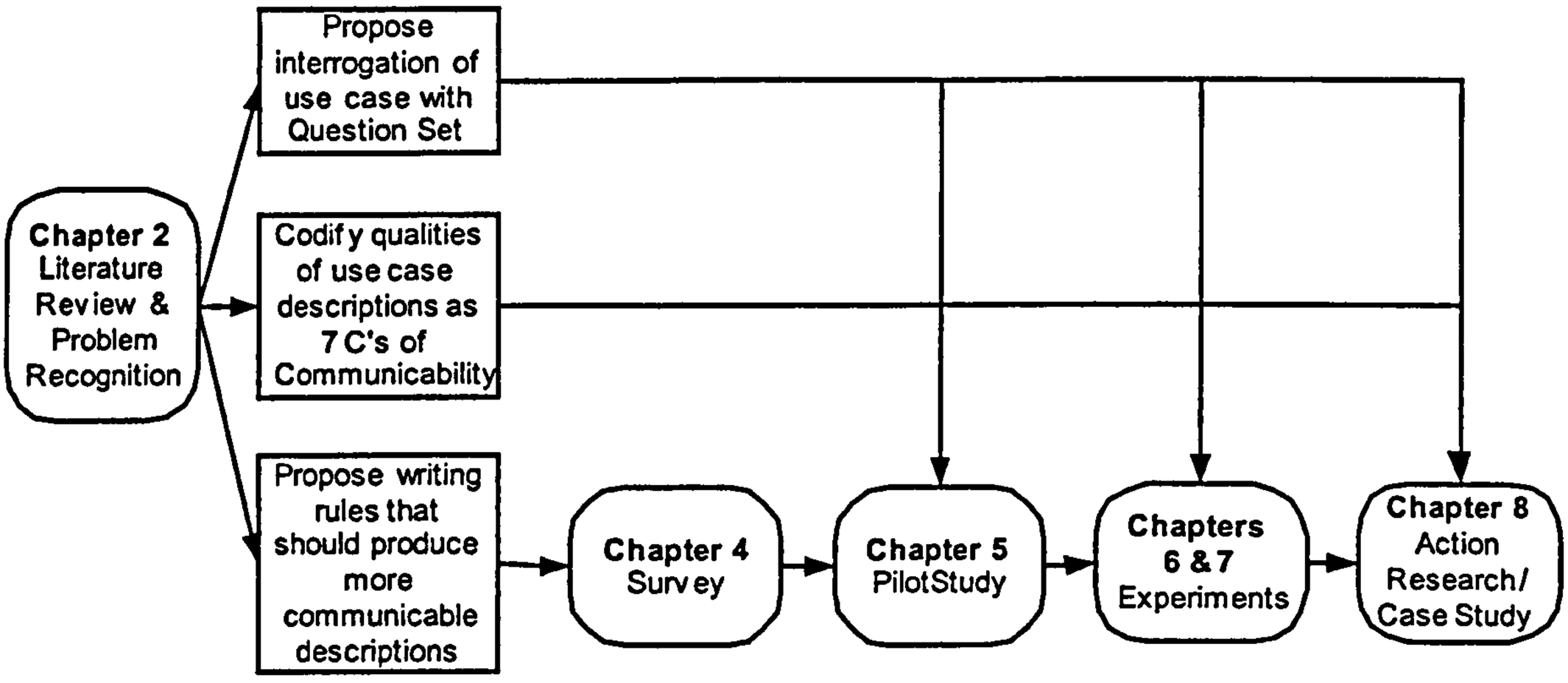


Figure 3-1. Overview of Research Methods Used

3.4 Case Studies

Case studies can be considered as 'research in the typical' because the researcher is amongst practitioners, observing how their techniques are used in real situations. Thus there is context to the study. However, there are risks because the researcher often is constrained by (potentially unpredictable) project events outside his control.

The influences of such project events can have an impact on the case study approach taken (Galliers and Land 1987). If one takes the DESMET approach (Pickard 1992) to case study design, hypotheses similar to that of an experimental design should be stated and the case study ought to be conducted like an experiment. This is not always possible because of confounding factors in the real world. Galliers and Land (1987) note workplaces are stressful environments that are noisy and do not provide complete information and do not lend themselves to an entirely controllable experiment. Yin (1994) has a slightly 'softer' approach. He proposes five components of a case study design:

"1. a study's questions,
2. its propositions, if any,

3. its unit(s) of analysis,
4. the logic linking the data to the propositions, and
5. the criteria for interpreting the findings." (p.20).

These allow for more flexibility in real-world environments, though it is still constrained by a framework. Chapter 8 takes this approach. The case study also contains large elements of the other suggested research approaches, especially action research and ethnography.

As figure 3-1 shows, it is the intention to apply all the proposed techniques in an industrial setting once they have been tested in experiments and considered worth studying further. Although there is a lack of control to their application in this setting, it is necessary so as to see whether the techniques stand up to the rigours of a real project and whether they might need refinement or reconsideration.

3.4.1 Action research

Action research "should not only be used to gain better understanding of the problems which arise in everyday practice, but actually set out to alter things" (Denscombe 1998, p.57). Action research should be practical, dealing with real world problems. Typically, action research has a cyclical nature such as, participate – evaluate – modify – participate etc. In this thesis, the case study (chapter 8) also does this because the author examined the situation 'As-is' and then proposed the 'To-be' situation as part of the project. In other words, the author was not a passive observer (in an ethnographic sense – section 3.4.2) and conducted much of the work himself as part of the project team. This is action research. The justification of this approach is that the case study company's project team did not have time to apply the techniques themselves due to project time constraints. Thus the author had to apply the techniques to the problem himself.

3.4.2 Ethnography

"Put simply, ethnography is a method of social investigation which involves a fieldworker studying some situation as a 'real time, real world' organisation of activities" (Hughes 2000). Davis and Hickey (2002) imply that "by its nature, requirements engineering is the investigation of how people do things currently" (p.107). Indeed, it strikes this author that the differences between ethnography and requirements engineering are not vast. Indeed, ethnography has been proposed as a tool for requirements engineering (Hughes et al. 1995) and for conducting software engineering research (Sim et al. 2001). Ethnographers examine 'As Is' situations, as should requirements engineers (though they are often driven to only describe the 'To Be' perspective because of project constraints). Ethnographers do not appear to have a standard representation mechanism, and neither do requirements engineers (what representation format goes into a requirements or specification document is often very loosely constrained or organisation dependent (Melchisedech 1998)). However, since the author was not a passive observer but actively involved in the case study, his approach is more action research than ethnographic.

3.5 Surveys

A survey can be defined as viewing "comprehensively and in detail. In another sense it refers specifically to the act of 'obtaining data for mapping'" (Denscombe 1998, p.6) and can be seen as 'research in the large'. The survey tends to quantitative empirical data and is wide-ranging. The survey's inherent disadvantage is that the sample does not always represent the population, for example, because of bias in sampling, in that the respondents differ from non-respondents. In any case, it was considered necessary to conduct a survey of grammar used by a number of authors and students to their use case descriptions to determine whether there were any grammar structures that occurred regularly and from a number of sources. If a grammar structure was to meet this aim, then it would be worth considering as part of the proposed writing rules. The survey is detailed in chapter 4, as seen in figure 3-1.

3.6 Ethics

Raven (2000) states that ethical considerations have become routinised in psychology; this might also be the case in software engineering. The pilot study and experiments reported in this thesis are student-based. The students were asked whether they wished to participate in the pilot and experiments prior to them occurring. Though this has the notion of informed consent, it might be considered that the students felt they were obliged to take part (Storey et al. 2001, Sieber 2001). The introductions to the experiment topics (use case descriptions and analysing descriptions for design) occurred during their normal lecture time. The subsequent experiments also occurred during the students' timetabled seminars. Thus they might have felt obliged to participate, though they were given the opportunity to withdraw yet again. Perhaps this author routinised, as Raven suggests, the consent issue. However, the students were informed during the whole experimental procedure, were only encouraged to participate (some chose to withdraw) and were given feedback about the pilot and experiments.

The industrial case study also raises some ethical issues. In this particular case, the author was asked not to disclose the name of the company. Thus it is called Company X and the project the author worked on is referred to as Project F. Nor was the author allowed to disclose the names of the servers used by the company's software applications. Generic names such as 'Front Office' and 'Print Server' were agreed as acceptable. The author had been trusted (Andrews and Pradhan 2001) with fine details of the workings of the company and its software products so care was taken not to disclose anything that was not permitted. Structured presentations and interviews were conducted with members of the IT team regarding the case study. Permission was granted to present and to interview and to use any feedback as part of this thesis (Sieber 2001). The only restriction was that subject anonymity be maintained. It was agreed that the subjects' job roles be stated since this had an impact on the research; that is, different job roles had different viewpoints about the project, the IT department and the company, thus presenting a more complete view of the problem.

3.7 Summary

This chapter has outlined the typical approaches to conducting research. It reports that this thesis uses a survey because the author wants to establish if there are common grammar structures that might be employed in a set of writing guidelines. A pilot study is chosen to assess whether its structure could be used in a larger experiment and to see if any of the heuristics might need some revision before submitting them to an experiment. Experiments are conducted to test the heuristics since it is necessary to present some empirical evidence as to their effectiveness. The heuristics are then applied in an industrial case study in an attempt to assess them in a real setting. There are also some ethical considerations. The author was careful to obtain consent from students and employees before conducting and presenting the research.

Chapter Four

The CP Use Case Writing Rules

4.1 Introduction

It has been suggested that there is little guidance on how to produce use cases and scenarios (Jarke 1999, Cockburn 2001, Anda and Jorgensen 2000). Upon examination of the literature (chapter 2) this appears to be the case. Only the CREWS project has presented a full set of authoring guidelines (Achour et al. 1999); Anda et al. (2001) present a modified version of these based on the findings of Cox and Phalp (2000c). Theories of discourse processes show that there are sentence structures that can improve text comprehension and speed understanding. These need to be captured in a set of writing heuristics. The proposed 7 C's of Communicability (section 2.7) describe qualities that combine to make a good use case description. One of the key qualities is Consistent Grammar. This goal needs to be expanded upon to provide more detailed guidance to the use case writer. From this, the author proposes the CP (Cox and Phalp) Use Case Writing Rules.

The chapter takes the following structure: the next section (4.2) shows how the use case writing heuristics are derived from the 7 C's of Communicability and other aspects of use case descriptions, as described in chapter 2. Section 4.3 describes a survey of grammar structures used by a number of authors to write use case descriptions and scenarios. This shows how often grammar structures have been used. Section 4.4 introduces the CP Use Case Writing Rules with examples of their usage. Section 4.5 discusses the outcomes of this chapter.

4.2 Developing Use Case Writing Heuristics

Table 4-1 proposes a list of heuristics for writing use case descriptions. Their derivation is shown from the 7 C's of Communicability and from other sources. The table also describes if and where the proposed heuristics are also found in the CREWS Guidelines

(section 2.4.1.1). It is shown that many potential style heuristics are also present in the CREWS Style Guidelines but that there are only two references to their Content Guidelines (points 12 and 13). One goal of this research is to refine the CREWS Guidelines and such a potential refinement is the identification of fewer required grammar structures. There are also a number of heuristics that are not contained in CREWS, some of which are derived from aspects of discourse processes, as discussed in section 2.5.

Proposed Writing Heuristics		Derivation (7 C or other)	Contained in CREWS?
1	Each sentence should be on a new numbered line.	e.g. Fowler (2000), Bray (2002), Consistent Structure	Yes (SG1, SG2)
2	Alternatives/exceptions should be in a separate section and sentence numbers should agree.	Consistent Structure Consideration of Alternatives	Yes (SG2, SG3)
3	Use present tense	Consistent Grammar	Yes (SG5)
4	Avoid pronouns	Consistent Grammar, Cogent	Yes (SG4)
5	Avoid adverbs	Consistent Grammar, Cogent	Yes (SG6)
6	Avoid adjectives	Consistent Grammar, Cogent	No
7	Only use negatives in alternative / exceptional flows of events	Consistent Grammar	Avoid negatives throughout (SG6)
8	Give explanations (where necessary provide further details)	e.g. Carroll (1995b), Fowler (1997)	No
9	Ensure coherence (local and global)	Coherent	No
10	Always employ the adjacency pair (Question -> Reply to Question)	Coherent, Cogent	No
11	Show related use cases by underlining their names	Cockburn (2001)	No
12	Allow only atomic events: subject verb object	Consistent Grammar; Graddol et al. (1994)	Yes (SG1, CG5)
13	Allow only atomic events: subject verb object prepositional phrase	Consistent Grammar; Cockburn (2001), Graham (1998)	Yes (SG1, CG1-3)

Table 4-1 Deriving Use Case Writing Heuristics

This section discusses each element of table 4-1 in turn.

1. Each sentence should be on a new numbered line. To adhere to a semi-formal structure (Jarke et al. 1998), it makes sense to place each sentence (event) separately and number it. This is a commonly held view amongst use case writers (e.g. Fowler 2000, Cockburn 2001) and relates to the 'C' *Consistent Structure.*
2. Alternatives/exceptions should be in a separate section and sentence numbers should agree. This relates to two of the 7 C's, namely *Consistent Structure* and *Consideration of Alternatives*. *Consistent Structure* asks that *Variations* (alternatives and exceptions) be kept in a separate section. *Consideration of Alternatives* checks there is this separate section and the *Numbering* attribute asks that numbering between the main flow and the alternative / exceptional flows is consistent.
3. Use present tense. This heuristic relates to the *Consistent Grammar* facet. This recognises that passive voice should be avoided. Since use case descriptions describe more general occurrences than scenarios, their language ought to report what happens in the ideal present.
4. Avoid pronouns. This is derived from the *Consistent Grammar* facet and from the *Cogent* facet. This means that proper nouns should be used. The risks of misunderstanding or misusing pronouns are described in section 2.5.3.
5. Avoid adverbs. An adverb is an element of *Grammar* that introduces subjectivity and therefore potential uncertainty, which should be avoided (perhaps raising an issue as to whether it is a *Rational Answer*).
6. Avoid adjectives. An adjective is an element of *Grammar* that introduces subjectivity and therefore potential uncertainty, which should be avoided (perhaps raising an issue as to whether it is a *Rational Answer*).
7. Only use negatives in alternative / exceptional flows of events. The CREWS Guidelines (Achour et al. 1999) recommend that negatives be avoided. This is sensible for the main flow of events where negation brings potential confusion and alternative events, thus there is a question of *Consistent Grammar*. However, negatives might need to be employed in alternative / exceptional sections to help describe the reason for the alternative or exception (Cockburn 2001).
8. Give explanations (where necessary provide further details). Certain use case and scenario writers (e.g. Carroll 1995b, Fowler 1997) provide different levels of detail in

descriptions. For instance, there are occasions when further explanation of an event in a use case might be required. For example,

> the operator enters the payment details

The reader might not be certain what payment details – they may vary dependent upon the type of customer or the account they hold, for instance. So a small explanation is added:

> The operator enters the payment details (*the payment details are: customer id, telephone number, email*).

9. Ensure coherence (local and global). This equates to the *Coherent* facet of the 7 C's and is discussed in sections 2.5 and 2.6.
10. Always (try to) employ the adjacency pair (Question -> Reply to Question). The application of the adjacency pair (input -> response to input, from the *question->reply to question* pair) should help improve understanding because the reader will expect a response to an input the same as they expect a reply to a question when one is asked. If there is no reply or system response this could confuse the mental model of the reader and further the complexity of the description. This corresponds to the *Cogent* and *Coherent* facets.
11. Show related use cases by underlining their names. When a use case needs to call on another use case the text should represent this in a simple format, instead of the standard UML approach:

> The operator calls the manager (<<extend>> use case Alert the Manager)

The author prefers Cockburn's suggestion (2001) to underline the use case title and have it embedded into the text of a normal event. This also has the benefit of appearing like a hyperlinked document or a web link. So the example would be re-written:

The operator alerts the manager.

12. Allow only atomic events: subject verb object. Graddol et al. (1994) state that the English language is an SVO (Subject Verb Object) language and this heuristic follows that lead. The concern over complexity and misinterpretation of sentences, especially second clauses, is not entirely simple to remove. However, the SVO structure succeeds because it is of singular clause in its definition, for example,

    The User selects Display Balance.

    There is no room for a sub-clause. This structure is also derived from *Consistent Grammar* because of its atomic nature (e.g. Achour et al. 1999, Fowler 2000, Bray 2002).

13. Allow only atomic events: subject verb object prepositional phrase. This allows more freedom of expression than solely SVO. It is derived from *Consistent Grammar* in that it is still an atomic event. The structure is similar to that as recommended by Graham (1998) and Cockburn (2001).

These are the key elements to emerge that address the use case description at the level of its structure. The first two points in table 4-1 relate more to the overall structure, in terms of a template; this is provided to give the writer an outline of where to start and flags the importance of exceptional and alternative flows.

It has been stated that "the contribution of syntax to comprehension is not particularly robust, and thus, people can be led by considerations of plausibility to construct mental models," (Garnham and Oakhill 1996, p.320). However, if there is no syntactic structure then there is only chaos.

The Structure Building Framework (Gernsbacher 1996, 1997) is used here to suggest the use case grammar structures (points 12 and 13 in table 4-1) should be built upon one another. Figure 4-1 describes this.

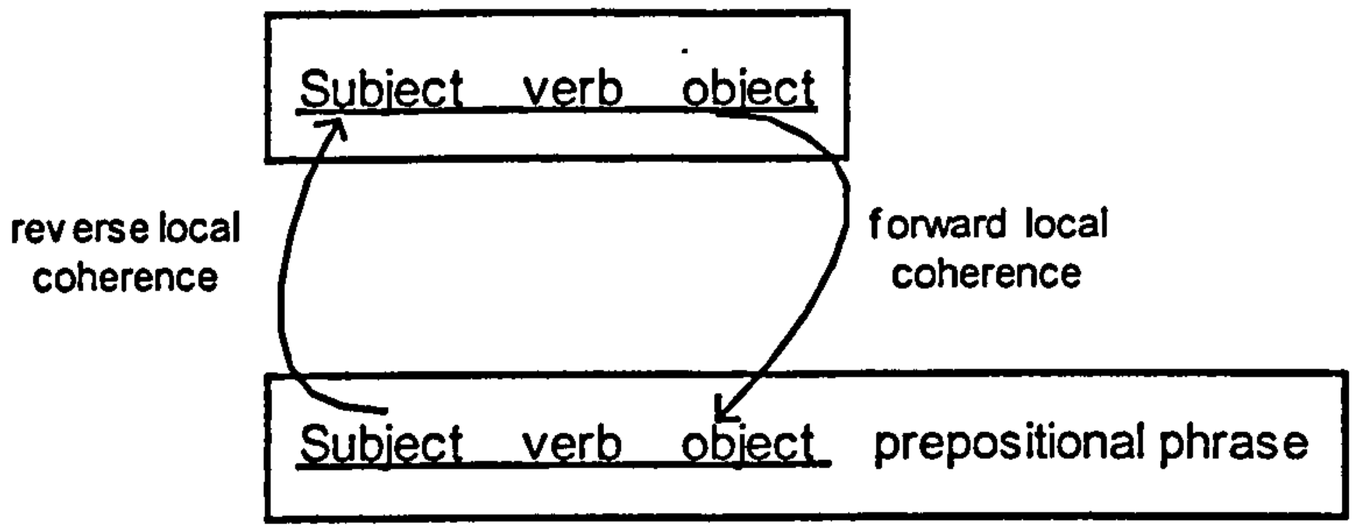


Figure 4-1. Example of forward and reverse local coherence

The first structure *subject verb object* should shape the inferences of the use case writer to predetermine the next structure rule or at least expect something similar. This inference is called *forward local coherence*. The use case writer infers the next structure, at least to a degree, in that the writer has comprehended and understood the previous rule. With the construction of a mental model foundation the writer can use that foundation to help structure the next model element which helps to speed up the process of comprehending the next rule.

So the second structure *subject verb object prepositional phrase* has, as part of it, the *subject verb object* from the first structure. There is, therefore, backward or *reverse local coherence* from the second structure to the first. In terms of Gernsbacher's Structure Building Framework, it is clear that the mental model foundations laid by the first structure are built upon by the second structure.

The next section describes a survey of grammar structures of individual events in use case descriptions and scenarios. Its aim is to identify whether there are any structures that repeat a number of times that might be considered as additions to (or confirmations of) the grammar structure heuristics suggested in section 4.2, notably points 12 and 13 in table 4-1.

4.3 Grammar Survey

When one compares the amount of work published on use cases, it is surprising that there is so little on the description. To make some sense of this, the author conducted a grammatical analysis of 152 use cases and scenarios; some 33 use cases from students (the students had not been exposed to any use case writing guidelines), and 119 from authors (published use cases and scenarios).

The use case descriptions were selected entirely randomly. The author simply sifted through texts and papers searching for descriptions to examine. In terms of the descriptions written by students, these were those written in seminars prior to the students being informed of any guidelines except basic concepts such as numbering events and placing alternative and exceptional flows in separate sections. Table 4-1 shows that these are suggested heuristics that are also contained in the CREWS Guidelines.

The complete set of results for this analysis is in Appendix B. Here a description of the overall results is given. The goal has been to identify the most common grammar structures used and to find if any structures appear to be pervasive to use cases and scenarios so that they might be employed in a writing rule set.

4.3.1 Results

Table 4-2 gives an overview of the results.

Number of use cases and scenarios examined	152
Number of events (sentences) identified	1913
Number of unique grammar structures identified	734

Table 4-2. Overview of number of use cases and events examined

Grammatical Structure	Count	%
Subject verb object	313	16.4
Subject verb direct object preposition indirect object	77	4
Subject verb indirect object preposition direct object	44	2
Verb object	35	1.8
Subject future object	34	1.8
Subject passive	34	1.8
Subject verb preposition object	29	1.5
<<use 'use case name'>>	28	1.5
Subject verb	27	1.4
Subject verb infinitive object	23	1.2
Subject verb object conjunction (and) object	22	1.2
Subject future direct object preposition indirect object	21	1.1
Subject verb preposition subject verb object	17	0.9
Subject passive preposition object	17	0.9
If subject verb object	16	0.8
Then	16	0.8
End if	14	0.7
If subject verb object then	13	0.7
Goto	13	0.7
Subject verb object past participle	12	0.6
Subject preposition object passive	11	0.5
Subject verb direct object preposition indirect object preposition indirect object	10	0.5
Else if subject verb object then	10	0.5
Subject verb preposition subject passive	10	0.5
Subject verb object passive	10	0.5
Verb object passive	10	0.5
End loop	10	0.5

Table 4-3. Most common grammar structures, count and percentage

Schneider and Winters (1998) describe 44 separate, differing use case descriptions. Other authors have just one (e.g. Booch et al. 1999). This certainly reflects in the overall number of types of structures (table 4-2) in that Schneider and Winters have some

structures that repeat a number of times that are not described by other authors (e.g. *subject future object:* the system will display the Entry screen). Table 4-3 describes the most commonly found grammatical structures in the use cases and scenarios.

There is only one grammatical structure whose occurrence is greater than 5%: *subject verb object.* The next is: *subject verb object preposition object* (6% occurrence, combining the second and third listed structures in table 4-3). This structure is similar to Graham's SVDPI (1998) and to Cockburn's *subject verb direct object prepositional phrase* (2001).

It is clear from the variations in grammar structures identified in the survey, some 734 unique structures, that the writers of use cases and scenarios have been fairly liberal in their consistency of structure. It is evident that different authors have their own preferred styles of writing.

Since this thesis proposes that some guidance is provided, the heuristics in table 4-1 will be organised into the CP Use Case Writing Rules (section 4.4) that take into consideration the outcomes of the survey.

The survey suggests that the structures used in writing descriptions and scenarios are very unconstrained, that there is enormous variety. This could hinder their understanding when read. Therefore, a set of writing rules might reduce the variety and constrain authors into producing more communicable descriptions.

4.3.2 Threats to Validity (Wohlin et al. 2000)

4.3.2.1 Population

The population of use case writers would be those who write them in industry and academia. A possible threat to whether this survey reflects current practice is the sample of use cases and scenarios. However, the published use cases and scenarios are from

twenty-four different sources (both academics and practitioners): Kulak and Guiney (2000), Jacobson (1995), Robertson (1995), Britton and Doake (2000), Wiegers (1999), Fowler (1997), Carroll (1995), Neilson (1995), Booch et al. (1999), Achour and Maiden (1999), Graham (1998), Rolland and Achour (1998a), Achour et al. (1999), Achour (1998a), Holbrook (1990), Cockburn (1997), Cox and Phalp (2000a), Fowler (2000), McGraw and Harbison (1997), Pooley and Stevens (1999), Eriksson and Penker (1998), Leite et al. (2000), Rosenberg (1999), Schneider and Winters (1998). The purpose of the survey is to examine writing styles in the hope of identifying common structures. It is the case, though, as stated, that some authors (e.g. Schneider and Winters 1998) provide a number of descriptions and some only a few (e.g. Booch et al. 1999).

4.3.2.2 Nature of the Problem (Host et al. 2000)

The use cases and scenarios describe a variety of domains and instances of use. For this reason, the results are not biased to a specific problem environment but a number of the published descriptions and those from students might be considered artificial (because they are out of context – see below) or 'toy' problems.

4.3.2.3 Setting (Robson 1993)

Use cases and scenarios are usually written in work environments under different pressures to those of authors and students. The published scenarios will in many cases have probably been polished to remove ambiguities and inconsistencies. Those written by students were not. The students were not novice writers – they had written many use cases before this point. Even in real situations it is probable that scenarios are revised and shaped dependent upon new information and customer validation. It is also possible that only one draft was written. There do not appear to be any different structure trends from students compared against authors. There is a wide range of structures used by both groups.

4.3.2.3 Internal Validity

A threat is the mis-identification of grammar. To reduce the likelihood of this, the author used these expert references: Leech (1991), Swan (1980), Murphy (1985), Webster's (1994) and Allen (1984).

4.3.2.4 History

Since this survey there have been further publications about use cases and scenarios, notably Cockburn (2001). If the survey were expanded to include the more recent works (2001 onwards) the results might be different because the use case writers' population would also be different. However, the sample size is large so it is unlikely that there would be dramatic differences to the outcome.

The next section describes the amalgamation of the thirteen heuristic elements described in section 4.2 into a set of writing rules.

4.4 The CP Use Case Writing Rules

The author proposes a set of writing rules from the heuristics described in section 4.2 and table 4-1. These are named the CP Use Case Writing Rules. The grammar survey (section 4.3) confirms that the CP Rules take the structures that occur the most. CP Structure 1 will be subject verb object, which has 16% occurrence. This structure (point 12 in table 4-1) also acts as a foundation to build upon the other CP Structure (subject verb object prepostion phrase), as described in figure 4-1.

In terms of how the grammar survey reflects on the CP Writing Rules and the CREWS Authoring Guidelines, figure 4-2 neatly shows that three quarters of the grammar structures identified were unrelated to those proposed by the CP Rules and the CREWS Guidelines. Figure 4-2 (left) shows the percentage CP Rules. CP Structure 1 has 16% representation, the same as CREWS CG5 in figure 4-2 (right).

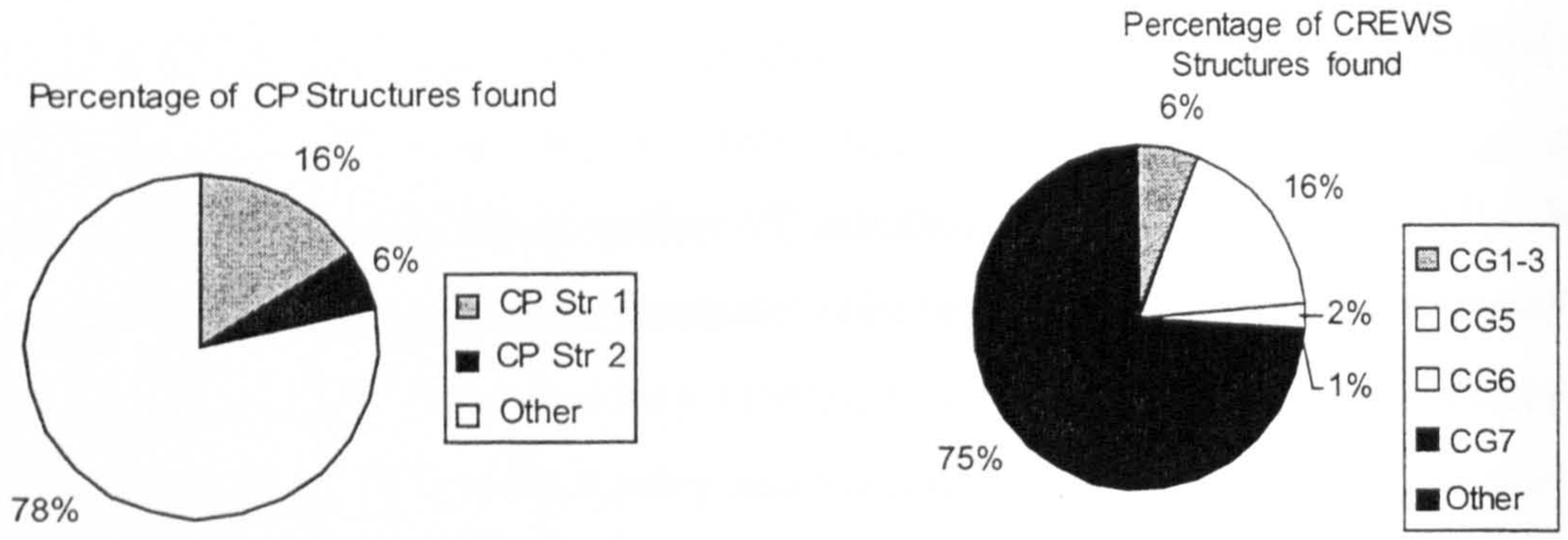


Figure 4-2. Percentage CP Use Case Structure Rules (left) and CREWS Use Case Guidelines (right) found in the survey

Figure 4-2 (left) shows the percentage of CP Structure 2 occurred in 6% of events and CREWS CG1-3 (similar to 'subject verb object prepositional phrase') also in 6% of events (right).

The other CREWS Content Guidelines (figure 4-2 (right)) that appear in the survey are CG6: 'If' <alternative assumption> 'then' <action>, which has an occurrence of 2%. The Guideline is taken to be any form of an *if... then...* statement in terms of the survey. Next is CG7: 'Loop' <repetition condition> 'do' <action>, which has 1% occurrence. CG4 and CG8 do not occur.

This rest of this section describes the CP Use Case Writing Rules as presented to participants in chapter 5:

To help you write use case descriptions, here are some writing rules that you should apply as much as you can. Each rule has an explanation to help you understand it. The rules are in two parts: General Style Rules and Specific Structure Rules.

1. General Style Rules.

These are applicable to all sentences in the use case description.

Style 1: Each sentence in the description should be on a new, numbered line. Alternatives and exceptions should be described in a section below the main description and the sentence numbers should agree. For example:

1. *The patient record appears on the screen.*
2. *The doctor enters the patient's new address.*

Alternatives

2. The doctor deletes the patient's record.

The alternatives go below the main flow and the sentence numbers agree (2 and 2).

Style 2: Avoid pronouns. (E.g. he, she, it, we, their etc). Use proper nouns and names instead. An example:

The patient stands next to the doctor.

He puts the prescription in his pocket.

Who is "he"? Whose pocket is "his"? So write proper nouns / names to be clear:

The patient stands next to the doctor.

The doctor puts the prescription in the patient's pocket.

Style 3: No adverbs or adjectives. These add unnecessary clutter to the description and give values that are difficult to measure. For example:

The doctor writes the prescription slowly.

Slowly is an adverb - we don't need to know how the doctor writes the prescription, just that he writes it.

The patient swallows the big pill.

big is an adjective - again, for a use case description it is unnecessary.

Style 4: Avoid negatives. (E.g. don't, can't, won't, not, no) Tell us what we *can* do rather than what we can't.

Style 5: Give explanations if necessary. Explanations should be enclosed in brackets. For example:

The librarian enters the borrower's details (details are: name, address, phone number, library card number).

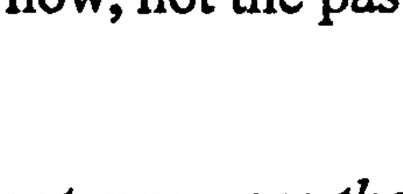
Don't overuse explanations. If you find the use case is full of explanations then you have too much information and have granularity problems - you need to abstract up a level or break the use case down into further smaller use cases.

Style 6: All verbs are in present tense format. The use case should describe events and actions in the here and now, not the past or the future. Some examples of present tense:

The operator presses the button.
The cashier enters the amount.

Style 7: There should be logical coherence throughout the description. That is, the sentence you are writing now should refer to something in the last sentence or a previous sentence, if possible. We understand the use case better this way.

1. The cat sits on the mat.



2. The mat belongs to Fred.

The *mat* in (2) coheres to *mat* in sentence (1).

Style 8: When an action occurs there should be a meaningful response to that action. For example, when there is an input there should be a response to that input somewhere in the use case, usually immediately. This makes sure we do not forget to respond to any action in the use case description.

1. *The doctor enters the patient's record identification number.*
2. *The system displays the patient's record.*

Sentence 2 gives an immediate response to sentence 1.

Style 9: Underline other use case names. When it is necessary to include a use case or have a use case extended by another use case, then write the use case name in the sentence and simply underline it. For example:

The user makes a request to <u>buy new product</u>.

2. Specific Structure Rules.

There are also some structures that you should apply for individual sentences. Again, try to use these structures as much as you can. Note that **verb** refers to present tense as described in **Style 6.**

Structure 1: Subject verb object. For example:

The operator presses the button.

Structure 2: Subject verb object prepositional phrase. Some examples:

The operator gives the tool to the mechanic.

*The builder puts the bricks **on top of the pile of rubbish.***

*The system reminds the operator **to save the open files.***

The **bold** text in the examples are prepositional phrases.

4.5 Discussion

This chapter has described the CP Use Case Writing Rules. It relates the CP Rules to the concepts described in chapter 2. The 7 C's of Communicability describe general identified qualities of comprehensible use case descriptions. This can be seen as 'comprehensible use case descriptions in the large'. To create those qualities in individual events in the use case description, further guidance is necessary. The proposed CP Rules hope to provide these qualities, thus enabling 'comprehensible use case descriptions in the small'. The grammar survey (section 4.3) shows there is an enormous variety in how to structure descriptions. The CP Structure Rules use the most common structures found in the survey to bring some guidance in how to structure descriptions, as is also the intention of the CREWS Guidelines, though it is the case that their Content Guidelines do not fare any better in the survey. The CP Rules first have to be tested and chapter 5 describes a pilot study to explore this.

Chapter Five

Pilot Study

5.1 Introduction

To assess the CP Use Case Writing Rules (section 4.4) a pilot study was conducted to find whether the CP Rules produce comprehensible use case descriptions. They are compared against the CREWS Guidelines (section 2.4.1.1), as opposed to a control group with no guidelines because it has been shown that guidelines produce better quality descriptions than no guidelines, which is a very hard control in practice (Achour et al. 1999, Cox and Phalp 2000b, Anda et al. 2001 – see section 2.4.1.2). The CREWS Guidelines are used as a comparison because they offer a complete set of writing guidelines. This chapter expands upon that reported by Cox et al. (2001).

5.2 Overview

The pilot is conducted in 2 parts:

1. Write a use case description according to the CP Rules or the CREWS Guidelines.
2. Read a use case written in the style of one of these approaches and answer questions on the comprehensibility of the use case completed in the first part.

5.2.1 Aims of the Pilot

This pilot has two main aims,

1. To assess whether the CP Rules have the potential to be a useful set of guidelines. This has a number of sub aims:
   - To assess whether the CP Rules help writers produce use cases in a similar time to the CREWS Guidelines.

- To assess whether the CP Rules help writers produce use cases of a similar length to the CREWS Guidelines.
- To assess whether participants use the CP Rules as often as the CREWS Guidelines.
- To assess whether the CP Rules use cases are as comprehensible as CREWS use cases.

2. To assess whether the structure of this pilot is feasible for an experiment. Sub aims of this are:
   - To determine whether the time allocated for writing the descriptions will be sufficient.
   - To assess whether student participants (as opposed to 'subjects', because this term is more suitable to formal experiments (Robson 1993)) can produce reasonable answers in the allocated time.

5.3 Pilot Design

The pilot takes the form of a preliminary experiment to assess how a larger experiment might later be conducted to formally test the CP Rules. But because this is a pilot, there are no formally stated hypotheses.

5.3.1 Course of the Pilot

Figure 5-1 gives an overview of the order of events.

Phases 1 and 2, the questionnaire and pre-test, occurred a week before the main study. This gave the author time to assess the quality of the participants based upon the pre-test and their experience from the questionnaire in order to allocate participants to treatments.

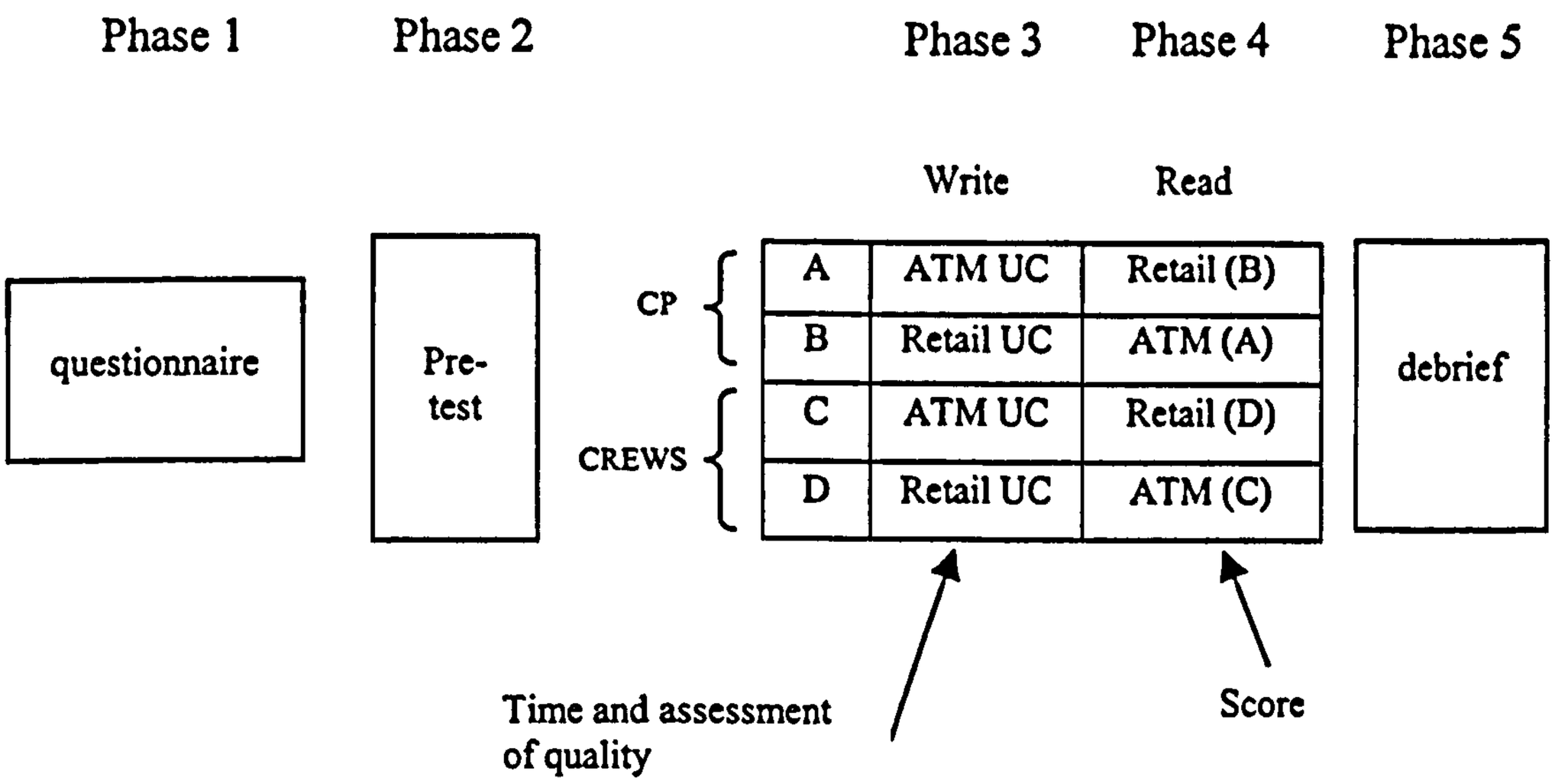


Figure 5-1. Overview of the pilot

5.3.1.1 Phase 1: Experience Questionnaire

The purpose of the experience questionnaire is two-fold. First, to establish the software engineering experiences of the participants both professionally and personally. Second, the questionnaire asks about the English language qualifications of the participants and, third, whether they have used a supermarket and an ATM. The rationale was to provide four groups of 'equal' ability in English and also to get a spread of 'equal' software engineering experience among the groups. Appendix C1 shows the questionnaire.

5.3.1.2 Phase 2: Pre-Test

This asked the participants to write a simple use case description. The goal of the pre-test was to further identify participant writing ability and understanding of use cases (after the introductory half-day seminar). The participants were blocked into levels of ability, experience and qualification and then randomly blocked into groups. The pre-test is in Appendix C2.

5.3.1.3 Phase 3: The Writing Part

The participants were presented with the Guidelines or Rules, without being told their origins; these were read out by the pilot assessors (the author and another lecturer) and questions or queries were answered. The participants were then presented with the tasks. These were read through and again any queries were answered by the assessors alone. The participants were then given 60 minutes to independently write the descriptions. They were allowed to ask questions but only to the assessors. When the time was up, all pilot material was collected before the participants left.

5.3.1.4 Phase 4: The Reading Part

Group A read the descriptions of Group B and B that of A's. Group C read Group D's description and Group D read C's (figure 5-1). This means that each group read a description written with the aid of the guidelines/rules they applied, however, in a different domain to the one they had written. The participants answered comprehension questions (Appendix C3) that related to the description they were reading but which also contained general questions applicable to all use case descriptions. The participants had thirty minutes to answer the questions.

5.3.1.5 Phase 5: Feedback and Debrief

The groups were asked about the study, the tasks and the rules or guidelines they employed. The groups were presented with both sets of writing guidelines and any questions were answered. This feedback phase lasted thirty minutes.

5.3.2 Pilot Participants

The participants were Masters students studying software engineering. There were 25 participants in total. There were therefore three groups of six participants and one group (A) of seven. One participant from group A was excluded from the final analysis, and

also from the second part of the pilot, since the standard of the participant's written English was not high enough, as identified in the pre-test. The participant was allowed to participate in the pilot because this took place in normal timetabled classes. For the second part of the pilot, a photocopy of the use case description written by another participant from Group B (randomly selected) was used. As such, in the final analysis, each group had six participants. The gender of the participants was male dominated (20 males, 5 females). The females were placed in different groups. The participant whose answers were discarded is female. This means that each group contained five males and one female.

The participants received a half-day introduction to use cases. This formed a normal part of their academic course. The participants were able to write practice descriptions and these were discussed as a group. The author was careful to avoid giving advice on the structure of descriptions save that each event should be numbered and that alternatives should be placed in a separate section. Both CREWS (Style Guideline 2) and CP (Style 1) recommend the numbering of events and a separation of alternatives from the main flow of events – no bias was shown to either set.

5.3.3 Tasks

All pilot tasks are found in Appendix C4.

5.4 Results

The next section discusses the sub aims for aim 1 in section 5.2.1. Section 5.4.1 discusses the time taken to write descriptions; section 5.4.2 considers the length of descriptions; section 5.4.3 describes how often the CP Rules and CREWS Guidelines were used and section 5.4.4 explores whether the CP use cases are as comprehensible as the CREWS Guidelines use cases. Section 5.4.5 explores other issues such as the logic of the use case descriptions examined and feedback from the questionnaires.

5.4.1 Time to Write Use Case Descriptions

This sub aim considers whether use cases written with the CP Rules are written as quickly as CREWS use cases. If it takes a much longer time to write descriptions with the CP Rules then they probably need some revision. Table 5-1 shows there is not much difference between the groups. Note that a number of participants used the whole 60 minutes. Upon examination of their descriptions, it was found that all main flows were complete and there were, in general, varying numbers of alternative and exceptional flows.

Gr. A CP ATM	T	L	Gr. C CREWS ATM	T	L	Gr. B CP Retail	T	L	Gr. D CREWS Retail	T	L
A1	60	11	C1	55	30	B1	56	13	D1	60	16
A2	55	26	C2	55	27	B2	60	6	D2	50	13
A3	60	15	C3	57	37	B3	45	28	D3	45	33
A4	45	34	C4	56	19	B4	60	5	D4	50	22
A5	50	35	C5	30	20	B5	60	15	D5	52	13
A6	53	12	C6	60	17	B6	55	24	D6	48	44
Median	54	20.5	Median	55.5	23.5	Median	58	14	Median	50	19

Table 5-1. Time taken (T) and number of steps in main flow (L)

5.4.2 Length of Descriptions

The suggestion is that the CP Rules should produce descriptions of a similar length to CREWS. The results show there is no major difference between the groups (table 5-1).

5.4.2.1 Correlation between Time and Length

There is a suspected correlation between time and length. Table 5-2 shows the Spearman correlation, as opposed to Pearson, which expects data to be on an interval scale (as calculated on DataDesk 6).

There is no significant correlation between time and length of use case description. The Spearman coefficient for 6 participants for a two-tailed test (that there is a significant correlation between time and length), with p =< 0.05, is 0.886 (Greene and D'Oliveira 1982). So although group B has high negative correlation, this is not significant.

Group	Correlation Time and Length
A	-0.725
B	-0.820
C	-0.232
D	-0.691

Table 5-2. Spearman correlation between Time and Length

The notion of length of use case and time taken to write as a measure of efficiency is perhaps suspect. What is missing is a consideration of whether the description is of good quality. That is, does the description provide a solution to the problem, is it well written, is it communicable, understandable? Without explicit consideration of these notions, it is not very probable that one can suggest a great deal from the results shown.

5.4.3 Counts of Rule / Guideline Occurrence

It is only possible to compare similar CP Rules and CREWS Guidelines (table 5-3). Appendix C5 provides the full data sets. Simple counts of the number of participants applying the Rules / Guidelines are shown. The normalised mean is also shown for each misusage (for Style) or usage (for Structure / Content) per participant across the groups (tables 5-4 to 5-9). In other words, each misuse / use made is divided by the number of steps in the main flow. For example, if one description main flow is only 2 steps in length and one step has a non-present tense structure, then the score shows 50 (percent).

CP Rule / CREWS Guideline
Style
Alternatives in main flow
Pronouns / Anaphors
Adverbs
Negatives
Non-present tense
CP Structure / CREWS Content
CP Structure 1 / CG5
CP Structure 2 / CGs 1-3

Table 5-3. Comparable elements of CP and CREWS

5.4.3.1 Alternatives in the main flow

Table 5-4 describes the average number of alternative flows used by the participants across the groups.

Group / Participant	1	2	3	4	5	6	Count of Participants Using
A	0	0	0	0	2.17	0	1
C	16.67	0	0	0	13.04	11.76	3
B	46.15	10	7.89	20	0	8	5
D	15	0	6.38	72.73	0	15.91	4

Table 5-4. Mean Count of Alternatives in Main Flow (shown as percentages)

The count shows that only one participant in group A used alternatives in the main flow. 5 out of 6 participants in group B used alternatives. Both sets of ATM descriptions have fewer alternatives in their main flows than the Retail descriptions. It is unclear why this should be the case since both tasks ask for a variety of actions to take place. However, the problem domain seems to be more important than the rule set.

5.4.3.2 Pronouns / Anaphoric references

Table 5-5 shows the mean count of pronouns that all groups used (though CREWS use the term anaphoric reference, their examples only include pronouns, so only pronouns are counted). It shows there was less usage of pronouns by the CREWS groups (C and D). It is unclear why CREWS did better though it might be the uncommon wording of the CREWS Style Guideline 4 (section 2.4.1.1). This uses the term 'anaphoric reference' instead of pronoun. Such unusual vocabulary might have focussed the attention of the reader more effectively than the common 'pronoun'.

Group / Participant	1	2	3	4	5	6	Count of Participants Using
A	0	12.9	6	20	4.35	0	4
C	3.33	0	0	0	0	0	1
B	7.69	40	2.63	0	0	8	4
D	0	0	0	9.09	3.85	0	2

Table 5-5. Mean Count of Anaphors (Pronouns) (shown as percentages)

5.4.3.3 Adverbs

Table 5-6 shows the mean count of adverb usage across the groups. Only one participant in CREWS groups C and D used adverbs. Whereas, five did in CP (four participants in group B and one in A), possibly suggesting that the format of the CREWS Style Guidelines made its participants more aware of avoiding adverbs.

Group / Participant	1	2	3	4	5	6	Count of Participants Using
A	0	3.23	0	0	0	0	1
C	0	0	4.55	0	0	0	1
B	0	10	5.26	0	3.33	16	4
D	0	0	0	0	0	0	0

Table 5-6. Mean Count of Adverbs (shown as percentages)

5.4.3.4 Negatives

Table 5-7 describes the mean count of negatives. There was a high usage of negatives in all groups. All participants with the retail task used negatives. Negatives should not be described in the main flow since this (generally) represents the 'happy day' scenario. Clearly, this guideline was not well heeded.

Group / Participant	1	2	3	4	5	6	Count of Participants Using
A	3.7	6.45	0	4.44	0	23.53	4
C	6.67	2.13	0	8	8.7	0	4
B	30.77	10	10.53	20	6.67	12	6
D	20	5.26	10.64	27.27	3.85	9.09	6

Table 5-7. Mean Count of Negatives (shown as percentages)

5.4.3.5 Non-present tense

Table 5-8 shows the usage of non-present tense throughout the descriptions. This attribute takes into consideration all tenses that are not present tense.

Group / Participant	1	2	3	4	5	6	Count of Participants Using
A	0	29.03	0	4.44	0	0	2
C	6.67	2.13	1.52	24	21.74	23.53	6
B	0	30	5.26	20	0	4	4
D	15	21.04	36.17	40.91	7.7	25	6

Table 5-8. Non-present tense (shown as percentages)

It can be seen that all CREWS use cases had non-present tenses in them. This could be due to the Content Guidelines allowing iterations and if... then statements, which can be as a result of a negative (for example, if not a then b).

5.4.3.6 Overall Results for Style Comparison

There were some differences between the groups. For instance, CREWS used fewer pronouns and adverbs than the CP descriptions. As stated, the difference in pronoun / anaphora usage might be because of the way the CREWS Guidelines are worded. In some cases, CP did better than CREWS, such as in avoidance of non-present tense. But overall there was little difference between the groups suggesting that the application domain is more important than the rule set. This is because there is the potential for two actors in the retail task (Customer and Cashier) and this would introduce more complex interactions than the one actor in the ATM task.

5.4.3.7 Comparing CP Structure Rules to CREWS Content Guidelines

Table 5-9 shows the mean counts of usage for the two comparable CP Structure Rules / CREWS Content Guidelines. Group A had almost twice the mean usage of CP Structure 1 compared to the other groups. Group B had the lowest usage. Interestingly, groups C and D had fairly similar usage of Content Guideline 5. It can be seen that CP performed as well as CREWS on all counts apart from group B's mean and medians for CP Structure 1. But then group B used CP Structure 2 more than the other groups. This might reflect the nature of the tasks though there is little difference in use between groups A and D of CP Structure 2 / CREWS Content Guidelines 1-3. It was stated in feedback by some participants that the CREWS Content Guidelines were difficult to understand because of their formal appearance.

5.4.3.8 Count of Rule Usage Conclusions

The CP Rules were applied about the same as the CREWS Guidelines. In terms of structure, the CP Structures were applied more than the CREWS Content Guidelines. It appears that the CP Rules therefore seem acceptable in terms of structure but might need some revision in style.

CP Structure 1		CREWS CG5		CP Structure 2		CREWS CG1-3	
A1	74.07	C1	23.33	A1	7.41	C1	10
A2	41.94	C2	65.96	A2	6.45	C2	0
A3	84	C3	54.55	A3	6	C3	6.07
A4	57.78	C4	8	A4	0	C4	4
A5	56.52	C5	21.74	A5	8.7	C5	0
A6	29.41	C6	11.76	A6	11.76	C6	0
B1	0	D1	40	B1	15.38	D1	15
B2	20	D2	26.32	B2	10	D2	21.05
B3	28.95	D3	34.04	B3	5.26	D3	0
B4	0	D4	22.73	B4	0	D4	0
B5	43.33	D5	61.54	B5	36.67	D5	3.85
B6	12	D6	22.73	B6	12	D6	2.27
Group Means				Group Means			
A	57.29	C	30.89	A	6.72	C	3.35
B	17.38	D	34.56	B	13.22	D	7.03
AB	37.33	CD	32.72	AB	9.97	CD	5.19
Group Medians				Group Medians			
A	57.15	C	22.54	A	6.93	C	2
B	16	D	30.18	B	11.00	D	3.06
AB	35.67	CD	24.82	AB	8.05	CD	3.06

Table 5-9. Comparing CP Structure 1 and 2 against CREWS CG5 and CG1-3 (shown as percentages)

5.4.4 Use Case Comprehension

This suggests that CP descriptions are as comprehensible as the CREWS descriptions. Section 5.4.4.1 assesses the use case descriptions by means of the 7 C's of Communicability suggested in section 2.7. Section 5.4.4.2 examines the completed questionnaires.

5.4.4.1 The 7 C's of Communicability

This is a relatively independent assessment of the use case descriptions because it has been suggested that relying on counts of rule usage alone does not necessarily deliver a

good quality description (Cox and Phalp 2000b). The independent assessment suggested by Cox and Phalp (2000b) is one that might be used by an academic when marking assignments. This marking approach might have a set of criteria that an assignment is evaluated against. This heuristic judgement is typical of how one might assess other software artefacts. For example, object-oriented designs are often assessed in a similar way, with a consideration of heuristics, such as coupling and cohesion (Budgen 1994), as are Role Activity Diagrams (Phalp and Shepperd 1999). Note that each facet is marked out of 10 and the scores are simply added to give totals out of 70, not 100 (since there are 7 C's).

The use case descriptions were blind marked by the assessors and averages were taken. The assessment criteria set out in section 2.7 (7 C's of Communicability) were used. Since the 7 C's and the CP Rules describe good qualities of use case descriptions, a number of these qualities are shared (see table 4-1). The Coherent facet is impacted by the CP Style Rules 7 and 8. The Consistent Grammer, Structure and Consideration of Alternatives facets concern themselves with grammar, sequencing, alternative sections etc. and are impacted by various CREWS Guidelines and CP Rules.

Table 5-10 shows that group A's descriptions are more comprehensible than group C's in table 5-11. A reason for this difference is that group A scores higher for the Cogent, Consistent Abstraction and Consistent Structure facets.

As an example of the marks awarded, participant C4 (table 5-11) scored 0 for Cogent because this description forces the reader into a possibly infinite loop, moving from 3 to 4, jumping to 9 and back to 3:

"3. The application prompts the Customer to select an option.

4. The Customer selects the Check Balance option. Go to 9.

...

9. The application displays the Customer's balance. Go to 3."

7 C's / Participants	A1	A2	A3	A4	A5	A6	Totals
Coverage	6	8	5	10	8	8	7.5
Cogent	6	5	5	7	6	5	5.67
Coherent	5	4	7	8	5	7	6
Consistent Abstraction	6	10	6	10	10	8	8.33
Consistent Structure	7	6	9	9	5	9	7.5
Consistent Grammar	5	0	0	3	4	5	2.83
Consideration of Alternatives	5	6	4	2	7	2	4.33
Totals	40	39	36	49	45	44	42

Table 5-10. 7 C's results for Group A (CP ATM)

7 C's / Participants	C1	C2	C3	C4	C5	C6	Totals
Coverage	8	4	10	6	6	2	6
Cogent	3	5	7	0	3	7	4.17
Coherent	2	6	8	4	0	4	4
Consistent Abstraction	0	10	2	8	4	2	4.33
Consistent Structure	5	7	10	1	2	2	4.5
Consistent Grammar	2	8	6	2	2	2	3.67
Consideration of Alternatives	3	8	4	4	0	0	3.17
Totals	23	48	47	25	17	19	30

Table 5-11. 7 C's results for Group C (CREWS ATM)

This affects its logical order, dependencies and the rationality of the answer. The participant who answered questions on this description identifies other problems: "Goto statements make it difficult to read. There is also no return card option (even at end). Customer stuck there forever." This suggests a need for tool support to prevent use of 'goto' statements.

Groups B and D (tables 5-12 and 5-13) scored exactly the same overall for the Retail task (30 out of 70). This suggests that the Retail use case is more difficult than the ATM. However, group C also scored 30 on the ATM task. The CREWS Guidelines can be said

to be generic, though they do not score particularly well. The CP Rules appear to be more successful for the ATM task than Retail.

7 C's / Participants	B1	B2	B3	B4	B5	B6	Totals
Coverage	4	4	8	2	6	8	5.33
Cogent	3	5	4	4	2	5	3.83
Coherent	4	6	4	2	2	6	4
Consistent Abstraction	2	2	10	8	6	8	6
Consistent Structure	3	5	5	5	4	4	4.33
Consistent Grammar	3	2	4	3	2	4	3
Consideration of Alternatives	2	6	7	0	4	0	3.17
Totals	21	30	42	24	26	35	30

Table 5-12. 7 C's results for Group B (CP Retail)

7 C's / Participants	D1	D2	D3	D4	D5	D6	Totals
Coverage	6	10	8	6	6	8	7.33
Cogent	5	5	4	3	5	2	3.83
Coherent	6	2	5	2	5	2	3.67
Consistent Abstraction	6	2	5	8	0	2	3.83
Consistent Structure	5	2	3	2	5	2	3.17
Consistent Grammar	5	4	2	1	6	5	3.83
Consideration of Alternatives	4	10	8	0	2	0	4
Totals	37	34	35	22	29	21	30

Table 5-13. 7 C's results for Group D (CREWS Retail)

Feedback suggests CREWS and CP should differentiate between the main flow of events and alternative flows. For example, allowing negatives and loops only in the alternative flows. Participants suggested that although CREWS SG3 allows for iteration and concurrency in the main flow, these should only occur in a separate (alternative / exceptional flows) section.

5.4.4.1.1 Assessing the Impact of the Results

The results indicate that CP use cases appear to be as communicable as the CREWS use cases. However, Group A scores higher marks than Group C. This might indicate that certain guidelines might suit certain problem frames (Jackson 2001). In this case, the CP Rules might be better suited to the ATM task than the Retail task. It might be possible to focus sets of guidelines to different domains. This though, is work for the future. More assessment of the CP Rules and the 7 C's has first to occur because this is only a pilot study with a small sample. A larger experiment might provide different outcomes or confirm the findings here (see chapter 6).

5.4.4.2 Reading the Use Case Description (phase 4)

The four groups were presented with descriptions to read and were then asked to answer questions on the logic and plausibility of the description. The questionnaire is in Appendix C3.

The participants were perhaps less critical of the descriptions than the markers. For instance, question 8 asks if the description is plausible. One reader suggests why A3's use case is not: "The customer never enters the PIN number; therefore, all of the transactions cannot be completed." The description itself states: "System validates card." Whether the Validate Card use case asks for a PIN is impossible to know. The description scores 5 for Coverage because of this ommission. The Cogent facet is also affected because this reduces the logic of the answer (how rational it is) and there are missing dependencies. Consistent Abstraction is also reduced since the Validate Card use case is internal design.

There were several questions about the logic of the description.

- Is there any illogical order to the use case? For example, does the customer get their cash before entering their PIN? (For those reading the ATM description.)

- Does there appear to be a logical flow to the order of events, that is, one event naturally follows another?
- Is the use case plausible?
- Are the alternative/exception paths viable?

Use case C4 gets stuck in a loop. The participant who read this description wrote: "There is... no return card option... Customer stuck there forever." This indicates there are problems with its logical flow. But in a general comment on the use case, the reader stated, "The statements are well written and follow a logical progression." Most of the use case does follow a logical progression but there is a Coverage problem here. Not enough information is presented to the reader. Perhaps the participant did not want to condemn a fellow student's work. This raises the wider issue of using students in pilots and, subsequently, in experiments. There is always a risk of personal politics biasing the outcome of the study when participants are asked to evaluate other participants' work.

The participants agreed that there were more illogical use cases than logical. The participants found ten CREWS descriptions with problems and five CP descriptions with problems. The CP Rules explicitly consider the logical order and coherence of descriptions (CP Style Rules 7 and 8) and this may be a reason for this difference, though group B's coherence scores (table 5-12) are similar to group C's (table 5-11) and D's (table 5-13). The doubt in the logic of the descriptions suggests a need for tool support for consistency checking.

5.4.4.2.1 Summary of Comprehension of Use Case Descriptions

The suggestion is that descriptions written with the CP Rules are as comprehensible as those written with the CREWS Guidelines. This does appear to be the case. When marking the descriptions with the 7 C's of Communicability, it has been shown that the CP Rules do score higher for the ATM task. When considering the comprehension and logic questionnaire, it appears to be the case that the CP Rules produce descriptions of at least as good quality as those of CREWS.

5.4.5 Assessing Aim 2

The second aim from section 5.2.1 considers whether the structure of the pilot is suitable for an experiment. This appears to be the case especially in consideration of the sub aims. The first of which considers whether there is enough time for the participants to complete the tasks. This does appear to be the case. Though a number of participants took the whole time for the task, they did manage to write complete main flows. The second sub aim considers whether the participants could produce acceptable answers in the time allowed. This appears to be the case though there are a number of poorer answers than expected. Perhaps the tasks were too complicated and need to be simplified for a larger experiment.

5.5 Threats to Validity

Though this is only a pilot study, it is worth considering validity threats because these have a direct impact on the aims of the pilot, as set out in section 5.2.1.

A threat is the sample size. With only six participants per group, it is possible that Group A had the six strongest participants. However, participants were randomly blocked to provide an 'equal' range of ability across the groups to try to avert this potential threat.

The pre-pilot questionnaire, pre-test and introduction to use cases occurred a week before the pilot. There is a risk that participants examined the use case literature perhaps biasing the results. This is an unknown quantity that could not be controlled. However, the results show that 3 out of 4 groups scored similar marks. Whether group A explored the literature is unknown.

The main part of the pilot (writing) lasted an hour. This was shown to be sufficient time to complete the task and present reasonable answers though a number of participants used all the time available (and produced complete main flows of events). Only one participant finished much before the rest (participant C5, who took 30 minutes). The participants

then had an hour break before the questionnaire part, which lasted only 30 minutes. This seemed insufficient time for boredom or over-enthusiasm to occur and no participants dropped out during the pilot.

The pilot design is similar to work previously conducted (Achour et al. 1999, Cox and Phalp 2000c); thus the format has been tested. The only new element is the comprehension questionnaire.

The ATM is often used to provide an example of use case descriptions (e.g. Kaindl 1998, Kosters et al. 2001, Rolland 2002). There is the risk that participants found similar examples in the literature. However, it is difficult to find tasks that participants are assumed to be familiar with. The Retail and ATM examples have been used by CREWS to experiment and describe their authoring guidelines (Rolland and Achour 1998, Achour et al. 1999). These therefore seemed suitable choices for this pilot. In any case, most participants produced reasonable answers to these problems. Some marks were quite low, however, but this was due more to structural and logic problems in constructing the description than in not understanding the task.

5.6 Conclusions

This pilot compares two sets of writing guidelines, the CREWS Use Case Authoring Guidelines and the CP Use Case Writing Rules. Two aims were presented in section 5.2.1. These related, firstly, to the CP Rules. This aim had a number of sub aims. Two sub aims assessed efficiency (time to write and length of main flow). The results are mixed and there is no overall difference in favour of either set of guidelines. There are also doubts about the validity of these sub aims without consideration of other measures, such as accuracy of the description. The third sub aim examines the number of times the Guidelines and Rules have been applied in the descriptions. This suggests the CP Rules are applied as much as CREWS. This is not the case for some of the Style comparisons, for example, anaphors (pronouns). However, different CP Rules are applied more than the CREWS Guidelines, especially CP Structure 1 (subject verb object), when compared

to CREWS CG5 (<agent><action><object>). Thus it is the case that CP is applied approximately as much as CREWS. The final sub aim explores whether the CP Rules produce as comprehensible use cases as the CREWS Guidelines. Group A (CP ATM) scores much higher than its CREWS equivalent (group C) overall, as well as the retail groups (B and D). However, there was no difference between groups B and D (and group C scored the same as B and D). Perhaps the problem type has an impact upon the CP Rules more than it does CREWS, which produced similar marks for both the ATM and the Retail tasks. As stated, it might be acceptable to reduce the complexity of the problems for the experiment.

The second aim addressed the structure of the pilot. It was found that there was sufficient time to produce completed answers. However, since a number of participants used the whole time (60 minutes) it might be necessary to reduce the complexity of the problem statements so that participants have more time to reflect on their work. This might reduce the number of errors made.

5.6.1 Lessons Learned from the Pilot

The aim of this pilot was to assess whether this study format is worth exploring further and whether there need to be some changes to the CP Rules and/or to its design.

The first aim was to assess whether the CP Rules have the potential to be a reasonable set of guidelines. This does appear to be the case for the Structure Rules. However, the Style Rules need some revision based on their use compared to CREWS. It is suggested that the format of the CREWS Style Guidelines might force participants to read them more carefully. The CP Style Rules should therefore be modelled on this format. The revised CP Rules are presented in section 5.6.2.

The second aim listed in section 5.2.1 considers whether the pilot design is sufficient for a larger experiment. This does appear to be the case. Having four groups was managable. Using two different tasks made the second part of the pilot easier because participants

answered questions on a different domain and were not biased by the memory of their own descriptions. The first sub aim assessed if there was sufficient time to complete the task of writing the description and the second sub aim aim assessed whether the participants could produce reasonable answers. Though some participants took all the allocated time, they did finish the descriptions and produced generally acceptable answers to the problem tasks. The participants also had no difficulties understanding the tasks (ATM and Retail). However, some marks were rather low. Indeed, groups B, C and D scored 30 out of 70 overall. This suggests that the descriptions were not that good. Writing a description is not easy and given that participants had only an hour and could not validate their work with anyone, the markers ought to take this into consideration in further studies.

5.6.2 Revised CP Rules

The CP Rules are revised based on the results of the pilot. The Structure Rules remain the same. The Style Rules, though, have been slightly reorganised, being reduced from 9 to 7. Styles 2, 3 and 4 have been combined into one (Style 3), since this reflects more the CREWS Style Guidelines format that performed well in the pilot. CP Style 6 (all verbs should be present tense format) has been elevated to Style 2 to reflect its importance. The following shows the new CP Rules set,

1. General Style Rules.

These are applicable to all sentences in the use case description.

Style 1: Each sentence in the description should be on a new, numbered line. Alternatives and exceptions should be described in a section below the main description and the sentence numbers should agree. For example:

Main Flow

1. *The patient record appears on the screen.*

2. The doctor enters the patient's new address.

Alternative Flow

2. The doctor deletes the patient's record.

The alternatives go below the main flow and the sentence numbers agree (2 and 2).

Style 2: All sentences are in present tense format. The use case should describe events and actions in the here and now, not the past or the future. Some examples:

The operator presses the button.
The checkout operator enters the amount.

Style 3: Avoid using adverbs and adjectives, these add unnecessary clutter to the description and give values that are difficult to quantify. Only use negatives in alternative and exceptional flows of events. Avoid using pronouns (E.g. he, she, it, we, their etc.). Examples:

The doctor writes the prescription slowly.

slowly is an adverb – we don't need to know how the doctor writes the prescription, just that *the doctor writes the prescription.*

The patient swallows the big pill.

big is an adjective and is unnecessary; you should write *the patient swallows the pill.*

The patient stands next to the doctor.
He puts the prescription in his pocket.

Who is "he"? Whose pocket is "his"? Write proper nouns / names instead:

The doctor puts the prescription in the patient's pocket.

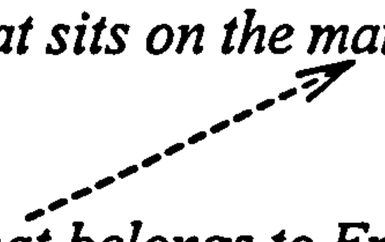
Style 4: Give explanations if necessary. Explanations should be enclosed in brackets:

The librarian enters the borrower's details (details are: name, address, phone number, library card number).

Don't overuse explanations. If you use too many explanations then you have too much information; you need to break the use case down into smaller use cases.

Style 5: There should be logical coherence throughout the description. The sentence you are writing now should refer to something in the last sentence or a previous sentence, if possible. We understand the use case better this way.

1. The cat sits on the mat.



2. The mat belongs to Fred.

The *mat* in (2) coheres to *mat* in sentence (1).

Style 6: When an action occurs there should be a meaningful response to that action. For example, when there is an input there should be a response to that input somewhere in the use case, usually immediately. This makes sure we do not forget to respond to any action in the use case description.

1. *The doctor enters the patient's record identification number.*
2. *The system displays the patient's record.*

Sentence 2 gives an immediate response to sentence 1.

Style 7: Underline other use case names.

The user makes a <u>new equipment request</u>.

When it is necessary to include a use case or have a use case extended by another use case, then write the use case name in the sentence and simply underline it.

2. Specific Structure Rules.

There are also some structures that you should apply for individual sentences. Again, try to use these structures as much as you can. Note that **verb** refers to present tense as described in **Style 2.**

Structure 1: Subject verb object.

The operator presses the button.

Structure 2: Subject verb object prepositional phrase.

*The operator gives the tool **to the mechanic.***

*The builder puts the bricks **on top of the pile of rubbish.***

*The system reminds the operator **to save all the open files.***

The **bold** text in the examples are prepositional phrases.

Chapter Six

Experiment 1: Comparing Guidelines for Writing Use Case Descriptions

6.1 Introduction

This chapter describes an experiment that follows up on the pilot study on the CP Use Case Writing Rules as described in chapter 5. The results from the pilot study indicated that the CP Rules were equally effective as the CREWS Use Case Authoring Guidelines, and more so with regard to the CP Structure Rules. This chapter expands on that reported by Phalp and Cox (2002).

6.1.1 Overview

The experimental goal is to compare the application of two sets of use case writing guidelines to ascertain if the CP Rules are used more than the CREWS Guidelines and are more comprehensible, since the CP Rules are derived in part from the 7 C's of Communicability (see section 2.7).

The guidelines are assessed by two means:

1. To count the number of times a guideline / rule has been applied (Hypothesis 1).
2. To evaluate the comprehensibility of the descriptions (Hypothesis 2).

6.1.2 Experimental Subjects

The subjects were software engineering and computing undergraduates. There were 60 experimental subjects, 15 per group. Although student subjects are not always considered typical of the general software engineering population, they are much closer to software engineering practitioners than the general public (Kitchenham et al. 2000). As Host et al. (2000) show, there is no significant difference between students and practitioners in performing lead-time impact assessment. (Though it is not claimed that subjects and

practitioners would perform equally well in this experiment because this has not been tested.)

Group	Guidelines	Use case task
A	CP Writing Rules	ATM use case
B	CP Writing Rules	Retail use case
C	CREWS Guidelines	ATM use case
D	CREWS Guidelines	Retail use case

Table 6-1. Experimental Groups, Guidelines, Tasks

Table 6-1 shows the experimental groups and tasks, as per the pilot (chapter 5). The subjects' seminar groups formed the basis for allocating subjects to experimental groups, such that seminar group 1 became experimental Group A, and so on. Despite possible reservations about having all the best subjects in the same group by chance, due to timetabling constraints it was impossible for the experimenter to pre-test and assess the experimental subjects as was performed in the pilot study. But because there are larger numbers this is less important.

6.2 Experimental Hypotheses

$H1_0$: There is no difference in the frequency of use of the CP Rules and the CREWS Guidelines.

H1: The CP Rules are used more frequently than the CREWS Guidelines.

$H2_0$: There is no difference in the comprehensibility of the CP Rules use cases and CREWS Guidelines use cases.

H2: The CP Rules use cases are more comprehensible than the CREWS Guidelines use cases.

One-tailed tests are applied because since the CP Rules performed as well as CREWS, and in the case of the ATM task, better than CREWS, in the pilot study (chapter 5), it is hoped that in an experiment the CP Rules will perform better than CREWS. The risk with a one-tailed test is that if CREWS performs significantly better than CP then this cannot be stated as a significant finding and the null hypothesis has to be accepted (Miller et al. 1997).

The comparison is made by comparing like tasks, that is, ATM against ATM and Retail against Retail. A comparison is also made of combined scores, groups AB (CP) against CD (CREWS).

6.3 Course of the Experiment

The subjects received instruction on writing use cases in the form of a lecture and seminar (1 hour each). They were presented with example use cases and asked to write a fairly simple use case as a practice. A 'model' answer was then discussed. Table 6-2 shows the time allocation for the experiment.

Task	Time (minutes)
Read guideline / rule set + problem	10
Write description	45
Debrief	5

Table 6-2. Time scale of experiment

The experimenter clarified any problems with the guidelines or tasks prior to the 45 minutes writing time. This was deemed sufficient time to think and write the description applying the guideline/rule sets. The experimental material was collected and a debrief (5 minutes) ensued to provide feedback. Because of time constraints for seminars, it was only possible to allocate 45 minutes for writing the description. This means that the experimental tasks had to be slightly simplified from that of the pilot study.

6.3.1 Experimental Tasks

The experimental tasks, as stated, have been shortened from the pilot to take into account the reduced writing time. The tasks are found in Appendix D1.

6.3.2 CREWS Guidelines / CP Rules

Subjects were provided with the CREWS Guidelines (see section 2.4.1.1) or the revised CP Rules (section 5.6.2). The subjects were unaware of the origins of the guidelines.

6.4 Experimental Results

6.4.1 Hypothesis 1

The hypothesis is tested by comparing like tasks (ATM against ATM, Retail against Retail). This is in case there is a bias towards one particular domain area. For example, in the pilot the CP ATM group did better than the other experimental groups. If similar results occur here, this would then strengthen the claim that the CP Rules are suited to that particular domain and perhaps also indicate they are not as generalisable as the CREWS Guidelines.

There are seven attributes that appear in both sets of guidelines / rules. These are shown in table 6-3. The Style attributes should be avoided and the Structure / Content attributes should be applied.

Appendix D4 shows the data tables for all attributes counted. The experiment tests comparable attributes only. The t-test is applied because there is the assumption that the amount of variability in the two experimental groups is equal: both groups are students who have studied the same undergraduate courses (Salkind 2000). Unpaired, one-tailed t-tests (using Data Desk 6 with the Bonferroni adjustment applied) were applied on the means of guideline abuse / use to test whether there was a difference between the samples

that applied the rule sets. A count was also made of the number of subjects who broke the guidelines (for Style). It is perhaps less important how often a guideline was broken by one subject compared to the number of subjects that broke that particular guideline. Each misuse / use made is divided by the number of steps in the main flow. For example, if one description main flow is only 2 steps in length and one step has a non-present tense structure, then the score shows 50 (percent).

CP Rule / CREWS Guideline
Style (attributes to avoid)
Alternatives in main flow
Pronouns / anaphors
Adverbs
Negatives
Non-present tense
Structure / Content (attributes to apply)
CP Structure 1 / CG5
CP Structure 2 / CGs1-3

Table 6-3. Comparable elements of CP and CREWS

6.4.2 Style Comparison

6.4.2.1 Alternatives in the Main Flow (CP Style 1, CREWS SGs 2 and 3)

Table 6-4 describes the mean number of alternatives in the main flow and the count (Ct) of subjects using alternatives in the main flow.

Group/Subject	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Ct
A	0	0	43	35	29	0	5	5	8	11	0	0	0	8	4	9
C	13	15	7	7	0	10	18	11	6	0	9	13	0	0	13	11
B	14	0	36	0	0	0	50	7	11	0	14	0	16	0	17	8
D	4	23	8	27	6	0	13	0	8	5	18	0	9	23	14	12
t-test (p =< 0.05)	A, C p = 0.33					B, D p = 0.46					AB, CD p = 0.35					

Table 6-4. Alternatives in the main flow (shown as percentages)

Although there is a cluster of high percentages of alternative flows in group A (A3-A5), there is no significant difference. Groups B and D have similar numbers of alternatives. The use of alternatives is quite high by all groups but there is an almost bimodal distribution with CP averaging 8.5 and CREWS higher at 11.5. A reason for this difference could be use of CREWS Content Guideline 6 (if... then), despite both CREWS Style Guidelines 2 and 3 recommending that variations (alternatives) be placed in a separate section. These can be considered as potentially contradictory. Indeed, feedback from the pilot indicated that this Content Guideline should be flagged as only usable in an alternative section.

As an example of high usage of alternatives, subject A5 (who has 29% alternatives in the main flow) lists all possible alternatives to entering a card PIN:

"2. Customer enters PIN.
2.1 Incorrect PIN, return to 2.
2.2 Incorrect PIN for the third time. Card retained by the ATM and Customer advised to seek assistance (End of Use Case).
3. PIN accepted by ATM, proceeds to main menu."

The subject should have written the two incorrect PIN choices (2.1, 2.2) as exceptions in a separate section below the main flow, as is stated in CP Style 1.

6.4.2.2 Pronouns (CP Style 3) / Anaphoric References (CREWS SG4)

Group/Subject	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Ct
A	17	0	0	0	14	0	0	0	0	0	0	5	0	10	27	5
C	0	0	0	0	0	0	0	0	0	0	8	17	0	0	0	2
B	0	3	0	13	25	0	0	14	0	0	4	0	0	0	0	5
D	0	0	0	0	0	0	6	5	8	0	0	0	0	0	13	4
t-test (p =< 0.05)	A, C p = 0.11					B, D p = 0.21					AB, CD p = 0.07					

Table 6-5. Pronouns and anaphoric references (shown as percentages)

Table 6-5 shows there is no statistically significant difference between the groups. However, it can be seen that CREWS use pronouns less than CP. It could be the case that the CREWS terminology forces the reader not to skim the text, whereas 'pronoun' is probably more recognisable. The pilot study shows similar results. However, 10 out of 30 CP subjects used pronouns compared to 6 out of 30 CREWS subjects. There is not a large difference between the groups in these terms.

Subject B5 (CP Retail) writes a pronoun in the sentence:

"14. Customer has coupon / vouchers and gives *them* to the cashier." [Author's italics.]

It would have been better to ignore this kind of event since use cases are about the specification task and not interactions between actors (see section 2.3.1). But as most subjects involved a Customer in the description as well as a Cashier, the subject could have written this sentence:

"14. The Customer gives coupons to the cashier" which would be in compliance with CP Structure Rule 2 (*subject verb object prepositional phrase*).

6.4.2.3 Adverbs (CP Style 3 and CREWS SG6)

Table 6-6 shows the mean number of usage of adverbs per description.

Group/Subject	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Ct
A	8	0	0	0	0	0	0	0	8	6	0	0	0	0	0	3
C	0	0	0	0	13	0	5	7	0	0	0	0	0	0	7	4
B	0	0	0	0	3	0	0	7	0	0	32	0	5	0	6	5
D	4	0	0	0	0	4	0	0	10	0	0	0	0	7	27	5
t-test (p =< 0.05)	A, C p = 0.69					B, D p = 0.49					AB, CD p = 0.58					

Table 6-6. Usage of adverbs (shown as percentages)

There is low usage throughout. However, subject D15 uses adverbs in 27% of events. Many of these adverbs could have been avoided by not placing alternatives in the main flow:

"8. If *that* is all the goods then subtotal the bill (go to 11)
9. If *there* are still more goods then continue to pass them over the bar code reader (then go to 8)." [Author's italics.]

If choice is removed, the adverbs are eliminated:

"# The cashier scans the barcodes of the products."

6.4.2.4 Negatives (CP Style 3, CREWS SG6)

Group/Subject	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Ct
A	0	25	14	0	0	0	0	0	0	0	0	0	0	4	0	3
C	0	6	5	10	4	9	14	11	13	0	0	8	5	5	0	11
B	14	0	9	0	0	0	10	0	11	0	14	0	11	0	11	7
D	4	0	8	5	10	4	6	5	8	10	18	18	27	0	13	13
t-test (p =< 0.05)	A, C p = 0.09					B, D p = 0.07					AB, CD p = 0.02					

Table 6-7. Usage of negatives (shown as percentages)

Table 6-7 shows negative usage across the groups. CREWS use cases have a higher usage of negatives. When comparing the number of subjects using negatives, the difference is clear; 10 out of 30 CP subjects use negatives compared to 24 out of 30 CREWS subjects. The medians of the groups are revealing, group A has a median of 0 and C a median of 5, group B a median of 0 and group D a median of 8. Combined medians for AB is 0 and CD 6. Clearly, CP uses negatives much less than CREWS. However, when inference tested within task, there is no significant difference. This, though, might be of practical significance (Kitchenham et al. 2000). When the groups are combined, however, there is a significant difference in favour of the CP Rules.

Figure 6-1 clearly shows the differences in negative usage. It shows that 20 subjects from group AB had no negatives in their description whereas group CD has only 6 subjects with no negatives. Subject A2's 25% is something of an outlier, as is subject D13 who had 27% negatives.

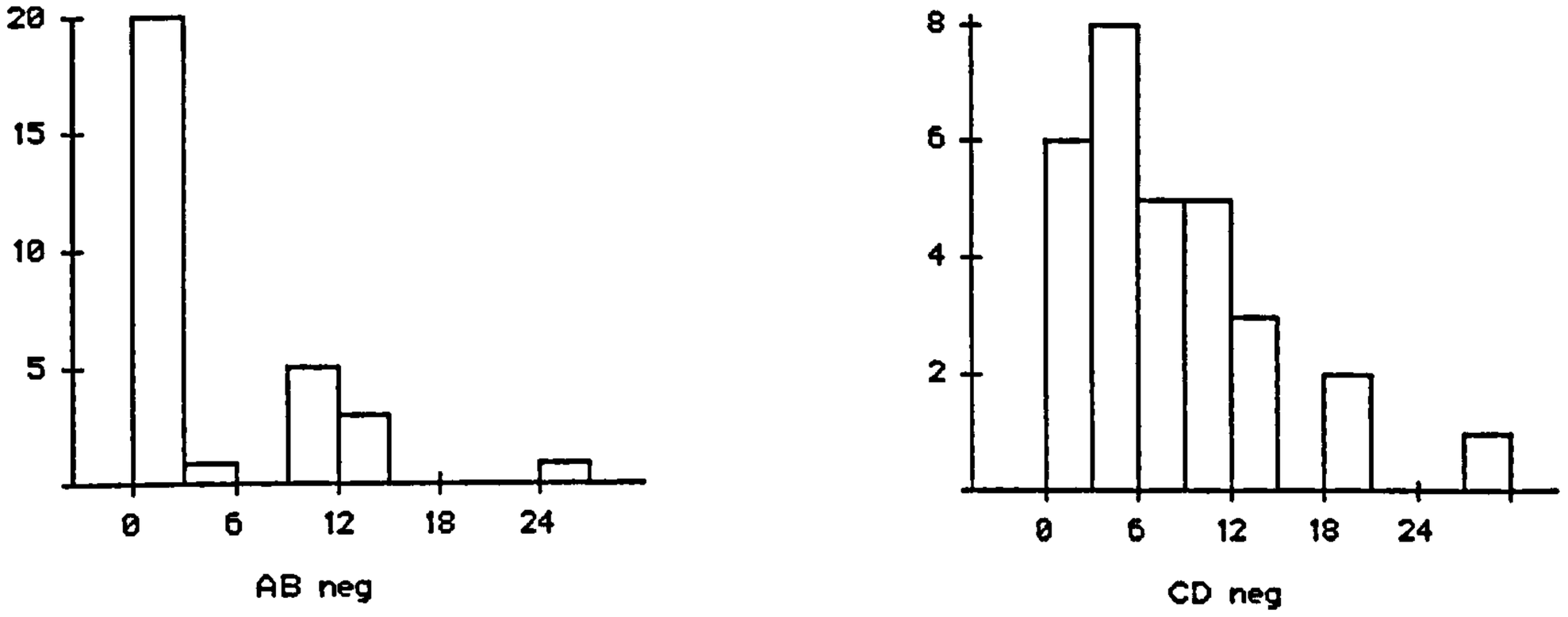


Figure 6-1. Histograms for groups AB (left) and CD (right) for use of negatives

Attribute	Mean	Median	StdDev	Min	Max	25th percentile	75th percentile
AB neg	4.1	0	6.6	0	25	0	10
CD neg	7.5	6	6.3	0	27	4	10

Table 6-8. Summary of data for groups AB and CD for use of negatives

Table 6-8 describes the data for groups AB and CD. The standard deviation is almost the same for both groups. Minimums and maximums are also similar. One difference is in the means, with CD having almost twice the mean as AB. The median has the largest difference as stated above. Both groups have 10 as the 75th percentile. Group AB has zero as its 25th percentile because the majority of its scores are zero.

Various CREWS Content Guidelines, such as CG6's *if... then*, allows the introduction of alternative flows, and hence negatives, into the main flow.

Subject D13 needlessly uses negatives because of this,

"6. If Customer has *no* more items then total bill." [Author's italics.]

This could be avoided by stating,

"# The cashier scans the barcodes of the products.
The cashier totals the bill."

6.4.2.5 Non-present tense (CP Style 2, CREWS SG5)

Active voice present simple tense is the only acceptable grammar structure. Table 6-9 shows usages of non-present tense.

Group/Subject	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Ct
A	33	8	0	19	57	8	53	32	0	17	15	19	38	3	8	13
C	4	25	10	30	9	0	9	15	6	0	63	13	11	30	67	13
B	6	6	9	20	22	3	24	14	11	0	11	10	16	12	17	14
D	4	8	0	23	5	27	0	5	92	15	18	0	18	7	27	12
t-test (p =< 0.05)	A, C p = 0.43					B, D p = 0.76					AB, CD p = 0.64					

Table 6-9. Non-present tense usage (shown as percentages)

Groups A and C have similar usage. Although there is no significant difference between the two groups, it can be considered important that both have a high percentage usage of non-present tenses. Perhaps the CP Rule and CREWS Guideline need reconsideration? The pilot study shows that non-present tenses are also used by a large number of participants. The grammar survey conducted in chapter 4 (section 4.4) shows that use case authors use a large range of tenses. Either the experimental tasks require use of other tenses or it is better to allow use case writers more freedom to use tenses that suit their particular problem context.

Subject D9 had 92% usage of non-present tenses in his/her use case description. 12 out of 13 sentences contained a non-present tense structure. For example,

"7. The operator *will* ask Customer how they *will* pay.

8. The operator *will* tell Machine how Customer *will* pay.

... " [Author's italics.]

As this description is specification (system level), event 7 should not have been written. The whole transaction could read:

"# The operator enters the Customer payment into the till."

6.4.3 CP Structure / CREWS Content Comparison

Table 6-3 shows only two comparable structures, CP Structure 1 against CREWS CG5, and CP Structure 2 against CREWS CG's 1-3.

6.4.3.1 CP Structure 1 / CREWS CG5

Table 6-10 describes the count of usage between the 4 experimental groups. Note the higher the figures in table 6-10 (and 6-12), the better the descriptions.

Group/Subject	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	33	33	0	4	29	50	47	16	83	61	20	33	62	70	31
C	35	25	48	10	65	50	45	33	69	62	18	46	79	20	33
B	6	33	45	30	6	57	24	14	22	29	36	30	42	12	22
D	4	0	23	0	25	38	35	5	15	0	18	45	18	0	13
t-test (p =< 0.05)	A, C p = 0.70					B, D p = 0.02					AB, CD p = 0.27				

Table 6-10. CP Str. 1 v CG5 (shown as percentages)

There is no significant difference between groups A and C (ATM task). Group B has a significantly higher usage of *subject verb object* than the equivalent from group D. The reason for this is unclear. Perhaps the semi-formal structure of the CREWS notation confused the subjects of group D? It is certainly the case in feedback that some subjects liked the CREWS Style Guidelines but not their Content Guidelines. Table 6-11

summarises the data for groups B and D. Group B's mean and median are almost double those of group D.

Attribute	Mean	Median	StdDev	Min	Max	25th percentile	75th percentile
B Str 1	27.2	29	14.45	6	57	16	35.25
D CG5	15.9	15	14.96	0	45	1	24.5

Table 6-11. Summary of data for groups B (CP Str 1) and D (CREWS CG5)

The 25th percentile for group B is much higher than for group D (16 to 1). The difference between the 75th percentiles is slightly less though group B is still higher than D (35.25 to 24.5). Figure 6-2 shows the difference between medians for the groups though there is a slight overlap in the confidence intervals (shaded areas), meaning there is no significant difference between the medians.

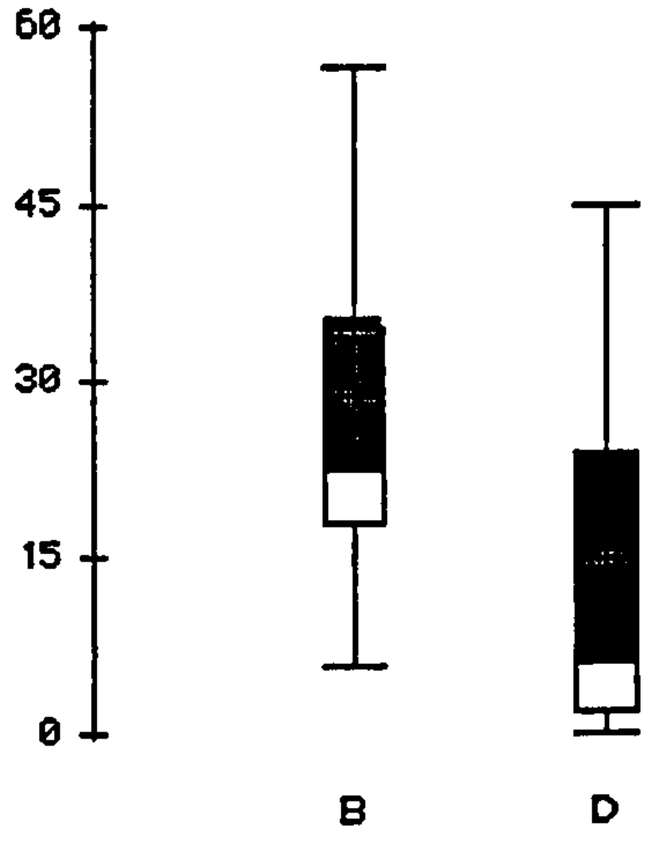


Figure 6-2. Box plots for groups B (CP Str 1) and D (CREWS CG5)

6.4.3.2 CP Structure 2 / CREWS CGs 1-3

Table 6-12 shows there is no significant difference between groups A and C. Group B, however, has a highly significant usage of CP Structure 2 compared to group D. The result reflects the differences found with CP Structure 1 v CG5. Group B appears to be

the strongest group in terms of applying the structures, although there is no overall difference in usage of style.

Group/Subject	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	0	0	0	0	0	33	5	37	0	0	0	5	0	23	8
C	8	13	5	3	4	23	0	7	6	15	0	4	0	0	0
B	50	33	36	17	22	29	29	14	33	14	18	40	16	29	17
D	21	15	38	23	15	0	6	35	16	25	18	0	0	0	0
t-test (α =< 0.05)	A, C p = 0.34					B, D p = 0.004					AB, CD p = 0.02				

Table 6-12. CP Str. 2 v CREWS CG1-3 (shown as percentages)

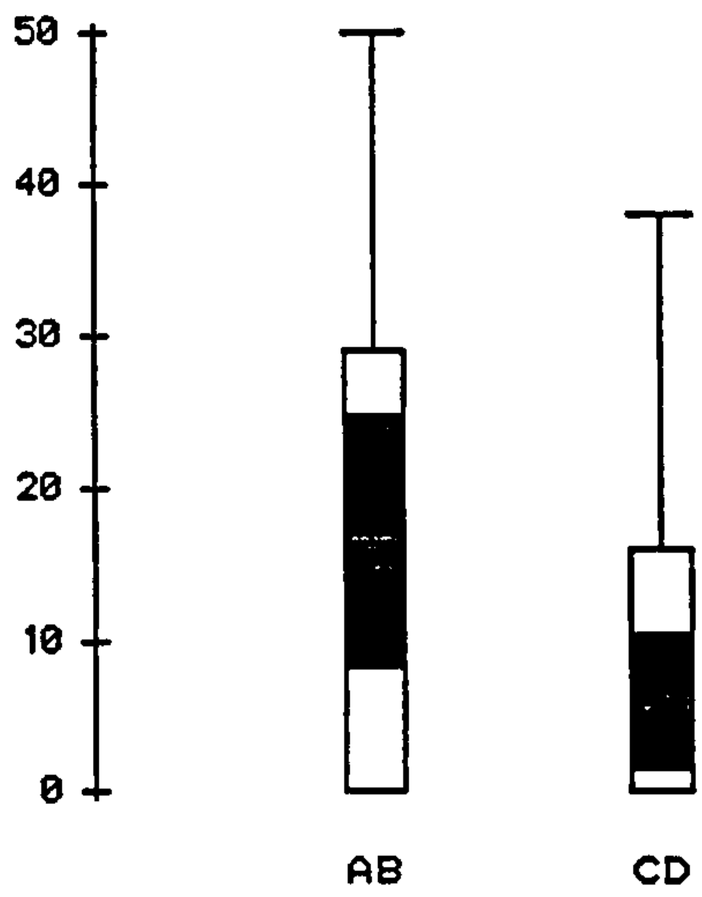


Figure 6-3. Box plots for groups AB (CP Str 2) and CD (CREWS CGs1-3)

Attribute	Mean	Median	StdDev	Min	Max	75th percentile
AB Str2	16.9	16.5	15.12	0	50	29
CD CG1-3	10	6	10.94	0	38	16

Table 6-13. Summary of data for groups AB (CP Str 2) and CD (CREWS CGs1-3)

CP Structure 2 is used significantly more than CREWS CG1-3. Table 6-13 shows that the median for AB is 16.5, which is higher than the median of group CD at 6. This could be due to the CREWS CG1-3's apparent restriction to three specific formats, whereas CP is not. This is depicted in the box plots of figure 6-3. However, the shaded areas on the

boxes overlap, indicating there is no significant difference between the medians. Table 6-13 shows that group AB also has a higher mean. There is a difference in standard deviation, showing that AB has a wider range of scores from the mean. There is a large difference in the 75th percentiles. Group AB has almost double the mark at the 75th percentile to group CD. The difference is compared against the pilot study. In the pilot it was the case that group A (CP ATM) outperformed the other groups. It is therefore difficult to assign differences to the task itself because group A should have performed better than B if the CP Rules appeared better suited to the ATM task than the Retail. It might be the case that group B was simply better than the other groups at writing descriptions or understanding problems. Since access to the subjects was restricted due to timetabling constraints, the author had no time to conduct pre-experimental questionnaires and tests to assess the ability of the subjects. Students are not assigned to seminar groups based upon ability so this is an unknown factor. However, with 15 subjects per group this is unlikely to be a major factor. Covariance between subject performance and their overall scores for the courses taken in their degree cannot be assessed because subject answers were only identified by a number to assure anonymity.

6.4.3.3 Hypothesis 1 Summary

Hypothesis 1 cannot be accepted because CP does not outperform CREWS in all aspects. However, group AB use significantly less negatives than group CD. In terms of structure, CP (Group B) significantly outperforms CREWS (Group D). Group AB applied CP Structure 2 significantly more than group CD the equivalent CREWS CG1-3. The null hypothesis is accepted because there is no overall significant difference between the use case writing sets.

6.5 Hypothesis Two Results

Hypothesis Two states that descriptions written with the CP Rules are significantly more comprehensible than CREWS descriptions. Judging how comprehensible descriptions might be is a highly subjective matter. The experimenters blind marked and then

averaged marks for the descriptions, applying the marking scheme described below. Mann-Whitney U tests were conducted on the data, rather than a parametric t-test because of the subjectivity of parts of the marking scheme.

6.5.1 Marking criteria for the 7 C's of Communicability

The pilot study's marking approach is one that might reflect the approach taken when marking an assignment. That is, separate sections are graded out of 10. These are summed to get a total mark (out of 70). This approach is continued here.

6.5.2 Comparing Groups A and C

Tables 6-14 and 6-15 show a summary of the grades awarded to groups A (CP ATM) and C (CREWS ATM). Appendix D2 shows the complete breakdown of marks for the groups.

Group A has the same mean total (47) as group C. (See Appendix D3 for examples of rationale for marks awarded.) Table 6-16 shows differences between the groups.

CP ATM (Group A)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Coverage	8	9	9	6	10	8	10	8	8	10	7	10	8	8	7
Cogent	10	3	7	6	3	6	9	9	6	5	6	10	6	10	9
Coherent	10	9	7	0	10	10	6	10	10	8	4	9	7	0	9
Consistent Abstraction	4	10	6	10	10	0	10	8	10	10	10	10	8	6	6
Consistent Structure	10	10	7	6	8	10	9	9	9	8	10	10	10	8	9
Consistent Grammar	3	6	9	5	5	9	0	4	9	6	7	5	5	5	1
Consideration of Alternatives	0	0	0	0	0	10	0	0	0	0	6	6	0	8	10
Total (out of 70)	45	47	45	33	46	53	44	48	52	47	50	60	44	45	51

Table 6-14. 7 C's marks awarded, Group A CP ATM

Overall there is no significant difference between the groups ($p = 0.65$). Neither are there any differences between the facets except for Consistent Structure. Group A's structure is highly significantly better than group C's ($p = 0.008$). Figure 6-4 shows histograms for

groups A and C for Consistent Structure. Note that the y scales for the histograms are different. Group A's is up to 6 subjects and C's up to 4 subjects. The majority of group A score 8 or more (13 out of 15), whereas just under half of C do (7 out of 15).

CREWS ATM (Group C)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Coverage	8	10	8	6	8	10	9	9	8	7	6	9	7	10	9
Cogent	2	9	9	3	9	9	9	9	8	4	3	9	6	8	7
Coherent	8	7	10	9	10	9	8	10	10	7	0	8	8	10	10
Consistent Abstraction	7	8	9	6	8	8	10	9	9	10	9	6	10	10	10
Consistent Structure	6	8	6	7	10	8	6	0	8	10	4	7	10	8	7
Consistent Grammar	7	3	5	0	4	8	4	1	7	10	0	1	7	3	8
Consideration of Alternatives	0	9	7	0	0	10	0	0	0	0	9	0	6	9	0
Total (out of 70)	38	54	54	31	49	62	46	38	50	48	31	40	54	58	51

Table 6-15. 7 C's marks awarded, Group C CREWS ATM

7 C	Mann-Whitney U test (p =< 0.05)
Coverage	p = 0.42
Cogent	p = 0.44
Coherent	p = 0.76
Consistent Abstraction	p = 0.51
Consistent Structure	**p = 0.008**
Consistent Grammar	p = 0.25
Consideration of Alternatives	p = 0.66
Total	p = 0.65

Table 6-16. Comparing A against C, Hypothesis 2

The median is higher for A than C (9 to 7, as seen in table 6-17). The 75th percentiles have a similar difference of 2, with Group A at 10 and C at 8, as does the 25th percentile: group A's is 8 and C's is 6. The standard deviation for group C is double that of A, showing a wider range of marks from the mean. Group C has 27 variations in its main flow whereas group A has only 17. Another reason for the difference is Group A has 6 sequencing problems (numbering of events incorrectly) and group C has twice as much with 13.

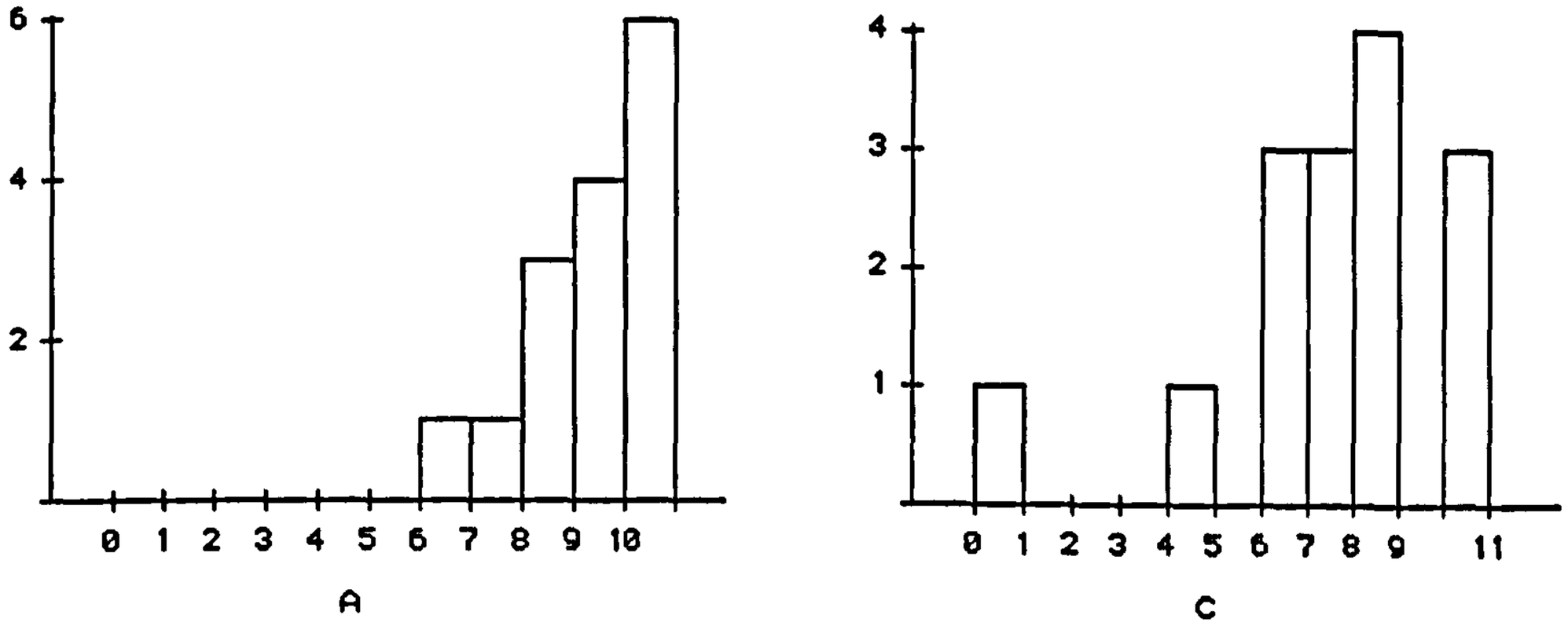


Figure 6-4. Histograms for Groups A (left) and C (right) of Consistent Structure marks

Attribute	Mean	Median	StdDev	Min	Max	25th percentile	75th percentile
A	8.86	9	1.25	6	10	8	10
C	7	7	2.56	0	10	6	8

Table 6-17. Summary of data for groups A and C Consistent Structure

Hypothesis 2 states that the CP Rules produce significantly more comprehensible use cases than the CREWS Guidelines. This is shown not to be the case for group A against C.

6.5.3 Comparing Groups B and D

Tables 6-18 and 6-19 show the results for groups B (CP Retail) and D (CREWS Retail). Here the mean of the totals slightly favours group B: 51 to 50. (See Appendix D3 for examples of justification for marks awarded.)

Though overall there is no significant difference, for the Cogent facet (table 6-20), there is for group B ($p = 0.04$). A major reason for the difference is that group D has 12 Text Order problems, with 9 out of 15 descriptions having at least one order problem, whereas Group B has 5 order problems, committed by only 3 subjects.

CP Retail (Group B)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Coverage	10	8	10	4	8	9	10	10	9	9	10	10	9	10	9
Cogent	8	9	10	10	10	10	7	8	10	10	4	10	10	10	10
Coherent	10	10	9	6	8	9	8	6	7	10	9	9	9	8	9
Consistent Abstraction	6	2	8	0	8	8	9	10	8	10	6	9	6	9	8
Consistent Structure	8	10	6	10	10	10	5	10	8	8	6	10	7	10	7
Consistent Grammar	7	7	8	0	0	9	4	5	6	10	0	9	4	8	4
Consideration of Alternatives	7	9	0	0	7	8	10	0	0	9	0	0	0	10	0
Total (out of 70)	56	55	51	30	51	63	53	49	48	66	35	57	45	65	47

Table 6-18. 7 C's marks awarded, Group B CP Retail

CREWS Retail (Group D)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Coverage	10	10	7	8	9	10	9	10	10	9	8	7	9	7	9
Cogent	10	9	8	6	9	8	9	10	8	8	6	9	10	10	7
Coherent	8	8	10	9	10	8	8	10	7	8	10	10	9	3	9
Consistent Abstraction	7	10	8	10	9	5	7	10	10	3	9	10	8	10	7
Consistent Structure	5	7	9	4	9	10	8	10	9	9	8	10	9	6	9
Consistent Grammar	7	9	9	4	7	0	8	7	0	1	6	8	5	8	0
Consideration of Alternatives	0	0	0	0	8	10	9	10	0	0	0	0	0	10	0
Total (out of 70)	47	53	51	41	61	51	58	67	44	38	47	54	50	54	41

Table 6-19. 7 C's marks awarded, Group D CREWS Retail

7 C	Mann-Whitney U test (p =< 0.05)
Coverage	p = 0.18
Cogent	**p = 0.04**
Coherent	p = 0.62
Consistent Abstraction	p = 0.87
Consistent Structure	p = 0.33
Consistent Grammar	P = 0.49
Consideration of Alternatives	P = 0.35
Total	P = 0.31

Table 6-20. Comparing B against D, Hypothesis 2

Table 6-21 shows there is only a small difference in all the values listed for the Cogent facet.

Attribute	Mean	Median	StdDev	Min	Max	25th percentile	75th percentile
B	9.1	10	1.71	4	10	8.25	10
D	8.5	9	1.36	6	10	8	9.75

Table 6-21. Summary of data for groups B and D for Cogent

The difference between the groups is described in figure 6-5. Group D's upper tail is level with group B's median score (10). Group B has one outlier, which scores 3 marks. Subject B11 scored poorly because it has 3 Text Order and 3 Rational Answer errors.

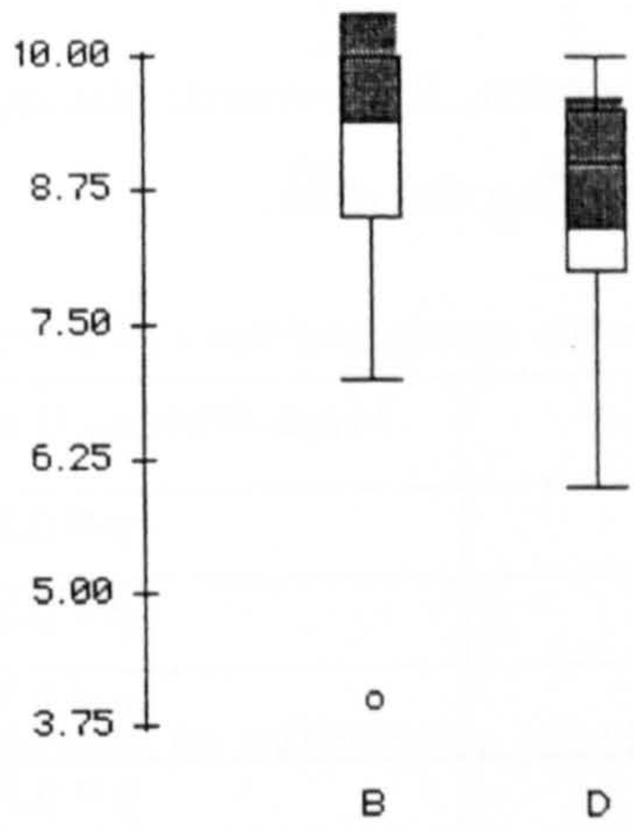


Figure 6-5. Box plots for groups B and D for Cogent

6.5.4 Summing up

There is no significant difference between groups A and C although group A has a highly significantly better Structure ($p = 0.008$). However, there is no significant difference between groups B and D for Structure ($p = 0.33$). 11 subjects from group B have 27 variations in the main flow and 8 subjects from group D have 23 variations. The Retail task states that customers can pay by cash, cheque or credit card (Appendix D1), suggesting a number of alternatives. The subjects were asked to consider payment by

cash alone. A number of alternatives in the main flow describe all payment options. The experimental task might be responsible for confusing these subjects.

Group B is significantly better than group D for the Cogent facet ($p = 0.04$). Group D had a number of Text Order problems. However, this result is not reflected in groups A and C, where $p = 0.44$. 7 subjects from group A had Text Order problems (9 in total), 9 subjects had Dependency problems (30 in total) and 3 subjects had Rational Answer mistakes (6 in total). 11 subjects from group C had Text Order problems (21 in total), 10 subjects had Dependency problems (24 in total) and 2 subjects had Rational Answer mistakes (2 in total).

6.5.5 Combining the Results

Table 6-22 shows combined group scores. It is worth stating that the results might be biased by unrecognised noise in combining the data.

7 C	Mann-Whitney U test ($p =< 0.05$)
Coverage	$p = 0.23$
Cogent	$p = 0.13$
Coherent	$p = 0.77$
Consistent Abstraction	$p = 0.84$
Consistent Structure	**$p = 0.03$**
Consistent Grammar	$p = 0.34$
Consideration of Alternatives	$p = 0.47$
Total	$p = 0.49$

Table 6-22. Comparing AB against CD, Hypothesis 2

It can be seen that there is a significant difference for group AB over CD in the Consistent Structure facet ($p = 0.03$). The overriding reason for this is the high performance of group A over C. There was no difference between groups B and D.

Figure 6-6 shows box plots for the combined totals of groups AB and CD. The medians for the combined groups are almost identical, with AB scoring slightly lower than CD. Interestingly, the range of marks for AB is narrower than for CD. This difference might mean that the CP Rules are less likely to produce poor use case descriptions compared to CREWS but both are likely to produce comparatively good scores.

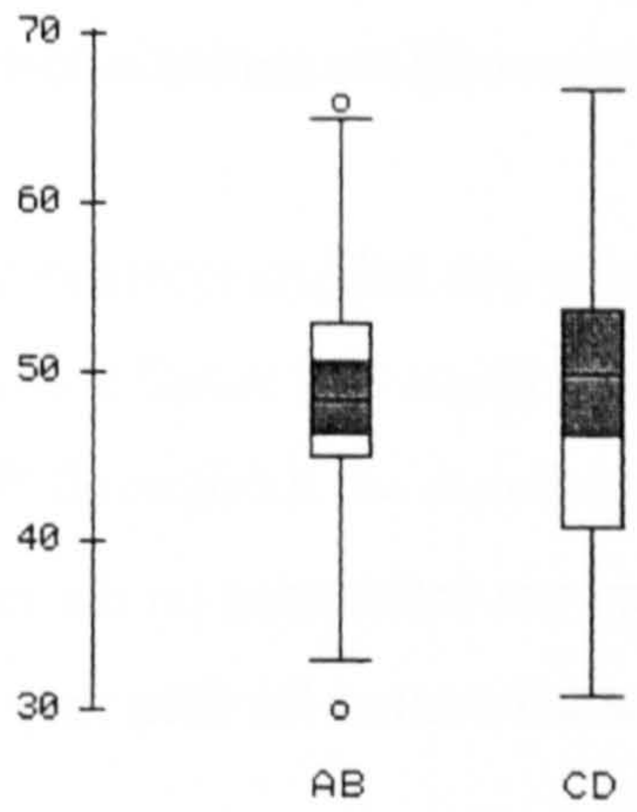


Figure 6-6. Box plots for combined groups AB and CD showing total marks

6.6 Threats to Validity

A comprehensive list of threats to validity is provided by Wohlin et al. (2000). This section discusses the perceived threats to this experiment only.

6.6.1 Conclusion Validity

Conclusion validity considers whether the conclusions drawn from the statistical results are valid or whether they might be biased by issues affecting the treatment and the outcome.

6.6.1.1 Random Heterogeneity of Subjects

This threat is reduced because the subjects are all undergraduate students on a computing course. However, some subjects performed worse than others. Four subjects from group C scored low marks for hypothesis two (below 40 marks). A post-hoc covariance analysis of student ability on their degree course to compare against the results here was considered to see whether the subjects performed unusually badly in the experiment; however, this was not possible because subject names were not recorded primarily to avoid bias in marking.

6.6.2 Internal Validity

Internal validity is threatened by unknown influences on the causal relationship between treatment and outcome. If these are not accounted for then they can invalidate the results.

6.6.2.1 History

In separate sessions that spanned a week after the introduction to use cases (due to timetabling restrictions), the subjects completed the experimental tasks. There is a risk that subjects, having participated in the experiment, would pass on any knowledge to those yet to take the experiment (causing diffusion of treatments). The only way to control this was to make sure no experimental material was taken from the location of the experiment.

There is no indication of an overall significant improvement in the results from group A to B and from group C to D except with regards to the CP Structures and CREWS Content Guidelines, which must in part be due to the nature of the different tasks. CP Structure 1 is used more in group A than B. CREWS CG5 is used more in group C than D. The opposite is true for CP Structure 2 / CREWS CG1-3, indicating that the Retail task has more complex interactions than the ATM.

6.6.2.2 Maturation

The experiment lasted an hour (the writing part 45 minutes). Concerns over boredom or over-enthusiasm were not considered significant. As such, no subjects dropped out of the experiment – mortality is not an issue. In terms of the experiment itself, one hour is not much time to learn a set of guidelines and write a use case description and in the wider industrial context would not be particularly realistic. However, due to timetabling pressures it was impossible to obtain more time with the subjects.

It is unknown whether any subjects studied use case descriptions between the lecture and the experiment itself. This is a factor that could not be controlled. The subjects were not informed of CREWS or CP throughout the duration the experiment.

6.6.3 Construct Validity

Are the results generalisable or is the experiment threatened by design problems or social issues? If these threats occur they can bias the outcome and invalidate the results.

6.6.3.1 Inadequate Preoperational Explication of Constructs (Wohlin et al. 2000)

The measures are clearly defined. Hypothesis 1 measures concern grammar and structural issues, which are understood concepts of the English language and the use case model. They are relatively easy to identify, although they can be numerous. Hypothesis 2 measures relate to established discourse process theory (coherence), software engineering (scope, span) and grammar.

6.6.4 External Validity

It is clear that the results of the experiment cannot be generalised to every software house employing use case descriptions. However, they might be considered representative of undergraduate computing and software engineering students. One cannot generalise to

software houses because of the time constraints on the experiment and, importantly, the prior knowledge brought to the task by experienced practitioners.

6.6.4.1 Nature of the Problem (Host et al. 2000)

The tasks are similar to those of the pilot study, though slightly shortened because of reduced available time with the subjects. It is also the case that CREWS have used the ATM example to explain their guidelines (Rolland and Achour 1998) and the Retail task in experimentation (Achour et al. 1999). The ATM is an ubiquitous software engineering problem (e.g. Kosters et al. 2001, Rolland 2002); it is therefore reasonable to use because of this apparent general acceptance through example in the literature. Both tasks should also be familiar to the subjects.

6.6.4.2 Setting (Robson 1993)

Use case descriptions should really be determined with the aid of stakeholders. It has been suggested that requirements engineers, systems analysts etc. tend to write descriptions without the direct involvement of the clients and that this is potentially risky (Ying 2001). As shown in Hofman and Lehner (2001), the most successful projects have close customer contact throughout the whole of the requirements process and that such customer input is vital to the success of projects. In this way the experiment is artificial. However, given all relevant facts and process (from elicitation already conducted), one would expect the analyst to write the first description and then validate this with clients. As such, although the experiment can be considered artificial for contextual reasons, the goal of the task is to write a description based on the information provided and the subject's pre-existing knowledge of the ATM or Retail domains. There are other aspects to consider regarding the setting of the experiment, for instance, the subjects might have rushed from other classes to participate in the experiment, thus potentially being unprepared for the experimental context. Timetabling restrictions did not allow the author to select times and locations that were convenient to all subjects. Indeed, the author was

unaware of the subjects' schedules. Background noise might also have been an issue but fortunately this was not the case in this experiment.

6.7 Conclusions

This chapter describes an experiment that compares two sets of use case writing guidelines, the CP Use Case Writing Rules and the CREWS Use Case Authoring Guidelines. The experimental subjects were asked to write a description (either ATM or Retail) with the help of the guideline set provided. The descriptions are compared by two means. Firstly, by counts of the number of times a Rule or Guideline has been used or broken, and secondly, by the comprehensibility of the descriptions as assessed by the 7 C's of Communicability.

Hypothesis 1 cannot be validated even though it is clear that the CP Structure Rules are used significantly more than the CREWS Content Guidelines in the Retail groups (B and D respectively). However, Groups A and C show no difference overall or individually. As such, the null hypothesis is accepted, that there is no significant difference between CREWS and CP.

Hypothesis 2 states that CP use cases are significantly more comprehensible than CREWS. This is shown not to be the case overall. The null hypothesis is accepted that there is no statistical difference between the sets in terms of comprehensibility. However, in terms of Consistent Structure group A (CP) is significantly better than group C (CREWS) ($p = 0.008$), as is group AB over CD ($p = 0.03$). This could be due to CP's smaller set of structure rules. Apart from the compared CREWS Content Guidelines and CG6, the other CREWS Content Guidelines were barely used. The significant difference is not reflected in groups B and D (the Retail task), thus the influence of group A on the combined scores is the significant factor.

Group B is significantly more Cogent than group D ($p = 0.04$). Group D has a large number of Text Order problems. Again, this result is not reflected for the ATM task

where there is no significant difference between the groups (A and C). Overall, there is no significant difference between the CP Rules and the CREWS Guidelines.

6.8 Discussion

It is interesting to note that the results from this experiment are different to those of the pilot study in that the pilot showed CP ATM (group A) to have the best descriptions. Groups B, C and D scored the same. In this experiment all groups scored reasonably similar (and higher) marks. However, group B (CP Retail) scored the highest overall total mean. Since the results of this experiment are opposite to those of the pilot, it appears to be the case that there is little difference in the performance of the CP Rules and the CREWS Guidelines overall. Because there is a lack of contextual setting for the experiments, that is, no contact with stakeholders and a small amount of time, it is perhaps unsurprising that there is only a small effect.

The results also suggest that there could be a high level of external variation that could not be controlled. As stated, it has not been possible to do a covariance test with course marks and this experiment (to assess, for example, if poor results here are mirrored by poor results in coursework) because subjects remained anonymous throughout. This raises an important issue: Just how good do experimenters expect student results to be? Since students are generally lacking in industry experience, perhaps experimenters have too high hopes as to student ability? This suggests that conducting experiments with students is difficult (because of time constraints, setting and a general lack of practical experience) and will not always yield results that are generalisable to the industry. However, they might be generalisable to the student population.

Chapter Seven

Experiment 2: Comprehension of Use Case Descriptions

7.1 Introduction

This chapter describes an experiment that builds upon the second part of the pilot study described in chapter 5 (section 5.4.4.2), that explored the comprehensibility of use case descriptions. This asked comprehension and logic questions about the description and found that descriptions written with the CP Rules were as comprehensible as those written with the CREWS Guidelines, though no statistical tests were applied. The experiment described in this chapter considers the comprehensibility of use case descriptions by focussing questions on a specific description. The chapter then continues by exploring the questioning approach outlined in chapter 2 (section 2.8) and suggests that by presenting the engineer with a set of general questions they can "interrogate" the use case description to elicit important design-oriented detail.

7.1.1 The Research Question

This chapter asks if CP Rules use case descriptions can elicit more information than a CREWS description. As suggested, this is done on two levels. Firstly, by asking comprehension questions specific to a description and, secondly, by exploring what specification and design information can be elicited from a description.

7.2 Experimental Design

7.2.1 Student Experience Pre-Experiment

Prior to the experiment, the subjects had attended a module on software design from their regular lecturer (not this author) and were attending a module in requirements engineering. Thus the subjects were reasonably versed in writing use case descriptions and drawing class diagrams. The subjects were asked as a seminar task to draw a class

diagram from two use cases without any guidance except the typical noun / verb search approach. After this seminar, the subjects attended an hour-long lecture on finding classes from use cases and about identifying dependencies between description events and identifying actors and "hidden" classes in the description.

7.2.2 Course of the Experiment

The subjects were presented with a description to read and were required to answer a set of questions about that description. These were,

- specific to the subject of the use case (part one)
- general – especially regarding dependencies between events (part two)
- general eliciting of actors and classes, their operations and attributes (part three)
- draw a class diagram of the problem (part four)

Subjects asked the experimenter any questions they had before the writing tasks. They then had forty-five minutes to answer the questions. This was considered enough time to answer parts one, two and three. Question four on drawing the diagram was employed as a stopgap to discourage subjects leaving early. Any different classes, operations or attributes found in the diagram were taken into consideration. All experimental materials were collected at the end of the experiment. All experimental material and full data sets are in Appendix E.

7.2.3 Experimental Treatments

There are two experimental treatments:

1. Answer questions about an ATM use case description written following the CP Use Case Writing Rules.
2. Answer questions about an ATM use case description written following the CREWS Use Case Authoring Guidelines.

The two descriptions were the highest scoring descriptions in the pilot study from their respective groups (see Appendix E2). The descriptions were therefore not identical. There is a risk that this might invalidate the experiment. However, if the information in the descriptions were manufactured to be identical then the experimental results would probably be identical too; that is, there would be no meaningful findings.

7.2.4 Experimental Subjects

There were 48 subjects who were undergraduates completing a software design module on a software engineering degree. The two experimental groups are shown in table 7-1, with 24 subjects per group.

Experimental Group	Experimental Treatment
A (24)	CP Use Case
B (24)	CREWS Use Case

Table 7-1. Experimental Groups and Treatments

The subjects were assigned to experimental groups based upon their seminar groups. For instance, seminar groups A, C and E were assigned the CP treatment (experimental group A), and seminar groups B, D and F the CREWS treatment (experimental group B). Though this is not an entirely random allocation, timetabling restrictions did not allow for more than this. The experimenter was unaware of the ability of the subjects prior to the experiment.

7.2.5 Experimental Hypotheses

There are four experimental hypotheses, based on the question types,

$H1_0$: There is no difference in the comprehension of use case descriptions written with the CP Rules and the CREWS Guidelines.

H1: Use cases written with the CP Rules are more comprehensible than use cases written with the CREWS Guidelines.

$H2_0$: There is no difference in the identification of dependencies between the CP Rules use cases and the CREWS use cases.

H2: CP Rules use cases allow for more identification of dependencies than CREWS use cases.

$H3_0$: There is no difference in the identification of actors between the CP Rules use cases and the CREWS use cases.

H3: CP Rules use cases allow for more identification of actors than CREWS use cases.

$H4_0$: There is no difference in the identification of classes between the CP Rules use cases and the CREWS use cases.

H4: CP Rules use cases allow for more identification of classes than CREWS use cases.

The hypotheses are one-tailed because since the CP ATM descriptions were more communicable than CREWS ATM descriptions from the pilot (section 5.4.4.1), it is expected that CP will perform better than CREWS here.

It is relatively straightforward to test hypothesis one. Subjects either answer each comprehension question correctly, make an assumption or answer incorrectly. A correctly answered question carries two marks, an assumption (when considered a reasonable assumption) carries 1 mark and an incorrect answer (invalid assumption, incorrect or unanswered) carries 0 marks. An unpaired, 1-tailed t-test is used to test for any significant difference between the experimental groups. Hypothesis two compares the number of identified dependencies by subjects with that of the experimenters' (the author and the subjects' lecturer) 'expert' answers, as identified prior to the experiment. The quality of

subject answers is poor, with a high number of unstated dependencies. This suggests that there might be an unequal distribution. The Mann-Whitney U test is therefore used instead of the t-test. Hypothesis three is tested (unpaired, 1-tailed t-test) by comparing the number of actors that are correctly identified by the experimental groups. Hypothesis four is tested (unpaired, 1-tailed t-test) by comparing the classes identified by the experimental groups.

Any acceptable dependencies (Hypothesis 2), actors (Hypothesis 3) or classes (Hypothesis 4) not identified by the experimenters prior to the experiment are considered valid.

7.3 Experimental Results

7.3.1 Hypothesis One: Specific Questions

All subjects received the same set of questions (see Appendix E1). Table E-1 in Appendix E4 shows the complete marks awarded to group A (CP) and table E-2 those to group B (CREWS). Table 7-2 shows no significant difference ($p = 0.94$) between the groups in answering the specific questions.

Result	Group A (CP)	Group B (CREWS)
Total	496	555
Mean total	20.67	23.13
Significance	$p = 0.94$	

Table 7-2. Totals and means for groups A and B

The CREWS description contains some internal design events; the CP use case does not. Thus, group B answered more questions factually than made assumptions. The range of marks was very narrow for both groups. Figure 7-1 shows histograms for both groups. Figure 7-1 (left) shows that the majority of scores for group A are from 15 to 24 marks,

whereas the majority for group B are from 20 to 30, showing that on average group B answered more correct questions or made more correct assumptions than group A.

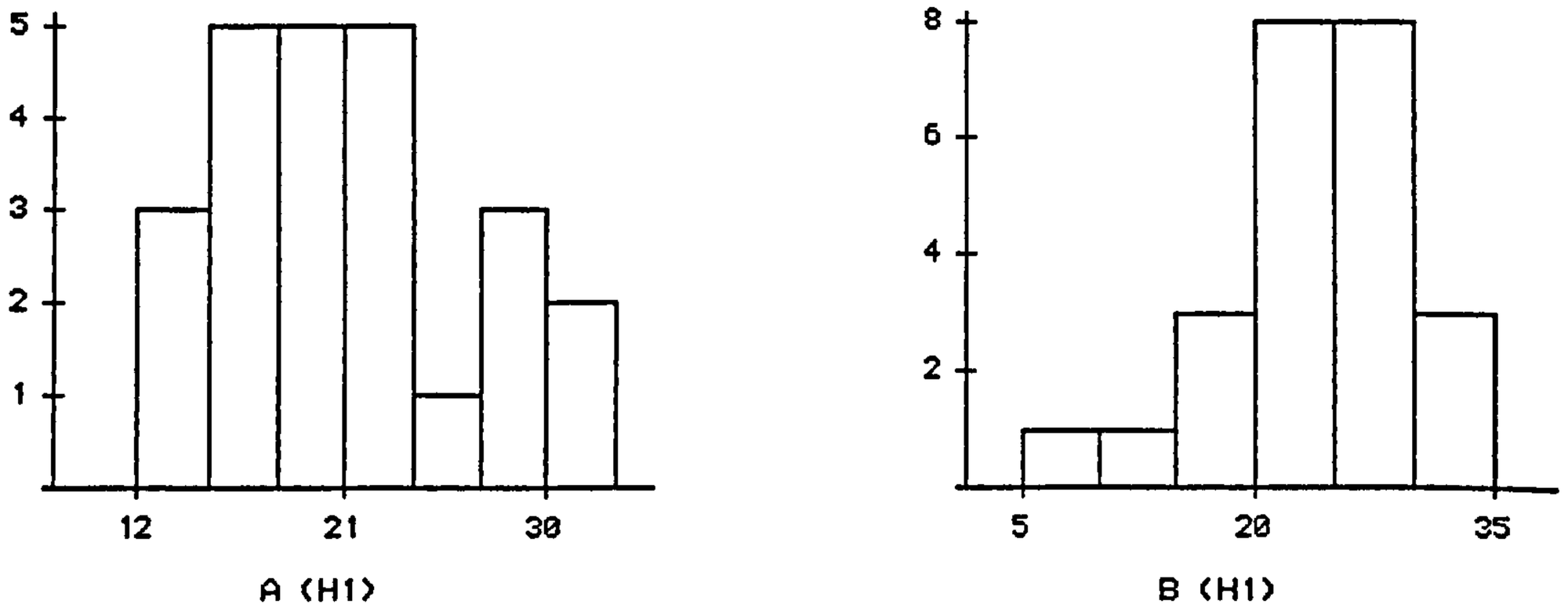


Figure 7-1. Histograms for groups A (left) and B (right), hypothesis one

7.3.1.1 Comments on the Answers Given

There is an obvious difference between the two descriptions and this does have an impact on the answers. The CP description does not contain internal design events, such as the ATM checking that the Customer's card is valid. The CREWS description does. It would therefore be expected that group B (CREWS) answer the specific questions better than group A (CP). The CP description does hold to the ideal of only describing actor-focused interactions at the interface of the machine (section 2.3.1). The CREWS description mixes abstractions. It is unsurprising, then, that group B scored better. In terms of discovering internal design issues, it might be better to write descriptions that take the CREWS viewpoint, so that a scenario (or use case) can describe actions in the problem environment, at the interface and also inside the machine itself. That is, the more there is relevant detail, the easier it is to answer design questions.

Only four group A subjects gave a correct answer to "Does the ATM validate the Customer's card?" (Answer: no). The other twenty subjects incorrectly stated that the ATM does do this. The description states:

"1. The Customer inserts their card.
2. The ATM requests the Customer's PIN."

There is no event stating that the card is validated. It is unclear why so many subjects got this answer wrong. Perhaps the author's assumption that the ATM is a familiar problem was incorrect. In comparison, group B does worse. The CREWS description states:

"1. The Customer inserts the credit card into the ATM card slot.
2. ATM validates card input."

Event 2 implies that the ATM checks on the status of what is entered into its card slot, that is, it checks the card. However, according to the vast majority of responses from group B, there is no checking mechanism. Only three subjects correctly state that the ATM does validitate the card, twenty-one subjects do not. One possible reason why there were so many incorrect answers is that the question is not precise enough and when the subject is presented with a complex (and relatively long) description, the subject perhaps searches for exact word matches only. When these matches are not clearly apparent, then answers might be incorrectly given. This, though, does not explain group A's majority response. Perhaps a lesson from this is that use case descriptions should be allowed to describe such internal events as card validation.

A question whose answer is stated in both descriptions, is "Where is the Customer's balance displayed?" Event 6 in the CP use case and event 9 in the CREWS use case describe this. Both groups score 46. That is, only one subject per group got this question wrong. Thus, when events are explicitly stated and worded in a similar way to the question, it is much easy to give a correct answer.

The questions mix abstractions, for instance, they ask what happens at the interface and also what might happen inside the machine. One question asks where the new PIN is stored once it has been changed. The CP description does not state where this occurs. Seven subjects correctly answer that the description does not state this. The rest provide

an assumption that the number is stored on the card or on the customer's bank account record. Only one subject from group B (B19) gets the answer wrong. This implies that if the reader of the description is the designer, it makes more sense to state exactly where internal actions occur. From the viewpoint of a customer, however, this is probably not that important. This suggests that descriptions should be written differently for different audiences.

7.3.1.2 Hypothesis One Conclusions

Hypothesis 1 states that the CP Rules description will be more comprehensible than the CREWS description. This is determined by the number of correct answers given, assumptions made and incorrect responses. Statistically, there is no significant difference between the groups ($p = 0.94$). As such, the hypothesis is rejected. Indeed, it is clear that the CREWS group scored higher marks than CP because the CREWS description contained some internal design events. This allowed group B subjects to correctly answer some of the questions. Group A tended to make assumptions because some of the answers were not explicit in the text.

The differences in answers given (and marks awarded) are dependent upon, in large part, the information that is presented in the description. There were few surprises with answers given and because of the nature of the questions, it is perhaps unsurprising that group B (CREWS) did better since their description was (accidentally) better suited to the questions posed. From the designer's perspective, when presented with such descriptions, they should be asking such questions about where and how a new PIN might be stored. That is, questions about how the use case is used in design ought to be considered. Unless the designer has a deep knowledge of the problem domain or 'question-description patterns' are archived, the notion of pre-preparing a set of questions that focuses on specific descriptions is suspicious. It implies that the designer knows what the description is going to be before the description has actually been written and validated. Nevertheless, it might be better to have a set of questions that are generally applicable to all use case descriptions. Instead of searching for specific information via specific

questions, the designer ought to focus on typical design issues, such as underlying processes, mapping to class structures or interfaces with other systems. In an attempt to explore this issue, the next section examines the answers provided by subjects to the second hypothesis.

7.3.2 Hypothesis Two: Identification of Dependencies between Events

Hypothesis 2 states that the CP description will enable identification of more dependencies than the CREWS description. The Mann-Whitney U test is used because there are large amounts of unidentified dependencies suggesting there might not be a normal distribution.

Dependencies are defined as the necessary conditions that must be displayed by entities involved in the description for an event to occur and to complete. In essence, these are pre- and post-conditions of events in the description. Full data sets for identification of dependencies are in Appendix E5 (the experimenters' 'expert' answers are in Appendix E3).

7.3.2.1 Group A (CP) Dependencies Identified

Group A identified one hundred and fifty-seven dependencies, an average of approximately 6.5 each (one subject, A17, did not find any, A19 scored 24 and A24 scored 16). None were identified for alternative and exceptional flows of events. Subjects possibly assumed that focus on the main flow of events was more important than any alternative sections. There were (at least) 30 unidentified pre- and post-conditions from the main flow (out of at least 68). ("At least" because some events have more than one pre- or post-condition.)

The major dependencies identified are as follows.

- Event 2 pre-condition: card is validated (9 subjects).

- Event 3 post-condition: PIN validated (14 subjects).
- Event 3 post-condition: ATM accesses Customer Account (2 subjects).
- Event 3 post-condition: ATM reads key input (1 subject). There were 17 correct identifications of post-conditions on this event.
- Event 4 pre-condition: valid PIN (12 subjects).
- Event 14 pre-condition: two new PINs are compared to be the same (7 subjects).
- Event 14 pre-condition: the new PIN is saved (4 subjects). 11 different dependencies were identified for this pre-condition.
- Event 22 pre-condition (1): withdrawal amount is less than balance (5 subjects).
- Event 22 pre-condition (2): the ATM checks it has enough money (7 subjects). 12 different dependencies were identified for this pre-condition.

Event 22's pre-condition (2) was not initially identified by the experimenters. It was considered a reasonable dependency and is therefore left in. This pre-condition might actually not relate to event 22. It would be more probable that if there were a shortage of notes, the ATM would inform the Customer before they chose an amount to withdraw, or it might not let the Customer attempt to withdraw any cash at all.

Other identified dependencies were few and far between. It can be argued that those listed above are key dependencies in the success of the description and one would expect them to be identified. However, overall, dependency identification was less than expected. This could be due to time constraints – there were many tasks to complete. It is also possible that subjects were unsure of what dependencies really meant and how one should identify them. This is questionable because the lecture prior to the experiment was focussed on this task and all subjects had attended that lecture.

7.3.2.2 Group B (CREWS) Dependencies Identified

Group B identified one hundred and forty-two dependencies, an average of approximately 6 per subject. Four subjects failed to identify any and subject B22

identified 32 dependencies. The nearest to this was B3 with 15. The most common dependencies identified were:

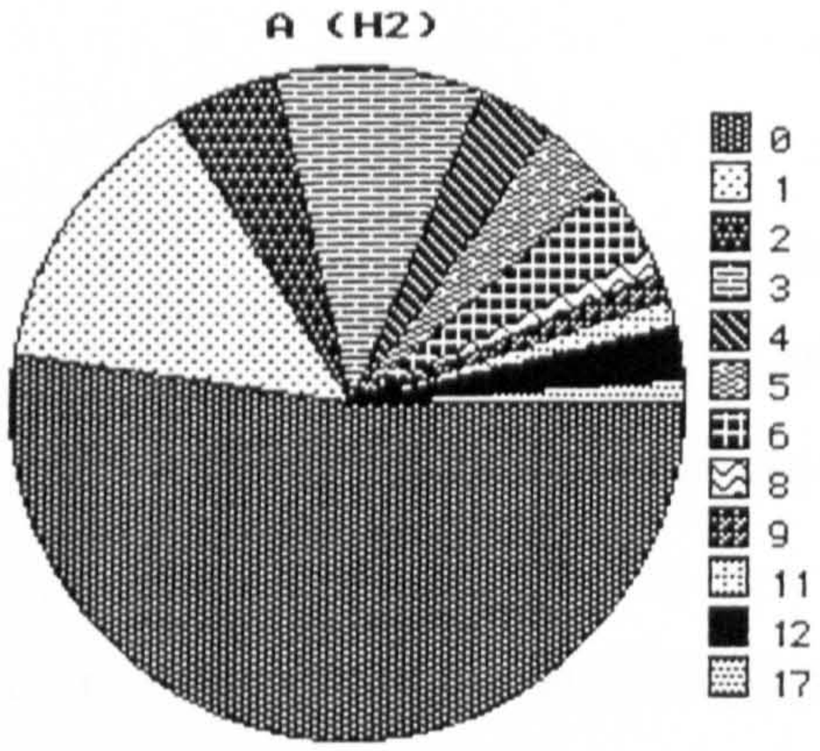
- Event 2 post-condition: card validated (10 subjects).
- Event 6 post-condition: link to Customer Account (9 subjects).
- Event 26 post-condition: withdrawal amount less than balance (10 subjects).

No other identified dependencies came close to double figures. Nor was a single dependency identified for Alternative and Exceptional flows. The three dependencies shown are among those identified by group A. These can be considered as either the most important dependencies for the subjects or the easiest to identify.

Six subjects identified a dependency that states that the ATM should check the amount of cash it has available before dispensing cash. This would occur, according to the majority, at event 30 (pre-condition). As stated above, this is a reasonable dependency to consider but it might occur before it allows the Customer to get so far into the Withdrawal service.

7.3.2.3 Discussion on Dependency Discovery

A Mann-Whitney U test shows there is no significant difference between the groups, $p = 0.12$ (ties omitted). Figure 7-2 describes the frequency of identified dependencies. Figure 7-2 (left) shows that just over half of the dependencies for the CP use case description are not identified. (Groups begin at 3 o'clock on the pie chart and are read clockwise.) That is, there are 40 events in the description, each with (at least) one pre- and post-condition, making 80 in total. 42 of these were not identified. As examples of those identified (figure 7-2 right), there are 11 dependencies that are identified once; 4 that are identifed twice etc. 17 post-conditions are identified for event 3, "The Customer enters their PIN". 12 pre-conditions are identified for event 4 "ATM displays the following options: Withdraw Cash; Withdraw Cash with Receipt; Check Balance; Order Statement; Make Deposit; Change PIN" and 12 pre-conditions for event 22 "The ATM displays a wait for cash message".

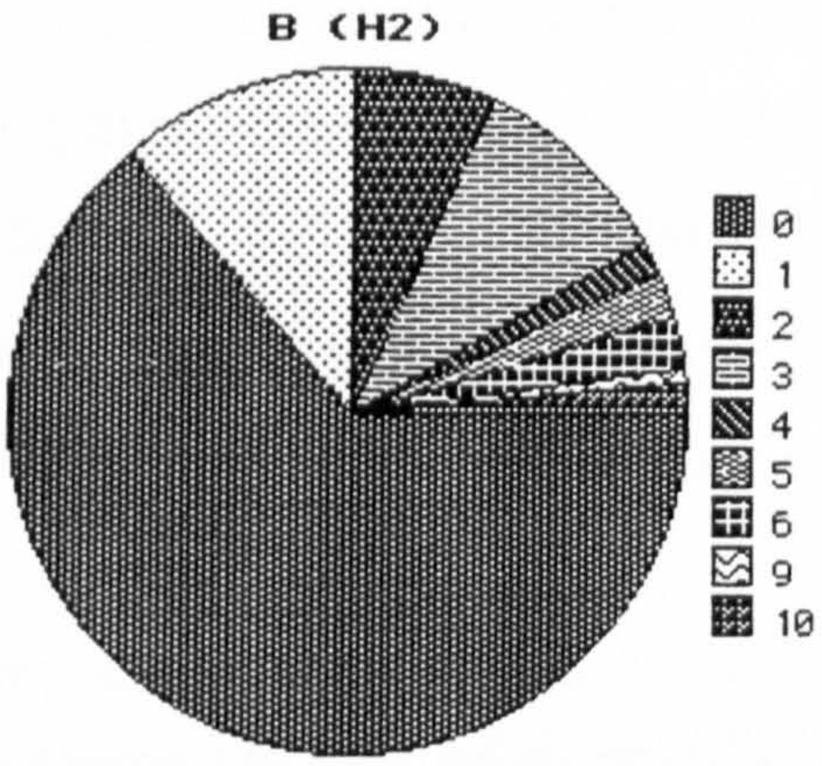


Frequency breakdown of A (H2)
No Selector

Total Cases	80
Number of Categories	12

Group	Count
0	42
1	11
2	4
3	8
4	3
5	3
6	3
8	1
9	1
11	1
12	2
17	1

Figure 7-2. Pie chart for group A (left) for frequency of identified dependencies (right)



Frequency breakdown of B (H2)
No Selector

Total Cases	132
Number of Categories	9

Group	Count
0	84
1	15
2	9
3	13
4	2
5	3
6	3
9	1
10	2

Figure 7-3. Pie chart for group B (left) for frequency of identified dependencies (right)

Figure 7-3 (left) shows that group B's subjects failed to identify approximately two thirds of dependencies but that overall there is approximately the same number of dependencies identified (48, figure 7-3 right) compared to group A (38, figure 7-2 right).

The identification of dependencies is not a particularly easy task, and it is rather time-consuming. This could be proposed as a reason why the experimental subjects did poorly. However, determining how the system is supposed to respond to events at the interface is a significant aspect of design. Examining dependencies is a way to facilitate this process.

There are advantages to exploration of dependencies. First, this acts as a validation mechanism of the use case description. The engineer has to be confident that the event under scrutiny is correct to enable movement to the next event. Second, the engineer would be keen to know how the system responds to its interface usage. There are also disadvantages to this approach and these are relatively obvious. That the work is time-consuming is apparent – the threat of "analysis paralysis" (Booch 1994, p.6) is real. Also, many of the dependencies could be conceived of as trivial, such as that the ATM responds with the right screen display dependent on the button pressed on the key pad. Overall, though, this needs further exploration as seen in chapter 8.

7.3.2.4 Hypothesis Two Conclusions

Hypothesis 2 states that more dependencies can be identified from a CP use case description than a CREWS description. Although group A (CP) finds more dependencies than group B (CREWS), the difference is not significant ($p = 0.12$), with group A averaging 6.5 per subject and group B, 6. The majority of dependencies identified by both groups related to retrieving information from a database, that the Customer's card is validated and that the amount withdrawn is less than the Customer's balance. These are perhaps the obvious dependencies. The 'everyday' dependencies that are also vital to the success of the machine, such as correctly responding to key presses, were not considered by many subjects of both groups. This implies that the subjects searched for the key or obvious dependencies only and did not consider the everyday dependencies as important or that they did not have enough time to discover all dependencies and as such focussed on the ones they considered more important. Perhaps the subjects realised the risk of potential "analysis paralysis" during this experiment and feared they would not have enough time to complete the rest of the tasks. If this was the case, then this also suggests that the experimenters' expectations of what could be achieved in the time available were too great.

7.3.3 Hypothesis Three: Discovering Actors

Hypothesis 3 states that CP use cases allow identification of more actors than CREWS use cases. Of course, if any missing actors were discovered then subjects were also expected to describe dependencies that would guarantee the actions of these actors in the description. There are, in fact, only two potential actors that could add to the descriptions. Primarily, it is expected that subjects would name the underlying Banking System that holds the Customer's Account details. The ATM should connect to this. There is also the potential actor that is the middleware between the ATM and the Banking System, which is a connection machine, called a "Consortium" by Rumbaugh et al. (1991, p.217). Many subjects name the bank's database as a class, as opposed to an actor (section 7.3.4). Subject answers are in Appendix E6.

7.3.3.1 Actors Identified by Group A

Group A (CP) perhaps have the disadvantage over group B in that A's description does not contain internal design events. As such, it is possible that group B will identify more 'missing' actors (not explicitly labelled 'actor' in the description) than group A. Table 7-3 shows those identified by groups A and B.

15 subjects out of 24 from group A identify Bank System Database as an actor and realise that this actor must be involved in retrieval and storage of the Customer's account details. Only one subject identified the connection machine between the ATM and the Bank. Subject A5 states: "The main actor that is missing is one that checks to see what card it is and then links to the bank with the relevant details."

'Missing' Actors	Group A Totals	Group B Totals
Bank System Database	15	17
Connection Machine	1	1
Other Actors	19	24

Table 7-3. Actors found by groups A and B

19 other actors were identified by 9 subjects. For example, 3 subjects identify the Cash Counter as an actor. A11 calls it a "dispensing mechanism", A19 a "money storage and dispense unit" and A21 a "Cash Counting Machine". It is debatable whether this is an actor. It is certainly not an external actor in the same sense as the Customer but it might be considered a system that interfaces with the ATM electronically, as does the Bank Database. It is more likely that this is a sub-system and at the level of abstraction of this description, should not be considered.

7.3.3.2 Actors Identified by Group B

Table 7-3 shows 17 out of 24 subjects from group B correctly identified the Bank System Database. This is only slightly better than group A (71% identification compared to 63%). Unsurprisingly, there is no significant difference between the groups ($p = 0.72$, unpaired, 1-tailed t-test).

It might be expected that group B would score 100% identification since event 6 states:

"6. ATM accesses Customer bank details from database".

However, the inverse might also be true in that since the database is mentioned in the description it could be assumed not to be missing at all. Only subject B21 identified the connection machine between the ATM and the Bank Database. This connection machine is implied since its function is stated:

"Other actors to the system would include the Server Bank Details Database holding information or either redirecting requests for a particular bank account of a particular account number, to the correct bank database server."

24 other actors were identified by 13 subjects of group B. The Cash Counter was identified by 3 subjects. 8 subjects identified Card as an actor. Again, it is arguable that Card is not an actor but is simply an entity used by the Customer actor to access the

ATM. If the card were seen to be an actor it would be wholly passive since the Customer puts the card into the ATM and removes it, the Card Reader reads the Card identification number and writes a new PIN to it if required. The Card is manipulated, as a key would be to unlock a door. However, such questioning of the function of the Card leads to a consideration of the information that can be written to it and read from it and can only be beneficial in understanding the mechanisms of the ATM.

7.3.3.3 Discussion on Actor Identification

Dependent upon the kind of description presented for examination, relevant actors can more, or less, be identified. The CREWS description (group B) contains much internal design and so more actors should be apparent. Interestingly, group A identified almost the same number of actors as group B and the majority of both identified the Bank System Database. It is also interesting that many actors were identified that are part of the ATM itself but are hardware components as opposed to human or software-based. The sensors indicating insertion or removal of objects to and from the ATM were identified by both groups. These entities are considered as classes in the class diagram (see next section) but they would have to be considered as sub-systems that interface together to perform the tasks of the ATM. So the notion of interrogating the use case description for any missing actors can raise design issues or resolve them.

7.3.4 Hypothesis Four: Identification of Classes

Appendix E7 describes a class diagram suggested by the experimenters that corresponds to both the CP and CREWS descriptions. The diagram does not include a connection machine ("Consortium") between the ATM and the Customer's bank as depicted by Rumbaugh et al. (1991, p.217) since the diagram primarily deals with the interface of the ATM itself, though, of course there is the necessary CustomerAccount class. If subjects suggest the connection machine, this would be a reasonable assumption. However, the focus is primarily on retrieving classes from the experimental treatments. Both descriptions explicitly state these classes are present: Customer, Card, Receipt and

(perhaps slightly obscurely) the OptionsScreen (called Options in A and OptionMenu in B). Group B's description also states there is a CardSlot and CustomerAccount. Some discovered classes are considered 'hidden' classes, because they are not explicit in the description; they are not nouns found in the text, for example, ReceiptPrinter, CashDispenser and ReceiptSlot.

7.3.4.1 Classes Found by Group A (CP)

Table 7-4 shows the total number of correct and erroneous classes identified by groups A and B that match the experimenters' class diagram. Complete sets of subject responses are in Appendix E6.

Elements	Group A (CP)	Group B (CREWS)
Correctly identified classes	41	77
Erroneous classes	31	53
Correctly identified operations	76	70
Misplaced operations	160	137
Correctly identified attributes	30	42
Misplaced attributes	66	44

Table 7-4. Groups A and B classes, operations, attributes identified

'Correctly identified operations' and 'correctly identified attributes' refer to those operations and attributes that are part of a correctly identified class. 'Misplaced' operations and attributes are those that are correct to the diagram but incorrectly located within the wrong class.

The results are not particularly encouraging for group A since the average correct class identification is less than two. The two most frequently identified classes were Customer (sometimes User) and Card (9 subjects identified these). In terms of the abstraction in this diagram, it is questionable whether the Customer and Card should be there at all. Indeed, this might be considered as "naïve real-world modelling" (Isoda 2001, p.155) and these

classes ought to be viewed as actors since at this level of abstraction the classes inside the machine that refer to real-world entities should only hold information about those real-world entities and not be a direct representation of them. (This is why these classes are labelled as coming from the Analysis Model in Appendix E7.) This suggests two possibilities. First, group A's subjects were not particularly good at object-oriented analysis. However, time constraints reduced the think time that the subjects had for class identification compared to an industrial context. Second, the identification of classes from use case descriptions might not be that obvious. The description might need internal design events to help identify classes. Though this author argued that internal design should be kept out of descriptions at this level of abstraction (section 2.3.1), this might need some re-thinking.

Seventeen subjects named the ATM as a class. The ATM is not a single design class but the system under design itself. There were 160 "misplaced operations". Many operations were assigned to the ATM class but should have been assigned to others. For instance, subject A2 states that the ATM should *produceReceipt()* and *distributeMoney()*. These operations should be assigned to elements of the ATM: the ReceiptPrinter class and the CashDispenser, which sends cash to the CashSlot class. 5 subjects identified PIN as a class but this is an attribute of Card.

7.3.4.2 Classes Found by Group B (CREWS)

Table 7-4 shows the total number of classes identified by group B that match the experimenters' class diagram. Group B correctly identified 77 classes, an average of 3 per subject. 16 subjects identified the two problem domain (analysis) classes Customer and Card. 21 subjects incorrectly identified the ATM as a class. Like group A, there were a large number of misplaced operations (137). This is perhaps unsurprising since a use case description is primarily about events, which can be roughly translated into operations. 9 subjects identified the Keypad as a class (compared to 0 from group A). This class was missed by the experimenters. Since there are classes such as the CashSlot,

CardSlot, ReceiptSlot and various Screens that are information and physical entity outputs, and Customer input is via the Keypad, this is an acceptable class.

7.3.4.3 Comparing Groups

Group B identified almost double the number of correct classes compared to group A (77 to 41). When an unpaired t-test is applied, hypothesis four is rejected ($p = 0.99$). This suggests that group B identified significantly more classes than A and reflects, to a degree, on the abstraction represented in the descriptions presented to both groups. Group A's description does not mention internal design considerations and the identification of design classes would be less apparent. Group B's description is a mix of abstractions aiding identification of design classes. However, in terms of the CustomerAccount class, there is equality of identification; it was identified five times by each group. One might have expected group B to do better than this. However, the difference between the groups is shown by the identification of other classes, not least that of the Customer and Card classes which group B identified almost twice as much as group A (16 subjects in group B to 9 in A).

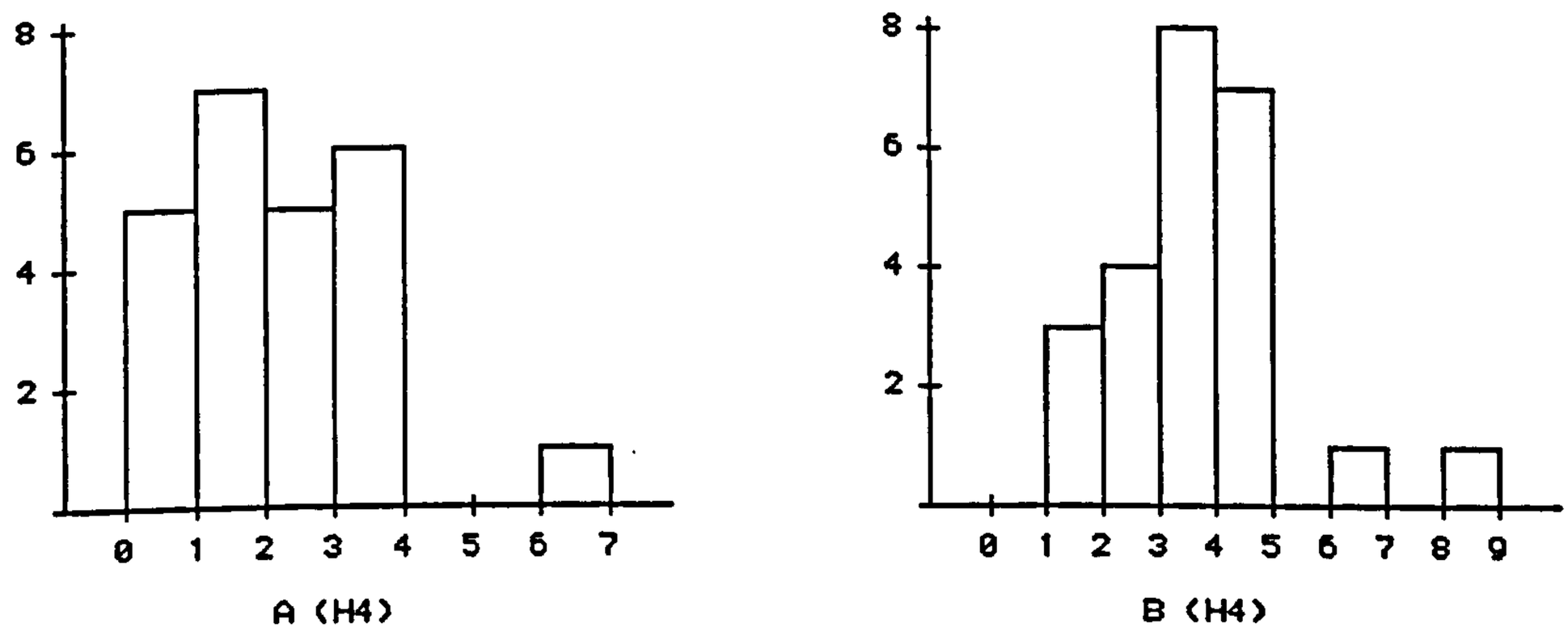


Figure 7-4. Histograms for groups A (left) and B (right) of number of classes identified

Figure 7-4 describes histograms for groups A and B. The histogram for group A (left) shows that all but one of its subjects identified 3 classes or less and 5 subjects identified none. It is disappointing that these subjects identified no classes – it might be that they

were not certain what a class might be in the context of the description. In comparison, all of group B's subjects (right) identified at least a single class. 9 subjects identified four or more classes.

Curiously, group A identified more correct operations (76) than group B (70) despite identifying many fewer classes, averaging 1.9 operations per class, compared to group B's 0.9 operations per class. However, group B identified more correct attributes (42) than group A (30), though averaged slightly less attributes per class (0.5) than group A (0.7). Per correct class, there is a rather low number of operations and especially attributes. The large numbers of operations and attributes identified overall by both groups (table 7-4) suggests that the subjects had enough time to answer the questions but that they were not particularly good at doing object-oriented analysis. This suggests, as with the conclusions of chapter six, that perhaps, the student experiments were not necessarily as successful as hoped and that they might not reflect the abilities of experienced practitioners.

7.3.4.4 Hypothesis Four Conclusions

Hypothesis 4 states that the CP description will lead to more discovered classes than the CREWS description. This is shown not to be the case. In fact, group B (CREWS) identified almost twice as many classes as group A ($p = 0.99$). This can be seen as a highly significant difference in favour of group B. The difference does appear to be aided by the level of abstraction described in the use case descriptions. The hypothesis is rejected.

7.4 Threats to Validity

7.4.1 Statistical Power

It was considered difficult to calculate the effect size from the pilot (the number of recommended experimental subjects depends on the effect size (Miller et al. 1997)). The

null hypothesis has been accepted for all four hypotheses. The recommended sample size per group is 30 as a starting point for generalizing results (Salkind 2000). The sample in this experiment is a little smaller with 24 subjects per group and the results are considered potentially acceptable for a student population with similar experiences, as opposed to the software engineering industry, since their practical experience would probably have an influence on the outcomes and one would expect them to identify more classes, actors, dependencies etc.

7.4.2 Internal Validity

7.4.2.1 History

Not all subjects participated in the experiment at the same time. The experiment occurred in separate sessions over a week, due to timetabling restrictions. There is a risk that subjects would pass on any knowledge to subjects yet to take part (causing diffusion of treatments). The only reasonable way to control this was by retrieving all experimental material at the end of each experiment session.

7.4.2.2 Maturation and Mortality

The experiment lasted an hour (the writing part 45 minutes). Concerns over boredom or over-enthusiasm were not considered significant. No subjects dropped out of the experiment. It is probably the case that subjects were under time pressure. After answering the specific questions (section 7.3.1), examining dependencies (section 7.3.2) and actors (section 7.3.3) it is possible that the subjects were over-tasked. However, there are a large number of classes and operations identified overall (section 7.3.4). There were over 200 classes named by subjects (although 84 were incorrect), over 440 operations and over 180 attributes identified. As such, subjects did appear to have time to answer the questions.

7.4.3 Construct Validity

Are the results generalisable or is the experiment threatened by design problems or social issues?

7.4.3.1 Inadequate Preoperational Explication of Constructs (Wohlin et al. 2000)

All hypotheses were significance tested. All measures were checked by both experimenters and a consensus agreed on the correct answers and also responses that were deemed correct but initially missed by the experimenters. It is unlikely, therefore, that the measures (correct answers to H1, dependencies in H2, actors in H3 and classes for H4) are incorrectly designed.

7.4.4 External Validity

The results of the experiment cannot be easily generalised to every software company employing use case descriptions, because software practitioners might be expected to be more experienced. However, the results might be generalisable of undergraduate software engineering students who have taken a course in software design for the first time.

7.4.4.1 Nature of the Problem (Host et al. 2000)

The ATM example is used because the CP ATM description scored the best marks in the pilot study (chapter 5) and for comparative purposes it was necessary to present the other experimental group with the best CREWS ATM description. The point of using these descriptions instead of fabricating one's own was simply to avoid introducing bias into the experiment. The ATM is still a popular choice of example for use case descriptions (e.g. Kosters et al. 2001, Rolland 2002) and use of an ATM was expected to be familiar to the subjects; it is thus considered valid to use here.

7.4.4.2 Setting (Robson 1993)

Engineers in software houses would probably interrogate descriptions and would probably have other distractions so that a complete interrogation might take place over a period of time. It is also probable that a practitioner would validate their answers with a colleague. Therefore, the classroom at a university is not a typical setting, though it is for conducting experiments. There are other aspects to consider regarding the setting of the experiment, for instance, the subjects might have rushed from other classes to participate in the experiment, thus potentially being unprepared for the experimental context. Timetabling restrictions did not allow the author to select times and locations that were convenient to all subjects. Indeed, the author was unaware of the subjects' schedules. Background noise might also have been an issue but fortunately this was not the case in this experiment.

7.5 Experiment Two Conclusions

This chapter described an experiment that compared the comprehension of two use case descriptions. Four hypotheses stated that the CP use case would, first, be more comprehensible than the CREWS use case; second, that it would find more dependencies than CREWS; third, that it would find more actors than CREWS; and fourth, that it would find more classes than CREWS. All hypotheses were rejected. It was found that CREWS did better than CP in all hypotheses except in the identification of dependencies.

7.6 Discussion

The results suggest that for more correct class identifications, descriptions should contain mixes in levels of abstraction and indeed, the CREWS scenario model (section 2.3) suggests this; this can be considered a description from the designer's viewpoint. The larger number of classes identified by group B points to this being the case. However, the author stated that descriptions should be written at the level they are supposed to represent (see section 2.3.1). As the OMG (2001) states, "use cases … indirectly state the

requirements the specified entity poses on its users; that is, *how they should interact* so the entity will be able to perform its services," (p.2-141, author's italics). The OMG, though, avoids being specific about mixing abstractions in one use case description (section 2.3.1). It appears to be the case that when a description is used by a designer, it might be better to contain internal design events since this is the kind of information the designer would look for. Indeed, it has been shown that designers think and move through different levels of abstraction when considering a design problem (Guindon et al. 1987) and thus if use case descriptions contain mixed abstraction levels, then this might aid the designer in constructing a design.

Not all classes were identified and there were a large number of incorrectly placed operations and attributes. It is clear, though, that the notion of interrogating descriptions in this way does yield a lot of design-oriented information. It also makes one think more about the system and how it relates to the problem environment. Even if descriptions are going to be at more than one level of abstraction, additional questions that enable better interrogation of aspects of descriptions are worth exploring further. This is taken up in an industial setting in the following chapter. This risk in moving towards a general set of questions, away from specific domains might be "naïve" (Potts 1993, p.26). However, the exploratory nature of this experiment and the subsequent case study allows for the expansion of aspects of the questions.

7.6.1 Expanding the Questions

The questions presented to experimental subjects need to be expanded upon to take into consideration further aspects of today's software systems. To have more coverage of issues of design, for instance, the questions ought to take into consideration connections to other software systems either via hardware ports or electronically. As such, it is important to ask questions about the system interfaces. This includes interface design solutions. Though it is recommended to avoid this if possible in analysis (Cockburn 2001), the interface has to be built eventually so it might be useful to take this into consideration. Since the goal of the questions is to ease the passage into design, it is also

important to consider what the system itself must do to guarantee success of the events in the descriptions. This approach also helps maintain traceability between the classes and descriptions. Figure 8.1.1 (Appendix F3) and the class model fragments in the second section of chapter 8 (section 8.5.2) show examples of this traceability (Lubars et al. 1993).

7.6.1.1 The Question Set

This section lists all questions. However, not all questions are relevant to all events. Some questions were identified from the literature survey (section 2.8). This indicated that identifying dependencies was important to validating specifications (Sutcliffe 1998). Sutcliffe suggested that dependency discovery would reveal internal design issues. This is thus explicitly addressed by both the Dependency and System questions. The Actor Set is derived from the questioning approach of Armour and Miller (2001). As stated above, modern computer systems are rarely standalone (Bray 2002), therefore, external interfaces (hardware and software) need to be addressed through the Interfaces Set. The Deriving Classes Set is derived from typical approaches such as noun and verb identification (e.g. Coad and Yourdon 1991, Arlow and Neustadt 2002).

7.6.1.2 Dependencies (pre- and post-conditions)

This is a key facet of the Question Set and a starting point for validating the use case description.

- Each event and resulting action in the use case is dependent upon what?

The pre- and post-conditions of each event ask what must guarantee its success. This forces the engineer to carefully consider whether all necessary information has been conveyed, acting as a validation mechanism.

- What dependencies do you have to assume, and where?

It might be the case that events are dependent upon internal mechanisms of the machine or upon other systems. These need to be noted and clarified. It is worth considering whether the Span of the description is sufficient.

- Are there timing constraints that the event has to adhere to? What happens if the timing parameters are broken?

This is fundamental to safety critical systems. Management of timing or other kinds of failures is vital (Sutcliffe 1998).

7.6.1.3 Actors

- Do you have to assume there are other actors involved in the system that are necessary to assure dependencies but are not stated in the use case?

Actors might not be defined in the description because they play a minimal role or are considered secondary. It is still important to know what roles these actors play so that the problem (or solution) can be specified more precisely.

- How and where do these actors link to the description?

It is important to know about how these actors are involved in the problem (and system) and what type of connection they might have, i.e. what is their interface?

- Is the actor a 'go-between' between the system and someone else?
- If so, what do we know about the Customer who uses the actor at the interface?

It is often the case that the actor directly interacting with the machine is not its target audience, but a go-between. It is uncertain how to describe the actor on whom the requirements are supposed to have an effect. Graham (1998) notes this as a particular

pitfall of the use case approach. Armour and Miller (2001) label the go-between actor a "facilitator" or "proxy" (p.11).

- Is the actor active or passive?

A passive actor might be a repository or record log. If it is active, what triggers the actor's action?

- Are any of the actors other systems?

This can be software or hardware. Knowing that there are connections to other systems and/or applications helps determine the boundary of the system and direct the designer's attention to the systems's interfaces.

7.6.1.4 Interfaces

- What kind of interaction is there between actor and system? E.g. completing a field at an interface, pressing a button on a machine, speaking to a system and getting a verbal response or a visual level of communication?

This will give the Interface designers some lead as to what kind of system the Customer requires.

- Does the use case impose any style of interface design?
- Does the use case suppose any style of interface design?

There are warnings about putting graphical components into a use case description, such as a button or a text field, because of the fear of slipping into design too early (e.g. Anton et al. 2001). In general, this is a good approach but at some point there must be a cross over from high-level abstractions down into the detail of the system. Interfaces need to be described as closely as possible to the envisioned final product.

- What interfaces are there to other systems?

Modern systems are rarely standalone. A fundamental task of specification is to determine interfaces to other systems, to describe what kind of interface that is (e.g. API) and if there is any middleware (e.g. CORBA, COM) and to consider the solution data structure that goes into and out of the machine (Bray 2002).

- If the description is internal design, what are the interfaces between sub-systems?

Use cases that are entirely internal to the system need to consider interfaces between subsystems. In the case study, there is a typical three-tiered architecture of interface, business model and database. It is important to know where and how these subsystems connect with each other.

7.6.1.5 System

- What does the system have to do to guarantee success of the event?

Only assumptions about design are made at this point. Focus should be on what processes the system might have to perform and what classes would be necessary to support those processes.

- Are these actions only internal to the system; that is, does the system have to employ an external device to make this action work?

Systems will have to connect to others, such as to the credit checking system in the case study (chapter 8). Consider how the system responds to external events in terms of the classes that might perform the tasks executed at the interface. Also, there needs to be consideration of the processing of those tasks, which might be internal design level use case descriptions, class models (for the static structure) and interaction models, once the static structure and descriptions are determined.

7.6.1.6 Deriving Classes

Once these questions have been considered then list the classes, processes and operations identified. Fit these to relevant classes. For instance, in the questions on dependencies, what conditions are necessary for the success of the event? This suggests operations rather than classes. This is also the case with the interfaces and system questions. The actor questions suggest classes but in terms of design these might not be of major value since the actors are all external to the system (except if the description is at an internal design level). Nonetheless is it still worth examining the actor answers because the possible classes do suggest attributes. Also, some problem domain classes may need to be represented by the machine in some way. Thus actor consideration will help in this task.

In identifying the classes, a typical approach is taken once the questioning is completed.

- Examine the answers to the previous questions. List all relevant entities that are possible classes. Label classes to their stereotype for easier identification.

The three stereotypes (often reserved for object identification) are *interface*, *control* and *entity* (Jacobson et al. 1992). Two others are added, *GUI* and *database*. The *GUI* label indicates that this class is concerned with a human user interface. *Interface* classes concern themselves with connecting to other software systems or hardware devices. These two types of class deal with the external specification of the machine.

The database classes act as a 'translator' between the underlying business model and the actual database. These classes make sure the data structures passed between the database and the business model are acceptable. (When implemented in Java, the database classes might form SQL queries that then use the standard JDBC API to connect to the actual database.)

Control and entity classes construct the underlying business model. Entity classes act as data stores that can be manipulated. Control classes act as the drivers of various

processes, where necessary. Some functionality is contained in the entity classes but these generally refer to accessing the attributes of those classes.

- Examine the answers to the above questions to identify services and attributes. Place them in the appropriate class.

As stated, many operations can be found in the Dependencies, Interfaces and System questions. Placing these into the correct classes is not a trivial matter but once classes have been identified, this makes it easier.

- If possible, consider structures such as inheritance, aggregation and association.

This is not easy to determine from use cases. Some obvious structures might be clear, such as different access levels to a security application. Designers should use their object-oriented expertise in helping determine such relationships.

- Do you have to invent any further classes?

It might be necessary to invent some additional classes to those already identified, which are needed to fulfil their tasks. There is nothing unusual about this since it is unlikely that every class for the design of the system can be found in the use case descriptions. Indeed, these questions are designed to find the hidden objects, as well as act as a validation of what is already there.

The Question Set is explored in the second part (section 8.5.2) of the case study in chapter 8.

Chapter Eight

Case Study

8.1 Introduction

This chapter describes an industrial study undertaken at a financial company in the City of London. The study's purpose is to further explore the use case description heuristics suggested in this thesis in an industrial environment on a live project because it presented the author with a larger scale problem in a real setting that involved a number of software engineers. Due to project time constraints only the Question Set (see section 7.6.1) was given the opportunity for full stakeholder validation (a stakeholder is defined as a member of the IT department who was involved in the case study). This chapter is divided into two parts. The first part relates to the company, the case study project and the use of the CP Rules and 7 C's on use case descriptions. The second part, section 8.5.2 onwards, relates to the discovery of design elements via the Question Set. The case study itself was complicated and produced some interesting findings, some of which are not entirely central to this thesis. Therefore, this chapter will remain narrowly defined, focussing on the issues of the thesis and only brushing over other aspects when considered necessary.

8.1.1 The Company

The chief business of Company X (named as such in this thesis to maintain its confidentiality) is the building and supporting of complex applications for online buying and selling of shares, stocks and investments, such as ISAs. The high quality of the online application, with its potential for the American market, led to a multinational finance company buying Company X just prior to the study.

Company X states that is an IT-driven company. IT plays a central role in what is produced and when. The demands on IT staff are high. Deadlines are short and

requirements are often sparse. Members of the IT department are domain experts and highly experienced in building the financial products Company X offers.

8.1.2 Case Study Process

The process for the case study began when the author was approached by a member of Company X's IT department to assess whether the author could offer them some requirements process guidance. There ensued a number of meetings with the Vice-President of the company and members of the IT department to determine if the author could offer Company X advice and to ascertain if Company X would allow the author to use any information from this as potential case study material.

Due to the impending purchase of Company X by the multinational, the case study was put on hold for three months. A further meeting with the Vice-President in February 2001 led to the author being asked to work on a current live project (called Project F for confidentiality, which is outlined in section 8.1.4). The author worked on the case study from March to July 2001.

The case study began with the author informally interviewing the Project F Project Manager. The interview notes are found in Appendix F2.1. The author also discussed the project informally with the Vice-President and the IT Development Manager as well as members of the Project F IT team. The author observed the print room at work at the start of the project (see Appendix F2.2). A formal meeting with the Project F team occurred half way through the project to discuss current progress and project issues, as described in Appendix F2.3.

The author gave two presentations outlining the case study findings to various members of the IT department, one half way through the project and one at the end. Feedback from these is presented in Appendix F2.4. Formal interviews with four members of the IT department were also conducted at the end of the project to provide stakeholder feedback. These are described in Appendix F2.4. Not all questions were answered by the

interviewees due to their field of expertise and because of the time they had available for interview. The author had planned to interview two other members of the IT department but because of personal and other work commitments (many related to Company X's purchase by the multinational), this was not possible.

The author was expected to feedback to the Project F Project Manager every week on progress. Feedback tended only to occur twice a month due to the other business commitments of the Project Manager and the state of progress on the case study at those times. The author did informally discuss the case study on a regular basis with the IT Development Manager and other members of the project team.

The IT team worked on a number of projects throughout this case study and as such could not dedicate as much time as they wished to the author and his work. This meant that feedback was not as forthcoming as hoped because they did not have the time. There were also serious concerns about the potential restructuring of the company after the multinational purchase. This meant that the IT department had a number of important considerations to occupy themselves besides their normal very busy schedules, including a major live release, and this author's work. Thus the author was somewhat at the "mercy of project events" outside his control (Potts 1993, p.27). The events relate as much to other projects and business concerns as they do to Project F.

The author concluded his case study work once an overall document had been produced for the Project F including process and design documents (see appendices F3 and F4).

8.1.3 The Company X Software Development Process

The following are comments made in structured interviews by members of the IT team. Full interviews are in Appendix F2.4. Each interviewee was asked:

Can you describe your current requirements analysis and design process?

Stakeholder: IT Development Manager:

"We have a rapid development approach... What we do depends on the size of the project and the time we have until delivery. We... get basic requirements... some business requirements... Then we code from here!"

Stakeholder: IT Quality Assurance Manager

"Now there are bigger projects with larger teams, processes are required (but are missing)."

Though Company X have successfully used scenarios in their requirements process (Cox 2000c) and individual developers do their own design work, often there is little more than a walkthrough of the requirements to sign them off before coding begins. This has been recognised as typical in small and medium enterprises (Kamsties et al. 1998). Now that Company X are part of the multinational, there are larger projects planned and an overall IT development process will probably be required. The work done by the author was used by Company X to assess how the techniques and processes used might fit Company X's wider development process. Though the author was working with Company X primarily to test the techniques suggested in this thesis, he was also using many other techniques (for example, Role Activity Diagrams (Ould 1995) to model processes). Thus the author's work was being assessed to see if there were techniques or processes that Company X could use *and* present to the multinational as evidence of a more detailed development process to help procure future project work.

8.1.4 The Case Study Project

The project is called Project F. This entails re-engineering the process of printing online customer account applications. The application packs, determined by the type of application, are then sent to the customers for signing.

The front end (web site) was not to be altered, except when new products are introduced. Only the printing process and the internal structure of the application required re-

engineering, with the goal of reducing printing time for each application batch. The in-house printing process is dependent upon the information received from the online application. The system determines and places applications in lists of pack types (called pack codes). There were 23 pack types in use at the start of the case study and this was expected to rise to 45 with new products and with the planned introduction of telephone applications. The number of pack codes is too high to have an efficient printing process and should be reduced especially as multiple pack types cannot be printed at once. An important goal is to reduce packs to only two types.

To reduce the number of packs, a re-engineering of the design of Project F was required. The few project documents made available to the author are presented in Appendix F1. The lack of design documents makes it difficult to understand how the application works save from the perspectives of the Customer and the Print Room staff, then to work inwards from there.

The author's role from the point of view of Company X was to model the current Project F process to allow for a re-engineering of the printing process and underlying design. In parallel, the IT team were also to re-engineer the actual application and comparisons were to be made between the author's and the IT team's efforts.

Since the aim of the study is to examine the heuristics suggested in the thesis, the details of the Project F process are described only to provide background to the application of the heuristics. The full details of the Project F processes, as modelled by the author, are found in Appendix F3.

8.2 Case Study Design

This study also contains amounts of action research as well as other more typical elements of the case study. The author acted as a member of the IT team working on Project F and as such this can be considered action research (Zuber-Skerritt 1996). To provide a structure, this section follows Yin's outline case study design (1994). The

author modelled from external processes and interfaces inwards. This "outside-in" (Potts 1993, p.22) analysis was appropriate because the external boundaries were already established.

8.2.1 The Study's Questions

The study initially set out to question how useful the CP Rules, 7 C's and Question Set are in an industrial setting.

8.2.2 The Study's Propositions

There are no propositions that could be tested for statistical significance. There is a comparison of the approaches taken by the IT team and the author, but since the IT team did not produce equivalent documentation, the comparison is mainly based on interviews and discussions.

The goal of the case study is to apply the techniques developed in the thesis to an industrial project. As such, the propositions suggested are,

1. The CP Rules are of value when writing use case descriptions in an industrial setting.
2. The 7 C's of Communicability are a useful technique for assessing the internal structure of the use case descriptions.
3. The Use Case Question Set helps identify classes and other design elements from use case descriptions.

The first two are entirely subjective to the author's viewpoint because time constraints did not afford the IT team the opportunity to employ use cases (members of the IT team were working on more than one project). The third propostion can be evaluated more conventionally since the Question Set received feedback.

8.2.3 The Units of Analysis

The units of analysis are the implementation processes and effectiveness of the CP Rules, the 7 C's of Communicability and the Question Set.

8.2.4 The Logic Linking the Data to the Propositions

As Yin (1994) states, this is a difficult task. Therefore, the majority of the logic linking the data to the propositions is provided by the author's viewpoint and stakeholder feedback in the form of interviews.

8.2.5 Criteria for Interpreting the Findings

There is critical feedback from stakeholders and this is drawn upon to provide a validation of the Question Set. However, the findings regarding the CP Rules and 7 C's are those observed by the author and thus the interpretation placed upon the findings is that of the author's. Galliers and Land (1987) suggest that descriptive research is a valid approach, and some of the work here follows that lead, with the caveat of author subjectivity kept in mind.

8.3 Project Goals

These are the Project F goals from Company X (Appendix F3).

G1. The print process has the capacity to print 1,000 Customer applications per day.

There is no proposed increase in hardware. As such, the software design will have to ease the print process by reducing pack types and make printing packs easier. A problem recognised by the multinational when acquiring Company X was the complexity and inherent risks of the printing process as-is.

G2. The selection of printing jobs is more efficient than the current system.

There have been no timing or complexity measures taken. However, the current process is very labour intensive and open to much human error. It has been suggested that the generation of print jobs and the selection of print trays should be automated. Automation is a long-term goal and not part of the suggested solution in this study.

G2.1. The design of the application process is to facilitate more effective generation of pack types to allow easier printing of applications.

This goal relates to the internal design of the system. The system is modelled from a business process viewpoint and from Customer interactions at the interface.

G1 might be considered a capacity requirement but G2 and G2.1 are too coarse-grained for this.

8.4 The Study

8.4.1 The Print Room

The job of the print room staff is to print applications. The example description (Print Room (pr) Use Case 2: Print Mailing List) describes this important process. It also shows application of the use case writing heuristics (CP Rules and 7 C's) in industrial practice. Appendix F3 contains all use case diagrams and corresponding use case descriptions of the print room processes. The use case diagrams were met with interest and considered a way to present key functions to the business managers:

"The use case diagram is an excellent diagram to present to the business as an overview of what's to be done" – IT Vice-President and Company X co-Founder (Appendix F2.4).

However, an assessment of the UML diagram notation is beyond the scope of this thesis since its primary focus is on descriptions.

prUC2: Print Mailing List (Pack Types)

Actors: Print Room Staff member(s), Printers

Context: It is either 9am, 12pm or 3pm and the Print Room staff are readying for a new print run to commence.

Pre-conditions: Files have been imported (copied) to the Access database from the Back Office; the Access Database is functioning normally.

Main flow of events:

1. The Print Room Staff member selects "Pack Type" to print from Pack Type list.
2. The Print Room Staff member selects "Print Mailing List".
3. The Print Room Staff member sees instructions for opening a Company X account in Microsoft Word.
4. The Print Room Staff member sees a mail merge in Microsoft Word.
5. The Print Room Staff member selects "Print" from the File menu.
6. The Print Room Staff member selects the printer to fit the Pack Type.
7. (Optional) The Print Room Staff member selects the number of document pages to print.
8. (Optional) The Print Room Staff member checks the correct forms are in the printer.
9. The Print Room Staff member starts the print run.
10. The Printer prints the documents.

Exceptional Flows

e6. The Print Room Staff member selects the wrong printer.

e6.1 The Print Room Staff member cancels the print run. (Here or 10?)

e6.2 The Print Room Staff member selects the correct printer.

e6.3 The Print Room Staff member restarts the print run.

e9. The print run does not complete.

e9.1 The Print Room Staff member cancels the current print run.

e9.2 The Print Room Staff member locates the cause of failure. (?)

e9.3 The Print Room Staff member rectifies the problem. How?

e9.4 The Print Room Staff member restarts the print run.

e10. The printer feeds papers from the wrong tray.

e10.1 The Print Room Staff member does what?

Post-conditions: The packs are printed.

<end of use case>

8.4.2 Assessing the Description

8.4.2.1 Use of the CP Use Case Writing Rules

Events 1 and 3 are rather long but they can be described as CP Structure 2 (subject verb object prepositional phrase). Event 3 could be better written:

3. The Print Room Staff member sees instructions *to open* a Company X account in Microsoft Word.

Events 3 and 4 are unusual because they describe what the actor sees. These are in the description because the Print Room staff indicated that if these are not visible then something is probably amiss. Events 7 and 8 are unusual because they contain '(Optional)', meaning this event can occur but does not have to if not required (Kulak and

Guiney 2000, p.93). Though the use of 'Optional' is not a CP Rule, it was considered expedient here to avoid writing a number of alternative scenarios just to accommodate one different event. The use case does adhere to the CP Rules though as shown this is not always possible. As a general guide the CP Rules are effective, but they occasionally need to be broken. Perhaps it is better to use the CP Rules for guidance only. If one religiously adheres to the Rules then the events might not be described in the most comprehensible form for the context at hand.

8.4.2.2 Validating the Description with the 7 C's of Communicability

Validation was an informal reading through the 7 C's heuristics as a checklist. There is an identified problem with the application of the 7 C's in that there is a large degree of subjective judgement involved in their assessment. Since many of the C's are semantic notions, this subjectiveness seems somewhat inevitable.

1) Coverage. There is a potential Span problem in the exceptional flow event e9.3, which is flagged with a 'How' question because it needs further investigation. The description should not normally be signed off without resolving this issue. There do not appear to be any Scope problems, though as stated, events 3 and 4 might be considered too much information since they are cognitive events. However, the context suggested that they be included though this is a subjective opinion. Span and Scope might be increased or decreased dependent upon the reader's viewpoint. Since the selection of pack type to printer is important (due to application form sizes) this might be necessary to expand upon. The Span of events 1 and 6 could be considered insufficient information to cover this important point. However, to go into further detail in this description might detract from its overall task of describing a print run. The details could be expanded upon elsewhere perhaps through a use case scenario. One might have to accept the context of the situation and state that the descriptions match the Coverage required by the context. If one were considering reusing use case descriptions this might be a drawback in that events 3 and 4 could be considered superfluous to the general task of printing applications.

2) Cogent. The Text Order of the description follows a logical path except exceptional flow e6.1, which questions whether this event should occur here or at 10. Since this is flagged, this is considered a reasonable answer at this early analysis stage but it should be resolved before being signed off. The description is an end-to-end transaction (Dependency) though exceptional flows e9.2, e9.3 and e10.1 need resolution. The description provides a Rational Answer though the exceptional flows are flagged for further study. These need to be resolved to avoid potential software or process failures (Alexander 2002b). Again, these are subjective considerations. For instance, are events 3 and 4 necessary in the logical order? Do events 7 and 8 confuse the answer because these events are not dependent upon the others? It is clear that the success of event 10 is dependent upon there being sufficient paper in trays and it can be argued that Customer satisfaction is dependent upon being sent the right number of pages to be able to complete the application (event 7). Thus events 7 and 8 are possibly more important to the success of the description than their 'Optional' state suggests. There is sufficient detail in the description to provide a rational answer for the context but there is the potential playing down of parts of the process important to its success.

3) Coherent. There is local coherence throughout the main flow because Print Room Staff member has been named in all events except 10. This refers to the Printer and documents. The Printer coheres to the 'print run' noun phrase of event 9. The exceptional flow e10.1 does not cohere to the exceptional flow e10. This exception is unresolved and its resolution might make it more coherent.

4) Consistent Abstraction. Most of the description is at the interaction level though there are some events that relate to the problem domain, for instance event 8. The mix of abstractions appears reasonable because the description is describing the actions of the actor when conducting a print run. This suggests, as do the results of chapter 7, that mixing abstraction levels is necessary to convey full meaning in the description. There is therefore some subjectivity in determining what level of abstraction one should show. Again, the situation should determine the abstraction required.

5) Consistent Structure. There are no Variations in the main flow, though the 'Optional' events might be considered alternatives. They are not quite the same as alternatives because they are not substitutes for other events. They could have been written in separate scenarios but this would mean repeating the description four times for one different event. There are no alternative flows listed though there are exceptional flows. There were no alternatives suggested – the process is quite constrained. There are no Sequence (numbering of events) mistakes. There is no subjectivity to this C.

6) Consistent Grammar. This relates primarily to the CP Style Rules 2 and 3 (see section 5.6.2). Grammar is consistent in accordance with these Rules. Each event uses present tense and there are no negatives, adjectives, adverbs or pronouns. The exception e10.1 shows variation to the Rules but this is a question and flagged for further study. Whether events such as e9.3, which ends with the How question should be counted or not as grammatically correct is a subjective judgement made by the use case writer. In this case, the author suggests that these not be counted as incorrect since they flag that further study has to be done to resolve these issues.

7) Consideration of Alternatives. The Separate section considers exceptions only. These are reasonably well described though there are four exceptions that are flagged as questionable or need further work (Viability). They are correctly Numbered. In general, the descriptions (Appendix F3) are lacking in detailed alternative and exceptional flows. This is a weakness but time constraints forced a rationalisation of the descriptions. That is, there was not enough time to elaborate upon all the potential alternative flows.

This example description is shown because it is typical of the other descriptions in terms of how it applies the CP Rules and the 7 C's. There are occasions when the CP Rules are insufficient to describe the event. In these cases, alternative structures are used that enable a better description of an event than the CP Rules would allow. When considering the 7 C's there is a degree of subjectivity that is determined by the perspective of the use case writer (in this case the author) and the context one finds oneself in. Each use case

description was assessed with the CP Rules and 7 C's. The level of detail of analysis described above can be applied to all other use case descriptions in this study.

8.5 The Design

The design is in two parts: external and internal. The external design (section 8.5.1) relates to use case descriptions. The internal design (section 8.5.2) explores the Question Set as described in section 7.6.1. See Appendix F3 for all other accompanying documentation.

8.5.1 External Design

An example interface ('i') description is presented. This is selected because it is typical of the detail of the interface descriptions in this study, though it is rather longer than most. Note that though the first two events somewhat contradict what is said in Appendix A4 (Use Case Title, Actors and Context), it is important in this context to consider precisely what the Customer does.

iUC1: Apply for an Individual Trading Account

Actors: Customer

Context: The Customer wants to open a trading account on the Company X website because the Customer wants to start trading in stocks and shares on the stock market.

Pre-conditions: The Customer logs on to the Company X web site; the Company X website is accessible.

Main Flow of Events

1. The Customer types www.Company X.com into the address bar of the web browser.

2. The Company X website appears on the screen.
3. The Customer selects "Apply Now".
4. The website takes the Customer to the Apply Screen.
5. The Customer reads the guide on how to apply.
6. The Customer sees the choice of Accounts (details: Trading Account or ISA).
7. The Customer selects "Trading Account"
8. The website takes the Customer to the Application Form (details: page 1 of 3)
9. The Customer selects the Country of Residence.
10. The Customer selects "Individual Trading Account".
11. The website asks, "Where possible do you prefer to deal in certificates?"
12. The Customer ticks the option.
13. The website asks how the Customer wishes to have interest and dividend paid.
14. The Customer selects "Cash".
15. The website asks which currency the Customer wishes any income paid in.
16. The Customer selects "GB Pound".
17. The website asks if the Customer would like to deal in UK registered warrants.
18. The Customer ignores the option.
19. The website asks if the Customer wants stock to be registered at a different address.
20. The Customer ignores the option.
21. The website informs Customer that a "contract note" will be sent on every trade.
22. The website asks the Customer if the Customer requires a contract note to be sent to a third party.
23. The Customer ignores the option.
24. Customer selects "continue".
25. The website takes the Customer to the second page of the Application form.
26. Website presents the Customer with "About the Primary Holder" screen.
27. Customer completes details.
28. Customer selects "Continue".
29. The website shows the completed page 2.
30. Customer selects "Continue".
31. The website goes to page 3 of the Application form.

32. The website presents the Customer with the Customer's details on the application.

33. The Customer selects "Apply".

34. The website presents the Customer with "Application" screen.

35. The Customer sees "Successful Submission" message.

36. The website displays the "Customer Number".

37. The Customer enters a numeric PIN twice.

38. The website presents the Customer with "Change Dealing Password".

39. The Customer enters a password.

40. The Customer clicks "Change" button.

41. The website presents the Customer with a Welcome Information screen.

Alternative Flows of Events

a20. Customer selects Register Stock at Different Address.

a23. Customer selects Send Contract Note to Third Party.

a33. Customer selects "Back".

a33.1 Website displays page 2 of application form.

a33.2 Customer makes changes to application.

a33.3 Customer clicks "Continue" (use case returns to event 33 in the main flow).

a33. Customer selects Transfer Stock

Post-conditions:

The Customer has successfully opened a Trading Account with Company X. The system has generated a unique Customer Number (event 36).

Note: Need to consider this exceptional flow in detail.

Exceptional Flow of events

e34. The website presents the Customer with "Failure" screen.

e34.1 The website informs the Customer of alternative application procedures.

<The Use Case Ends>

This description is long but this is not unusual in industry (Leibundgut 2002), though it somewhat contradicts Cockburn's suggested 10 line maximum (2001). This is because the author wanted to know what the Customer actually does at the interface; that is, the describing of the interaction between the actor and the system, which is a key purpose of the use case description (OMG 2001). It would have been easy to write a short description such as,

1. Customer selects Apply for Individual Trading Account.
2. Customer completes details.
3. System presents unique trading number.

But this would have been almost valueless for the task of developing a design for the application because the author would have had to elicit the details contained in the full description in any case.

8.5.1.1 CP Rule Evaluation

The description employs the CP Rules where possible. However, there are events that contain long phrases direct from the website and include tenses such as future passive (e.g. event 21). Most descriptions encountered in texts do not contain such detail for fear of slipping into interface design (Antón et al. 2001). In this case, though, the product already exists. There are few design documents. One of the author's tasks was to produce a design (section 8.1.4) and it appeared reasonable to take such a detailed approach. Of course, linking the activities at the interface with the internal design is not straightforward.

A number of events are underlined. This is CP Style 7, which shows links to include or extend use cases. This appeared to work well though sometimes it was not possible to phrase the event to match the use case name whilst adhering to the CP Structure Rules. For instance, event 27 has a link to this use case, 'Customer completes details'. The actual name of this use case is Complete Customer Details (iUC7). However, this is not a CP Structure. Though the electronic document is hyperlinked and iUC7 notes its origin and where to return to in the calling use case (see Appendix F3), there is the potential for confusion over use case names. The use case name is typically a verb-noun construction (see Appendix A4). The CP Structure Rules appear not to be flexible enough to always write the use case name in its correct grammar format.

8.5.1.2 The 7 C's Validation

The following describes a 7 C's check on the description iUC1: Apply for an Individual Trading Account. This is less documented than the Print Room use case description analysis in section 8.4.2.1 to save space. The key points of interest are listed only.

1) Coverage. There are no Span problems except in the exceptional flow of events where there is a reference to examine the 'Failure' Screen further because the exact details of the Failure Screen need to be documented so that this Customer might be able to apply by another route. One can argue that there is too much detail (Scope) because of the clicking and ticking of options. However, all the details are appropriate to the context of this description (Alexander 2002a) because they are necessary to be able to determine the exact type of application the Customer has applied for and what potential options there are.

2) Cogent. The Text Order of the description follows a logical path – the event order matches that of the web interface at the time of the study. Dependencies are fine – the use case describes an end-to-end transaction. The only doubt is the exceptional flow but this is flagged for further study. The description provides a Rational Answer since it is taken from the website.

3) Coherent. This is hard to achieve because the website dictates the order of events and the language used. However, there is local and global coherence throughout.

4) Consistent Abstraction. The entire description is at the interface specification level and as such can be considered consistent.

5) Consistent Structure. There are no Variations in the main flow, though the ability of the actor to select different options is possible. There are Alternative Flows missing, e.g. event 12: although appearing relatively insignificant, the impact on how the Customer and Company X conduct business is important. This should thus be documented. Sequence (numbering of events) is correct.

6) Consistent Grammar. Grammar is not always consistent. There are modal verbs (e.g. event 13) and future passive tense (e.g. event 21). The author was constrained to use these structures in an attempt to represent the information at the interface as closely as possible.

7) Consideration of Alternatives. Though there is a Separate section some alternatives are missing which ought to be considered for a more complete description. For example, events 12, 14, 16 and 18 probably require further explicit consideration of alternatives because these determine whether the Customer wants to be paid in cash or dividends, to deal in certificates etc. and are very important to the Customer's account set up. The alternatives are Viable and correctly Numbered.

8.5.1.3 Proposition Evaluation

As stated in section 8.2, much of the evaluation undertaken has been by the author alone. Thus, the first proposition in section 8.2.2 is difficult to evaluate.

1. The CP Rules are of value when writing use case descriptions in an industrial setting.

Though this might sound more like conviction, the author applied the Rules as much as possible. There are, though, occasions where the CP Rules are insufficient. There are only two allowable CP Structures. To provide a long list of structures such as the CREWS Content Guidelines (section 2.4.1.1.2) was not considered worthwhile because of the infrequency of their use (see section 4.4). This implies that the CP Rules are not a complete set. However, it would be difficult to provide a complete set of all grammar structures for all situations and it would be rather unwieldy; see, for example, the large number of different structures identified in the use case grammar survey (section 4.3). The two Structures provided are considered sufficient for the majority of situations. There are 277 events (or sentences / lines) in the main flows of use case descriptions in this study, of which 35 do not conform to the CP Rules. That is, 13% of structures are non-CP. Table 8-1 shows the most common occurrences of these.

Source	Structure	Count
pUC2 (x2), iUC9, iUC10	'Optional'	4
iUC6 (x2), iUC8	Subject verb negative infinitive object	3
bUC4	Subject verb object verb infinitive object	3

Table 8-1. Recurring non-CP Structures in Use Case Descriptions

The use of 'Optional' occurred four times. This structure occurs in the process-oriented print room use case prUC2 (section 8.4.1) and in the interface use cases iUC9 and iUC10 (Appendix F3). It is therefore not restricted to one type of use case and might be worth introducing to the CP Rules. However, four occurrences out of 277 is not a particularly high count.

The second structure listed in table 8-1 is 'subject verb negative infinitive object'. An example of its use is from iUC6: Register Customer, event 18.

Customer selects to not receive junk mail
<subject> <verb> <negative infinitive> <object>

This structure occurred three times in two interface descriptions. It represents the deselection (*unticking* a pre-selected check box) of, in this case, receiving junk mail. This de-selection (as opposed to actively choosing to receive junk mail) appears to be more and more typical on modern websites. This structure seems suited to detailed interface interaction and is too specific to include in the CP Rules. However, it might be interesting to conduct a study on a number of websites to identify typical grammar structures in interface descriptions so as to develop a set of Rules specific to this abstraction.

The third structure in table 8-1 occurred three times in bUC4: Generate Pack Code. Event 10 provides an example,

The Business Layer determines the Customer wants to transfer stock.

<subject> <verb> <object> <verb><infinitive><object>

This structure is local to this description alone and should not be considered for the CP Rules. There are a number of other structures that occur once or twice in the descriptions. This suggests that if one needs to occasionally use another structure, then use it.

The second proposition states

> 2. The 7 C's of Communicability are a useful technique for assessing the internal structure of the use case descriptions.

It is the author's opinion that this is the case because there were a number of problems identified with the use case descriptions, examples of which are described in sections 8.4.2.1 and 8.5.1.2. The 7 C's proved a reasonable checking mechanism but because of their subjectivity it might be a matter of opinion as to how well structured a description is. There is also the issue of how much time one is prepared to give to assess the internal structure of the descriptions manually. Perhaps an automated version would be useful. However, since there is subjectivity to the heuristics it is unclear how this might be done.

8.5.1.4 Length of Descriptions

The above description (iUC1, section 8.5.1) might be considered long in Cockburn's terms. He suggests that descriptions have no need to be more than 10 lines in length (2001). The histogram in figure 8-1 shows the lengths of the main flows of events for use case descriptions in this study. The majority of descriptions (17 out of 26) are less than 10 lines in length. Two descriptions are exactly 10 lines. The longer descriptions tend to be the interface descriptions that portray user actions at the interface in fine detail. There are, though, 7 descriptions that are longer than Cockburn's suggestion. It seems prudent to write descriptions that describe the problem at hand. For instance, Leibundgut (2002) states that descriptions in his company tend to be three sides in length. Each description is no doubt more than 10 sentences. Nevertheless, the mean number of events in the main flows of the descriptions in this study is approximately 11. The median is much lower at only 5.

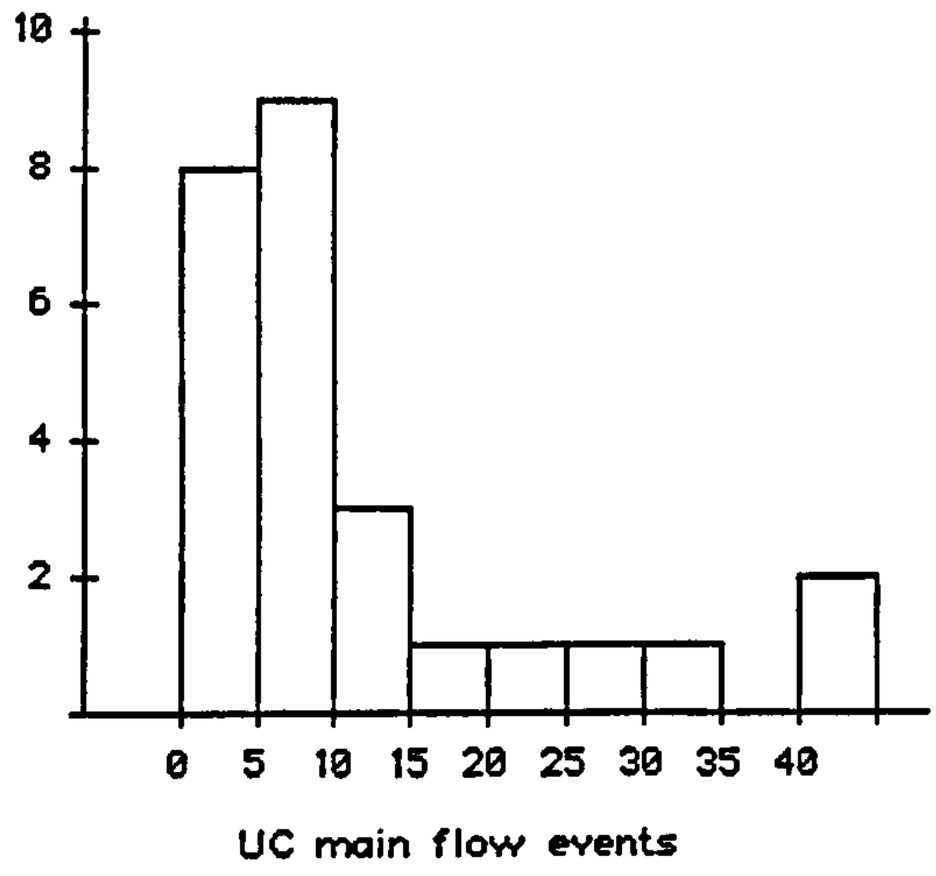


Figure 8-1. Lengths of Main Flows of Events for Use Case Descriptions

On the whole, the descriptions are close to Cockburn's 10 lines. This might, therefore, be considered a reasonable guideline. However, one should be aware that this guideline cannot always be adhered to. Figure 8-1 emphasizes that the use case description shown in section 8.5.1 is something of an outlier, along with iUC2, which has 43 lines. The nearest to these is iUC8 with 30 lines (see Appendix F3).

8.5.1.5 Use Case Feedback

There was no feedback with regards the CP Rules and the 7 C's because time constraints on the project did not allow the IT team to develop any use cases (at least, none that were formally documented or shown to the author). However, there was positive feedback to the descriptions produced by the author and this is described here because it offers insight into ideal and actual industrial practice within this company. Appendix F2.4 gives complete feedback.

This comment comes from feedback after the author gave a presentation of his work.

"The descriptions are very important for the finer details and the step-by-step operations. This is very useful for both IT and QA." - Company co-Founder and IT Vice-President.

QA refers to the Quality Assurance department. This suggests that use case descriptions could have a potentially important role to play in the IT and QA departments at Company X. When interviewed at the end of the study, the QA Manager reiterated the Vice-President's comment:

"[The use case descriptions are] definitely a good thing for QA and testing as a basis for test development and user acceptance tests."

When asked if there were any weaknesses with use case descriptions, he stated there were none. The IT Development Manager also considered the descriptions as a useful testing tool when providing feedback after the final presentation made by the author.

"Use cases [are] good for testing and for reshaping the application process at the interface. The online application process is too long and no one has ever produced this kind of document to see the exact steps required to complete an

application. The trouble is time constraints on getting this done – we don't have time to do all this" – IT Development Manager.

Here is one major problem. Time. Despite the recognition from members of the IT department of the role use case descriptions can play, firstly, to document the processes and, secondly, to use those descriptions to test and reshape applications, there is not enough time to do it. The IT Development Manager is actually at the 'coal face' of software construction and product delivery in the company. Although the QA Manager and the Vice-President are well aware of how development is done, they were not building the actual software at the time of the study and had a perhaps more idealistic view. The IT Development Manager delivered the practical view that although the descriptions would be a very good tool, they would not have time to write them. Indeed, in the structured interview the QA Manager recognised that much of the author's approach might not be taken up because,

"the work culture needs to change a bit [before] this can occur."

This appears to be a significant barrier to introducing the tools used by the author to Company X. The recognition of their usefulness was there, but the work culture of the IT department would make take up difficult.

A further, interesting comment came from the IT Development Manager in the structured interview at the end of the study:

"The descriptions are good for test scripts. They capture lots of detail. [And are] actually similar to how a servlet is written! This is dangerous because it might encourage jumping from a use case straight into code. It's also frightening to see that there's loads to do! Descriptions are good for showing what's going on in the system."

The resemblance to how servlets are written is interesting. In terms of internal design (section 8.5.2), this might be useful because the description acts as a guide for how the system is to work. But as stated, developers might jump from the use case into code without consideration of design.

The benefit to Company X of the author's descriptions was the realisation that the Customer application process is too long. Unfortunately, there are legal constraints (as well as time) and the steps probably have to remain.

This section has described elements of the external design. There is no documentation to compare this work against. As the IT Development Manager stated,

"...no one has ever produced this kind of document to see the exact steps required to complete an application."

The rest of the case study considers how use case descriptions can be exploited to discover classes and other elements of design by application of the Use Case Question Set.

8.5.2 Internal Design

The CP Rules and 7 C's were used in constructing the internal design descriptions. However, since the use of the CP Rules and 7 C's has been discussed above, the focus of the chapter from here is on interrogating use case descriptions. This draws upon the full Question Set as described in section 7.6.1.

8.5.2.1 Interrogating Use Case Descriptions

One of the most difficult aspects of the use case approach is the transition from descriptions to classes (section 2.8). Chapter 7 describes an experiment that tests whether questioning use case descriptions yields information to enable easier passage towards

design. It appears that there is scope for a general set of questions to interrogate the descriptions. This section explores the role of questioning use case descriptions to help identify elements of design.

8.5.2.2 Question Set Process

Each event of the use case descriptions was interrogated with the Question Set. This is a very meticulous approach but is considered necessary in assigning operations to classes (Insfran et al. 2002). It is time consuming and many events yielded little (especially for the interface descriptions). The descriptions often took about half a day to interrogate and always required revision. Though this was time consuming it did allow the author to think considerably about the use case description and its subsequent design. In many cases much information was gleaned. There follows examples from both ends of the spectrum.

8.5.2.2.1 Example Use Case Description Fragment 1

This example fragment comes from a description above:

iUC1: Applying for Individual Trading Account

...

35. The Customer sees "Successful Submission" message.
36. The website displays the "Customer Number".
37. The Customer enters a numeric PIN twice.

...

The interrogation is applied to event 36.

Event 36. The website displays the "Customer Number"

Pre-condition: Customer number generated how? Does this mean that an account has been set up in the system? Stored on database or in run-time memory or both? Is a Pack Code generated here? Where is this stored?

Post-condition: Customer reads number

Interface: text

System: Number generated. Must be drawn from a list of Customer numbers to avoid repetition. Pack Code generated as well – is the Pack Code reflected in the Customer Number? Should it be? It needs to be on the Customer Record somewhere.

Actors: Customer, subsystem actor e.g. customer number list generator?

Classes: Customer (rename as Primary Holder? Entity / GUI) – stores application details of the Customer; Customer Number (database, control – to put Customer number to the GUI).

The pre- and post-conditions are the Dependencies Set. The pre-condition is straightforward: it is important to know where the Customer number is generated. This is also the responsibility of the system and is discussed further under that heading. The class diagrams for the interface use case descriptions are in Appendix F3, figure 8.1.1.

8.5.2.2.2 Example Use Case Description Fragment 2

This comes from an internal design use case: bUC1: Determine Type of Application. The event examined is line 14.

...

13. The Business Layer generates a pack code for the application.

14. The Business Layer increments the pack code print list.

...

Event 14 The Business Layer increments the pack code print list.

Pre-condition: Pack Code generated

Post-condition: Pack code stored in database.

Interface: Link class(es) to Oracle Database.

System: Must check that pack code and details sent to correct file in the database or added to correct linked list Pack List and Pack Type

Actors: Database

Classes: PackCoder <<control>>, Pack Code <<entity>> ?? (Or is the Pack Code a list of attributes drawn from objects that at run-time get bundled into an object of its own?), Pack Code <<entity>> - a linked list that would deal with duplication etc. at run-time before storing in the database.

Figure 8-2 depicts part of a class diagram, showing where event 14 has an impact upon the design (the complete diagram is figure 8.2.1 Appendix F3). The key part of this figure is the identification of the PackCoder class. This enables new PackCode classes to be added to the PackType linked list in run-time memory. The PackCode links with Database classes (these are control classes that prepare the Customer's application details to be transferred into the Oracle Database).

There are options described in the Class Set on how the Pack Codes are generated and stored. It is reasonable to assume that the state of the attributes of each application will decide the pack code. These are then held in a linked list. The System is responsible for correctly filing the generated Pack Code object.

Occasionally, Customers double click the interface Apply button. This generates duplicate applications in Front Office. The solution to removing duplicates, as suggested by the Project F IT team is to 'dedupe' (remove duplicate files) in the database via the Maintenance Screen (figure 8-3). This runs the risk of clogging up run-time memory and storage space with unnecessary applications (a small risk, granted) but also leaves the deduping to human judgement. In the author's design solution, the risk of failing to dedupe is automatically removed. When discussing this with the Systems Analyst, he pointed out:

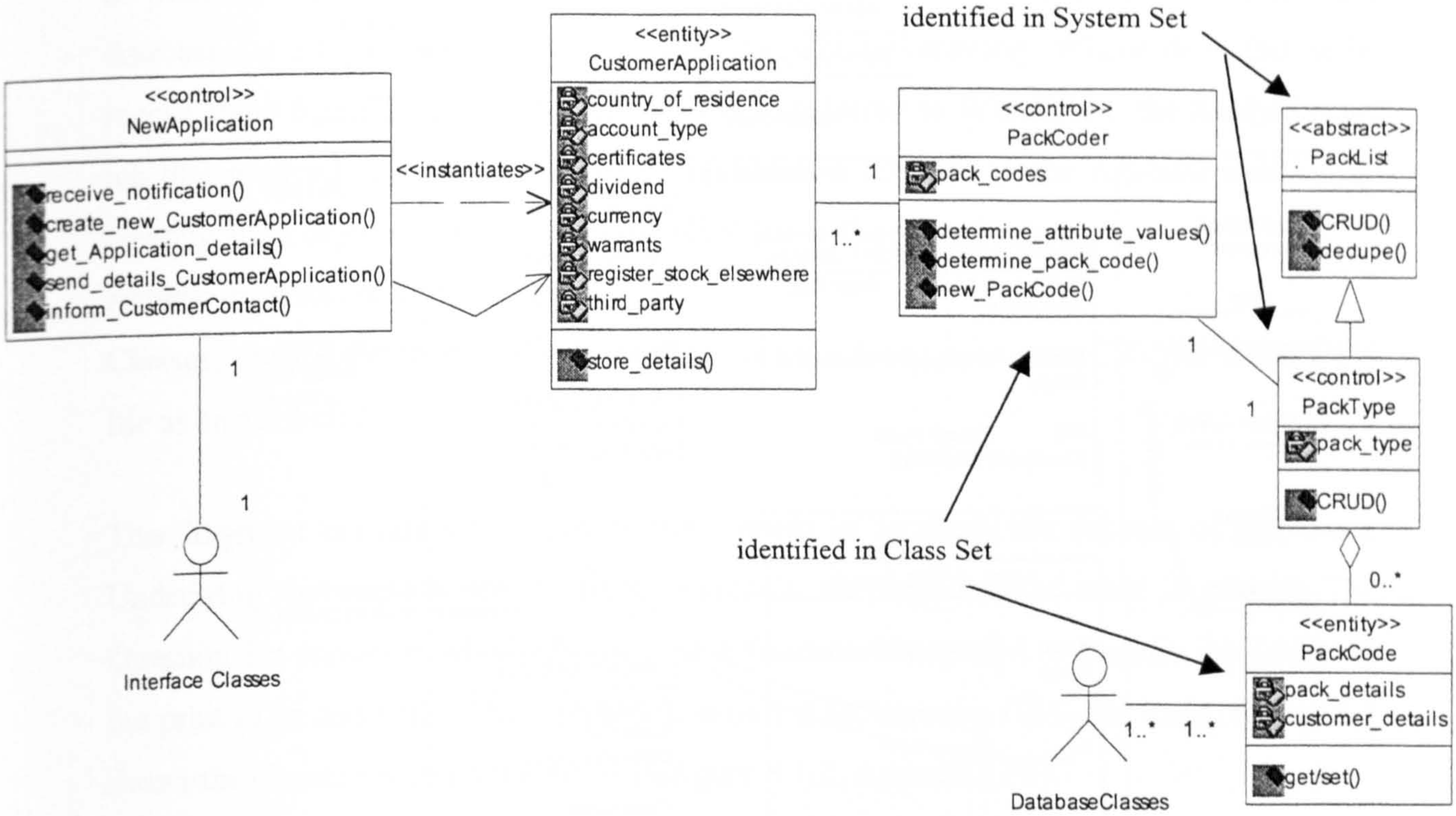


Figure 8-2. (Part of) Business Model Class Diagram

"Changes to applications come in over the phone so details get altered when they are already in the database (for example, if a Customer changes their address). Sometimes a new record is created and the old one is unaltered. It's simpler to remove duplicates only from the database" (Appendix F2.3).

It might be even wiser to dedupe both at run-time *and* when maintaining or preparing to print customer applications. This is reflected in the class diagram fragment above (figure 8-2) and in the use case description iUC9: Maintain Pack Codes event 4:

4. (Optional) The Database Maintenance Staff dedupes the records.

See figure 8.1.2 in Appendix F3 for the related class diagram.

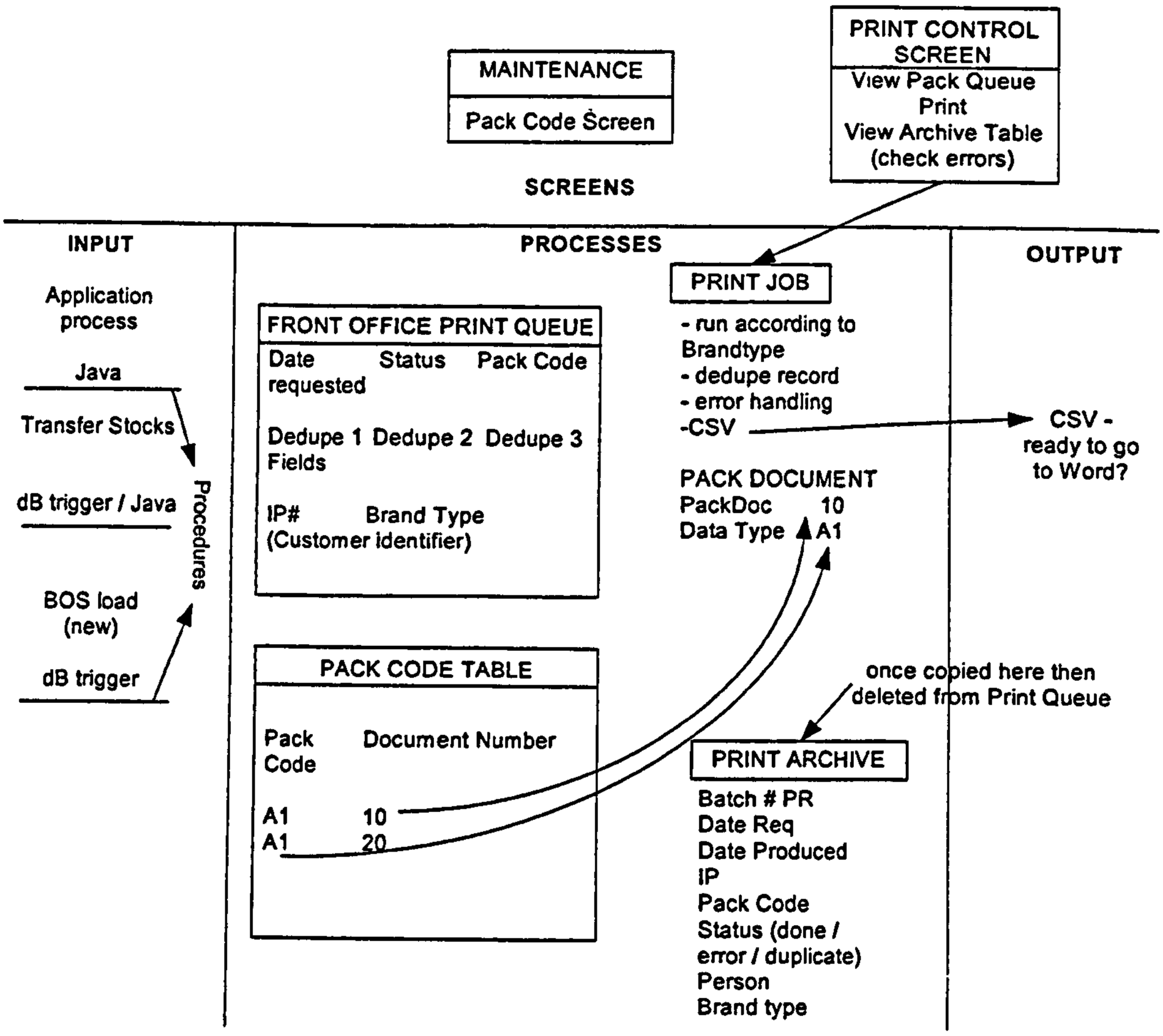


Figure 8-3. Internal design as suggested by Project F IT team

8.5.2.2.3 Example Use Case Description Fragment 3

The third example iUC10: Print Packs, event 9.

Event 9: Print Room Staff selects Print.

Pre-conditions: Mail merge complete

Post-conditions: Print selected

Interface: Word interfaces with the Printer automatically. However, Word's printer driver might need to be manipulated to allow automated printer selection and collation of printed materials.

System: Batch print job (Word file) and send to Printer. The Front Office needs a record of what has been printed for its audit trail. This means that records must be stored in the database for all pack types and jobs that are sent for printing. Where does this audit record come from? Is it the CSV file that is transferred to Word? Is it the batched print job that is saved? It might be better to just have a text file in the database rather than creating audit objects. I suspect that the CSV file is the place to start.

Actors: Print Room Staff, Word

Classes: <<database>> Print Pack should contain a function to save CSV for audit; CSV file as an attribute?

This fragment considers the role of the System in securing the success of the event. Underlying the event is the desire to maintain an audit trail of what is printed. The Question Set considers whether a class or a function is required to transfer the details of the print to an audit file. The simplest solution is to store the CSV file itself. Figure 8-4 shows the classes involved (taken from figure 8.1.2, Appendix F3).

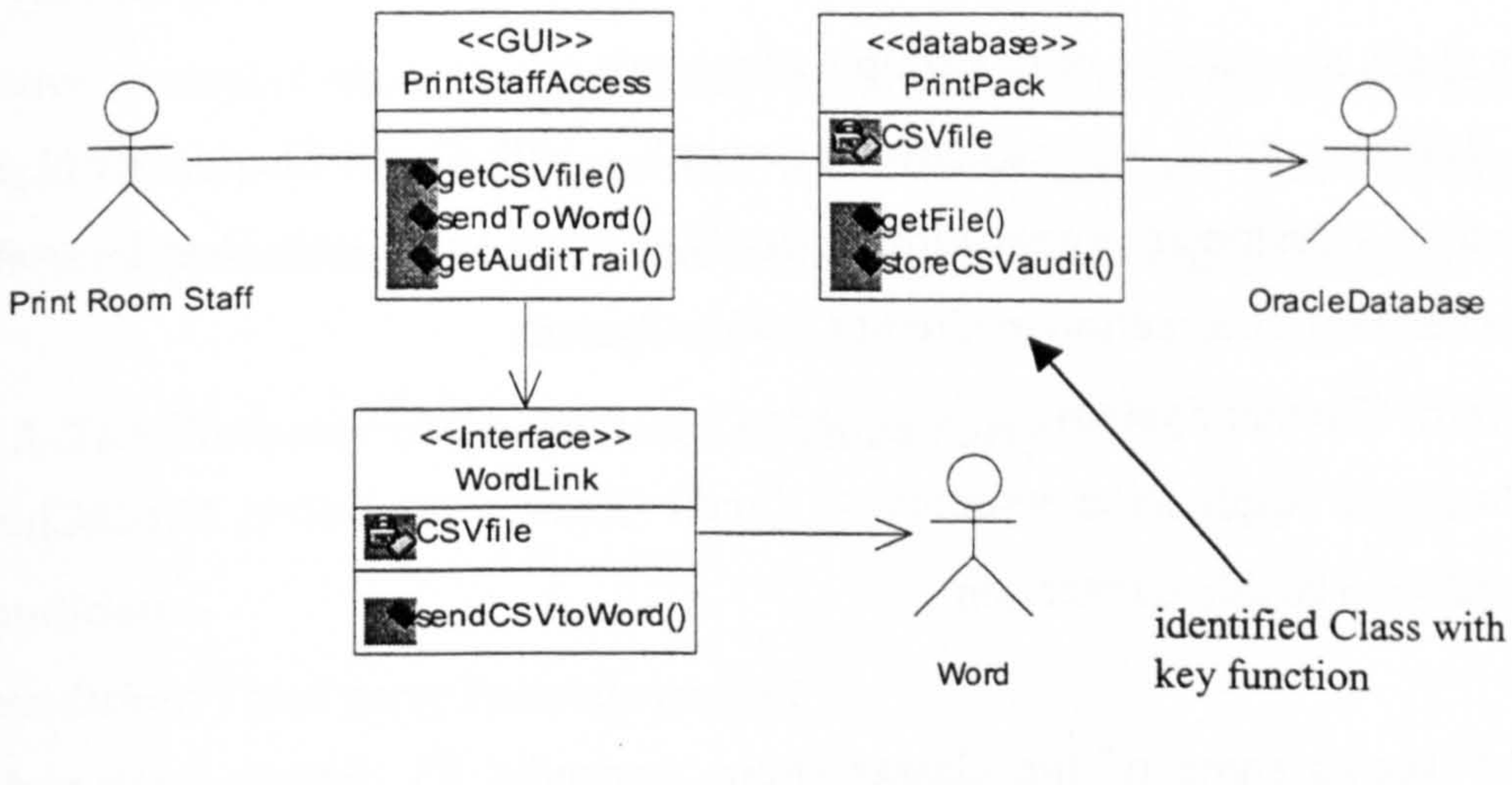


Figure 8-4. Printing and Auditing

The PrintStaffAccess class represents a GUI component that allows the retrieval of CSV files; these are then sent to Word. The Print Room Staff has the option to create an audit trail through the operation *storeCSVaudit()* in the PrintPack class. It might also be that a

copy of the CSV file is made and stored in the database, removing the option to create an audit file.

8.5.2.2.4 Example Use Case Description Fragment 4

The next example is event 2 from bUC2: Check Credit Status.

Event 2: The Business Layer sends the Customer's bank details to the Credit Checker System.

Pre-Condition: Connection made with Credit Checker System. This should involve an interface connector object that might need some middleware. A CreditCheck <<control>> object has retrieved the necessary information to send Customer bank details (from CustomerBankDetails <<entity>>) and Credit Checker System in state to receive them.

Post-Condition: Customer bank details sent; received by Credit Checker.

Interface: There should be a dedicated connection to the Credit Checking Agency system since this check should take only a couple of seconds.

System: Business model locates Customer records – with Credit Check control object – from memory (Customer Application <<entity>> class and Bank Details <<entity>> class) and has sent information to Credit Checker System.

Actor: Credit Checker System

Class: Customer Application <<entity>>, Credit Check <<control>>, CreditCheckLink <<interface>> to enable connection.

Figure 8-5 shows some of the classes (from Appendix F3, figures 8.1.1 and 8.2.1) involved in this event. The pre-condition is of interest because there must be a connection to the Credit Checker System. The Interface Set also suggests this. For speed of throughput this ought to be a dedicated line. A CreditCheckLink class is necessary to begin the connection process to the CreditCheckerSystem.

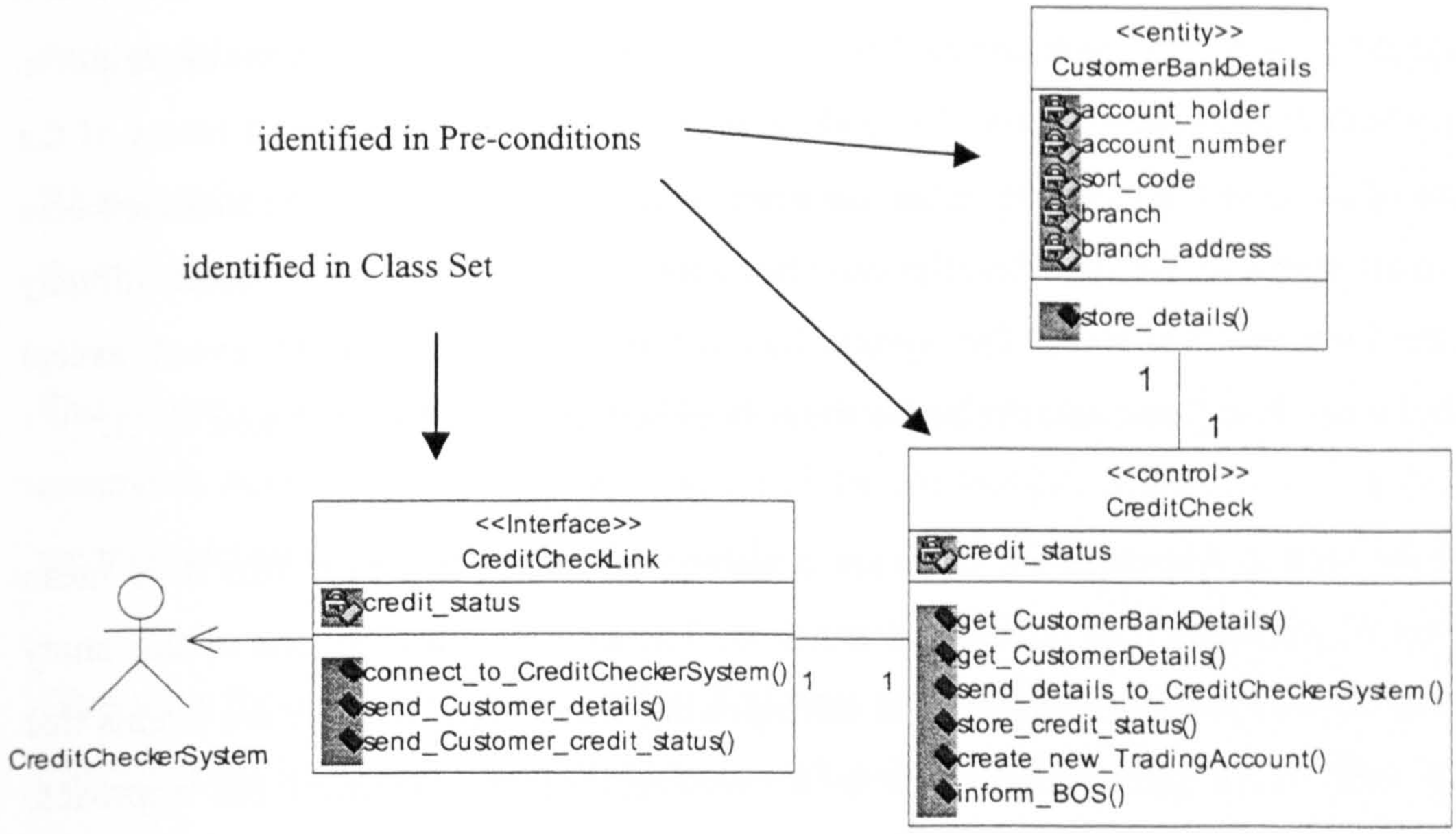


Figure 8-5. (Part of) Class Model for Project F

8.5.2.2.5 Example Use Case Description Fragment 5

The above examples show that the Question Set yields interesting design information. This final example (<<extend>> use case iUC5: Send Contract Notes to Third Party, event 3 (see Appendices F3 and F4)) shows that very little can also be yielded.

Event 3. The Customer enters Postcode of Third Party.

Pre-condition: -

Post-condition: Third party Postcode entered

Interface: edit box

System: enable edit box

Actor: Customer

Classes: Primary Holder / Secondary Holder, Third Party

There is no pre-condition because the Third Party details can be completed in any order. Arrival at this screen is predetermined by the pre-condition of event 1 in the description. The post-condition states that the Postcode is entered. This makes sense in terms of the success of the event. The only Interface concern is a GUI component. It is included here (and in all interface use case descriptions) because the descriptions were taken directly from the Company X website. The System has to guarantee nothing in this event, except enable the edit box. The classes relate to those depicted in Appendix F3, figure 8.1.1.

As can be seen in Appendix F4, there are a number of events like this. This does mean that a lot of work might be considered fruitless. The Interface descriptions in this study are taken directly from the website. It is unsurprising, therefore, that there are events that do not yield much information. This is a weakness of the Question Set approach. Predicting which events might not yield much useful information, however, might not be straightforward. For example, in the interface description iUC1 (section 8.5.1) there are number of tick / select options. These might not appear to reveal much information at face value but actually they determine the exact details of the Customer's account and are vital to its successful set up.

8.6 Question Set Feedback

Full feedback notes are in Appendix F2.4; this section only provides a summary. The following two comments were made during presentations and the rest from structured interviews conducted at the end of the study.

"The Question Set raises questions about the design and important issues of complexity such as how we manage the uniqueness of the Customer Number and is there sufficient check to avoid a duplicate record?" - IT Development Manager.

This refers to the first two examples (sections 8.5.2.2.1 and 8.5.2.2.2). Although the Customer Number is uniquely generated, it appears there are ways in which it can be

duplicated. Is deduping from the database alone the best way to perform this function? Perhaps this could also be done at run-time.

"Questioning the use cases is very detailed and good to help in the design" - Company co-Founder and IT Vice-President.

This is a general comment made in feedback but is nonetheless gratifying. The following comments were made in interviews – some of the comments are general and brief, others are more detailed. Interviewees were asked,

Give your thoughts on the Question Set that are used to interrogate the use case descriptions, including weaknesses as well as strengths.

On Dependencies Question Set (pre- and post-conditions):

"Good points: Pre- and post- very handy to define where you start and end and also helps you bound your system. So programming is constrained by the pre- and post- conditions." – Software Developer.

This comment concerns implementation although the Dependencies were initially conceived as a specification validation mechanism. It is interesting that there is direct reference to code, as suggested by the IT Development Manager (section 8.5.1.5).

"Bad points: Too precise at times. This might impose stricter conditions than really required i.e. we might have a "very bad scenario" as opposed to the "happy day scenario". It all depends on the robustness required for the system being built. Overly constrained at times. There's a risk of introducing more errors by dealing with lots of constraints that might not be necessary. For instance, there might be an overly constrained requirement that deals with system failure risks that might occur two times a year. To code these risks out of the system means a vast amount of programming effort and lots of lines of code. This introduces the

risk of introducing many more errors that occur more frequently than the frequency of occurrence of the original risk." – Software Developer.

The interviewee later noted that if a system is safety critical, it would be worth the extra effort in coding out the risks.

On Interfaces Question Set (the comments are also applicable to whole Question Set technique):

"Over-analysis could lead to performance problems [in the IT department, not with the speed of the application itself]." – Software Developer.

This is a comment made by many members of the IT department. They do not have the time to do all this detailed design work.

On Actors Question Set:

"These questions give good perspective to look at interactions to other systems. It's hard to anthropomorphise and imagine yourself to be another system." – Software Developer.

This reflects on comments made by the author on typical approaches to discovering classes in object-oriented analysis.

On System Question Set:

"From a business perspective people don't care about this. But for development it is important. It's vital to know what the interfaces are and how to connect to the right systems. Technically this is valid." – Software Developer.

The Question Set was never meant to be part of the business perspective but this is interesting.

General comments from interviews:

"[I] use these sort of questions. They are a good thing to do. If you read a load of use cases you would ask these kind of questions. I would use the questions in business analysis. Timing constraints and interfaces would be asked mainly subconsciously.

"There's no downside to questioning. In the context of discussion with users these are very useful, especially in building a rapour with them and getting the necessary information about the system. The same approach applies to the use cases themselves." – Systems Analyst.

Interestingly, the Systems Analyst and Software Developer see some of the Question Set from a business viewpoint. This would be worth exploring further. The IT Quality Assurance Manager provides a viewpoint from the testing perspective:

"[Question Set is an] Essential activity. If you go down the road of doing some of this before coding, then coding would be a lot easier and quicker, instead of the usual approach of hacking prototypes. Changing the working mentality would be a drawback for their introduction. The approach is good and thorough and it reflects the depth of detail needed from a testing perspective."

Again, the work culture of the company would make the take up of the Question Set difficult. Combined with time constraints, this would be doubly difficult, although the stakeholders recognise the potential of the tool.

8.6.1 Question Set Conclusions

In terms of Company X's work culture, it is unlikely that they would use the Question Set so rigorously, if at all. However, the technique was well received. It is clear that it has a future but also that it needs some refinement. The future success of the approach probably depends on four aspects:

- The more detailed the description the less useful the Question Set.

The interface use cases are highly detailed. The primary reason for this is that the descriptions were taken directly from the website. This leads to too many events that relate filling in text boxes and ticking check boxes. The perceived wisdom is that this kind of information should not be in descriptions (Cockburn 2001). But in this case study there were no stated requirements, and little analysis and design to originally work from, so this seemed the best approach. More fundamentally,

- the Question Set should be focussed on the important events in the descriptions.

Determining the key events in descriptions is probably not as easy as one might imagine. In terms of the interface descriptions, it is easier to tell what is trivial and what is not. But with more abstract descriptions, such as the Underlying Business Model and Database level use cases in this study, many events describe a significant action for the system or the actor. However, Insfran et al. (2002) suggest that each sentence in each description be examined.

- The Question Set should have a more data-centric focus.

Use cases are process and function-oriented. The Question Set should explicitly draw out data and attribute requirements to suit the data-driven Company X product. Indeed,

- The Question Set ought to consider domain specific details.

A failure of the Set in this study is its ignorance of data within descriptions. It is conjectured that with the aid of problem frames (Jackson 2001), the Question Set can be more explicitly focussed to types of application, however, this is work for elsewhere.

The case study Proposition 3 (section 8.2.2) states

> 3. The Use Case Question Set helps identify classes and other design elements from use case descriptions.

From feedback, it is clear that the Question Set is a useful technique. These results should not be over-emphasised, however, since this is only one study. The Question Set as-is carries the risk of over-analysis; if the technique is streamlined, as suggested, it might prove more useful. This suggests that further work needs to be undertaken to assess how the Question Set can be effectively refined as proposed.

8.7 Qualitative Analysis of Presentation and Interview Feedback

This section briefly considers some of the issues raised in this chapter, namely that Company X do not have time to conduct such detailed analysis and design and that the working culture of the IT department is also a major hinderance to doing this work. Presentation feedback and the four structured interviews provide the source of data (Appendix F2.4). The author conducted a qualitative data analysis (Miles and Huberman 1994) on the feedback to investigate two categories that appeared to be an issue for the IT department of Company X:

- Time – time constraints do not allow for such work to take place,
- Culture – the work culture of the IT department hinders such work.

These have been identified as potential barriers to the take up of the techniques suggested by the author (section 8.5.1.5). In coding the text for these (using the qualitative data analysis tool, QSR Nud*ist v.4), a number of other issues appeared.

- Business – Company X's Business and Marketing departments are blamed for failures by IT; this was found to divide into four sub-categories: requirements problems, lack of general IT knowledge, lack of knowledge of what the IT department is doing and process problems,
- Requirements – the identification of current requirements engineering problems within the IT department,
- Process – the identification of current software development process problems within the IT department.

Table 8-2 shows the number of occurrences of these categories and the number of sources. The highest occurrence is Business (16 in total). The Business and Marketing departments were blamed for problems with delivered requirements, for no delivered requirements, for not understanding business processes, for not knowing what the IT department is doing and a general lack of IT-oriented knowledge. There appears to be no regard for the Business department especially because the IT department perceived the Business department's role as the providers of requirements and specification documents so that IT can build the products from these documents.

Time constraints are mentioned 11 times in all feedback. Interestingly, the suggested work culture problems were only mentioned 3 times from 2 sources. Perhaps this is not such a barrier? Or the work culture is so ingrained that it is not perceived to be a problem? It appeared to the author that the major issues stopping the conducting of detailed analysis and design were the lack of requirements and specification documents to work from. Little analysis and design can occur when the problem has not been defined. This led to even tighter time schedules whilst IT tried to specify exactly what was required from their Business and Marketing departments. Indeed, one stakeholder within the IT department told the author (informally) that he would no longer agree to start a project unless he had first received a signed off, complete technical specification (as opposed to a telephone conversation, a few scribbled lines on a note or an email that was meant to represent such a document).

Categories		Occurrences	Sources
Time		11	5
Culture		3	2
Business	Requirements	6	3
	IT Knowledge	4	4
	IT Dept Work	3	4
	Process	3	4
Requirements		5	3
Process		5	5

Table 8-2. Identified work problems by stakeholders from presentation feedback and interviews

Recognised requirements and process problems within the IT department itself (as opposed to Business) were also identified (5 each). Without interviewing the Business and Marketing departmental members, the arguments presented here are one-sided.

8.8 Discussion

Though the techniques explored in this thesis are evaluated, there is no stakeholder feedback for the first two techniques (CP Use Case Writing Rules and 7 C's of Communicability) because the author was working in parallel with the Project F team more than working with them and time constraints did not allow the Project F team to produce any use case descriptions. Despite this, the author considers that the techniques are useful based upon his own experience of them. However, since the CP Rules are broken a number of times because the context of the description dictated events rather than the Rule set, they should not be religiously adhered to. Tool support would help in structuring descriptions but this needs to be flexible to cope with the occasions when the Rules are not sufficient. The 7 C's acted as a checklist to rapidly assess the description for any major faults. The subjectivity in assessing whether the descriptions comply with a number of the 7 C's makes it difficult to categorically state the usefulness of them. But as a guide, they are useful, if time consuming. The third technique (Use Case Question Set) received positive feedback. The Question Set raised important design issues, though there

were stakeholder concerns regarding the time taken to do such a detailed analysis. The author agrees. They appeared to work better on shorter / less detailed use cases. The Question Set did help identify 33 classes and give them the responsibility for the events of the 26 identified use case descriptions. The complexity and management of this number of classes is double that for an ATM. (The ATM example in chapter 7 and Appendix E7 identifies 15 classes.) The heuristics are therefore reasonably robust even on this larger real case study though time issues are a potential problem.

Feedback to the case study as a whole (see Appendix F2.4) and with regard the overall process of the work conducted was also favourable and considered similar to the work undertaken by the Project F IT team:

"This is very close to the actual work carried out on the project. We have very similar understandings of the problem and proposed designs. [We] want to integrate both sets of work carried out to present as a full documentation set to the business [the multinational]." - Project F Project Manager.

The case study was relatively successful though there is a lack of stakeholder feedback for the CP Rules and the 7 C's. The Question Set received some positive feedback and there appears to be scope for its further development. One of the most important issues to come out of the study is the lack of time software developers have to deliver projects. Though many of the techniques used in the study by the author were well received, and Company X could see where they would be useful to them, it is unlikely that any will be taken up in a formal capacity because time constraints do not allow it.

Though it is recommended in software engineering texts (e.g. Pressman 1997) that detailed requirements analysis and design occur to help deliver the right product on schedule, this did not occur in this case study in the classical sense. The Project F product was completed and released despite the apparent lack of a classical development process. Short delivery schedules do not allow Company X to use a more formal software development process though they recognise the need to do so.

The last word as to the ideal and the actual approach at Company X, when reflecting on the case study, is left to the IT Development Manager:

"It would be good to model every current system in IT with [the author's approach] so we have details of everything – if we took this as a project in itself – and then every new project could expand on the model, rather than straight on the actual system."

Chapter Nine

Conclusions

9.1 Aims of the Thesis

The lack of guidance in writing comprehensible use case descriptions and in moving from analysis to design are recognised problems for industry (Weidenhaupt et al. 1998). A survey of the literature supports the claim that there is little detailed guidance available. The aim of the thesis is to suggest guidance in these areas and to evaluate their efficacy. The aims, as set out in section 1.4, were:

1. To make textual use case descriptions more understandable.
2. To present guidelines that help the software engineer in extracting relevant specification and design information from use case descriptions.

With regards the first aim, important qualities of use case descriptions in terms of their comprehensibility and their structure were established. The 7 C's of Communicability were derived from suggested best practice and theories of text comprehension (chapter 2). To help engineers write use case descriptions, a small set of writing guidelines was presented. The CP Use Case Writing Rules were derived from the 7 C's and suggested best practice (chapter 4). The CP Rules were evaluated through a pilot study, experimentation and an industrial case study. The comprehensibility of the descriptions produced was assessed by application of the 7 C's (chapters 5, 6 and 8).

The second aim explored how descriptions can be questioned to elicit elements of specification and design, such as external interfaces to other systems and design classes. The pilot study explored the general comprehension and plausibility of the use case descriptions written. This was developed into a larger experiment in chapter 7 that examined what could be elicited from a description by both questions specific to that description and by more general questions applicable to all descriptions. Out of these, the

Use Case Question Set emerged. This was examined further in the industrial study (chapter 8).

9.2 Results

9.2.1 Aim 1: Comprehension of Use Case Descriptions

The CP Rules were compared against the CREWS Use Case Authoring Guidelines (Achour et al. 1999). Comparisons were made of counts of application of these guidelines and of the comprehensibility of the description to the reader (as assessed by the 7 C's of Communicability). Table 9-1 shows the overall results from the pilot study and experiment in terms of the communicability of the use case descriptions.

	Pilot Study (24 subjects, 6 per group)		Experiment (60 subjects, 15 per group)	
Applic.	CP	CREWS	CP	CREWS
ATM	$M_{11} > M_{12}$		$M_{11} = M_{12}$	
Retail	$M_{21} = M_{22}$		$M_{21} > M_{22}$ (not significant)	

Table 9-1. Comparing Pilot and Experiment Results

(M_{11} explanation: M = mean; the first number, in this case $_1$ = treatment 1 (here the ATM task); the second number, in this case $_1$ = group 1 (here the CP group).)

The results show that CP mean scores (M) are better than or equal to CREWS (table 9-1). For example, for the pilot ATM domain, CP's higher mean (M) score indicates that the CP Rules use case descriptions were more comprehensible than CREWS. However, in the experiment, there was no difference between the means for the ATM treatment. In fact, the results of the experiment are opposite to those of the pilot, indicating an amount of external variability might have affected the results, such as time constraints and lack of contextual setting.

In terms of the individual facets of the 7 C's, in the experiment CP (groups A and B) had a significantly better Consistent Structure score than CREWS. Group B (CP Retail) scored significantly better for the Cogent facet than group D. However, these differences are not reflected across the groups.

In the industrial case study (chapter 8), the author applied the CP Rules to the use case descriptions where possible. However, 13% of events in main flows did not conform to the Rules showing that not all possible structures could be covered by the Rules. Any set that did cover all possible structures would be enormous and probably unusable (as evidence, see section 4.3 that found 734 unique grammar structures from a survey of 150 use cases and scenarios). The 7 C's identified problems with some descriptions. However, to search in depth for all use case problems might run the risk of over analysis. But if the application domain were safety critical, one should conduct such in depth analysis. Though both received no stakeholder feedback due in part to project time constraints, their evaluation was conducted as action research. This was subjective to the author's viewpoint (which is biased to the extent that he would like them to be as successful as possible because they are his invention). Both the CP Rules and the 7 C's would probably benefit from tool support.

9.2.2 Aim 2: Interrogating Descriptions

Experimental results (chapter 7) suggest that if descriptions are a mix of abstractions (they contain both external *and* internal design), more classes will be identified. From the designer's viewpoint, the more detail there is in the description, the more he will be able to correctly answer design questions he might have for the descriptions (as opposed to assume correctness).

The Question Set used in the case study was expanded from the experiment to consider the importance of interfaces and the responsibility of the system in guaranteeing success of events. Feedback from the study was positive and it was recognised that this is a useful technique in determining the details of design. There were, though, concerns over time

constraints in employing the Question Set. The Question Set might be more useful when applied to key events only and have a focus on data as well as function.

9.3 Issues Raised in the Thesis

Three important issues emerged during the thesis: the abstraction levels in use case descriptions, expectations when conducting student experiments and the time that companies have to actually conduct detailed requirements analysis and design.

9.3.1 On Abstraction in Use Case Descriptions

Throughout the thesis, the author has attempted to defend the position that abstraction is important in use case descriptions, namely that internal design events should not appear in interaction level descriptions. Though use case descriptions were originally not meant to encompass internal design (Mattingly and Rao 1998), many authors have shown internal design in their descriptions in the literature (see section 2.3.1).

The experimental results in chapter 7 show that a description containing design information will reveal more design details. The lesson that comes out of this is that descriptions should be written with their audience in mind. That is, if the intent of the description is to show actor actions at the interface from the user's perspective, then internal design should not be described. If, however, the audience of the description is the designer, then internal design should be included. Indeed, Insfran et al. (2002) do this by describing a three-column template for use case descriptions that encompasses the problem domain, the system interface and internal design. They state that for the designer, the last column, internal design, is important. It might therefore appear reasonable to have different abstraction levels in a single description, especially if this were implemented into a case tool.

9.3.2 On Student Experiments

A pilot study and two student experiments are described in this work. The results are interesting but the quality of the subject's work was sometimes less than anticipated. For instance, in chapter 7 on average subjects only found 2 classes (group A) or 3 classes (group B). This suggests that either it is difficult to identify classes from descriptions or that the subjects were not very good at doing so. A combination of both is probably the case.

Some of the descriptions in the pilot study and use case writing experiment scored low marks and failed to implement a large number of the guidelines provided. This suggests that either the subjects ignored the guidelines, applied them only occasionally or are not very experienced in writing descriptions.

Clearly student subjects have limited experience in writing descriptions and in finding classes from descriptions and this is a reason for some of the poor results described. This lack of experience can be applied to the wider context that students will probably not always achieve equally good results as practitioners. Since the use of practitioners is not always possible, experimenters might have to lower their expectations as to the ability of their subjects and should not expect large effects from their results.

9.3.3 Industrial Time Pressures

During the case study, it emerged that although Company X understood the need to conduct detailed requirements, analysis and design work, they simply did not have time to do this because of project deadlines. To extend project deadlines would probably mean losing the company's competitive advantage with the potential for losing business. Indeed, some stakeholders informally stated to the author that they had to deliver or they would go out of business and if that meant working around the clock then so be it. Software engineering authors have stressed the need to conduct detailed analysis and design (e.g. Pressman 1997, Insfran et al. 2002), otherwise the final product might not be

accepted by the customer. Members of the IT department at Company X realised the potential benefits of using some of the techniques applied by the author during the study but also realised they had almost no time to use them. Perhaps a balance between 'quick' analysis and design techniques and slightly extended project schedules is necessary. However, that would mean persuading a whole industry to change its working practice. It has been shown that when time is spent on conducting a requirements engineering phase, there is a 'quality for free' factor (Phalp 1995), in that the delivered product better meets customer needs than when there is only a token requirements effort. So it might be better to convince companies to spend the time on requirements engineering rather than developing 'quick' techniques. How this issue can be resolved is left open.

9.4 Validity Threats

It would have been beneficial to experiment further with all the suggested tools with a greater number of subjects, practitioners as well as students, over a number of different domains to prepare the CP Rules more for the case study. Unfortunately, such windows of opportunity are rarely opened. The format of the pilot study was reasonably successful and suggested a formal experiment could be conducted (chapters 5 and 6). Though the results show no overall advantage to CP over CREWS, they do show that the shorter CP set produces as comprehensible descriptions as the CREWS Guidelines in two different domains. This suggests the CP Rules might be expected to produce reasonable results in a case study.

The Question Set was exploratory in nature, relatively roughly defined and the results of the experiment in chapter 7 were slightly disappointing because the number of identified dependencies and classes was lower than expected. Further experimentation might have produced better results to prepare a more comprehensive or possibly streamlined Question Set. More industrial studies in different domains would have more tightly defined the Question Set. However, the outcome of the case study shows that it has potential for use in industry.

9.5 Future Work

The thesis has introduced and experimented with techniques to help use case writers and designers. The techniques are not established and have only proved to be reasonably useful thus far. More experimentation is certainly required and this has been planned (for the CP Rules). Experiments in different domains also need to be explored to build a body of work. With regard the experience and ability of student subjects, it might be wiser to select those who are either final year undergraduate or Masters students since they have more experience and software engineering knowledge than, say, second year undergraduates. Further experimentation should also consider the use of practitioners as subjects to test the heuristics, as stated, in a number of different domains to see if the heuristics can be more refined or tailored to those specific domains. From this it would be useful to determine how generally applicable the heuristics are or whether it is necessary to focus the heuristics to different types of domains. It might be interesting to assess the benefits and costs of these heuristics in practice in industry.

To be more effective, the CP Rules need to be automated. This tool should be able to check and correct sentence structures, employing both the CP Rules and the 7 C's as a validity check. The use case description meta-model (section 2.6.6) could be implemented in XML, for instance, as a potential framework for a case tool.

The Question Set proved to be a relatively successful tool. It would be interesting to introduce the Questions into the case tool to try to automatically generate relevant classes; whether this is feasible should be explored.

The industrial study (chapter 8) raised many interesting topics for further study. Finding a formal, automated mapping from role activity diagrams to use cases would be a useful tool, and, indeed, this work is being currently undertaken. Putting problem frames into practice is also worthwhile since there has not been enough research as yet to show their usefulness and utility. An interesting study might be the feeding of use case descriptions and scenarios into an optical reader or tool that might identify key words – verbs, nouns

etc. – that relate to specific problems. A taxonomy might then be developed to recognise aspects that most commonly associated to different problem frames. It has been shown that identifying single frames is relatively straightforward (Phalp and Cox 2000a, 2000b) but that multi-frame identification is more difficult. This proposed tool might help in determining multi-frame problems. This would then help develop domain specific Question Sets.

Chapter 8 indicates that a process emerged within which the tools and techniques used in the case study seemed to fit (Phalp and Cox 2001). Further industrial studies would be needed to test whether these techniques can be generally ordered in this manner.

9.6 The Contribution of this Work

The thesis explores many issues and is rather wide-ranging. Problems with use case description writing and using descriptions to find elements of design are identified. The author proposes heuristics to try to relieve those problems: three tools that are explored through experimentation and an industrial case study. These are the 7 C's of Communicability, the CP Use Case Writing Rules, and the Use Case Question Set.

The literature shows that many have considered, to varying degrees, what structures one should employ in writing use case descriptions. The 7 C's summarise this by describing qualities that make a good textual use case description in terms of how comprehensible it is to the reader. For this, the 7 C's draw upon theories of discourse processes as well as suggested software engineering best practice. The 7 C's are semantic notions (in the main), however, and as such are subjective to the reader's perspective. It is also time consuming to check each description for 7 C's compliance so tool support would seem the next step.

To help the use case writer produce descriptions that are comprehensible to the reader, the CP Rules are presented. These are derived from the 7 C's and from suggested best practice in use case writing. The CP Rules are compared against another complete writing

guideline set (CREWS Use Case Authoring Guidelines – section 2.4.1.1). The experimental results show that the CP Rules produce as comprehensible descriptions as the CREWS Guidelines and with a smaller structure set. In the industrial study, it was shown that the CP Rules were applied most of the time but occasionally were not wide-ranging enough to cover all structures. However, they were used in 87% of events in the main flows of descriptions. The CP Rules are therefore a contribution because it is shown they can be applied in a real industrial study on different types of descriptions from environmental to interaction to internal design. It is hoped that they would be even more effective through tool support.

The scope of grammar structures used in descriptions cannot be entirely controlled and thus the notion of imposing such a grammar is questionable. However, some imposition is necessary to bring a degree of consistency or standardisation to the description – if reuse or use case patterns are a serious goal; the CP Rules coupled with the 7 C's will be of benefit in this endeavour.

The thesis also suggests that there is a lack of detailed guidance in progressing from analysis to design in terms of exploiting use case descriptions for elements of specification and design. The Use Case Question Set provides designers different focussed questions (on dependencies, interfaces, actors, system) to interrogate use case descriptions at the individual event level to help them identify those elements of specification and internal design. The Question Set emerged from experimentation on the comprehension of use case descriptions which suggested that it was worthwhile developing a set of questions generally applicable to all use case descriptions (chapter 7). They were applied relatively successfully in the industrial study and were considered an important design tool by the case study stakeholders though there was concern over the time it would take to conduct this design work. Though there was recognition that the Question Set needs refinement, it has the potential to become a useful addition to the designer's toolkit.

APPENDICES

Appendix A

Additional Use Case and Scenario Information

A1. Scenarios: Abstraction and Purpose

In the early 1990s the Human Computer Interaction (HCI) community debated the meaning and usage of the scenario. These differences stem from the fact that "scenario" has become a buzzword (Campbell 1992a). The "scenario" word has been applied to just about everything to do with requirements, software and the rest of the world at large. In the early 1990s it became clear to the HCI community – as it now has to the software engineering community – that there were scenarios for different purposes and that to label everything a scenario was really too simplistic. Campbell stated that the scenario should not be so generically applied. He describes "scenario" in HCI to mean: "representative instances of interaction between user and system," (p.6) which isn't too far from a use case scenario (an instance of a use case). Campbell states that scenarios are used for four purposes (and as such classifies them): scenarios to illustrate the system ("illustrations" or "demonstration examples"), scenarios for evaluation ("evaluation tasks"), scenarios for design ("user tasks for design") and scenarios to test theories ("test cases") (pp.6-7). The HCI community's response was staunch, if not harsh. Kyng (1992) criticised Campbell, accusing him of going against the free spirit of naming conventions in industry. Kyng was probably right to state that the genericity of the term 'scenario' is reasonable when at a coarse level of granularity. In defence of Campbell, Kyng's (then stated) view that one can name anything anything has been a problem in the software engineering world and Campbell's deeper aim was to attempt a classification of scenarios (Campbell 1992b). Others wished Campbell luck in his quest but considered he might be "tilting at windmills" (Young and Barnard 1992, p.10) or that scenarios could not be classified since a single scenario might consist of multiple classifiably different elements (Wright 1992) or that there were more types of scenario than Campbell initially suggested (Reisner 1992).

Benner gives some credence to Campbell's views, stating that scenarios are used for four different purposes:

1. Describing and clarifying the relevant properties of the application domain
2. Uncovering system requirements

3. Evaluating design alternatives, and
4. Validating designs

Scenarios are defined as "partial descriptions of system and environment behaviour arising in restricted situations," (Benner et al. 1993, p.119). Basically, they describe scenarios in terms of states and transitions. The approach to states and transitions is a popular method of describing scenarios of business processes for instance with Role Activity Diagrams (Ould 1995) and the UML Activity Diagrams (e.g. Booch et al. 1999).

Though HCI is not the focus of this thesis, there is a cross over from HCI to software engineering. Both disciplines have to concern themselves with the human-machine interface and also the problem domain. This cross over is represented in many works relating HCI and software engineering, for instance, the CREWS project and Sutcliffe's work (1997, 1998). Carroll and Rosson (1992) revert to the more general scenario definition when they state a "scenario is a description (in text, in a storyboard etc) of the activities a user might engage in while pursuing a particular concern," (p.185). Kuutti (1995) sees that scenarios can encompass not only the HCI domain of visual representations but also the internal design world of computer systems, for instance, in terms of interacting objects (Rubin and Goldberg 1992).

A2. Scenarios in Software Engineering – some definitions and discussion

Since scenarios are used more and more in software engineering, the software community should set about trying to pinpoint what a scenario actually is, in as precise a form or description as possible. Turning to the literature, this should give a clear, unified and agreed explicit scenario definition, because after all, requirements have to be right and the following specification and design precise. What one finds is a mix of general and specific definitions. Sommerville and Sawyer (1997) provide both specific: "Scenarios are examples of interaction sessions which are concerned with a single type of interaction between an end user and the system" (p.99) and general: "Scenarios can be thought of as stories which explain how the system is used," (p.99). Potts et al. (1994) are both general and specific: "In a broad sense, a scenario is simply a proposed specific use of a system... More specifically, a scenario is a description of one or more end-to-end transactions involving the required system and its environment," (p.23). It might be wiser to tighten Potts's definition by stating that a scenario is one, and only one, such specific transaction. Potts's relatively specific description is similar to Jacobson's description of a use case (Jacobson et al. 1995, see section 2.1.2) raising, unsurprisingly, the question of whether there is a difference between a scenario and a use case.

Indeed, Booch (1994) did not distinguish between scenarios and use cases. He defined a scenario (read: use case) from a variety of perspectives: "A scenario provides an outline of activities that signifies some system behaviour. Scenarios document decisions about requirements or designs, provide a focus of communication about a system's semantics, and can serve as a blueprint for detailed implementation" (p.4). Booch (2000) has since differentiated scenarios from use cases in that he uses competing use case diagrams and scenario diagrams to model a system. The difference between these two diagrams is a use case is a general description of behaviour and a scenario is an instance of that use case's behaviour (Booch et al. 1999). Rumbaugh (1994) states that a scenario is a "specific history of specific events exchanged among system and actors," (p.8) but does not say whether this can be multi-transactional. Rumbaugh does not distinguish between a use case and a scenario but is in fact describing a use-case scenario (that is, an instance of a use case).

Karat (1995) describes a scenario from a usability engineering perspective: "A scenario is a narrative form that describes someone trying to do something in some environment. As such it is a description of a context, which contains information about users, tasks and environment," (p.112). Bødker (2000), from a similar usability perspective, is equally encompassing in her definition of the core characteristics of the scenario. Scenarios are "hypothetical, selective, bound, connected and assessable" and that their purpose is "to present and situate solutions, to illustrate alternative solutions [and] to identify potential problems" (p.63). Use cases should be no less than this although there is no reason why use case descriptions cannot describe existing transactions. How selective one is in describing descriptions is another matter. One of the most generic descriptions of a scenario's purpose is given by Weidenhaupt et al. (1998) who state that scenarios are used in requirements engineering to "present ways to use a system to accomplish some desired function" (p.34). Sutcliffe (1998), though, is precise in his definition, "For our purposes, scenarios are defined as 'facts describing an existing system and its environment including the behaviour of agents and sufficient context information to allow discovery and validation of system requirements'" (p.49). Sutcliffe then gives a slightly more generic definition in his next sentence: "Scenarios are instances of actual experience with a system captured from users" (p.49). From the perspective of this thesis, the use cases described in the case study describe an existing system. Scenarios are difficult to precisely define and Sutcliffe recognises this point. That is why he says "For our purposes". Antón and Potts (1998) are typically generic in stating that scenarios "are descriptions of concrete system behaviours," (p.219) and so can be internal as well as external to the system. Jarke (1999) is also general is his description of a scenario: "A *scenario* describes (textually or graphically) a possible set of events that might reasonably take place" (p.4). This generic definition could equally apply to the use case. Jarke complicates the definition issue when he states that "a *scene* captures the same [scenario] in some form of multimedia" (p.4). This distinction can be considered an unnecessary complication, though Campbell might disagree. Storyboards, for instance, consist of scenes from a story or scenario, yet storyboards are regarded as a good scenario and prototyping approach (e.g. McGraw and Harbison 1997). Jarke (1999) then goes further by describing the purpose of a scenario (or scene): "Its purpose is to *stimulate and document thinking* about current problems, possible occurrences, assumptions relating these occurrences, action opportunities and risks," (p.4).

Jarke's points are significant in that they define what scenarios are used for. In fact, scenarios appear to be used for different purposes, as Campbell (1992a) previously suggested. One of these purposes provides a rationale for the experiments described in the thesis: scenarios, and specific to this work, use cases, are primarily a communication tool among stakeholders. If everyone is speaking the same language, that is, stakeholders have a shared understanding, then there this more confidence that the requirements will be right (Graham 1998, Alexander 1998).

Scenarios are context specific. Even their definition appears to be context specific. It is only on a very generic level that any agreement can be made. This is usually something like 'scenarios are stories / descriptions of things / events / behaviours that might / do take place between a system and a user / between a user and their environment'.

A3. Where Scenarios Fit into the Software Process

To try to make some sense of the myriad scenario-based approaches to software engineering it is necessary to try to map these approaches to what might be considered a "typical" software engineering process. This will perhaps go some way to helping solve the appropriateness issue though this following section is by no means attempting to state that this is a formal solution, or *the* answer, to the appropriateness problem. This section elaborates on work presented in (Cox 2000a).

Figure A-1 describes a high-level Requirements Engineering (RE) process model. The process model, adapted and extended from (Bray 2002), is split into two types of design: external design (edesign) and internal design (idesign). Bray's model does not show a validation phase because this is applicable across the whole process. Indeed, this could be considered a weakness of the process model since it is regarded as fundamental to project success to review and validate the requirements (and presumably the specification) (Lawrence et al. 2001). As such, this validation/testing task is now added to the model. The process model describes the major phases of the RE process, how they are linked and the documents that should emerge from the phases. (The black dot on the Specification phase linking to HMI design indicates that Human-Machine Interface (HMI) design is part of the Specification phase. The notation is borrowed from Jackson's context diagrams (Jackson 1995).) Each phase is iterative.

Bray notes that publications on requirements engineering often ignore elicitation notes as an important document. As requirements changes arise or mistakes are uncovered there is often no helpful documentation to provide guidance because elicitation notes tend to be thrown away. This is a key concern for this thesis because a major focus of this work is on writing and using use case descriptions, which might be early passes of these such documents. The notion of documenting design rationale, which can be extended back to requirements via use cases is gathering further acceptance (e.g. Dutoit and Paech 2000, 2001, 2002).

There is a problem with this attempted mapping: scenario-based design has generally been happier talking about fitting a context than fitting a process (see examples in Carroll (1995a)). Also, the approaches talk about bridging a gap, with scenarios acting as a "middle-level abstraction" (Carroll 1995b, p.15) between certain phases (notably informal elicitation/analysis and formal specification). This means that scenarios fall

across various phase boundaries and are therefore quite difficult to categorize. The following lists are not exhaustive - they merely offer an insight into what the various authors suggest their approaches cover and also where the approaches might be useful.

(The internal design elements described in figure A-1 have been added onto 'Structural Design' by this author and are hopefully sufficiently detailed to give an overview of these tasks.)

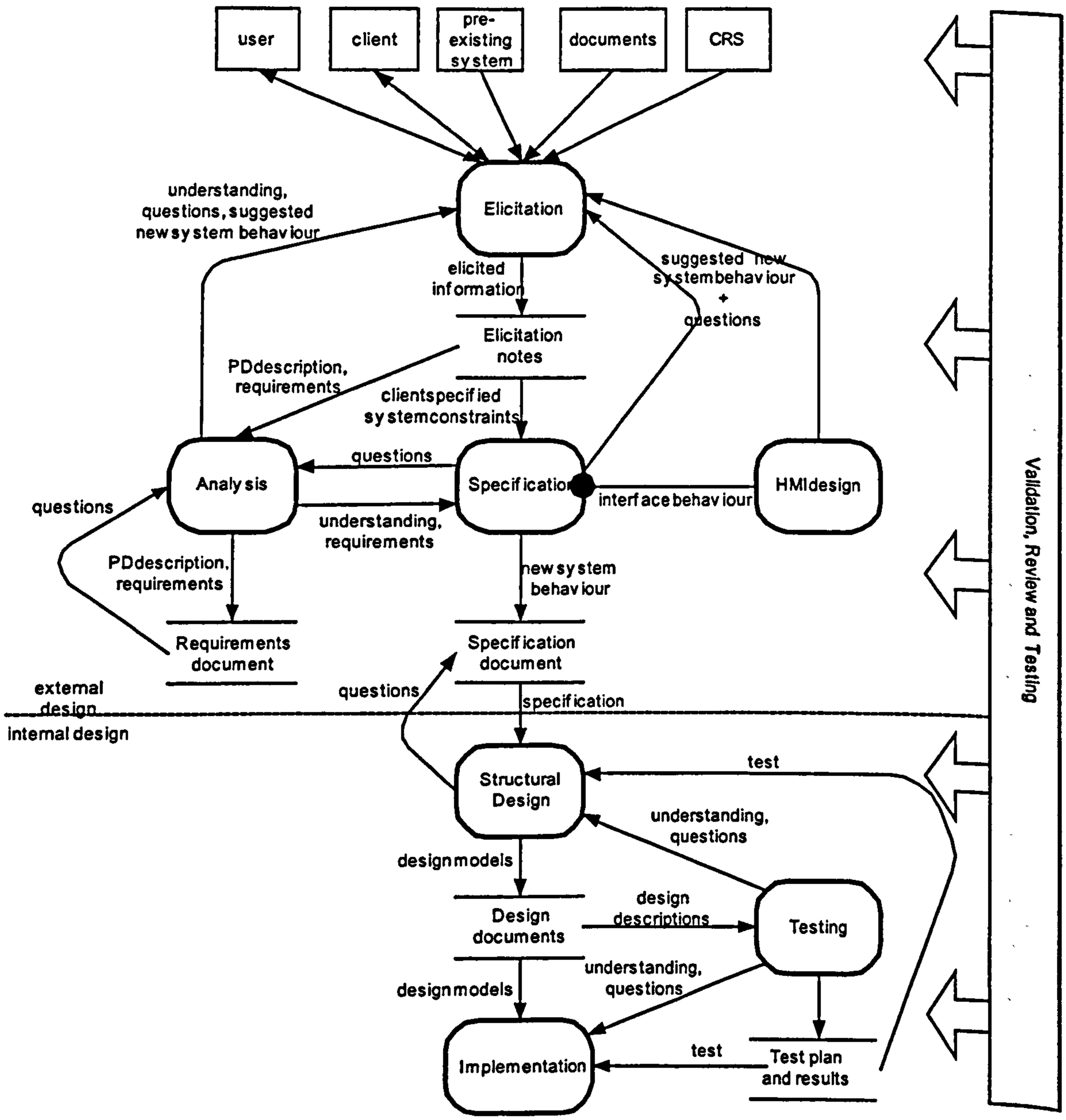


Figure A-1. "ideal" RE process model, adapted from Bray (2002, p.31)

Project Inception (Business Process Task Analysis)

Booch et al. (1999) apply use cases to describe business processes. Haumer et al. (1998) use the CREWS-EVE tool to animate scenarios that describe the current system. Nielson (1995) depicts a Diary scenario, part of his usability engineering process, which captures early scenarios. The Muller et al. (1995) CARD approach can focus on work processes. Carroll et al. (1998) propose a project charter which includes system envisionment scenarios. McGraw and Harbison's (1997) Scenario-Based Engineering Process (SEP) – uses scenarios to model problem space and work processes. Carroll and Rosson (1992) have use-scenarios that are task analysis descriptions of what occurs in the problem environment. Ould's (1995) Role Activity Diagrams can be used to describe business process scenarios.

Requirements Elicitation

Booch et al. (1999) use cases are considered an elicitation tool. Rolland et al (1998b) elicit goals from previously elicited <goal, scenario> couplets. Haumer et al. (1998) use the CREWS-EVE tool to elicit requirements from scenarios. As part of Nielson's (1995) usability engineering process, scenarios are used to support brainstorming. Johnson et al. (1995) develop scenarios for requirements gathering (elicitation); these focus on user tasks. Dzida and Freitag (1998) show context scenarios are an aid to elicitation. Holbrook's (1990) Scenario Based Requirements Elicitation (SBRE) is aimed at elicitation of requirements via scenarios. Carroll et al. (1998) have an ethnographic approach, including the "Inventory of extant workplace" (p.1168) scenario. McGraw and Harbison's (1997) SEP scenarios take the form of storyboards and animation. Carroll and Rosson's (1992) use-scenarios are task analysis descriptions of envisioned behaviour. Benner et al. (1993) use scenarios to uncover requirements and for describing the problem domain.

Requirements Analysis

Booch et al. (1999) employ use cases to describe and analyse functional requirements. Rolland et al. (1998b) <goal, scenario> couplet is used for analysis. Stiemerling and Cremers (1998) have textual scenario use in requirements analysis. Dzida and Freitag's (1998) context scenario provides an early analysis of the requirements. Holbrook's (1990) SBRE approach is suited to analysis. McGraw and Harbison's (1997) SEP scenarios feed requirements analysis via demos/ prototypes. Nardi (1992) states that

scenarios are developed to help understand requirements. Carroll and Rosson's (1992 use-scenarios are analysed via psychological claims. Haumer et al. (1998) analyse scenarios via Message Sequence Charts. Benner et al. (1993) use scenarios to evaluate (analyse) requirements. Feblowitz and Greenspan (1998) use scenarios to determine COTS acquisition.

Requirements Discovery (i.e. inventing new requirements)

For Booch et al. (1999) use case analysis can lead to discovery of new use cases and as such, possible new requirements. Rolland et al (1998b) analyse the <goal, scenario> couplet to discover new goals (and requirements). Muller et al. (1995) PICTIVE envisions system work from CARD scenarios. McGraw and Harbison's (1997) SEP vision scenarios can help in requirements discovery.

Specification

Booch et al. (1999) use-case descriptions are considered specification-oriented (Jackson 1998). Heymans and Dubois (1998) develop formal specifications from scenarios. Karat (1995) used scenarios to help develop the user guide, which also mapped into the specification. Carroll and Rosson's (1992) use-scenarios are used in the (external) design of new system. Wirfs-Brock's, (1995) scenarios (use cases) show Input/Output to actors.

HMI design (part of specification)

As part of Nielson's (1995) usability engineering process, scenarios are developed for User Interface ideas evaluation and prototyping. Erickson (1995) has work scenarios to enable vision prototypes and usage scenarios to build detailed scenarios. Kyng's (1995) scenarios are used to feed system mock-ups. MacLean and McKerlie (1995) envision scenarios (text) / evaluator scenarios (prototype) scenarios. Johnson et al. (1995) prototype scenarios are test beds for development. Rosson and Carroll (1995) use scenarios to generate an executable prototype. Campbell (1992a) suggests scenarios for hypothetical system use/illustrative excerpts. Stiemerling and Cremers (1998) use scenarios for functional evaluation. Dzida and Freitag (1998) use scenarios for design. Carroll et al (1998) scenarios are developed to integrate initial vision, trade-off analysis and technology analysis for COTS prototypes. McGraw and Harbison's (1997) SEP uses scenarios as demonstrators/prototypes. Nardi (1992) uses scenarios to help develop prototypes. Carroll and Rosson's (1992) use-scenarios are used in (external) design of

new systems. Hsia and Yaung's (1988) Scenario Generator tool builds prototype screens. Rosenberg (1999) states use cases should be used to help define the User Manual and as such the graphical interface.

Validation

Benner et al. (1993) use scenarios in validation and evaluation. Booch et al. (1999) descriptions are used to validate requirements. Rolland et al (1998b) use the <goal, scenario> couplet to validate requirements. Haumer et al. (1998) get user validation by viewing animation scenarios. Heymans and Dubois (1998) generate scenarios to perform validation checks on specifications. Nielson's (1995) usability engineering process develops tasks in user testing. Kyng (1995) has exploration/requirements scenarios (provide a link between the use scenario and design) for developers; explanation scenarios hypothesise about elements of the system. Campbell (1992a) categorises scenarios for evaluation of system function. Stiemerling and Cremers (1998) uses scenarios for evaluation (designers only). Dzida and Freitag (1998) use scenarios of a prototype for validation. McGraw and Harbison's (1997) SEP has user validation of scenarios across all phases. Sutcliffe et al. (1998) validate system requirements using scenarios through validation frames.

Structural (internal) Design

Booch et al. (1999) use cases help determine class structures through interaction diagrams and activity diagrams. Kösters et al. (2000), Homrighausen et al. (2002) extend UML Activity Diagrams of use cases to help in the design of class structures. Rosson and Carroll (1995) use scenarios to generate executable prototypes and SmallTalk objects. Robertson (1995) examines scenarios by propositional analysis to find classes and objects. Wirfs-Brock (1995) states that use cases can describe the system's internal design by the system's response to the use case.

Implementation

Jacobson et al. (1992) use cases are implemented in the implementation phase. Booch et al. (1999) use cases are realized.

Testing

Campbell (1992a) categorises scenarios as test cases (this was a popular view of the use case descriptions from the case study – chapter 8). Jacobson et al. (1992), Booch et al. (1999) employ use cases as test cases. Nielson (1995) develops tasks in user testing. Dunsmore et al. (1999) propose the use of scenarios to help comprehend code to enable better detection of defects in programming.

Documentation (across all phases)

Carey and Rusli (1995) design scenarios by analogy – they reuse existing scenarios to improve similar systems in the future. Jacobson et al. (1992) use case driven process allows for use case impact across the whole process and as such, use cases documentation is used in all phases. Jacobson et al. (1999), Krutchen (2000) show the Rational Unified Process is use case driven. As such, use case documentation has an impact across all phases.

Maintenance

Jarke et al. (1998) use scenarios in a goal driven change process for envisioning new scenarios from a current (legacy) system.

Upon examination of the various scenario approaches, many show they are relevant to requirements process phases not shown in figure A-1. *Project Inception* relates more to business processes and task analysis than to traditional requirements engineering. (The Rational Unified Process also calls the first phase in its project lifecycle *inception* (Jacobson et al. 1999).) The phase *Requirements Discovery* is not shown in figure A-1. This is because it can occur at any time in the RE process and it has been suggested that this isn't a phase at all. The point about having a separate section for discovery is that discovery can occur at any time in a requirements process and not just in the requirements elicitation phase. In any case, the distinction is perhaps over-engineered and the critics are probably right (Alexander 2000b).

Note that some approaches cross many of the phases (Booch et al. 1999, Rolland et al. 1998b, McGraw and Harbison 1997, Haumer et al. 1998) and some only appear in a few or just one phase (Kyng 1995, Hsia and Yaung 1988). However, none of the approaches spans all the phases except possibly McGraw and Harbison (1997). They describe a

scenario-based process that focuses on virtually every phase of the RE process. Examination and validation of this promising and complex process still needs occur. Booch et al. (1999) describe use cases. They appear to be applicable to many phases of the RE process. However, some have argued that there are inherent weaknesses in the notation, as well as essentially being specification driven (Jackson 1998, Cox and Phalp 1999, 2000a). That is to say, there do appear to be flaws in the use case approach in that its generality of applicability falls into the trap of many other so-called panaceas.

The list shows that there is the scope to choose a scenario-based approach at every phase. If a developer opts to use a scenario approach then they can select one from the list in the knowledge that the approach is useful, for example, in elicitation or analysis. What the developer might struggle with is knowledge of how to apply the approach, especially if selecting an approach applicable to just one phase. The developer will have to spend time understanding and applying the approach (if new to it). If a developer chooses to use a scenario approach from project inception, what happens when the scenario approach is not applicable to a later specific phase?

The scenario approaches bunch mainly in four of the phases: requirements elicitation, requirements analysis, HMI design and validation. Most of the scenario approaches within the HMI design phase are from the HCI world. The more software engineering-oriented approaches (Booch et al. 1999, Rolland et al. 1998a, Haumer et al. 1998, Heymans and Dubois 1998) do not concern themselves with the actual user interface, although use cases can be used to design and validate user processes at the interface because the use cases should help define the user manual (Rosenberg 1999). In fact, Rosenberg states that an error developers shouldn't make is to ignore the Graphical User Interface (or prototypes) when deciding use case descriptions.

Goal modelling is becoming accepted as a fundamental part of the RE process (Kavakli 2002) in that the requirements must satisfy the business and user goals for the system: "Goals are a logical mechanism for identifying, organising and justifying software requirements," (Antón 1996, p.136). Indeed, Jarke and Kurki-Suonio (1998) state that a focus on user goals is vital in developing scenarios. Jarke (1999) claims "that a scenario-based approach, at least for large projects, is inextricably linked to explicit capture of a goal/requirements hierarchy," (p.5). Antón et al. (2000, 2001) show that if goals are not

identified then there is the danger of not identifying requirements. The case study (chapter 8) had only two stated goals and one subgoal and unsurprisingly, there was also a lack of requirements. Figure A-1 does not explicitly consider goal modelling as a requirements activity. Perhaps it should.

A4. Use Case Title, Actors and Context

It is accepted that each use case must have a title. Indeed, it is absurd that one shouldn't. It is the case that readers will first lay a foundation for the mental model (Gernsbacher 1996) they are about to construct for the use case from its title. People take more time to read the first words of a sentence or clause (Aaronson and Ferres 1983) (and in this case, that would be the use case title), or the first sentence of a paragraph (Haberlandt 1984) because they are constructing mental representations or models from which they can more rapidly infer the following words or sentences. Indeed, Long et al. (1996) recount a classic experiment conducted by Bransford and Johnson (1973) that emphasises the need for a highly relevant title to text.

> "If readers are unable to make inferences that connect explicit information in a text to relevant world knowledge, they feel as though they have not comprehended the text and have difficulty remembering it. In their classic experiment, Bransford and Johnson (1973) asked readers to comprehend and recall paragraphs that contained a number of vague referring expressions. These passages elicited low comprehension ratings and poor recall. Comprehension ratings and recall dramatically improved by providing readers with a title that evoked relevant world knowledge." (Long et al. 1996, pp.189-190).

Thus a use case title that is highly relevant and somehow invokes world knowledge is of great importance. It is assumed that those who have to read and validate use cases have that world knowledge, otherwise their contribution to comprehending that use case would be small.

Kulak and Guiney (2000) suggest that

> "When you're considering use case names, it's a good idea to run them through the following filters. Use case names
>
> - Should conform to verb-noun construction: mail checks, determine legibility, trace shipment, print letter
> - Can contain adjectives or adverbs

- Should not be instances of classes and should not contain any situation-specific data
- Should not be tied to an organisation structure, paper forms, or computer implementation: enter form 104-B, complete approval window, get approval from immediate supervisor in Accounting Department
- Should not use "weak verbs" that do not describe the action: *process, complete, do, track*" (pp.77-78).

These are highly sensible rules and ones that should always be followed, if possible. This author, though, is unsure about the use of adjectives and adverbs since they introduce unnecessary quantifiers and therefore uncertainty.

Rosenberg (1999) gives a simple grammar structure to use case names. "A use case is most effectively stated from the perspective of the user as ***a present-tense verb phrase in active voice.*** For instance, one use case within a hospital system might be called Admit Patient, while a portfolio system is likely to contain use cases named Do Trade Entry, Update Portfolio Information, and Generate Reports" (p.39). It is a wise idea to consider the title of the use case and make sure that it describes the goal of the use case.

The general rule for naming use cases, then, is the verb-noun phrase (Cockburn 2001) for example, Make Book Order. Henderson-Sellers (1997) has raised a cautionary word about the current trend of turning nouns into verbs and verbs into nouns. One must take great care in naming the use case. It is left to the writer to be able to differentiate between verbs and nouns. Thus the use case name should represent the **goal** that the use case is trying to fulfil.

Actors need careful naming. They are essentially roles and as such, names of people should be avoided. Focus should be on the role and its responsibility. Actors should play a very meaningful role in either inputting or receiving output or both to the use case. Kulak and Guiney (2000) provide a list of good and bad actor names. Good names: pension clerk, sales supervisor, production accountant, customer service representative. Poor actor names: clerk, third-level supervisor, data entry clerk #165, Eddie "The Dawg" Taylor (p.78). Along with a name, the actor should have a brief specification explaining

its role. Schneider and Winters (1998) put it thus, "Each actor needs a descriptive name and a brief description that is one or two sentences long" (p.17). For example, Appendix F3 provides actor specifications.

Haberlandt (1984) suggests that great care is taken in forming the first sentence of a paragraph; it is also equally vital to get the first sentence correctly formed in the section on contextual information. It must be made clear who the actor is, what their goal is and why they want to instantiate this particular use case; that is, what do they expect to get out of the use case?

Readers are more likely to comprehend the first sentence in a paragraph, or the first clause in a sentence as a conveyor for the meaning of the rest of the paragraph or sentence. These parts are more accessible to people than the rest of the sentence – they recall the first clause, not the middle or end clause (Gernsbacher 1996, pp.290-291). Thus the first interaction in the use case is very important. It is not worth starting with sentences such as "The user starts the application" or "The user logs on to the system".

Appendix B

Use Case Grammar Analysis

B1. Most common structures

Most repeated structures from 1913 sentences, count and percentage

Subject	Verb	object					313	16.40%					
subject	verb	direct obj	Preposition	Indirect Obj			77	4%					
subject	verb	InDirect Obj	Preposition	Direct Obj			44	2%					
verb	object						35	1.80%					
subject	future	object					34	1.80%					
subject	passive						34	1.80%					
Subject	Verb	Preposition	Object				29	1.50%					
<<use use case name>>							28	1.50%					
subject	verb						27	1.40%					
subject	verb	object	Infinitive	object			23	1.20%					
subject	Verb	object	Conjunction	object			22	1.20%					
subject	future	direct obj	Preposition	Indirect Obj			21	1.10%					
subject	verb	Preposition	Subject	verb	Object		17	0.90%					
subject	passive	Preposition	Object			17	0.90%						
IF	subject	verb	Object			16	0.80%						
THEN		16	0.80%										
end if		14	0.70%										
IF	subject	verb	Object	THEN			13	0.70%					
GOTO		13	0.70%										
subject	verb	object	past participle				12	0.60%					
subject	preposition	object	Passive		11	0.50%							
subject	verb	direct obj	Preposition	Indirect Obj	Preposition	indirect obj	10	0.50%					
ELSE	IF	subject	Verb	object	THEN		10	0.50%					
subject	verb	Preposition	Subject	passive		10	0.50%						
subject	verb	object	Passive		10	0.50%							
verb	object	passive		10	0.50%								
end loop		10	0.50%										

B2. Count of different structures

Count of occurrence of structures

Subject	Verb	InDirect Obj	Prep	Gerrund	Direct Object	5						
Subject	Verb	Direct Obj	Infinitive	Article	Adjective	Indirect Obj	1					
Subject	Verb	Preposition	Object	29								
Subject	Verb	object	Gerrund	object	8							
Subject	Verb	object	313									
Subject	Verb	Preposition	Indirect Obj	Direct Obj	1							
Subject	Verb	Adjective	Object	1								
Subject	Verb	InDirect Obj	Gerrund	Preposition	Indirect obj	Preposition	Direct Obj	1				
Preposition	Determiner	object	Past Participle		1							
Subject	Verb	Object	Conjunction	Object	Adjective	1						
Subject	Verb	Object	Conjunction	Object	22							
Subject	Verb	Direct Obj	Preposition	Indirect Obj	Conjunction	if-clause	Passive	1				
Subject	Verb	Object	Verb	Conjunction	Verb	Object	1					
article	noun	adverb	article	subject	verb	object	1					
article	subject	verb	article	Indirect Obj	infinitive	article	Direct Obj	1				
article	subject	verb	article	Object	2							
article	subject	verb	noun	2								
IF	indirect obj	verb	noun	Article	subject	verb	Direct Obj	infinitive	article	possessive noun list and conjunction	3	
article	subject	verb	Object	infinitive	article	noun phrases	1					
article	subject	verb	article	object list	1							
article	subject	verb	Object	adverb	object	verb	adverb	1				
article	subject	verb	Object	possessive noun	adverb	article	subject	verb	article	object	possessive nouns	adverb	conjunction	verb	prep	noun	1

article	subject	modal	verb	Article	Indirect obj	Infinitive	article	noun	prep	adjective	uncountable noun		1
Preposition	adjective	Noun	Preposition	noun	article	subject	verb	article	object	infinite	determiner	countable noun	1
IF	article	subject	Verb	noun	preposition	adjective	noun	article	subject	verb	object	1	
subject	verb	Object	Preposition	Gerrund	infinitive	article	object	prepos	agent	1			
adjective	object	1											
noun	object	verb	passive infinitive	Preposition	Conjunction	Preposition	2						
article	object	verb	passive infinitive	1									
noun	preposition	Noun	1										
IF	subject	passive	4										
subject	verb	adverb	Preposition	Indirect Obj	preposition	article	Direct Obj	1					
IF	subject	verb	adverb	verb	preposition	Object	1						
IF	subject	verb	negative	adverb	1								
subject	verb	Preposition	Indirect Obj	Article	adverb	direct obj	conjunction	direct obj					
verb	preposition	object	5										
subject	verb	InDirect Obj	Preposition	Article	Direct Object	1							
IF	subject	verb	Object	16									
object	preposition	Object	1										
subject	verb	InDirect Obj	direct obj	4									
adverb	subject	verb	Indirect Obj	Article	Direct Object	conjunction	article	noun	prep	pronoun	verb	passive	phrasal verb	prepositi on	(else)	1
IF	subject	negative	Preposition	Article	object	1										
subject	verb	InDirect Obj	Preposition	Direct Obj	44											
IF	indirect obj	verb	noun	noun(s)	passive	Preposition	Direct Obj	conjunct	indirect obj	passive	prep	adverb	prep	article	adverb	noun	1
IF	indirect obj	verb	noun	Indirect Obj	passive	infinitive	prep	object	adverb	passive	prep	adverb	prep	article	adverb	noun	1
subject	adverb	Infinitive	prep	determiner	object	adverb	1										

subject	verb	InDirect Obj	Preposition	Indirect Obj	Conjunction	direct obj		4
subject	verb	InDirect Obj	adjective	Preposition	Direct Object		2	
subject	verb	InDirect Obj	Preposition	Direct Obj list		1		
subject	verb	object list	9					
if	subject	Verb	Object	or	Object	1		
else	2							
subject	verb	adverb	negative	Object	Adjective	Infinitive	object	1
negative	subject	passive	3					
subject	past	Preposition	Object	object list	1			
subject	verb	object	Infinitive	object	OR	verb	object	2
subject	verb	object	OR	2				
subject	verb	InDirect Obj	possessive of	Indirect Obj	preposition	direct obj		1
subject	verb	determiner 'any'	Gerrund	object	2			
subject	verb	determiner 'all'	object	passive	conjunction	object		2
subject	verb	InDirect Obj	Infinitive	Indirect Obj	Direct Object	verb	Infinitive	1
subject	verb	object	Conjunction	subject	Verb	Infinitive	1	
subject	verb	object	verb	adjective	adverb	past	1	
subject	verb	direct obj	Preposition	Indirect Obj	77			
subject	verb	negative	Object	or	object	1		
subject	verb	object	Infinitive	object	23			
subject	verb	negative	adjective	Preposition	object	past	1	
subject	verb	Object	verb	Adjective	1			
subject	verb	Object	Infinitive	preposition	Object	Infinitive	object	OR	infinitive	1
subject	verb	Infinitive	Preposition	object	1			
if	subject	verb	Infinitive	object	Verb	Preposition	object	1

subject	aux verb	negative	verb	Object	3							
object	negative	verb	1									
negative	determiner 'all'		object	passive	2							
subject	verb	Object	passive	gerrund	conjunction	verb	object	2				
verb	Preposition	direct obj	Preposition	Indirect obj	2							
object	verb	passive	1									
subject	verb	object	conjunction	Object	preposition	Object	1					
subject	verb	object	present passive	perfect	1							
subject	passive	Infinitive	object	5								
subject	verb	object	past participle	12								
subject	verb	past participle	Object	verb	Adjective	1						
subject	verb	object	Infinitive	direct obj	passive	Infinitive	preposition	Indirect obj	2			
subject	verb	InDirect Obj	Infinitive	direct obj	Conjunction	direct obj	conjunction	direct obj	conjunction	direct obj	1	
subject	verb	object	anaphor	verb	Adjective	1						
subject	verb	object	Infinitive	object list	1							
subject	verb	object	EITHER	by	object	OR	adverb	1				
IF	subject	verb	Object	(noun phrase)	THEN	subject	verb	negative	object	passive	adverb	Indirect	adverb	object	1
IF	subject	verb	Object	subject	Verb	negative	object	THEN	1						
subject	verb	Object	Object	Conjunction	Object	1									
IF	subject	verb	Object	THEN	13										
subject	verb	direct obj	Preposition	object list	1										
subject	verb	InDirect Obj	adverb	direct obj	Verb	negative	adjective	preposition	object	1					
subject	verb	object	past participle	conjunction	determiner 'any'		object list	1							
subject	verb	object	passive	10											
subject	verb	direct obj	Preposition	Indirect Obj	gerrund	Indirect obj	infinitive	1							

subject	verb	object	conjunct ion	Object	passive	infinitive	1				
subject	verb	adverb	Object	verb	negative	adjectiv e	prepositi on	article	noun	past participle	2
subject	verb	InDirect Obj	Preposit ion	adverb	Direct Object	verb	adjectiv e	1			
subject	verb	object	Preposit ion	1							
verb	pronoun	Preposit ion	Object	<<extend>> use case		OR	1				
phrasal verb	object	conjunct ion	object	passive	1						
subject	verb	adverb	Object	conjunct ion	object	past participl e	verb	object	past participle	1	
subject	verb	InDirect Obj	adjectiv e	Preposit ion	Direct Object	or	direct obj	1			
subject	verb	InDirect Obj	Infinitive	direct obj	Conjunc tion	direct obj	GOTO	1			
subject	verb	object	Infinitive	9							
phrasal verb	InDirect Obj	negative	Preposit ion	Direct Object	OR	1					
verb	object	GOTO	1								
subject	modal verb	article	Indirect Obj	Preposit ion	Direct Object	infinitive	1				
subject	modal verb	object	Infinitive	1							
IF	object	GOTO	3								
direct obj	past participl e	gerrund	Indirect Obj	conjunct ion	Direct Object	passive	prepositi on	indirect obj	IF	adjectiv e	THEN	verb	1
subject	passive	Preposit ion	object	17							
subject	verb	InDirect Obj	Preposit ion	direct obj	Conjunc tion	verb	direct obj	passive	passive	infinitive	1
subject	verb	InDirect Obj	Preposit ion	direct obj	passive	2					
IF	determi ner 'all'	adjectiv e	GOTO	1							
IF	determi ner 'all'	OR	determiner 'some'		verb	adjectiv e	subject	verb	object	OR	object	1
subject	Verb	IF	Object	modal	passive	infinitive	1				
Verb	negative	modal	passive	infinitive	GOTO	1					

future	passive	GOTO				1								
subject	passive	object				4								
IF	subject	aux negative verb		Infinitive	object		1							
IF	subject	aux negative verb		Infinitive	object	conjunct ion	modal	passive infinitive		THEN		1		
IF	subject	verb	Object	THEN	subject	passive		1						
subject	verb	object	passive infinitive			7								
IF	object	THEN	subject	verb	object	gerrund	object		1					
IF	indirect obj	Preposit ion	direct object	THEN		1								
subject	verb	object	verb	Object		3								
subject	passive	Preposit ion	object	Infinitive		1								
subject	verb	subject	object	adverb	Verb	passive infinitive			1					
subject	verb	InDirect Obj	conjunct ion	Indirect Obj	prepositi on	direct obj		2						
subject	verb	InDirect Obj	IF	direct object	passive	prepositi on	verb		1					
subject	verb	InDirect Obj	adverb	passive	prepositi on	direct obj		1						
subject	verb	object	Infinitive	Object	OR		1							
IF	subject	verb	Object	conjunct ion	subject	passive	object	subject	passive	prepositi on	subject	possessive object	infinitive	1
subject	verb	InDirect Obj	past participl e	Preposit ion	Direct Object		2							
subject	verb	object	passive infinitive	IF	subject	present continuous		1						
subject	verb	direct obj	Preposit ion	Indirect Obj	prepositi on	indirect obj		10						
IF	subject	verb	Object	GOTO		2								
IF	subject	negative	verb	Indirect Obj	Direct Object	modal	passive	prepositi on	indirect obj	e.g. noun list		1		
IF	object	verb	adjectiv e	GOTO		1								
subject	verb	direct obj	Preposit ion	Indirect Obj	GOTO		2							
subject	verb	object	negative	adjectiv e		1								
subject	verb	InDirect Obj	Preposit ion	direct obj	negative	adjectiv e	infinitive	adverb	direct obj	verb		1		
subject	verb	InDirect	Infinitive	direct	prepositi	indirect		7						

subject	verb	Obj object	subject	obj passive	on prepositi on	obj 1							
IF	subject	verb	direct object	Indirect Obj	future infinitive	WHETH ER	Indirect obj	future passive	prepositi on	Indirect ob	OR	Indirect obj	1
IF	subject	verb	Object	subject	Verb	object	pronoun	passive	infinitive	1			
IF	prepositi on	noun	subject	future	object	GOTO	1						
IF	indirect obj	subject	future	direct obj	GOTO	1							
subject	future	object	IF	subject	future	object	1						
IF	object	6											
GOTO	13												
subject	future	direct obj	Preposit ion	Indirect Obj	21								
subject	future	InDirect Obj	direct object	1									
subject	future	InDirect Obj	Preposit ion	direct object	adverb	past particip l e	1						
subject	future	Preposit ion	Object	conjunct ion	Verb	pronoun	verb	adverb	present perfect		Indirect obj	prepositi on	direct object 1
subject	future	Preposit ion	Object	IF	adjectiv e	1							
subject	verb	27											
subject	verb	object	passive infinitive	Preposit ion	4								
subject	verb	object	GOTO	9 ?									
IF	subject	verb	Object	THEN	subject	verb	object	9					
IF	subject	verb	Object	THEN	subject	verb	direct obj	prepositi on	Indirect obj	3			
subject	verb	InDirect Obj	Preposit ion	direct obj	gerrund	object	2						
END	6												
subject	verb	negative	adjectiv e	Preposit ion	object	1							
GOTO	ELSE	1											
subject	verb	negative	past particip l e	Preposit ion	object	1							
prepositi on	adjectiv e	noun	subject	passive	prepositi on	object	1						

subject	verb	Preposition	Object	Preposition	gerrund	object		1							
subject	passive	Infinitive	Indirect Obj	Preposition	Direct Object		1								
subject	verb	Infinitive	WHETHER	object	Verb	adjective	OR	adjective		1					
subject	passive	Preposition	Indirect Obj	Preposition	WHETHER	OR	negative	direct obj	future infinitive	OR	WHETHER	determiner 'any'	direct obj	passive	1
subject	passive	Preposition	direct object	adverb	modal	passive	preposition	indirect obj	1						
determiner 'all'	subject	Preposition	object	conjunction	object	passive	1								
subject	verb	Preposition	gerrund	Preposition	Direct Object	preposition	indirect obj	1							
IF	subject	verb	object	(comma)	subject	verb	object	6							
FOR	object	past participle	1												
subject	verb	object	conjunction	IF	subject	passive	1								
subject	verb	conjunction	verb	object	2										
object	modal	passive	Preposition	1											
subject	passive infinitive	Preposition	object	passive	1										
object	negative	adjective	2												
subject	verb	object	conjunction	Verb	2										
IF	direct obj	verb	negative	adjective	preposition	past participle		indirect obj	THEN	pronoun	passive	prepo	direct obj	verb	prepos	indirect obj	1
subject	verb	infinitive	indirect obj	preposition	determiner 'any'	direct obj	2										
subject	verb	InDirect Obj	Preposition	direct obj	conjunction	direct obj	conjunction	direct obj	1								
subject	verb	object	conjunction	object	conjunction	object	1										
subject	verb	object	conjunction	verb	object	1											
subject	verb	object	verb	negative	adjective	1											
subject	verb	object	adverb	3													
subject	*verb*	*IF*	*object*	*passive*	2												
object	*auxiliary*	*negative*	*verb*	*passive infinitive*	1												

object	future passive		1									
subject	verb	InDirect Obj	Preposition	direct obj	passive	infinitive		3				
IF	object	THEN	verb	object	gerrund	object		1				
IF	object	THEN	verb	conjunction	verb	object		1				
subject	verb	object	conjunction	verb	preposition	direct obj	preposition	indirect obj		1		
IF	object	negative	adjective	THEN	1							
IF	subject	auxillary	negative	verb	infinitive	object	THEN	2				
IF	object	THEN	subject	verb	adjective	1						
IF	direct obj	Preposition	Indirect obj	conjunction	Indirect obj	verb	adjective	adverb	noun phrase	THEN	GOTO	1
subject	verb	direct obj	verb	adjective	preposition	indirect obj	1					
IF	subject	present continuous		object	THEN	GOTO	1					
IF	subject	verb	object	THEN	subject	verb	preposition	object	2			
IF	subject	verb	object	THEN	subject	verb	direct obj	preposition	verb	preposition	indirect obj	1
IF	subject	verb	object	THEN	GOTO	2						
subject	verb	negative	adjective	Infinitive	1							
subject	auxillary	negative	past participle	Preposition	object	GOTO	1					
subject	verb	negative	adjective	Preposition	object	1						
subject	verb	object	IF	Object	2							
subject	verb	direct obj	Preposition	Indirect obj	conjunction	verb	Indirect obj	preposition	direct obj	3		
IF	adjective	object	GOTO	1								
subject	direct obj	passive	Preposition	Indirect obj	1							
subject	verb	direct obj	gerrund	Indirect obj	preposition	gerrund	IF	passive	1			
subject	verb	object	passive	Preposition	object	2						
subject	preposition	gerrund	object	1								

negative	on direct obj	Preposition	indirect obj		1											
verb	object	infinitive	object		1											
verb	object	conjunction	verb	object		1										
verb	indirect obj	Preposition	direct obj		3											
verb	object	infinitive	object	OR		1										
subject	passive	object	passive		3											
subject	passive	IF	adjective		1											
IF	subject	negative	passive		1											
subject	passive	34														
verb	object	35														
subject	verb	object	gerrund		1											
subject	verb	object	gerrund	GOTO		1										
negative	object	passive	GOTO	1												
subject	auxillary	negative	verb	Infinitive	object	1										
subject	verb	InDirect Obj	conjunction	Indirect obj	preposition	direct obj	GOTO	1								
IF	negative	object	subject	modal	verb	object	preposition	verb	1							
IF	subject	negative	modal	verb	(comma)	subject	verb	object	1							
subject	verb	GOTO	3													
subject	verb	object	gerrund	subject	verb	adverb	adverb	verb	negative	adjective	object	1				
IF	subject	verb	direct obj	Preposition	indirect obj	1										
IF	object	subject	verb	object	1											
subject	verb	Preposition	Indirect obj	preposition	direct obj	7										
IF	object	subject	verb	object	GOTO	1										
IF	object	subject	verb	Indirect obj	preposition	direct obj	1									
IF	subject	verb	Indirect object	Preposition	Indirect obj	direct obj	THEN	subject	verb	Indirect obj	conjunction	Indirect obj	direct obj	verb	prepos	object	1
subject	verb	direct obj	Preposition	Indirect obj	conjunction	subject	gerrund	object	1							
subject	verb	object	Infinitive	object	GOTO	1										

negative	determiner 'more'	subject	object	infinitive	GOTO	1							
negative	object	GOTO	1										
subject	verb	object	conjunction	verb	object	GOTO	1						
subject	verb	object	conjunction	verb	object	IF	adjective	1					
subject	verb	object	IF	subject	verb	object	4						
subject	verb	object	past participle	adverb	object	preposition	adverb	object	conjunction	indirect obj	preposition	direct obj	1
subject	verb	EITHER	1										
infinitive	object	past participle	OR	1									
subject	EITHER	2											
verb	object	Infinitive	OR	1									
verb	object	past participle	conjunction	verb	object	1							
subject	verb	direct obj	Preposition	Indirect obj	passive infinitive	indirect obj	conjunction	indirect obj	1				
subject	verb	object	conjunction	verb	preposition	object	1						
subject	auxiliary	negative	verb	object	8								
subject	verb	object	verb	infinitive	object	OR	GOTO	1					
subject	verb	infinitive	object	9									
subject	verb	object	subject	verb	object	conjunction	object	infinitive	1				
subject	verb	direct obj	Preposition	pronoun	indirect obj	1							
subject	passive	Preposition	object	infinitive	subject	infinitive	preposition	object	1				
adverb	subject	verb	object	subject	verb	Indirect obj	preposition	indirect obj	preposition	direct obj	1		
IF	subject	negative	verb	adverb	(comma)	subject	verb	object	adverb	1			
IF	subject	negative	verb	(comma)	subject	verb	indirect obj	preposition	noun	preposition	direct obj	1	
preposition	determiner 'all'	direct obj	Preposition	indirect obj	passive	(comma)	subject	verb	direct obj	preposition	indirect obj	1	
subject	verb	Preposition	object	IF	subject	verb	object	1					

OTHER WISE	subject	verb	Preposit ion	object	IF	subject	verb	object	2					
subject	passive	verb	Infinitive	direct obj	prepositi on	indirect obj	1							
subject	passive	conjunct ion	passive	Preposit ion	subject	prepositi on	object	1						
IF	subject	present perfect	object	(comma)	subject	verb	object	1						
ELSE	GOTO	1												
subject	verb	negative	object	2										
subject	verb	Preposit ion	object	Preposit ion	adverb	adjectiv e	verb	OR	subject	prepositi on	pronoun	object	1	
OR	GOTO	direct obj	Preposit ion	indirect obj	1									
subject	EITHER	verb	object	1										
OR	subject	verb	direct obj	Preposit ion	indirect obj	1								
subject	modal auxillary	verb	1											
subject	negative	modal auxillary	verb	1										
GOTO	object	OR	object	1										
subject	verb	object	Infinitive	subject	prepositi on	object	2							
loop	subject	passive	infinitive	1										
do	1													
THEN	subject	verb	object	2										
subject	verb	direct obj	Preposit ion	indirect obj	verb	object	conjunct ion	verb	object	1				
THEN	subject	verb	object	conjunct ion	object	1								
subject	verb	direct obj	Preposit ion	indirect obj	conjunct ion	verb	object	2						
subject	adjectiv e	infinitive	object	1										
continue	IF	object	THEN	object	prepositi on	1								
subject	verb	direct obj	verb	direct obj	prepositi on	indirect obj	1							
continue	subject	verb	object	1										
IF	object	THEN	GOTO	1										
subject	verb	direct	adjectiv	prepositi	indirect	1								

		obj	e	on	obj						
ELSE	subject	verb	direct obj	adjectiv e	prepositi on	indirect obj					1
verb	prepositi on	determi ner 'all'	subject	past participl e		1					
switch	subject	verb	direct obj	Preposit ion	indirect obj		1				
subject	verb	object	infinitive	prepositi on	indirect obj	prepositi on	direct obj				1
switch	object		1								
object		6									
subject	verb	WHETH ER	object	conjunct ion	object		2				
subject	verb	WHETH ER	object		1						
subject	verb	WHETH ER	adjectiv e	subject	passive infinitive	prepositi on	object				1
IF	subject	verb	negative	object	(comma)	subject	verb	infinitive	object	GOTO	1
IF	subject	verb	Infinitive	object		2					
subject	verb	object	Infinitive	subject	verb	object		2			
subject	verb	object	IF	adverb	verb	determiner 'any'	object	passive	Infinitive		1
subject	verb	IF	subject	verb	passive	Infinitive					1
subject	verb	negative	Infinitive	object		1					
subject	verb	passive infinitive	verb	gerrund		1					
subject	auxillary	infinitive	Preposit ion	object	conjunct ion	verb	object				1
subject	verb	object	conjunct ion	subject	verb	object		1			
subject	negative auxillary	modal	verb		2						
subject	verb	object	Infinitive	object	past participl e		1				
subject	verb	object	IF	determi ner 'any'	object	passive		1			
subject	verb	negative	object	passive		1					
subject	verb	object	Infinitive	subject	past participl e		1				

subject	verb	Preposition	gerrund	object	conjunction	gerrund	subject	object	1								
subject	verb	Preposition	gerrund	pronoun	direct obj	preposition	indirect obj	conjunction	gerrund	pronoun	prepositi on	verb	1				
subject	verb	Preposition	gerrund	object	adverb	verb	subject	verb	object	1							
subject	verb	object	conjunction	verb	object	preposition	object	1									
subject	verb	Preposition	gerrund	object	conjunction	subject	object	2									
subject	conjunction	subject	verb	preposition	object	3											
subject	verb	Preposition	gerrund	object	7												
subject	verb	Preposition	gerrund	indirect obj	preposition	direct obj	4										
subject	verb	direct obj	conjunction	direct obj	preposition	indirect obj	preposition	object	1								
subject	verb	direct obj	Preposition	indirect obj	conjunction	indirect obj	conjunction	verb	object	prepositi on	direct obj	modal auxillary	verb	prepositi on	directo	prep	object	1
object list	1																
subject	verb	object	conjunction	subject	verb	preposition	object	1									
subject	verb	Preposition	gerrund	direct obj	preposition	Indirect obj	conjunction	Indirect obj	4								
preposition	subject	verb	object	subject	verb	pronoun	preposition	pronoun	object	1							
subject	verb	Preposition	object	verb	infinitive	object	1										
subject	verb	Preposition	gerrund	indirect obj	preposition	direct obj	preposition	gerrund	Indirect ob	conjunction	indirect ob	(example list given)	1				
subject	verb	direct obj	Preposition	indirect obj	preposition	indirect obj	conjunction	verb	indirect ob	adverb	direct obj	modal auxillary	present perfect	1			
preposition	object	IF	negative	subject	past passive	preposition	object	(comma)	pronoun	object	auxillary	negative verb	1				
subject	verb	Preposition	indirect obj	conjunction	indirect obj	present perfect		prepositi on	direct obj	1							
subject	verb	object	adverb	adjective	infinitive	object	1										
subject	verb	Preposition	gerrund	object	conjunction	verb	direct obj	prepositi on	Indirect ob	1							
subject	conjunction	subject	verb	object	verb	adjective	1										

subject	verb	Preposition	gerrund	direct obj	direct obj	direct obj	conjunction	direct obj	conjunction	subject	verb	adjective	infinitive		1
subject	verb	Preposition	gerrund	object	infinitive	object	conjunction	verb	pronoun	object		1			
subject	verb	direct obj	Preposition	indirect obj	conjunction	subject	verb	direct obj	preposition	indirect obj	preposition	indirect obj		1	
subject	verb	InDirect Obj	indirect obj	preposition	direct obj		1								
subject	verb	object	conjunction	subject	verb	pronoun	passive	infinitive	preposition	object		1			
subject	auxiliary	adverb	past participle	(passive)		1									
subject	verb	Preposition	gerrund	object	conjunction	gerrund	indirect obj	indirect obj	preposition	direct obj		1			
subject	verb	pronoun	object		1										
subject	verb	Preposition	gerrund	conjunction	subject	verb	object		1						
subject	verb	Preposition	direct obj	preposition	indirect obj	preposition	object	indirect obj	conjunction	Indirect obj		1			
subject	verb	adjective	object		1										
subject	verb	Preposition	gerrund	object	conjunction	gerrund	indirect obj	direct obj	verb	preposition	pronoun	object	OR	object	1
subject	verb	Preposition	gerrund	object	conjunction	object	conjunction	preposition	gerrund	indirect obj	preposition	direct obj	verb		1
subject	verb	Preposition	object	verb	object	conjunction	object	conjunction	verb	direct obj	preposition	indirect obj		1	
subject	verb	Preposition	gerrund	object	infinitive	direct obj	preposition	indirect obj	past participle		preposition	object		1	
preposition	subject	verb	Preposition	object	subject	verb	object	conjunction	object		1				
subject	verb	conjunction	verb	object	conjunction	preposition	pronoun	verb	object	verb	object		1		
subject	verb	Preposition	gerrund	direct obj	OR	direct obj	preposition	indirect obj	past participle	object		1			
preposition	subject	verb	Preposition	object	subject	verb	object		1						
subject	verb	Preposition	gerrund	object	infinitive	preposition	object	past participle			1				
subject	verb	Preposition	object	verb	gerrund	object	conjunction	object		1					
subject	passive		preposition	object		1									

	continuous		on												
subject	verb	pronoun	direct obj	preposition	Indirect obj		4								
subject	verb	direct obj	Preposition	Indirect obj	conjunction	verb		IF	subject	verb	object		1		
IF	subject	verb	object	(comma)	subject	verb		indirect obj	preposition	pronoun	direct obj		1		
gerrund	preposition	object		2											
IF	subject	verb	object	(comma)	subject	verb		object	infinitive	object		1			
subject	verb	object	conjunction	subject	future	direct obj		preposition	indirect obj	conjunction	verb	direct obj	preposition	indirect obj	1
gerrund	preposition	object	Infinitive		1										
subject	verb	object	conjunction	verb	object	passive		infinitive		1					
subject	verb	direct obj	Preposition	Indirect obj	preposition	indirect obj		preposition	indirect obj	preposition	indirect obj		1		
subject	passive	passive	infinitive		1										
subject	passive	Preposition	indirect obj	preposition	direct obj		1								
subject	passive	conjunction	passive	preposition	object		1								
IF	subject	verb	negative	object	subordinating conjunction		subject	preposition	object	verb	negative	adjective	OR	(continued on next line)	
(continued from previous line)			IF	pronoun	verb	object	THEN	subject	passive	preposition	object		1		
IF	subject	verb	object	subject	future passive	preposition	object	subject	passive	object	infinitive	object		1	
negative	passive	infinitive		1											
IF	subject	auxillary	negative	verb	object	subject	gerrund	object	passive	preposition	object	conjunction	subject	passive	1
subject	modal	adverb	verb	direct obj	preposition	adverb	indirect obj	preposition	pronoun	passive	preposition	object		1	
subject	future	Preposition	direct obj	preposition	indirect obj	conjunction	indirect obj	preposition	direct obj		1				
subject	past	conjunction	pronoun	object	past	object	preposition	object		1					
subject	past	infinitive	object		1										
subject	past	direct obj	Infinitive	indirect obj		1									
subject	*past*	*Preposition*	*object*	*conjunction*	*past*	*direct obj*		*preposition*	*indirect obj*		1				

subject	past	object	conjunction	subject	modal	verb	object		1
subject	past	object	conjunction	past	object				1
subject	past	object	IF	subject	modal	verb	object		1
preposition	object	subject	past	object	preposition	conjunction	past	object	1
subject	past	object	gerrund	preposition	object				1
subject	passive past	object							1
preposition	subject	past	pronoun	past	object	past	pronoun	conjunction past	1
subject	verb	OR	verb	preposition	object				3
subject	verb	object	adverb	preposition	adjective	object			1
subject	verb	determiner 'which'		preposition	object	adverb	verb	object	1
subject	verb	IF	adverb	verb	determiner 'any'	object			1
subject	verb	InDirect Obj	adverb	direct obj	preposition	Indirect obj	direct obj		1
subject	verb	object	conjunction	Indirect obj	conjunction	Indirect obj	preposition	direct obj	2
subject	verb	object	Preposition	adverb	preposition	object			1
subject	verb	determiner 'what'		object	verb				1
subject	verb	object	conjunction	pronoun	preposition	subject	verb	object	1
subject	verb	InDirect Obj	Preposition	adjective	direct obj	preposition	adverb	object	1
subject	verb	negative	auxillary	Infinitive	object	preposition	pronoun	object	preposition object past participle 1
subject	verb	InDirect Obj	direct obj	preposition	Indirect obj	preposition	Indirect obj	pronoun	verb preposition indirect obj direct obj 1
subject	verb	InDirect Obj	direct obj	preposition	pronoun	preposition	Indefinite	pronoun	object 1
object	adverb	subject	verb	direct obj	preposition	Indirect obj			1
subject	passive	adverb	pronoun	pronoun	passive	preposition	object		1
subject	verb	object	conjunct	modal	verb	conjunct	subject	verb	object 2

subject	negative	passive	ion Preposit ion	past particip le	prepositi on	ion object	1				
subject	verb	adverb	1								
subject	object	verb	Preposit ion	object	passive	1					
subject	passive	Preposit ion	pronoun	relative pronoun	passive	prepositi on	object	1			
subject	object	passive	1								
subject	verb	conjunct ion	object	verb	negative	adverb	prepositi on	object	1		
subject	verb	direct obj	prepositi on	indirect obj	prepositi on	subject	verb	passive infinitive	1		
subject	verb	object	conjunct ion	subject	present passive	perfect	1				
subject	verb	object	conjunct ion	verb	object	adverb	prepositi on	object	1		
verb	indirect obj	OR	indirect obj	pronoun	verb	direct obj	1				
verb	subject	verb	object	1							
verb	object	IF	pronoun	verb	indirect obj	OR	indirect obj	prepositi on	direct obj	1	
verb	object	OR	object	1							
verb	indirect obj	Preposit ion	direct obj	pronoun	verb	prepositi on	object	1			
verb	object	OR	verb	infinitive	object	1					
verb	object	IF	pronoun	verb	infinitive	adverb	object	OR	verb	1	
verb	infinitive	adverb	object	1							
verb	object	adverb	1								
subject	present continuous		object	1							
subject	passive	Preposit ion	indirect obj	prepositi on	indirect obj	prepositi on	direct obj	conjunct ion	modal	verb	direct obj	prepositi on	determiner 'many'	object	Continue on next
	continued from previous:		conjunct ion	verb	direct obj	prepositi on	indirect obj	(example given)		1					
determiner 'this'		subject	verb	infinitive	prepositi on	determiner 'several'		object	conjunct ion	verb	adverb	pronoun	verb	object	1
determiner 'each'		Preposit ion	pronoun	object	verb	prepositi on	object	1							
subject	present continuous		adverb	indirect obj	prepositi on	direct obj	1								

subject	conjunction	subject	verb	object	past participle	preposition	gerrund	1
adverb	subject	auxillary	passive infinitive	preposition	direct obj	preposition	indirect obj	1
subject	present perfect		object	gerrund	direct obj	preposition	indirect obj	1
subject	passive	Preposition	adjective	object	(semi-colon)	preposition	determiner 'some'	preposition	pronoun	verb	adverb	object	1
subject	verb	direct obj	Preposition	indirect obj	direct obj	preposition	indirect obj	conjunction	direct obj	preposition	indirect obj	1
subject	adverb	verb	object	conjunction	verb	object	preposition	adverb	subject	verb	object	adverb	gerrund	direct obj	indirect obj	conjunction	indirect obj	1
subject	verb	object	conjunction	preposition	pronoun	object	pronoun	verb	object	conjunction	adverb	object	1					
subject	verb	adverb	object	conjunction	object	verb	adjective	object	1									
subject	verb	object	verb	preposition	object	conjunction	adverb	adverb	object	1								
subject	modal	verb	adjective	object	1													
subject	verb	direct obj	adjective	preposition	indirect obj	1												
preposition	object	subject	verb	adverb	subject	verb	determiner 'such'	adjective	indirect obj	preposition	direct obj	(colon)	object	conjunction	object	1		
subject	verb	direct obj	Preposition	indirect obj	conjunction	adverb	verb	object	1									
subject	verb	Preposition	object	conjunction	pronoun	object	adverb	subject	verb	adjective	object	(object list)	1					
subject	verb	object	pronoun	object	preposition	object	conjunction	object	preposition	object	conjunction	object	conjunction		object	conjunction		
	(continued from previous)		object	preposition	object	conjunction	object	1										
subject	verb	infinitive	indirect obj	adverb	direct obj	preposition	gerrund	conjunction	gerrund	direct obj	preposition	Indirect obj	1					
past	direct obj	Preposition	indirect obj	adverb	past	direct obj	preposition	indirect obj	conjunction	pronoun	past	infinitive	passive past continuous		(continued on next line)			
	(continued from previous)		adverb	Indirect obj	preposition	direct obj	1											
verb	adverb	pronoun	verb	direct obj	adverb	pronoun	Indirect obj	1										
pronoun	subject	(example given)	list	past	adverb	preposition	verb	infinitive	indirect obj	infinitive	conjunction	pronoun	past	pronoun	object	1		
subject	past	Preposition	pronoun	object	conjunction	past	object	(explanation)	conjunction	past	object	conjunction	past	preposition		object	1	

subject	past	pronoun	object	1												
subject	past	conjunction	pronoun	past	object	adverb	adverb	pronoun	past	indirect obj	preposition	direct obj	verb	object	1	
pronoun	subject	past	Preposition	pronoun	direct obj	past participle	indirect obj	preposition	Indirect obj	(explanation given)		1				
subject	past	direct obj	Preposition	indirect obj	adverb	subject	past	preposition	object	1						
verb	adverb	subject	past	object	conjunction	past	adjective	object	preposition	pronoun	adverb	subject	past	direct obj	preposition	indirect obj	1
adverb	verb	adverb	subject	past	adverb	negative	'never'	past	preposition	object	adverb	subject	past	object	preposition	object	1
subject	verb	Preposition	agent	verb	direct obj	preposition	indirect obj	1								
subject	modal	Preposition	verb	direct obj	preposition	indirect obj	1									
subject	verb	Preposition	object	infinitive	IF	subject	verb	object	1							
IF	subject	verb	object	(comma)	subject	verb	object	adverb	gerrund	object	1					
subject	modal	verb	object	preposition	object	preposition	gerrund	object	adverb	gerrund	object	1				
negative	subject	passive	Preposition	object	1											
subject	modal	verb	object	adverb	preposition	gerrund	object	conjunction	verb	object	1					
IF	subject	verb	object	(comma)	subject	verb	1									
IF	subject	verb	object	(comma)	subject	verb	object	gerrund	subject	preposition	gerrund	preposition	direct obj	preposition	indirect obj	1
verb	preposition	negative	determiner 'more'	subject	passive	infinitive	1									
subject	verb	direct obj	Preposition	determiner 'all'	indirect obj	1										
subject	modal	negative	verb	object	3											
subject	modal	verb	Preposition	direct obj	preposition	indirect obj	3									
subject	verb	object	conjunction	verb	direct obj	preposition	indirect obj	OR	indirect obj	1						
IF	subject	modal	negative	verb	object	(comma)	subject	modal	verb	direct obj	preposition	Indirect obj	1			
IF	subject	passive	adjective	object	(comma)	subject	modal	verb	object	conjunction	verb	object	infinitive	Indirect obj	preposition	direct obj	1

subject	modal	verb	object	7													
subject	modal	verb	preposition	indirect obj	direct obj	1											
IF	subject	verb	preposition	object	(comma)	subject	verb	direct obj	preposition	indirect obj	1						
preposition	subject	verb	direct obj	preposition	indirect obj	conjunction	verb	direct obj	preposition	indirect obj	1						
IF	subject	verb	preposition	object	(comma)	subject	modal	verb	direct obj	preposition	indirect obj	conjunction	verb	direct obj	preposition	indirect obj	1
subject	modal	negative	verb	infinitive	direct obj	preposition	Indirect obj	1									
subject	modal	verb	object	conjunction	subject	verb	direct obj	preposition	indirect obj	1							
verb	4																
subject	verb	IF	subject	verb	object	9											
IF	subject	verb	object	verb	determiner	object	conjunction	preposition	subject	verb	object	1					
subject	preposition	object	passive	preposition	direct obj	preposition	indirect obj	subject	verb	preposition	direct obj	preposition	indirect obj	(continued on next line)			
	(continued from previous)	conjunction	subject	verb	IF	subject	verb	object	2								
IF	subject	verb	direct obj	preposition	indirect obj	(comma)	verb	determiner	object	conjunction	preposition	subject	verb	object	1		
IF	subject	verb	object	preposition	object	preposition	subject	verb	direct obj	preposition	indirect obj	1					
subject	verb	IF	subject	verb	direct obj	preposition	indirect obj	1									
adverb	IF	subject	verb	negative	past participle	preposition	object	subject	verb	direct obj	preposition	indirect obj	1				
THEN	16																
subject	preposition	object	passive	11													
IF	subject	past passive	1														
subject	preposition	object	passive	preposition	indirect obj	preposition	direct obj	6									
IF	subject	past passive	Preposition	indirect obj	preposition	direct obj	3										
determiner	preposition	object	conjunction	preposition	subject	verb	object	2									

IF	subject	verb	object	preposition	object	2						
subject	verb	indirect obj	indirect obj	preposition	object list	1						
subject	verb	object	verb	direct obj	preposition	indirect obj	1					
subject	modal	verb	Preposition	object	3							
THEN	subject	verb	direct obj	preposition	indirect obj	conjunction	subject	verb	direct obj	preposition	indirect obj	1
subject	preposition	object	passive	preposition	object	1						
IF	subject	verb	negative	preposition	object	THEN	1					
subject	verb	determiner 'which'		object	present continuous		infinitive	preposition	object	1		
subject	object	conjunction	subject	object	passive	preposition	object	1				
IF	subject	conjunction	subject	verb	adjective	preposition	object	THEN	1			
subject	verb	adverb	direct obj	preposition	indirect obj	1						
subject	verb	Preposition	indirect obj	preposition	direct obj	passive	infinitive	2				
subject	verb	direct obj	Preposition	indirect obj	conjunction	subject	object	gerrund	object	1		
subject	verb	object	infinitive	IF	subject	verb	determiner 'any'		indirect obj	preposition	direct obj	1
IF	negative	(comma)	subject	verb	direct obj	preposition	indirect obj	conjunction	verb	object	1	
subject	verb	object	verb	direct obj	preposition	indirect obj	conjunction	verb	direct obj	preposition	indirect obj	1
preposition	object	subject	verb	object	conjunction	verb	indirect obj	preposition	indirect obj	preposition	direct obj	1
subject	verb	object	infinitive	IF	subject	object	passive	preposition	determiner 'any'		object	1
IF	negative	(comma)	subject	passive	preposition	object	1					
preposition	subject	passive	object	conjunction	subject	preposition	object	passive	object	1		
subject	verb	direct obj	Preposition	indirect obj	conjunction	preposition	object	subject	verb	preposition	Indirect obj	preposition	direct obj	1
subject	*verb*	*object*	*verb*	*preposition*	*object*	1						

past subject passive 1
subject verb object verb object verb determiner 'all' object verb object 2
negative subject verb indirect obj preposition direct obj 1
what question 1
subject verb determiner 'some' object 1
subject verb direct obj Preposition Indirect obj adverb preposition object passive infinitive 1
IF subject object (comma) verb preposition object 1
IF negative verb object 1
subject verb object OR object 2
subject verb Preposition object conjunction modal verb preposition object 2
subject negative modal verb object 1
subject verb object verb object conjunction verb object 2
subject verb object (hyphen) subject verb preposition object conjunction verb pronoun verb passive infinitive 1
preposition negative gerrund object 1
IF subject verb object subject verb direct obj preposition indirect obj ELSE object 1
object OR object 1
IF subject verb object (comma) object 1
negative subject verb 1
subject verb object conjunction verb passive direct obj preposition indirect obj 1
adverb subject modal infinitive subject verb object conjunction object conjunction verb object 1
subject verb Indirect obj Preposition direct obj conjunction verb object adverb conjunction verb object 1
subject verb object conjunction verb object infinitive 1
preposition object subject verb infinitive object 1
verb subject infinitive object conjunction verb 1
subject verb object object conjunction Indirect obj preposition direct obj 1
subject modal verb OR verb object 1

verb	preposition	object	subject	passive	preposition	preposition	object		1
subject	verb	object	verb	object	conjunction	verb	preposition	object	1
subject	conjunction	subject	passive	1					
subject	preposition	object	verb	1					
preposition	object	determiner	subject	preposition	object	conjunction	object	verb	1
adverb	subject	verb	object	conjunction	object	1			
subject	verb	Preposition	object	subject	preposition	object	1		
subject	passive	Preposition	object	object list	1				
verb	infinitive	object	Preposition	subject	preposition	infinitive	1		
negative	verb	Preposition	direct obj	preposition	indirect obj	(example given)	1		
subject	verb	Preposition	object	infinitive	direct obj	preposition	indirect obj	1	
verb	determiner	object	1						
subject	verb	Preposition	object	conjunction	verb	preposition	object	1	
subject	future passive	Preposition	object	conjunction	verb	preposition	object	1	
verb	direct obj	Preposition	indirect obj	preposition	subject	passive	preposition	object	1
adverb	verb	indirect obj	conjunction	indirect obj	preposition	indirect obj	preposition	direct obj	1
subject	verb	verb	verb	verb	object	1			
verb	preposition	indirect obj	Preposition	indirect obj	preposition	verb	preposition	direct obj	1
gerrund	indirect obj	Preposition	object	subject	past	infinitive	object	gerrund	preposition	object	1
gerrund	indirect obj	Preposition	direct obj	subject	past	object	conjunction	subject	past passive	preposition	object	1
subject	past	indirect obj	Preposition	direct obj	past	subject	preposition	object	infinitive	conjunction	past	subject	preposition	object	(continued on next line)
	(continued from previous line)			conjunction	subject	past passive	preposition	object	1						
adverb	*object*	*infinitive*	*gerrund*	*subject*	*OR*	*object*	*subject*	*past*	1						

subject past passive Preposition object conjunction subject past direct obj preposition Indirect obj 1

adverb preposition subject past object conjunction object 1

subject past object conjunction past object conjunction object modal passive Infinitive conjunction past object 1

subject past Preposition Indirect obj preposition direct obj Infinitive subject preposition object conjunction object 1

adverb subject past subject past Indirect obj preposition direct obj conjunction subject past passive object Infinitive 1

subject past gerrund direct obj preposition Indirect obj past preposition object conjunction object conjunction preposition Indirect obj preposition direct obj 1

subject past passive Preposition object gerrund object Infinitive subject object object conjunction object conjunction object 1

subject past passive Preposition object past conjunction subject past object Infinitive object conjunction verb object 1

adverb subject past subject past passive Infinitive object pronoun object past passive 1

adverb subject past adverb verb preposition object 1

subject past continuous object conjunction gerrund gerrund Indirect obj preposition direct obj IF passive 1

adverb subject past passive OR passive adjective Infinitive subject past passive preposition object OR object gerrund preposition object (continued on next)

(continued from previous line) conjunction past preposition object 1

gerrund object subject past passive 1

preposition direct obj Preposition Indirect obj subject conjunction subject preposition object past passive preposition object 1

subject verb conjunction direct obj preposition Indirect obj conjunction subject auxillary negative verb direct obj preposition Indirect obj 2

subject verb object UNLESS subject verb object adverb object subject verb object 2

IF subject verb subject verb object verb direct obj preposition Indirect obj preposition Indirect obj 1

adverb subject verb object 2

IF subject verb subject verb WHETHER adverb verb direct obj preposition Indirect obj <<include>> use case adverb subject verb Infinitive object 1

IF subject passive subject verb Infinitive object 1

adverb subject verb object verb Indirect obj preposition Indirect obj preposition direct obj conjunction verb preposition object passive Infinitive preposition object 1

subject verb Infinitive direct obj preposition Indirect obj

subject	verb	WHETHER	adverb	verb	direct obj	preposition	indirect obj	1									
IF	object	subject	verb	infinitive	object	1											
adverb	subject	verb	direct obj	preposition	indirect obj	present passive	perfect	gerrund	direct obj	preposition	Indirect obj	1					
subject	verb	Preposition	object	passive	infinitive	preposition	object	1									
adverb	passive	Preposition	gerrund	object	IF	subject	verb	direct obj	preposition	Indirect obj	(continued on next line)						
	(continued from previous line)		adverb	IF	subject	present continuous passive				preposition	object	subject	passive	infinitive	object	1	
past participle	Indirect obj	Preposition	direct obj	subject	verb	indirect obj	preposition	direct obj	preposition	direct obj	preposition	indirect obj	1				
IF	subject	verb	object	subject	verb	object	preposition	object	past	infinitive	preposition	object	1				
IF	subject	verb	object	THEN	subject	verb	object	passive	adverb	subject	verb	negative	1				
IF	subject	verb	negative	object	1												
passive	object	subject	passive	1													
subject	passive	Preposition	object	conjunction	object	2											
subject	passive	Infinitive	passive	3													
subject	preposition	object	passive	adverb	preposition	subject	past	preposition	object	1							
subject	passive	conjunction	modal	passive	infinitive	1											
verb	direct obj	Preposition	Indirect obj	Indirect obj	conjunction	Indirect obj	preposition	direct obj	1								
subject	passive	IF	subject	preposition	object	passive	preposition	object	1								
IF	negative	subject	modal	passive	infinitive	preposition	subject	modal	passive	infinitive	1						
subject	modal	verb	infinitive	conjunction	verb	Indirect obj	indirect obj	conjunction	indirect obj	preposition	direct obj	1					
subject	passive	conjunction	passive	preposition	object	adverb	preposition	subject	past	preposition	object	1					
subject	passive	Preposition	object	adverb	preposition	object	OR	object	conjunction	subject	present perfect	1					
subject	preposition	object	passive	conjunction	subject	modal	passive	infinitive	1								
verb	object	adverb	negative	subject	modal	passive	infinitive	1									
IF	subject	past passive	Preposition	object	THEN	verb	object	1									

subject	verb	pronoun	subject	verb	indirect obj	prepositi on	direct obj		2					
subject	OR	subject	modal	negative	verb	object		1						
subject	modal	passive	infinitive		1									
object	prepositi on	subject	verb		1									
subject	verb	indirect obj	Preposit ion	direct obj	infinitive	prepositi on	object		1					
IF	subject	Preposit ion	object	negative	passive	THEN	verb	prepositi on	object		5			
subject	verb	Preposit ion	object	subject	object		4							
subject	OR	subject	verb	prepositi on	object		3							
subject	OR	subject	verb	direct obj	prepositi on	indirect obj		8						
subject	OR	subject	verb	object		2								
subject	verb	indirect obj	Preposit ion	indirect obj	prepositi on	indirect obj	conjunct ion	direct obj		3				
IF	subject	verb	object	THEN	subject	verb	object	conjunct ion	object	past	prepositi on	object		3
subject	verb	object	past	prepositi on	object	conjunct ion	object		3					
verb	subject	modal	passive infinitive	adjectiv e	prepositi on	object		6						
subject	verb	direct obj	Preposit ion	indirect obj	prepositi on	subject	gerrund	object		1				
IF	subject	present passive	perfect	THEN	subject	verb	direct obj	prepositi on	indirect obj	prepositi on	indirect obj	gerrund	object	1
IF	subject	present passive	perfect	THEN	subject	verb	direct obj	prepositi on	indirect obj		1			
IF	subject	OR	subject	negative modal	verb	object	THEN	verb	object		1			
IF	subject	modal	passive infinitive	THEN	verb	object		1						
IF	subject	Preposit ion	object	verb	THEN	verb	object		1					
IF	subject	negative	past passive	THEN	verb	object		1						
adverb	subject	verb	object	passive	prepositi on	indirect obj	prepositi on	direct obj		1				
subject	adverb	verb	object	pronoun	verb	prepositi on	direct obj	prepositi on	indirect obj					
adverb	determiner 'all'		subject	verb	prepositi	object	subject	verb	object	infinitive	prepositi	object		1

subject	verb	object	Preposition	gerrund	on verb	preposition	object	adverb		1			on				
subject	verb	object	conjunction	subject	past	verb	object		1								
adverb	subject	verb	object	verb	gerrund	passive	infinitive	conjunction	verb		preposition	direct obj	preposition	indirect obj		1	
adverb	subject	verb	gerrund	object	subject	verb	object	verb	indirect obj		preposition	direct obj		1			
subject	verb	object	conjunction	adverb	verb	object	adverb		2								
subject	verb	object	infinitive	subject	preposition	object	adverb	verb	object	OR	object		1				
adverb	subject	verb	object	past	preposition	direct obj	preposition	indirect obj		1							
subject	verb	object	pronoun	verb	preposition	direct obj	preposition	indirect obj		1							
adverb	determiner 'all'		subject	verb	object	subject	verb	object	infinitive	subject	infinitive	object	conjunction	verb	preposition	object	1
subject	verb	object	infinitive	direct obj	preposition	indirect obj	past	preposition	object								
IF	subject	adverb	verb	(comma)	subject	verb	object	infinitive	object		1						
IF	subject	past	Preposition	object	auxillary	negative	verb	preposition	object	subject	verb	preposition	object		1		
subject	verb	indirect obj	Preposition	direct obj	conjunction	subject	verb	object		1							
adverb	subject	present perfect	object	subject	verb	indirect obj	preposition	direct obj		1							
IF	subject	verb	(comma)	verb	object	verb	object	conjunction	verb	direct obj	preposition	indirect obj		4			
IF	subject	past	Preposition	object	present perfect passive (negative with adverb)				subject	verb	object		1				
subject	verb	object	conjunction	object	preposition	gerrund	object		2								
adverb	object	subject	verb	preposition	object	pronoun	subject	verb	infinitive	object	OR	object	infinitive	preposition	object		1
subject	auxillary	verb	object	adverb	object	negative	object	adverb	object	conjunction	object						
preposition	subject	present perfect continuous		object	subject	verb	object	preposition	gerrund		1						
IF	subject	negative auxillary	verb	Indirect obj	direct obj	past	(comma)	subject	verb	object	infinitive	object		1			
subject	*verb*	*Preposition*	*subject*	*verb*	*object*		*17*										

IF	subject	verb	adverb	object	(comma)	subject	future	object	conjunction	object		2		
subject	future	indirect obj	conjunction	indirect obj	preposition	direct obj		2						
subject	future	gerrund	object	past	preposition	subject	passive		1					
subject	future	object		34										
subject	future	object	future	object	adverb	gerrund	conjunction	future	indirect obj	preposition	direct obj		3	
preposition	subject	passive	subject	passive	object	subject	passive	preposition	object	conjunction	subject	verb		3
IF	subject	verb	object	(comma)	subject	future	object	infinitive	object		2			
subject	modal	verb	object	object	OR	object		2						
IF	subject	past	object		2									
IF	subject	past	object	OR	object		1							
subject	future	direct obj	Preposition	indirect obj	passive	infinitive		1						
FOR	subject	passive		1										
subject	future	object	conjunction	object		2								
WHILE	subject	verb	object		3									
adverb	object	Preposition	gerrund	object	subject	modal	verb	object		1				
subject	negative	passive	conjunction	subject	verb		3							
preposition	subject	verb	object	infinitive	object	subject	verb	direct obj	preposition	indirect obj	conjunction	verb	subject	object	1
adverb	subject	passive	Preposition	object	infinitive	object		1						
subject	verb	Preposition	subject	verb	object	infinitive	object		1					
subject	passive	infinitive	object	conjunction	subject	verb		1						
subject	future	object	Preposition	gerrund	indirect obj	preposition	direct obj		3					
subject	future	Preposition	object		1									
IF	subject	past	object	conjunction	object		3							
<<use use case name>>			28											
WHILE	subject	auxillary	negative	object		1								

ELSE	IF	subject	verb	object	THEN	10				
ELSE	IF	subject	verb	object	conjunction	subject	verb	object	THEN	2
end if	14									
end loop	10									
FOR	subject	passive	loop	1						
subject	future	indirect obj	Preposition	indirect obj	preposition	direct obj	2			
subject	verb	object	conjunction	verb	indirect obj	preposition	direct obj	1		
subject	passive	object	subject	passive	preposition	object	conjunction	subject	verb	1
subject	modal	verb	object	adverb	preposition	gerrund	object	1		
subject	verb	Preposition	subject	passive	10					
subject	verb	indirect obj	Preposition	indirect obj	preposition	direct obj	past	preposition	object	2
subject	verb	object	pronoun	modal	verb	adverb	indirect obj	indirect obj	conjunction	direct obj	2
subject	verb	Preposition	subject	Infinitive	object	passive	3			
subject	future	object	conjunction	object	OR	object	preposition	object	2	
subject	future	object	infinitive	2						
subject	future	object	conjunction	subject	verb	5				
subject	verb	object	OR	object	conjunction	object	2			
subject	future	direct obj	Preposition	indirect obj	gerrund	adverb	object	conjunction	object	2
subject	verb	object	conjunction	subject	verb	7				
subject	future	direct obj	Preposition	Indirect obj	Infinitive	Indirect obj	preposition	Indirect obj	preposition	direct obj	4
OTHERWISE	4									
IF	subject	negative	present passive	perfect	THEN	1				
subject	passive	adverb	indirect obj	preposition	negative	indirect obj	preposition	direct obj	1	
subject	verb	object	past	preposition	indirect obj	preposition	direct obj	conjunction	subject	verb	2

FOR	determiner 'all'	direct obj	preposition	indirect obj	loop	1					
IF	subject	verb	object	pronoun	negative	passive	conjunction	negative	passive	THEN	2
FOR	object	loop	2								
verb	object	conjunction	object	preposition	object	1					
IF	subject	passive	object	2							
subject	verb	direct obj	Preposition	indirect obj	conjunction	subject	verb	object	passive	infinitive	preposition	2
subject	verb	Preposition	subject	verb	object	conjunction	subject	present passive	perfect	2	
subject	verb	object	past	object	conjunction	verb	direct obj	preposition	indirect obj	1	
FOR	subject	past	object	(comma)	verb	subject	object	1			
subject	verb	object	past	preposition	direct obj	preposition	indirect obj	1			
subject	verb	Preposition	subject	preposition	object	passive	2				
subject	verb	object	infinitive	object	past	object	1				
FOR	subject	past	2								
FOR	subject	past	object	1							
subject	verb	Preposition	subject	verb	object	passive	infinitive	1			
WHILE	subject	modal	negative	verb	object	loop	1				
subject	verb	direct obj	Preposition	indirect obj	conjunction	subject	verb	1			
adverb	preposition	gerrund	object	subject	modal	verb	object	1			
subject	future	object	infinitive	object	5						
subject	future	indirect obj	Preposition	indirect obj	preposition	direct obj					
IF	subject	negative	present passive	perfect	1						
subject	verb	Preposition	subject	verb	conjunction	object	passive	infinitive	1		
FOR	direct obj	Preposition	indirect obj	loop	1						
subject	verb	indirect obj	Preposition	direct obj	conjunction	negative	verb	1			

B3. Grammar of all descriptions

Grammatical Structures in Use (where there is 'verb' this represents present simple tense - otherwise the tense will be named)
Cases

1

Subject	Verb	InDirect Obj	Prep	Gerrund	Direct Object		
Subject	Verb	Direct Obj	Infinitive	Article	Adjective	Indirect Obj	
Subject	Verb	Preposition	Object				
Subject	Verb	InDirect Obj	Gerrund	Direct Obj			
Subject	Verb	object					
Subject	Verb	Preposition	Indirect Obj	Direct Obj			
Subject	Verb	Adjective	Object				
Subject	Verb	InDirect Obj	Gerrund	Preposition	indirect obj	Preposition	Direct Obj
Subject	Verb	Object					
Preposition	Determiner	object	Past Participle	(For each piece of work requested:) a loop here			
Subject	Verb	Object					
Subject	Verb	Object	Conjunction	Object	Adjective		
Subject	Verb	Object	Conjunction	Object			
Subject	Verb	Object					
Subject	Verb	Object					
Subject	Verb	Object					
Subject	Verb	Object					
Subject	Verb	Direct Obj	Preposition	Indirect Obj	Conjunction	if-clause	Passive
Subject	Verb	Object					
Subject	Verb	Object	Verb	Conjunction	Verb ~	Object	

Exceptional flows are very long sentences with umpteen clauses and conjunctions

2

article	noun	adverb	article	subject	verb	object	(start)
article	subject	verb	article	Indirect Obj	infinitive	article	Direct Obj
article	*subject*	*verb*	*article*	*Object*			
article	*subject*	*verb*	*noun*				

IF	Indirect obj	verb	noun	Article	subject	verb	Direct Obj	Infinitive	article	possessive noun list and conjunction						
IF	Indirect obj	verb	noun	Article	subject	verb	Direct Obj	Infinitive	article	possessive noun list and conjunction						
IF	Indirect obj	verb	noun	Article	subject	verb	Direct Obj	Infinitive	article	possessive noun list and conjunction						
article	subject	verb	noun													
article	subject	verb	Object	Infinitive	article	noun phrases										
article	subject	verb	article	object list												
article	subject	verb	article	Object	(end use case)											
article	subject	verb	Object	adverb	object	verb	adverb									
article	subject	verb	Object	possessive	noun	adverb	article	subject	verb	article	object	possessive nouns	adverb	conjunction	verb	prep noun
Preposition	adjective	Noun	Preposition	noun	article	subject	verb	article	object	Infinite	determiner	countable noun				
article	subject	modal	verb	Article	Indirect obj	Infinitive	article	noun	prep	adjective	uncountable noun					
IF	article	subject	Verb	noun	prepostion	adjective	noun	article	subject	verb	object					
3																
subject	verb	Object	Preposition	Gerrund	Infinitive	article	object	prepos	agent							
subject	verb	Preposition	Object	(loop) A1 requests from A2...												
adjective	object	<<extend>>														
noun	object	verb	passive infinitive	Preposition	Conjunction	Preposition										
article	object	verb	passive infinitive													
noun	preposition	Noun														
IF	subject	passive														
subject	verb	adverb	Preposition	Indirect Obj	prepostion	article	Direct Obj									
IF	subject	verb	adverb	verb	prepostion	Object										
IF	subject	verb	negative	adverb	<<extend>>											
subject	verb	Preposition	Indirect Obj	Article	adverb	direct obj	conjunction	direct obj								
verb	preposition	object														
subject	verb	InDirect Obj	Preposition	Article	Direct Object											
IF	subject	verb	Object		<<include>>		(if)									
object	preposition	Object														
subject	verb	InDirect	direct obj													

adverb	subject	Obj verb	Indirect Obj	Article	Direct Object	conjunction	article	noun	prep	pronoun	verb	passive	phrasal verb	preposition	(else)
IF	subject	negative	Preposition	Article	object	(loop)									
subject	verb	InDirect Obj	Preposition	Direct Obj											
IF	indirect obj	verb	noun	noun(s)	passive	Preposition	Direct Obj	conjunct	indirect obj	passive	prep	adverb	prep	article	adverb	noun
IF	indirect obj	verb	noun	Indirect Obj	passive	infinitive	prep	object	adverb	passive	prep	adverb	prep	article	adverb	noun
subject	adverb	infinite	prep	determiner	object	adverb									
	4														
subject	verb	Object													
subject	verb	Object													
subject	verb	InDirect Obj	Preposition	Direct Obj											
subject	verb	Object													
subject	verb	InDirect Obj	Preposition	Direct Obj	Conjunction	direct obj									
subject	verb	Object	Conjunction	Object											
subject	verb	Object													
subject	verb	InDirect Obj	Gerrund	Direct Obj											
subject	verb	InDirect Obj	adjective	Preposition	Direct Object										
subject	verb	Object													
subject	verb	InDirect Obj	adjective	Preposition	Direct Object										
subject	verb	InDirect Obj	Preposition	Direct Obj list											
subject	verb	object list													
subject	verb	InDirect Obj	Preposition	direct obj											
subject	verb	object													
	if	subject	Verb	Object	or	Object									
		subject	Verb	Object											
	else														
		<<extend> >	verb noun UC name												
subject	*verb*	*object*													
subject	*verb*	*Preposition Object*													

subject	verb	adverb	negative	Object	Adjective	infinitive	object
subject	verb	InDirect Obj	Preposition	Gerrund	Direct Object		
subject	verb	object					
negative	subject	passive					
subject	verb	object					
subject	verb	object					
subject	past	Preposition	Object	object list			
5							
subject	verb	object	Infinitive	object	OR	verb	object
subject	verb	object	OR				
subject	verb	InDirect Obj	possessive of	Indirect Obj	prepostion	direct obj	
subject	verb	object	(if any)				
subject	verb	determiner	'any'	Gerrund	object	(if required)	
subject	verb	determiner	'any'	Gerrund	object		
subject	verb	determiner	'all'	object	passive	conjunction	object
subject	verb	InDirect Obj	Infinitive	Indirect Obj	Direct Object	verb	Infinitive
subject	verb	object	Conjunction	subject	Verb	infinitive	
subject	verb	object					
subject	verb	object list					
subject	verb	determiner	'all'	object	passive	conjunction	object
subject	verb	object	verb	adjective	adverb	past	
subject	verb	object					
subject	verb	direct obj	Preposition	Indirect Obj			
subject	verb	object					
(end of use case)		subject	verb	object			
subject	verb	negative	Object	or	object		
subject	verb	object	infinitive	object			
subject	verb	negative	adjective	Preposition	object	past	
subject	verb	Object	verb	Adjective			
subject	verb	Object	infinitive	prepostion	Object	infinitive	object OR infinitive
subject	verb	Infinitive	Preposition	object			
if	subject	verb	Infinitive	object	Verb	Prepositio	object

						n		
subject	aux verb	negative	verb	Object				
	subject	verb	Object					
object	negative	verb						
	subject	verb	Object					
subject	Verb	InDirect Obj	prep 'by'	direct obj				
	subject	verb	Object					
negative	determiner	'all'	object	passive				
subject	verb	Object	passive	gerrund	conjunction	verb	object	
verb	Preposition	direct obj	Preposition	indirect obj				
negative	determiner	'all'	object	passive				
	subject	verb	Object	passive	gerrund	conjunction	verb	object
	verb	Preposition	direct obj	Preposition	indirect obj			

6

subject	verb	object					
object	verb	passive					
subject	verb	object					
subject	verb	object	conjunction	Object	prepostion	Object	
subject	verb	direct obj	Preposition	Indirect Obj			
subject	verb	object	present perfect	passive			
subject	verb	direct obj		Indirect Obj			
subject	verb	object					
subject	passive	Infinitive	object				
subject	verb	object					
subject	verb	object	past participle				
subject	verb	object	past participle				
subject	verb	object	past participle				
subject	verb	object	past participle				
subject	verb	InDirect Obj	Preposition	direct obj			
subject	verb	past participle	Object	verb	Adjective		

7

subject	*verb*	*object*	*Infinitive*	*direct obj*	*passive Infinitive*	*preposition*	*Indirect obj*

subject	verb	object												
subject	verb	InDirect Obj	Infinitive	direct obj	Conjunction	direct obj	conjunction	direct obj	conjunction	direct obj				
subject	verb	object	conjunction	object										
subject	verb	object												
subject	verb	object	anaphor	verb	Adjective									
subject	verb	object	Infinitive	object list										
subject	verb	object	EITHER	by	object	OR	adverb							
IF	subject	verb	Object	(noun phrase)	THEN	subject	verb	negative	object	passive	adverb	Indirect	adverb	object
subject	verb	object												
IF	subject	verb	Object	subject	Verb	negative	object	THEN						
	subject	verb	Object	Object	Conjunction	Object								
subject	verb	object												
IF	subject	verb	Object	THEN										
	subject	verb	Indirect Obj	Infinitive	Direct Object									
subject	verb	InDirect Obj	Gerrund	direct object										
subject	verb	object	(y/n)											
subject	verb	object												
IF	subject	verb	Object	THEN										
	subject	verb	Preposition	Object										
	subject	verb	Object											
	subject	verb	Object											
IF	subject	verb	Object	THEN										
	subject	verb	Object	(y/n)										
	subject	verb	Object											
	subject	verb	Object	(<<extend>> use case name)										
subject	verb	direct obj	Preposition	object list										
subject	verb	InDirect Obj	adverb	direct obj	Verb	negative	adjective	preposition	object	(<<extend>> use case name)				
subject	verb	object	(<<extend>> use case name)											
long sentence - 4 clauses														
subject	verb	object	(<<extend>> use case name)											
8														
subject	verb	object												
subject	verb	object	conjunction	Object										

subject	verb	object	conjunction	Object						
subject	verb	object								
subject	verb	object	past participle	conjunctio n	determiner 'any'		object list			
subject	verb	object	passive							
subject	verb	direct obj	Preposition	Indirect Obj	gerrund	indirect obj	Infinitive			
subject	verb	object	conjunction	Object	passive Infinitive					
subject	verb	adverb	Object	verb	negative	adjective	prepositi on	article	noun	past participle
subject	verb	InDirect Obj	Preposition	adverb	Direct Object	verb	adjective			
subject	verb	object	Preposition							
	verb	pronoun	Preposition	object	<<extend>> use case		OR			
	phrasal verb	object	conjunction	object	passive					
subject	verb	adverb	Object	conjunctio n	object	past participle	verb	object	past participle	
subject	verb	InDirect Obj	adjective	Prepositio n	Direct Object	or	direct obj			
subject	verb	InDirect Obj	Infinitive	direct obj	Conjunction	direct obj	GOTO			
subject	verb	adverb	Object	verb	negative	adjective	prepositi on	article	noun	past participle
subject	verb	object	Infinitive							
	phrasal verb	InDirect Obj	negative	Prepositio n	Direct Object	OR				
	verb	object	GOTO							
subject	verb	object	passive							
subject	verb	object								
subject	verb	object	passive							
9										
subject	verb	object								
subject	modal verb	article	Indirect Obj	Prepositio n	Direct Object	Infinitive				
subject	modal verb	object	Infinitive							
IF	object	GOTO								
subject	verb	object								
direct obj	past participle	gerrund	Indirect Obj	conjunctio n	Direct Object	passive	prepositi on	Indirect obj	IF	adjective THEN verb

subject	passive	Preposition	object									
subject	verb	InDirect Obj	Preposition	direct obj	Conjunction	verb	direct obj	passive	passive infinitive			
subject	verb	InDirect Obj	Preposition	direct obj	passive							
	IF	determiner 'all'	adjective	GOTO								
	IF	determiner 'all'	OR	determiner 'some'		verb	adjective	subject	verb	object	OR	object
subject	Verb	IF	Object	modal	passive infinitive							
Verb	negative	modal	passive infinitive		GOTO							
	future	passive	GOTO									
subject	passive	object										
long sentence - 6 clauses												
IF	subject	aux negative verb		infinitive	object	<<extend>> use case						
IF	subject	aux negative verb		infinitive	object	conjunction	modal	passive infinitive		THEN	<<extend>> use case	
IF	subject	verb	Object	THEN	subject	passive	<<extend>> use case					
10												
subject	verb	object	passive infinitive									
subject	verb	object	<<include>> use case									
subject	verb	object	passive	<<include>> use case								
			infinitive									
subject	verb	object	<<include>> use case									
subject	verb	object	<<include>> use case									
subject	verb	object	passive infinitive									
IF	object	THEN	subject	verb	object	gerrund	object					
IF	Indirect obj	Preposition	direct object	THEN	<<extend>> use case							
subject	verb	object										
subject	verb	object	verb	Object								
subject	verb	object	verb	Object								
subject	verb	InDirect Obj	Preposition	direct obj								
11												
subject	verb	object										
subject	passive	Preposition	object	Infinitive								
subject	verb	subject	object	adverb	Verb	passive infinitive						
subject	verb	object										

subject	verb	object											
subject	verb	InDirect Obj	Preposition	direct object									
subject	verb	InDirect Obj	conjunction	Indirect Obj	preposition	direct obj							
subject	verb	InDirect Obj	conjunction	Indirect Obj	preposition	direct obj							
subject	verb	InDirect Obj	Preposition	direct object									
subject	verb	InDirect Obj	IF	direct object	passive	preposition	verb						
subject	verb	InDirect Obj	adverb	passive	preposition	direct obj	<<extend>> use case						
subject	verb	InDirect Obj	Preposition	direct object									
subject	verb	object	Infinitive	Object	<<extend>> use case		OR						
	subject	verb	Object	<<extend>> use case									
IF	subject	verb	Object	conjunction	subject	passive	object	subject	passive	preposition	subject	possessive	infinitive
subject	verb	InDirect Obj	past participle	Preposition	Direct Object	<<extend>> use case							
subject	verb	object	OR										
	subject	verb	Object										
subject	verb	object											
subject	verb	object	passive infinitive	IF	subject	present continuous							
12													
subject	verb	direct obj	Preposition	Indirect Obj									
subject	verb	object	<<include>> use case										
subject	verb	direct obj	Preposition	Indirect Obj	preposition	indirect obj							
IF	subject	verb	Object	GOTO									
IF	subject	negative	verb	Indirect Obj	Direct Object	modal	passive	preposition	Indirect obj	e.g. noun list			
IF	object	verb	adjective	GOTO									
ELSE	<<extend>> use case												
subject	verb	object	passive	<<extend>> use case									
subject	*verb*	*direct obj*	*Preposition*	*Indirect Obj*									
subject	*verb*	*direct obj*	*Preposition*	*Indirect Obj*	GOTO								
IF	*subject*	*verb*	*Object*	*GOTO*									

subject	verb	object	negative	adjective									
subject	verb	InDirect Obj	Preposition	direct obj	negative	adjective	infinitive	adverb	direct obj	verb	<<extend>> use case		
subject	verb	direct obj	Preposition	Indirect Obj									
subject	verb	InDirect Obj	Infinitive	direct obj	preposition	Indirect obj							
subject	verb	object											
subject	verb	object											
subject	verb	object	subject	passive	preposition								
subject	verb	object											
IF	subject	verb	Indirect Obj	direct obj	future infinitive	WHETHER	direct obj	future passive	preposition	Indirect ob	OR	indirect obj	<<extend>> use case
IF	subject	verb	Object	subject	Verb	object	pronoun	passive	infinitive				
IF	preposition	noun	subject	future	object	GOTO							
IF	Indirect obj	subject	future	direct obj	GOTO								
subject	future	object	IF	subject	future	object							
	IF	object	<<extend>> use case										
		GOTO											
subject	future	direct obj	Preposition	Indirect Obj	list of <<extend>> use cases								
subject	future	direct obj	Preposition	Indirect Obj									
subject	future	InDirect Obj	direct object										
subject	future	InDirect Obj	Preposition	direct object	adverb	past participle	<<extend>> use case						
subject	future	Preposition	Object	conjunction	Verb	pronoun	verb	adverb	present perfect		indirect obj	preposition	direct object
subject	future	Preposition	Object	IF	adjective								
subject	verb												
13													
subject	verb	object											
subject	verb	object	passive infinitive	Preposition									
subject	verb	object	passive infinitive	Preposition									
subject	verb	object	passive infinitive	Preposition									
subject	verb	object	passive infinitive	Preposition									

subject	verb	object												
subject	verb	object												
subject	verb	object	<<include>> use case											
subject	verb	object	GOTO											
subject	verb	object												
subject	verb	object												
subject	verb	object												
IF	subject	verb	Object	THEN	subject	verb	object							
subject	verb	object												
IF	subject	verb	Object	THEN	subject	verb	object							
IF	subject	verb	Object	THEN	subject	verb	direct obj	preposition	indirect obj					
IF	subject	verb	Object	THEN	subject	verb	direct obj	preposition	indirect obj					
IF	subject	verb	Object	THEN	subject	verb	object							
subject	verb	InDirect Obj	Preposition	direct obj	gerrund	object								
END														
subject	verb	negative	adjective	Preposition	object									
IF	subject	verb	Object	THEN	subject	verb	direct obj	preposition	indirect obj					
GOTO	ELSE													
subject	verb	negative	past participle	Preposition	object									
	14													
subject	passive	infinitive	Object											
preposition	adjective	noun	subject	passive	preposition	object								
subject	verb	Preposition	Object	Preposition	gerrund	object								
subject	verb	object												
subject	passive	Preposition	Object											
subject	passive	infinitive	Indirect Obj	Preposition	Direct Object									
subject	verb	infinitive	WHETHER	object	Verb	adjective	OR	adjective						
subject	passive	Preposition	Indirect Obj	Preposition	WHETHER	OR	negative	direct obj	future infinitive	OR	WHETHER	determiner 'any'	direct obj	passive
subject	*passive*	*Preposition*	*direct object*	*adverb*	*modal*	*passive*	*preposition*	*Indirect obj*						

determiner 'all'	subject	Preposition	object	conjunction	object	passive	
15							
subject	verb	Preposition	gerrund	Preposition	Direct Object	prepositio n	Indirect obj
subject	verb	object	Infinitive	object			
subject	verb	Preposition	object				
subject	verb	InDirect Obj	gerrund	direct obj			
subject	verb	object					
subject	verb	object list					
subject	verb	object					
subject	verb	InDirect Obj	gerrund	direct obj			
IF	subject	verb	object	(comma)	subject	verb	object
subject	verb	object					
subject	verb	object					
FOR	object	past participle					
	subject	verb	object				
	subject	verb	object list				
	subject	verb	object list				
	subject	verb	object				
	subject	verb	direct object	Preposition	Indirect obj	prepositio n	Indirect obj
subject	verb	object					
subject	verb	object					
subject	verb	object	conjunction	IF	subject	passive	
subject	verb	object					
subject	verb	conjunction	verb	object			
subject	verb	object					
	object	modal	passive	Preposition			
	subject	passive infinitive	Preposition	object	passive		
subject	verb	object					
	<<extend>> use case						
object	negative	adjective					
	IF	subject	verb	object	(comma)	subject	verb object
	IF	subject	verb	object	(comma)	subject	verb object
	IF	subject	verb	object			
		subject	verb	object			
		subject	verb	object			
		subject	verb	object			
	subject	verb	object				

	subject	verb	object													
	subject	verb	object	conjunction	Verb											
object	negative	adjective														
	IF	subject	verb	object	(comma)	subject	verb	object								
	IF	subject	verb	object	(comma)	subject	verb	object								
	IF	subject	verb	object												
		subject	verb	object												
		subject	verb	object												
		subject	verb	object												
	subject	verb	object													
	subject	verb	object													
	subject	verb	object	conjunction	Verb											
	IF	direct obj	verb	negative	adjective	preposition	past participle		indirect obj	THEN	pronoun	passive	prepo	direct obj	verb	prepos	indirect obj
subject	verb	infinitive	object	Preposition	determiner 'any'	object										
	subject	verb	object													
	subject	verb	object													

16

subject	verb	object						
subject	verb	object						
subject	verb	InDirect Obj	Preposition	direct obj	conjunction	direct obj	conjunction	direct obj
subject	verb	object	conjunction	object	conjunction	object		
subject	verb	object						
subject	verb	InDirect Obj	Preposition	direct obj				
subject	verb	object						
subject	verb	object						
subject	verb	object						
subject	verb	Preposition	object					
subject	verb	object						
subject	verb	object	conjunction	verb	object			
subject	verb	object	verb	negative	adjective			
subject	verb	object						
subject	*verb*	*object*						
subject	*verb*	*object*	*past participle*					
subject	*verb*	*object*	*past participle*					

subject	verb	object	past participle					
subject	verb	object	past participle					
subject	verb	object	adverb					
subject	verb	object						
subject	verb	object						
subject	verb	object						
subject	verb	object						

17

subject	verb	object	Infinitive	object	passive infinitive	prepositio n	object
subject	verb	object (anaphorl)					
subject	verb	object					
subject	verb	InDirect Obj	Preposition	direct obj	<<include>> use case		
subject	verb	IF	object	passive			
	object	auxillary	negative	verb	passive infinitive		
	object	future passive					
subject	verb	InDirect Obj	Preposition	direct obj	passive infinitive		
subject	verb	object	Infinitive	object			
subject	verb	object					
subject	verb	object					
	IF	object	THEN	verb	object	gerrund	object	<<include>> use case
	IF	object	THEN	verb	conjunction	verb	object	<<include>> use case
subject	verb	object					
subject	verb	object					
subject	verb	object	conjunction	verb	preposition	direct obj	prepositi on	indirect obj
IF	object	negative	adjective	THEN	<<extend>> use case			
IF	subject	auxillary	negative	verb	infinitive	object	THEN	<<extend>> use case
IF	subject	auxillary	negative	verb	infinitive	object	THEN	<<extend>> use case
IF	object	THEN	subject	verb	adjective	<<extend>> use case		

18

subject	verb	object			
IF	subject	verb	object	THEN	<<extend>> use case
IF	subject	verb	object	THEN	<<extend>> use case
IF	subject	verb	object	THEN	<<extend>> use case

subject	verb	object									
subject	verb	object									
subject	verb	object									
subject	verb	object									
subject	verb	object									
IF	subject	verb	object	THEN	subject	verb	object				
subject	verb	object									
subject	verb	object									
IF	direct obj	Preposition	Indirect obj	conjunction	Indirect obj	verb	adjective	adverb	noun phrase	THEN	GOTO
subject	verb	object									
subject	verb	object									
subject	verb	direct obj	verb	adjective	preposition	Indirect obj					
subject	verb	InDirect Obj	Preposition	direct obj							
subject	verb	object									
IF	subject	present continuous		object	THEN	GOTO					
subject	verb	object									
subject	verb	object									
IF	subject	verb	object	THEN	subject	verb	preposition	object	<<extend>> use case		
IF	subject	verb	object	THEN	subject	verb	direct obj	preposition	verb	preposition	Indirect obj
subject	verb	object									
IF	subject	verb	object	THEN	subject	verb	object				
IF	subject	verb	object	THEN	GOTO						
IF	subject	verb	object	THEN	END						
subject	verb	object									
IF	subject	verb	object	THEN	<<extend>> use case						
subject	verb	object									
IF	subject	verb	object	THEN	subject	verb	object				
IF	subject	verb	object	THEN	subject	verb	object				
subject	verb	Preposition	object								
END											
subject	*verb*	*negative*	*adjective*	*Infinitive*							
subject	*verb*	*object*	*adverb*								

subject	auxiliary	negative	past participle	Prepositio n	object	GOTO			
subject	verb	negative	adjective	Prepositio n	object				
IF	subject	verb	object	THEN	subject	verb	object		
IF	subject	verb	object	THEN	GOTO				
END									
19									
subject	verb	object	IF	object	<<extend>> use case				
subject	verb	object	(example given)	<<extend>> use case					
subject	verb	direct obj	Preposition	indirect obj	conjunction	verb	indirect obj	prepositi on	direct obj
IF	adjective	object	GOTO						
subject	verb	InDirect Obj	Preposition	direct obj					
subject	verb	object							
subject	verb	object	(example given)						
subject	direct obj	passive	Preposition	indirect obj	<<extend>> use case				
subject	verb	direct obj	gerrund	indirect obj	preposition	gerrund	IF	passive	
subject	verb	InDirect Obj	Preposition	direct obj	(example given)				
subject	verb	object							
subject	verb	Preposition	object						
subject	verb	object	passive	Prepositio n	object				
subject	verb	object							
subject	verb	object	conjunction	verb	object				
subject	verb								
subject	verb	InDirect Obj	Preposition	direct obj					
subject	prepositio n	gerrund	object						
negative	direct obj	Preposition	indirect obj						
20									
verb	object	infinitive	object						
verb	object	conjunction	verb	object					
verb	indirect obj	Preposition	direct obj						
verb	object	infinitive	object	OR					

	GOTO				
subject	verb	conjunction	verb	object	
subject	verb	object	conjunction	verb	object
subject	passive	object	passive		
subject	passive	IF	adjective		
subject	passive	object	passive		
subject	passive	Preposition	object		
IF	subject	negative	passive	<<extend>> use case	
subject	passive	object	passive		
subject	passive				
subject	passive				
subject	passive				
verb	object				

21

subject	verb	Preposition	object		
subject	verb	object	gerrund		
subject	verb	object	Infinitive		
subject	verb	object	GOTO		
subject	verb	object	passive infinitive	<<extend>> use case	
subject	verb	object			
subject	verb				
subject	verb	InDirect Obj	Preposition	direct obj	
subject	verb	object	gerrund	GOTO	
subject	verb	object			
subject	verb	object	<<extend>> use case		
subject	verb	Preposition	object	<<extend>> use case	
subject	verb	object	<<extend>> use case		
subject	verb	object			
subject	verb	InDirect Obj	direct obj		
subject	verb				
negative	*object*	*passive*	GOTO		
subject	verb	<<extend>> use case			
subject	*auxillary*	*negative*	*verb*	*Infinitive*	*object*	*<<extend>> use case*

22

subject	verb	InDirect Obj	Preposition	direct obj								
subject	verb	InDirect Obj	conjunction	Indirect obj	preposition	direct obj	GOTO					
IF	negative	object	subject	modal	verb	object	preposition	verb				
IF	subject	negative	modal	verb	(comma)	subject	verb	object				
subject	verb	GOTO										
subject	verb	object	gerrund	subject	verb	adverb	adverb	verb	negative	adjective	object	
subject	verb	InDirect Obj	Preposition	direct obj								
subject	verb	InDirect Obj	Preposition	direct obj								
subject	verb	object list										
subject	verb	object										
IF	subject	verb	direct obj	Preposition	Indirect obj							
	subject	verb	object	GOTO								
IF	object	subject	verb	object								
	subject	verb	Preposition	Indirect obj	preposition	direct obj						
	IF	object	subject	verb	object	GOTO						
	IF	object	GOTO									
	IF	object	<<extend>> use case									
subject	verb	InDirect Obj	Preposition	direct obj								
	IF	object	GOTO									
	IF	object	subject	verb	Indirect obj	preposition	direct obj					
subject	verb	object										
subject	verb	InDirect Obj	Preposition	direct obj								
subject	verb											
23												
subject	verb	direct obj	Preposition	indirect obj								
IF	subject	verb	indirect object	Preposition	direct obj	noun	THEN	subject	verb	indirect obj	conjunction	indirect obj	direct obj	verb	preposition	object
subject	verb	direct obj	Preposition	indirect obj	conjunction	subject	gerrund	object								
subject	verb	object	Infinitive	object	GOTO											
negative	subject	passive														
negative	determiner	'more'	subject	object	infinitive	GOTO										
subject	verb	object														

subject	verb	object										
negative	object	GOTO										
subject	verb	object	conjunction	verb	object	GOTO						
subject	verb	object	GOTO									
subject	verb	object	conjunction	object								
subject	verb	InDirect Obj	direct obj									
negative	subject	passive										
subject	verb	direct obj	Preposition	Indirect obj	GOTO							
subject	verb	object	conjunction	verb	object	IF	adjective <<extend>> use case					
24												
subject	verb	InDirect Obj	Preposition	direct obj								
subject	verb	object										
subject	verb	InDirect Obj	Preposition	direct obj	(explanation given)							
subject	verb	object	IF	subject	verb	object						
subject	verb	object	conjunction	verb	object							
subject	verb	direct obj	Preposition	indirect obj	preposition	indirect obj						
GOTO												
subject	verb	object										
subject	verb	object	past participle	adverb	object	preposition	adverb	object	conjunction	indirect obj	preposition	direct obj
subject	verb	EITHER										
	infinitive	object	past participle	OR								
	verb	object										
subject	EITHER											
	verb	object	Infinitive	OR								
	verb	object	<<extend>> use case									
subject	verb	Preposition	object									
subject	EITHER											
	verb	object	past participle	conjunction	verb	object						
subject	verb	direct obj	Preposition	indirect obj	conjunction	verb	direct obj	preposition	indirect obj			
subject	verb	object										
subject	verb	object	conjunction	verb	object							
subject	verb	direct obj	Preposition	Indirect obj	passive infinitive	indirect obj	conjunction	indirect obj				
subject	*verb*	*object*	*conjunction*	*verb*	*preposition*	*object*						

subject	verb										
subject	verb	object	conjunction	object							
GOTO											
subject	auxillary	negative	verb	object							
subject	verb	object	verb	infinitive	object	OR	GOTO				
subject	verb	infinitive	object								
subject	verb	object	subject	verb	object	conjunctio n	object	infinitive			

25

subject	verb	direct obj	Preposition	pronoun	indirect obj						
subject	verb	direct obj	Preposition	indirect obj							
subject	passive	Preposition	object	infinitive	subject	infinitive	preposition	object			
adverb	subject	verb	object	subject	verb	Indirect obj	preposition	Indirect obj	preposition	direct obj	
IF	subject	negative	verb	adverb	(comma)	subject	verb	object	adverb		
IF	subject	negative	verb	(comma)	subject	verb	Indirect obj	prepositi on	noun	prepositi on	direct obj
subject	verb	object									
subject	verb	object	GOTO								
prepositio n	determiner 'all'	direct obj	Preposition	indirect obj	passive	(comma)	subject	verb	direct obj	prepositi on	indirect obj
subject	verb	object									
subject	verb	InDirect Obj	Preposition	direct obj							
subject	verb	object list									
subject	verb	Preposition	object	IF	subject	verb	object				
OTHERWI SE	subject	verb	Preposition	object	IF	subject	verb	object			
OTHERWI SE	subject	verb	Preposition	object	IF	subject	verb	object			
subject	passive	verb	Infinitive	direct obj	preposition	indirect obj					
subject	passive	conjunction	passive	Preposition	subject	preposition	object				
IF	subject	present perfect	object	(comma)	subject	verb	object				
ELSE	GOTO										

26

subject	verb	InDirect Obj	Preposition	direct obj

subject	verb	InDirect Obj	Preposition	direct obj								
subject	verb	object	GOTO									
subject	verb	negative	object	<<extend>> use case								
subject	verb	object										
subject	verb	GOTO										
subject	verb	GOTO										
subject	verb	object										
subject	verb	object										
subject	verb	object	GOTO									
subject	verb	object	<<extend>> use case									
subject	verb	object										
subject	verb	object	passive infinitive									
subject	verb	object										
subject	verb											
subject	verb	object										
subject	verb	object										
subject	verb	object										
subject	verb	Preposition	object	Preposition	adverb	adjective	verb	OR	subject	preposition	pronoun	object
subject	verb	object										
subject	verb	object	past participle									
subject	verb	object										
subject	verb	object	Infinitive	object								
subject	verb	InDirect Obj	Preposition	direct obj								
subject	verb	object										
27												
subject	verb	object										
OR	GOTO	direct obj	Preposition	indirect obj								
subject	verb	object	GOTO	loop								
subject	verb	object										
subject	verb	object										
subject	verb	object										
subject	verb	object	conjunction	object								
subject	verb	object	Infinitive									
subject	EITHER	verb	object									

	subject	verb	object	Infinitive			
	subject	verb	direct obj	Preposition	indirect obj	conjunction	indirect obj
OR	subject	verb	direct obj	Preposition	indirect obj		
subject	verb	direct obj	Preposition	indirect obj			
subject	verb	direct obj	Preposition	indirect obj			
subject	verb	direct obj	Preposition	indirect obj			
subject	verb	object	GOTO	loop			
subject	verb	direct obj	Preposition	indirect obj			
subject	modal auxillary	verb					
verb	object						
subject	verb	direct obj	Preposition	indirect obj			
verb	object						
subject	negative	modal auxillary	verb				
verb	object						
subject	verb	object					
subject	verb	object					
GOTO	object	OR	object				

28

subject	verb	object	Infinitive	object			
subject	verb	object					
subject	verb	object					
subject	verb	InDirect Obj	Preposition	indirect obj	preposition	direct object	
subject	verb	object					
subject	verb	direct obj	Preposition	indirect obj			
subject	verb	object					
subject	verb	object					
subject	verb	object					
subject	verb	direct obj	Preposition	indirect obj			
subject	verb	object	Infinitive	object			

29

subject	verb	object	Infinitive	subject	preposition	object
subject	verb					
loop	subject	passive infinitive				

	do								
		subject	verb	object					
		subject	verb	object	conjunction	object			
subject	verb	object	Infinitive	object					
subject	verb	object	conjunction	verb	preposition	object			
subject	verb	object							
IF	object								
THEN	subject	verb	object						
subject	verb	object	conjunction	verb	object				
subject	verb	direct obj	Preposition	Indirect obj	verb	object	conjunction	verb	object
IF	object								
THEN	subject	verb	object	conjunction	object				
subject	verb	object							
subject	verb	direct obj	Preposition	Indirect obj	conjunction	verb	object		
IF	object								
THEN	subject	verb	object						
subject	verb	object	past participle						
IF	subject	verb	object	THEN	subject	verb	object		
subject	verb	object							
subject	verb	object							
subject	verb	object							
subject	verb	object							
subject	verb	object							
30									
subject	verb	Preposition	object						
subject	verb	object							
subject	auxillary	negative	verb	object					
	GOTO								
subject	adjective	infinitive	object						
subject	verb	object							
verb	object								
continue	*IF*	*object*	THEN	object	*preposition*				
	subject	*passive*							
subject	*verb*	*InDirect Obj*	*Preposition*	*direct obj*					

subject	verb	Preposition	object					
subject	verb	object						
verb	object							
verb	object							
verb	object							
subject	verb	direct obj	verb	direct obj	preposition	indirect obj		
subject	verb	Preposition	object					
subject	verb	object	IF	object				
continue	subject	verb	object					
IF	object	THEN	GOTO					
subject	verb	object						
subject	verb	object						
GOTO								
31								
IF	object							
	subject	verb	object					
	subject	verb	direct obj	adjective	preposition	indirect obj		
ELSE	subject	verb	direct obj	adjective	preposition	indirect obj		
verb	prepositio n	determiner 'all'	subject	past participle				
switch	subject	verb	direct obj	Prepositio n	indirect obj			
	subject	verb	direct obj	Prepositio n	indirect obj			
	subject	verb	direct obj	Prepositio n	indirect obj			
	subject	verb	object	infinitive	preposition	indirect obj	prepositi on	direct obj
subject	verb	object						
subject	verb	object						
subject	verb	object						
subject	verb	object						
switch	object							
	object							
		subject	verb	WHETHE R	object	conjunctio n	object	
	object							
		subject	verb	WHETHE	object			

				R						
	object									
		subject	verb	WHETHE R	object	conjunctio n	object			
subject	verb	object	passive Infinitive							
subject	verb	WHETHER	adjective	subject	passive infinitive	prepositio n	object			
subject	verb	object								
subject	verb	direct obj	Preposition	indirect obj	gerrund	object				
32										
subject	verb	object	Infinitive	subject	preposition	object				
IF	subject	verb	negative	object	(comma)	subject	verb	infinitive	object	GOTO
IF	subject	verb	object	(comma)	subject	verb	object			
subject	verb	infinitive	object							
subject	verb	object	Infinitive	object	OR	verb	object			
IF	subject	verb	Infinitive	object						
	subject	verb	object							
	subject	verb	object	Infinitive	subject	verb	object			
	GOTO									
IF	subject	verb	Infinitive	object						
	subject	verb	object							
subject	verb	object	IF	adverb	verb	determiner 'any'		object	passive infinitive	
subject	verb	IF	subject	verb	passive infinitive					
subject	verb	infinitive	object							
	subject	verb	object							
	subject	verb	object	Infinitive	subject	verb	object			
	GOTO									
subject	verb	negative	Infinitive	object						
subject	verb	object								
subject	verb	object								
GOTO										
subject	verb	object								
subject	verb	object								
subject	verb	object	passive							
subject	verb	passive	verb	gerrund						
		Infinitive								
subject	*verb*	*direct obj*	*Preposition*	*indirect obj*						

subject	verb	direct obj	Preposition	Indirect obj			
subject	verb	direct obj	Preposition	Indirect obj			
subject	auxillary	Infinitive	Preposition	object	conjunction	verb	object
subject	verb	object					
subject	verb	Preposition	object				
subject	verb	object					
subject	verb	object					
subject	verb	object	conjunction	subject	verb	object	
subject	verb	object	Infinitive				
subject	verb	object	Infinitive				
subject	negative auxillary	modal	verb				
subject	verb	Infinitive	object				
subject	verb	object					
subject	verb	object	Infinitive	object	past participle		
subject	verb	object	IF	determiner 'any'	object	passive	
subject	verb	IF	object	passive			
subject	verb	object	passive				
subject	verb	object	Infinitive	object			
subject	verb	object					
subject	verb	object	past participle				
GOTO							
subject	verb	negative	object	passive			
subject	verb	direct obj	Preposition	Indirect obj			
subject	verb	object	Infinitive	object	(example list given)		
subject	verb	object					
subject	verb	InDirect Obj	direct obj				
subject	verb	object					
subject	verb	object					
subject	verb	object					
subject	verb	object					
GOTO							
subject	verb	object					
subject	verb	object	Infinitive	direct obj	preposition	Indirect obj	
subject	verb	object	Infinitive	object			

subject	verb	object										
subject	verb	object										
subject	verb	object										
GOTO												
subject	verb	object	Infinitive	object								
subject	verb	object										
subject	verb	Infinitive	object									
subject	verb	object										
subject	verb	Infinitive	object									
subject	verb	Infinitive	object									
subject	verb	object										
subject	verb	object	Infinitive	subject	past participle							
subject	negative modal auxillary		verb									
subject	verb	Infinitive	object									
subject	verb	object										
subject	verb	object	past participle									

33 Kulak & Guiney 2000 (Nearly all of K&G's alternative/exception paths are long single sentences with multiple clauses)

subject	verb	object										
subject	verb	Preposition	gerrund	object	conjunction	gerrund	subject	object				
subject	verb	object	Infinitive									
subject	verb	Preposition	gerrund	pronoun	direct obj	preposition	indirect obj	conjunction	gerrund	pronoun	preposition	verb
subject	verb	object										
subject	verb	InDirect Obj	Preposition	gerrund	direct obj							
subject	verb	Preposition	gerrund	object	adverb	verb	subject	verb	object			
subject	verb	object	conjunction	verb	object	preposition	object					
subject	verb	Preposition	gerrund	object	conjunction	subject	object					
subject	verb	Preposition	indirect obj	preposition	direct obj							
subject	verb	Preposition	gerrund	object	conjunction	subject	object					
subject	conjunction	subject	verb	preposition	object							
subject	*verb*	*Preposition*	*gerrund*	*object*								
subject	*verb*	*object*	*passive*									

subject	verb	Preposition	gerrund	object													
subject	conjunction	subject	verb	preposition	object												
subject	verb	Preposition	gerrund	indirect obj	preposition	direct obj											
subject	conjunction	subject	verb	preposition	object												
subject	verb	Preposition	gerrund	object													
34 Kulak & Guiney 2000																	
subject	verb	Preposition	direct obj	preposition	indirect obj												
subject	verb	direct obj	conjunction	direct obj	preposition	indirect obj	preposition	object									
subject	verb	direct obj	Preposition	indirect obj	conjunction	indirect obj	conjunction	verb	object	preposition	direct obj	modal auxillary	verb	preposition	direc to	prep	object
subject	verb object list	Preposition	object														
subject	verb	object	conjunction	subject	verb	preposition	object										
subject	verb	Preposition	gerrund	direct obj	preposition	indirect obj	conjunction	indirect obj									
preposition	subject	verb	object	subject	verb	pronoun	preposition	pronoun	object								
35 Kulak & Guiney 2000																	
subject	verb	Preposition	object	verb	infinitive	object											
subject	verb	Preposition	gerrund	indirect obj	preposition	direct obj	preposition	gerrund	indirect ob	conjunction	indirect ob	(example list given)					
subject	verb	direct obj	Preposition	indirect obj	preposition	indirect obj	conjunction	verb	indirect ob	adverb	direct obj	modal auxillary	present perfect				
preposition	object	IF	negative	subject	past passive	preposition	object	(comma)	pronoun	object	auxillary	negative	verb				
36 Kulak & Guiney 2000																	
subject	verb	Preposition	indirect obj	conjunction	indirect obj	present perfect		preposition	direct obj								
subject	verb	object	passive														
subject	verb	Preposition	gerrund	direct obj	preposition	indirect obj											
subject	verb	object	adverb	adjective	infinitive	object											
subject	verb	Preposition	gerrund	direct obj	preposition	indirect obj	conjunction	indirect obj									
subject	verb	direct obj	Preposition	indirect obj													
subject	verb	Preposition	gerrund	direct obj	preposition	indirect obj											

subject	verb	object											
subject	verb	Preposition	gerrund	direct obj	preposition	indirect obj	conjunction	Indirect obj					
subject	verb	object											
subject	verb	Preposition	gerrund	object	conjunction	verb	direct obj	preposition	indirect ob				

37 Kulak & Guiney 2000

subject	conjunction	subject	verb	object	verb	adjective							
subject	verb	Preposition	gerrund	direct obj	direct obj	direct obj	conjunction	direct obj	conjunction	subject	verb	adjective	infinitive
subject	verb	object											
subject	verb	Preposition	gerrund	object	infinitive	object	conjunction	verb	pronoun	object			
subject	verb	direct obj	Preposition	indirect obj	conjunction	subject	verb	direct obj	preposition	indirect obj	preposition	indirect obj	
subject	auxillary	negative	verb	object									
subject	verb	InDirect Obj	indirect obj	prepositio n	direct obj								
subject	verb	object	conjunction	subject	verb	pronoun	passive	infinitive		preposition	object		
subject	verb	Preposition	object										
subject	auxillary	adverb	past participle	(passive)									
subject	verb	Preposition	gerrund	object	conjunction	gerrund	indirect obj	indirect obj		preposition	direct obj		
subject	verb	pronoun	object										
subject	verb	Preposition	gerrund	conjunctio n	subject	verb	object						

38 Kulak & Guiney 2000

subject	verb	Preposition	indirect obj	verb	direct obj								
subject	verb	Preposition	direct obj	prepositio n	Indirect obj	prepositio n	object	indirect obj	conjunction	indirect obj			
subject	verb	adjective	object										
subject	verb	Preposition	gerrund	direct obj	preposition	indirect obj							
subject	verb	Preposition	gerrund	object	conjunction	gerrund	indirect obj	direct obj	verb	preposition	pronoun	object	OR object
subject	verb	Preposition	gerrund	object									

39 Kulak & Guiney 2000

subject	*verb*	*object*											
subject	*verb*	*Preposition*	*gerrund*	*object*	*conjunction*	*object*	*conjunction*	*preposition*	*gerrund*	*indirect obj*	*preposition*	*direct obj*	*verb*

subject	verb	Preposition	object	verb	object	conjunction	object	conjunction	verb	direct obj	preposition	Indirect obj
subject	verb	Preposition	gerrund	object	infinitive	direct obj	preposition	Indirect obj	past participle		preposition	object
subject	verb	object										
subject	verb	Preposition	gerrund	object								
preposition	subject	verb	Preposition	object	subject	verb	object	conjunction	object			
subject	verb	Preposition	gerrund	direct obj	preposition	Indirect obj	conjunction	object				
subject	verb	conjunction	verb	object	conjunction	preposition	pronoun	verb	object	verb	object	
subject	verb	Preposition	gerrund	direct obj	OR	direct obj	preposition	Indirect obj	past participle	object		
preposition	subject	verb	Preposition	object	subject	verb	object					
subject	verb	Preposition	gerrund	object								
subject	verb	object	past participle	preposition	object							
subject	verb	Preposition	object									
subject	verb	Preposition	gerrund	object	infinitive	preposition	object	past participle				

40 Kulak & Guiney 2000

subject	verb	Preposition	object	verb	gerrund	object	conjunction	object	
	(optional)	subject	verb	Indirect obj	preposition	direct obj	passive		
subject	verb	Preposition	gerrund	object					

41 Jacobson 1995

subject	passive continuous		preposition	object						
subject	verb	pronoun	direct obj	preposition	Indirect obj					
subject	verb	direct obj	Preposition	Indirect obj	conjunction	verb	IF	subject	verb	object
IF	subject	verb	object	(comma)	subject	verb	Indirect obj	preposition	pronoun	direct obj
gerrund	preposition	object								
subject	verb	pronoun	object							
IF	subject	verb	object	(comma)	subject	verb	object	infinitive	object	
gerrund	preposition	object								

subject	verb	object	conjunction	subject	future	direct obj	preposition	Indirect obj	conjunction	verb	direct obj	preposition	Indirect obj

subject	verb	passive	future passive	preposition	object									
gerrund	preposition	object	Infinitive											
subject	verb	object	conjunction	verb	object	passive infinitive								
subject	verb	direct obj	Preposition	indirect obj	preposition	indirect obj	preposition	indirect obj	preposition	indirect obj				
subject	passive	passive infinitive												
subject	passive	Preposition	indirect obj	preposition	direct obj									
subject	passive	conjunction	passive	preposition	object									
subject	passive	Preposition	object											
subject	verb	negative	object											
IF	subject	verb	negative	object	subordinating conjunction		subject	prepositi on	object	verb	negative	adjective	OR	(continued on next line)
(continued from previous line)			IF	pronoun	verb	object	THEN	subject	passive	prepositi on	object			
IF	subject	verb	object	subject	future passive	preposition	object	subject	passive	object	infinitive	object		
negative	passive	infinitive												
IF	subject	auxillary	negative	verb	object	subject	gerrund	object	passive	prepositi on	object	conjuncti on	subj ect	passive
object														
subject	modal	adverb	verb	direct obj	preposition	adverb	indirect obj	prepositi on	pronoun	passive	prepositi on	object		
subject	future	Preposition	direct obj	preposition	indirect obj	conjunction	indirect obj	prepositi on	direct obj					

41 Robertson 1995

subject	past	conjunction	pronoun	object	past	object	prepositi on	object
subject	past	infinitive	object					
subject	past	direct obj	Infinitive	indirect obj				
subject	past	Preposition	object	conjunction	past	direct obj	prepositi on	indirect obj
subject	past	object	conjunction	subject	modal	verb	object	
subject	past	object	conjunction	past	object			
subject	past	object	IF	subject	modal	verb	object	
preposition	object	subject	past	object	preposition	conjunction	past	object

subject	past	object	gerrund	preposition	object			
subject	passive past	object						
preposition	subject	past	pronoun	past	object	past	pronoun	conjunction past

42 Britton and Doake, 2000

subject	verb	OR	verb	preposition	object			
subject	verb	object	adverb	preposition	adjective	object		
subject	verb	determiner 'which'		preposition	object	adverb	verb	object
subject	verb	IF	adverb	verb	determiner 'any'	object	(explanation given why)	
subject	verb	InDirect Obj	adverb	direct obj	preposition	indirect obj	direct obj	
subject	verb	object						
subject	verb	Preposition	object					
subject	verb	object	conjunction	indirect obj	conjunction	indirect obj	preposition	direct obj

43 Britton and Doake, 2000

subject	verb	OR	verb	preposition	object			
subject	verb	object	Preposition	adverb	preposition	object		
subject	verb	determiner 'what'		object	verb			
subject	verb	object	conjunction	pronoun	preposition	subject	verb	object
subject	verb	object						
subject	verb	object						
subject	verb	object	conjunction	indirect obj	conjunction	indirect obj	preposition	direct obj

44 Britton and Doake, 2000

subject	verb	OR	verb	preposition	object							
subject	verb	InDirect Obj	Preposition	adjective	direct obj	preposition	adverb	object				
subject	verb	negative	auxillary	infinitive	object	preposition	pronoun	object	preposition	object	past participle	
subject	verb	InDirect Obj	direct obj	preposition	indirect obj	preposition	indirect obj	pronoun	verb	preposition	indirect obj	direct obj
subject	verb	object										
subject	verb	InDirect	direct obj	preposition	pronoun	preposition	indefinite	pronoun	object			

		Obj		n		n			
subject	verb								
object	adverb	subject	verb	direct obj	preposition	indirect obj			

45 Britton and Doake, 2000

subject	verb	Preposition	indirect obj	prepositio n	direct obj				
subject	verb	pronoun	direct obj	prepositio n	indirect obj				
subject	passive	adverb	pronoun	pronoun	passive	prepositio n	object		
subject	passive								
subject	verb	Preposition	object						
subject	passive								

46 Britton and Doake, 2000

subject	verb	object	conjunction	modal	verb	conjunctio n	subject	verb	object
subject	verb	Preposition	indirect obj	prepositio n	direct obj				
subject	object	verb	passive						
subject	verb	pronoun	direct obj	prepositio n	indirect obj				
subject	negative	passive	Preposition	past participle	preposition	object			
subject	passive								
subject	verb	adverb							

47 Britton and Doake, 2000

subject	verb	object	conjunction	modal	verb	conjunctio n	subject	verb	object
subject	verb	Preposition	indirect obj	prepositio n	direct obj				
subject	object	verb	Preposition	object	passive				
subject	verb	pronoun	direct obj	prepositio n	indirect obj				
subject	passive	Preposition	pronoun	relative pronoun	passive	prepositio n	object		
subject	object	passive							
subject	passive								
subject	verb	Preposition	object						
subject	object	verb	passive						
subject	passive								

48 Britton and Doake, 2000

subject	passive	infinitive	object						
subject	verb	conjunction	object	verb	negative	adverb	preposition	object	
subject	verb	direct obj	preposition	indirect obj	preposition	subject	verb	passive	infinitive
subject	passive	Preposition	object						
subject	verb	object	conjunction	subject	present perfect passive				

49 Britton and Doake, 2000

subject	passive	infinitive	object						
subject	verb	object	conjunction	verb	object	adverb	preposition	object	
subject	verb	InDirect Obj	preposition	direct obj					

50 Wiegers, 1999

verb	indirect obj	OR	indirect obj	pronoun	verb	direct obj			
verb	subject	verb	object						
verb	object	IF	pronoun	verb	indirect obj	OR	indirect obj	preposition	direct obj
verb	object	OR	object						
verb	indirect obj	Preposition	direct obj	pronoun	verb	preposition	object		
subject	verb	infinitive	indirect obj	preposition	determiner 'any'	direct obj			
verb	object	OR	verb	infinitive	object				
verb	object								
verb	object	IF	pronoun	verb	infinitive	adverb	object	OR	verb
verb	infinitive	adverb	object						
verb	object	adverb							

51 Fowler, 1997

subject	present continuous		object												
subject	passive	Preposition	indirect obj	preposition	indirect obj	preposition	direct obj	conjunction	modal	verb	direct obj	preposition	determiner 'many'		object	Continue on next
	continued from previous:		conjunction	verb	direct obj	preposition	indirect obj	(example given)							
determiner 'this'		subject	verb	infinitive	preposition	determiner 'several'		object	conjunction	verb	adverb	pronoun	verb	object	
determiner 'each'		Preposition	pronoun	object	verb	preposition	object								

52 Fowler, 1997

subject	present continuous		adverb	indirect obj	preposition	direct obj												
subject	conjunctio n	subject	verb	object	past participle	prepositio n	gerrund											
adverb	subject	auxillary	passive infinitive	prepositio n	direct obj	prepositio n	indirect obj											

53 Carroll, 1995

subject	present perfect		object	gerrund	direct obj	prepositio n	indirect obj											
subject	verb	object																
subject	passive	Preposition	adjective	object	(semi-colon)	prepositio n	determiner 'some'		prepositi on	pronoun	verb	adverb	object					
subject	verb	direct obj	Preposition	indirect obj	direct obj	prepositio n	indirect obj	conjuncti on	direct obj	prepositi on	indirect obj							
subject	adverb	verb	object	conjunctio n	verb	object	prepositi on	adverb	subject	verb	object	adverb	gerr und	direc t obj	indirect obj	conjunction	Indirect obj	
subject	verb	object	conjunction	prepositio n	pronoun	object	pronoun	verb	object	conjunctl on	adverb	object						
subject	verb	adverb	object	conjunctio n	object	verb	adjective	object										
subject	verb	object	verb	prepositio n	object	conjunctio n	adverb	adverb	object									
subject	modal	verb	adjective	object	(example given)													
subject	verb	direct obj	adjective	prepositio n	indirect obj													
prepositio n	object	subject	verb	adverb	subject	verb	determiner 'such'		adjective	indirect obj	prepositi on	direct obj	(colo n)	obje ct	conjunct ion	obje ct		
subject	verb	direct obj	Preposition	indirect obj	conjunction	adverb	verb	object										
subject	verb	Preposition	object	conjunctio n	pronoun	object	adverb	subject	verb	adjective	object	(object list)						
subject	verb	object	pronoun	object	preposition	object	conjunctl on	object	prepositi on	object	conjunctl on	object	conjunctio n	object	conjunction			
	(continued from previous)		object	prepositio n	object	conjunctio n	object											
subject	verb	infinitive	indirect obj	adverb	direct obj	prepositio n	gerrund	conjuncti on	gerrund	direct obj	prepositi on	Indirect obj						

54 Nielsen, 1995

past	direct obj	Preposition	indirect obj	adverb	past	direct obj	prepositi on	Indirect obj	conjuncti on	pronoun	past	infinitive	passive past continuous		(continued on next line)			
	(continued from previous)		adverb	Indirect obj	preposition	direct obj												
verb	adverb	pronoun	verb	direct obj	adverb	pronoun	Indirect obj											
pronoun	*subject*	*(example list given)*		*past*	*adverb*	*prepositio n*	*verb*	*infinitive*	*indirect obj*	*infinitive*	*conjuncti on*	*pronoun*	*past*	*pron oun*	*object*			
subject	*past*	*Preposition*	*pronoun*	*object*	*conjunction*	*past*	*object*	*(explanat*	*conjuncti*	*past*	*object*	*conjuncti*	*past*	*preposition*		*obje*		

								lon)	on		on			ct	
subject	past	pronoun	object												
subject	past	conjunction	pronoun	past	object	adverb	adverb	pronoun	past	indirect obj	preposition	direct obj	verb	object	
pronoun	subject	past	Preposition	pronoun	direct obj	past participle	indirect obj	preposition	indirect obj	(explanation given)					
subject	past	direct obj	Preposition	indirect obj	adverb	subject	past	preposition	object						
verb	adverb	subject	past	object	conjunction	past	adjective	object	preposition	pronoun	adverb	subject	past	direct obj	preposition	indirect obj
adverb	verb	adverb	subject	past	adverb	negative 'never'		past	preposition	object	adverb	subject	past	object	preposition	object

55 Booch et al, 1999

subject	verb	Preposition	agent	verb	direct obj	preposition	indirect obj								
subject	modal	Preposition	verb	direct obj	preposition	indirect obj									
subject	verb	object	Preposition	gerrund	object										
subject	verb	Preposition	object	infinitive	IF	subject	verb	object							
IF	subject	verb	object	(comma)	subject	verb	object	adverb	gerrund	object					
subject	modal	verb	object	preposition	object	preposition	gerrund	object	adverb	gerrund	object				
negative	subject	passive	Preposition	object											
subject	modal	verb	object	adverb	preposition	gerrund	object	conjunction	verb	object					
IF	subject	verb	object	(comma)	subject	verb									
IF	subject	verb	object	(comma)	subject	verb	object	gerrund	subject	preposition	gerrund	preposition	direct obj	preposition	indirect obj

56 CREWS, 1999 (Achour and Maiden)

subject	verb	object			
subject	verb	direct obj	Preposition	indirect obj	
IF	subject	verb	object	THEN	
verb	preposition	negative	determiner 'more'	subject	passive infinitive
subject	verb	object	gerrund	object	
IF	subject	passive			
subject	verb	object	conjunction	object	
subject	verb	direct obj	Preposition	indirect obj	
subject	verb	object			
subject	verb	direct obj	Preposition	indirect obj	
subject	verb	direct obj	Preposition	indirect obj	

subject	verb	direct obj	Preposition	indirect obj												
subject	verb	object														
subject	verb	object														
subject	verb	direct obj	Preposition	indirect obj												
subject	verb	direct obj	Preposition	determiner 'all'	indirect obj											
subject	modal	negative	verb	object												
subject	modal	negative	verb	object												
subject	modal	verb	Preposition	direct obj	preposition	indirect obj										
subject	modal	negative	verb	object	(examples given)											
subject	verb	direct obj	Preposition	indirect obj												
subject	verb	object	conjunction	verb	direct obj	preposition	indirect obj	OR	indirect obj							
subject	verb	object	infinitive	object												
IF	subject	modal	negative	verb	object	(comma)	subject	modal	verb	direct obj	preposition	indirect obj				
IF	subject	passive	adjective	object	(comma)	subject	modal	verb	object	conjunction	verb	object	infinitive	indirect obj	preposition	direct obj
subject	modal	verb	object													
subject	modal	verb	direct obj	preposition	indirect obj											
subject	modal	verb	preposition	indirect obj	direct obj											
IF	subject	verb	preposition	object	(comma)	subject	verb	direct obj	preposition	indirect obj						
preposition	subject	verb	direct obj	preposition	indirect obj	conjunction	verb	direct obj	preposition	indirect obj						
IF	subject	verb	preposition	object	(comma)	subject	modal	verb	direct obj	preposition	indirect obj	conjunction	verb	direct obj	preposition	indirect obj
subject	modal	negative	verb	infinitive	direct obj	preposition	indirect obj									
subject	modal	verb	object	conjunction	subject	verb	direct obj	preposition	indirect obj							
subject	modal	verb	object													

57 Graham, 1998 (task scripts)

verb	object		
verb	direct obj	Preposition	indirect obj
verb	object		
verb	object		
verb	object		
verb	object		
verb			

verb

58 Rolland and Achour, 1998

subject	verb	direct obj	Preposition	Indirect obj									
subject	verb	IF	subject	verb	object								
IF	subject	verb	object	verb	determiner	object	conjunction	preposition	subject	verb	object		
subject	preposition	object	passive	preposition	direct obj	preposition	indirect obj	subject	verb	preposition	direct obj	preposition	indirect obj (continued on next line)
	(continued from previous)	conjunction	subject	verb	IF	subject	verb	object					
IF	subject	verb	direct obj	preposition	indirect obj	(comma)	verb	determiner	object	conjunction	preposition	subject	verb object
subject	preposition	object	passive	preposition	direct obj	preposition	indirect obj	subject	verb	preposition	direct obj	preposition	indirect obj (continued on next line)
	(continued from previous)	conjunction	subject	verb	IF	subject	verb	object					
IF	subject	verb	object	preposition	object	preposition	subject	verb	direct obj	preposition	indirect obj		
subject	verb	object	IF	subject	verb	object							
subject	verb	IF	subject	verb	direct obj	preposition	indirect obj						
adverb	IF	subject	verb	negative	past participle	preposition	object	subject	verb	direct obj	preposition	indirect obj	

59 Rolland and Achour, 1998

subject	verb	direct obj	Preposition	Indirect obj	
subject	verb	IF	subject	verb	object
IF	subject	verb	object		
THEN					
subject	preposition	object	passive		
subject	verb	object			
IF	subject	verb	object		
THEN					
subject	preposition	object	passive		
subject	verb	object			
IF	subject	verb	object		
THEN					

subject	verb	object					
IF	subject	past passive					
THEN							
subject	passive						
subject	verb	object					

60 Rolland and Achour, 1998

subject	verb	direct obj	Preposition	Indirect obj			
subject	verb	IF	subject	verb	object		
IF	subject	verb	object				
THEN							
subject	prepositio n	object	passive	prepositio n	Indirect obj	prepositio n	direct obj
subject	verb	direct obj	Preposition	Indirect obj			
IF	subject	verb	object				
THEN							
subject	prepositio n	object	passive	prepositio n	Indirect obj	prepositio n	direct obj
subject	verb	direct obj	Preposition	Indirect obj			
IF	subject	verb	object				
THEN							
subject	verb	direct obj	Preposition	Indirect obj			
IF	subject	past passive	Preposition	Indirect obj	preposition	direct obj	
THEN							
subject	passive	Preposition	object				
subject	verb	direct obj	Preposition	Indirect obj			

61 Rolland and Achour, 1998

subject	verb	direct obj	Preposition	Indirect obj			
subject	verb	IF	subject	verb	object		
IF	subject	verb	object				
THEN							
subject	prepositio n	object	passive	prepositio n	Indirect obj	prepositio n	direct obj
subject	verb	direct obj	Preposition	Indirect obj			
subject	verb	IF	subject	verb	object		
IF	subject	verb	object				
THEN							

subject	preposition	object	passive	preposition	indirect obj	preposition	direct obj
subject	verb	direct obj	Preposition	indirect obj			
subject	verb	IF	subject	verb	object		
IF	subject	verb	object				
THEN							
subject	verb	direct obj	Preposition	indirect obj			
subject	verb	object	IF	subject	verb	object	
subject	verb	direct obj	Preposition	indirect obj			
IF	subject	past passive	Preposition	indirect obj	preposition	direct obj	
THEN							
subject	verb	direct obj	Preposition	indirect obj			
subject	verb	direct obj	Preposition	indirect obj			

62 Rolland and Achour, 1998

subject	verb	direct obj	Preposition	indirect obj			
subject	verb	IF	subject	verb	object		
IF	subject	verb	object				
THEN							
verb							

subject	preposition	object	passive	preposition	indirect obj	preposition	direct obj
subject	verb	direct obj	Preposition	indirect obj			
subject	verb	IF	subject	verb	object		
determiner	preposition	object	conjunction	preposition	subject	verb	object
IF	subject	verb	object	preposition	object		
THEN							
verb							

subject	preposition	object	passive	preposition	indirect obj	preposition	direct obj
subject	verb	direct obj	Preposition	indirect obj			
subject	verb	IF	subject	verb	object		
determiner	preposition	object	conjunction	preposition	subject	verb	object
IF	subject	verb	object	preposition	object		
THEN							

subject	verb	direct obj	Preposition	indirect obj		
subject	verb	object	IF	subject	verb	object
subject	verb	direct obj	Preposition	indirect obj		
IF	subject	past passive	Preposition	indirect obj	preposition	direct obj
THEN						
subject	verb	direct obj	Preposition	indirect obj		
subject	verb	direct obj	Preposition	indirect obj		

63 Achour et al, 1999

subject	verb	Preposition	object			
subject	verb	object list				
subject	verb	indirect obj	indirect obj	preposition	object list	
subject	verb	Preposition	object			
subject	verb	object	verb	direct obj	preposition	indirect obj
subject	verb	direct obj	Preposition	indirect obj		
subject	verb	object				
subject	modal	verb	object			
subject	modal	verb	object			
subject	modal	verb	object			
subject	modal	verb	object			
subject	modal	verb	Preposition	object		
subject	modal	verb	Preposition	object		
subject	modal	verb	Preposition	object		

64 Achour, 1998

subject	verb	direct obj	Preposition	indirect obj							
THEN	subject	verb	direct obj	preposition	indirect obj	conjunction	subject	verb	direct obj	preposition	indirect obj

65 Achour, 1998

subject	preposition	object	passive	preposition	object						
IF	subject	verb	negative	preposition	object	THEN					
subject	verb	object									
subject	verb	determiner 'which'		object	present continuous		infinitive	preposition	object		
subject	*object*	*conjunction*	*subject*	*object*	*passive*	*preposition*	*object*				

IF	subject	conjunction	subject	verb	adjective	preposition	object	THEN
subject	verb	object	passive infinitive					
subject	modal	verb	direct obj	preposition	indirect obj			
subject	verb	adverb	direct obj	preposition	indirect obj			

66 Holbrook, 1990

subject	verb	Preposition	indirect obj	preposition	direct obj	passive infinitive					
subject	verb	direct obj	Preposition	indirect obj	conjunction	subject	object	gerrund	object		
subject	verb	direct obj	Preposition	indirect obj							
subject	verb	object	infinitive	IF	subject	verb	determiner 'any'		indirect obj	preposition	direct obj
IF	negative	(comma)	subject	verb	direct obj	preposition	indirect obj	conjunction	verb	object	
subject	verb	object	verb	direct obj	preposition	indirect obj	conjunction	verb	direct obj	preposition	indirect obj
preposition	object	subject	verb	object	conjunction	verb	indirect obj	preposition	indirect obj	preposition	direct obj

67 Holbrook, 1990

subject	verb	Preposition	indirect obj	preposition	direct obj	passive infinitive					
subject	verb	indirect obj	Preposition	direct obj	conjunction	verb	object				
subject	verb	direct obj	Preposition	indirect obj							
subject	verb	object	infinitive	IF	subject	object	passive	preposition	determiner 'any'		object
IF	negative	(comma)	subject	passive	preposition	object					
preposition	subject	passive	object	conjunction	subject	preposition	object	passive	object		
subject	verb	direct obj	Preposition	Indirect obj	conjunction	preposition	object	subject	verb	preposition	indirect obj	preposition	direct obj

68 Cockburn, 1997

subject	verb	direct obj	Preposition	indirect obj
subject	verb	object	verb	object
subject	verb	object	infinitive	object
subject	verb	object	verb	preposition	object
subject	verb	object		
past	subject	passive		
subject	verb	object		

subject	verb	object							
subject	auxillary	negative	verb	object					
subject	verb	object	verb	object	verb	determiner 'all'	object	verb	object
negative	subject	verb	indirect obj	preposition	direct obj				
what question									
subject	verb	object							
subject	verb	object	verb	object	verb	determiner 'all'	object	verb	object
subject	verb	determiner 'some'		object					
subject	verb	direct obj	Preposition	indirect obj	adverb	preposition	object	passive infinitive	

69 Cox and Phalp, 2000

subject	verb	object	Preposition	gerrund	object							
IF	subject	object	(comma)	verb	preposition	object						
IF	negative	verb	object									
subject	verb	Preposition	object									
subject	verb	object	OR	object								
subject	verb	object										
subject	verb	object										
subject	verb	object										
subject	verb	Preposition	object	conjunction	modal	verb	preposition	object				
subject	negative	modal	verb	object								
subject	verb	object	<<extend>> use case									
subject	verb	object	verb	object	conjunction	verb	object					
subject	verb	object	(hyphen)	subject	verb	preposition	object	conjunction	verb	pronoun	verb	passive infinitive
subject	passive	Preposition	object									
subject	verb	object	<<extend>> use case									

70

<<include>> use case		preposition	negative	gerrund	object						
subject	verb	direct obj	Preposition	indirect obj							
IF	subject	verb	object	THEN	subject	verb	preposition	object			
subject	verb	object	OR	object							
IF	subject	verb	object	subject	verb	direct obj	preposition	indirect obj	ELSE	object	
subject	verb	object									

object											
object	OR	object									
subject	verb	object									
IF	subject	verb	object	(comma)	object						
subject	verb	Indirect obj	Preposition	direct obj							
subject	verb	object									
subject	verb	Indirect obj	Preposition	direct obj							
subject	verb	object	<<include>> use case								
subject	verb	object									
subject	verb	object									
GOTO											
subject	verb	Preposition	object	conjunctio n	modal	verb	prepositi on	object			
subject	verb	object	verb	object	conjunction	verb	object				
negative	subject	verb									
subject	modal	verb	object	<<extend>> use case							

71 Fowler, 2000

subject	verb	object	conjunction	verb	passive	direct obj	prepositi on	Indirect obj			
adverb	subject	modal	infinitive	subject	verb	object	conjuncti on	object	conjuncti on	verb	object
subject	verb	Indirect obj	Preposition	direct obj	conjunction	verb	object	adverb	conjuncti on	verb	object

72 Fowler, 2000

subject	verb	object	conjunction	verb	object	infinitive		
subject	verb	Preposition	object					
subject	verb	Preposition	object	(example given)				
subject	verb	object	gerrund	object				
subject	verb	Preposition	object					
subject	verb	object						
subject	verb	object	adverb					
subject	verb	direct obj	Preposition	Indirect obj				
prepositio n	object	subject	verb	infinitive	object			
verb	subject	infinitive	object	conjunctio n	verb			
subject	verb	object	object	conjunctio n	Indirect obj	prepositio n	direct obj	

subject	modal	verb	OR	verb	object			
verb	preposition	indirect obj	Preposition	direct obj				

73 McGraw and Harbison, 1997

verb	preposition	object	subject	passive	preposition	preposition	object	
subject	verb	object	verb	object	conjunction	verb	preposition	object
subject	conjunction	subject	passive					
subject	verb	object	conjunction	object				
subject	preposition	object	verb					
subject	verb	object						
subject	passive	Preposition	object					
preposition	object	determiner	subject	preposition	object	conjunction	object	verb
adverb	subject	verb	object	conjunction	object			
subject	verb	Preposition	object	subject	preposition	object		
subject	passive	Preposition	object					

74 McGraw and Harbison, 1997

subject	verb	Preposition	object			
subject	verb	Preposition	object			
subject	passive	Preposition	object			
subject	passive	Preposition	object			
subject	passive	Preposition	object			
subject	passive					
subject	verb	object				
subject	verb	indirect obj	Preposition	indirect obj	preposition	direct obj
subject	verb	object				
subject	verb	object				

75 McGraw and Harbison, 1997

subject	verb	direct obj	Preposition	indirect obj	preposition	indirect obj
subject	passive	Preposition	object	object list		
verb	infinitive	object	Preposition	subject	preposition	infinitive
subject	*verb*	*object*				
verb	*direct obj*	*Preposition*	*indirect obj*			

negative	verb	Preposition	direct obj	prepositio n	indirect obj	(example given)								
subject	passive													
subject	verb	object												
subject	verb	Preposition	object	Infinitive	direct obj	prepositio n	indirect obj							
verb	object													
verb	determiner	object												
subject	verb	object	Infinitive	object	(example given)									
subject	verb	object												
subject	verb	Preposition	object	conjunctio n	verb	prepositio n	object							
subject	future passive	Preposition	object	conjunctio n	verb	prepositio n	object							
subject	passive	Preposition	object											
verb	direct obj	Preposition	indirect obj	prepositio n	subject	passive	prepositi on	object						
adverb	verb	indirect obj	conjunction	indirect obj	preposition	indirect obj	prepositi on	direct obj						
subject	verb	verb	verb	verb	object									
verb	prepositio n	indirect obj	Preposition	indirect obj	preposition	verb	prepositi on	direct obj						
76 McGraw and Harbison, 1997														
gerrund	indirect obj	Preposition	object	subject	past	Infinitive	object	gerrund	prepositi on	object				
gerrund	indirect obj	Preposition	direct obj	subject	past	object	conjuncti on	subject	past passive	prepositi on	object			
subject	past	indirect obj	Preposition	direct obj	past	subject	prepositi on	object	infinitive	conjuncti on	past	subject	preposition	object	(continued on next line)
	(continued from previous line)			conjunctio n	subject	past passive	prepositi on	object						
adverb	object	infinitive	gerrund	subject	OR	object	subject	past						
subject	past passive	Preposition	object	conjunctio n	subject	past	direct obj	prepositi on	indirect obj					
adverb	prepositio n	subject	past	object	conjunction	object								
subject	past	object	conjunction	past	object	conjunctio n	object	modal	passive	infinitive	conjuncti on	past	object	
subject	past	Preposition	indirect obj	prepositio n	direct obj	infinitive	subject	prepositi on	object	conjuncti on	object			
adverb	subject	past	subject	past	indirect obj	prepositio n	direct obj	conjuncti on	subject	past passive	object	infinitive		

subject	past	gerrund	direct obj	preposition	indirect obj	past	preposition	object	conjunction	object	conjunction	preposition	indirect obj	preposition	direct obj		
subject	past passive	Preposition	object	gerrund	object	infinitive	subject	object	object	conjunction	object	conjunction	object				
subject	past passive	Preposition	object	past	conjunction	subject	past	object	infinitive	object	conjunction	verb	object				
adverb	subject	past	subject	past passive	infinitive	object	pronoun	object	past passive								
adverb	subject	past	adverb	verb	preposition	object											
subject	past continuous		object	conjunction	gerrund	gerrund	indirect obj	preposition	direct obj	IF	passive						
adverb	subject	past passive	OR	passive	adjective	infinitive	subject	past passive	preposition	object	OR	object	gerrund	preposition	object	(continued on next)	
	(continued from previous line)			conjunction	past	preposition	object										
gerrund	object	subject	past passive														
preposition	direct obj	Preposition	indirect obj	subject	conjunction	subject	preposition	object	past passive	preposition	object						

77 Pooley and Stevens, 1999

subject	verb	object															
subject	verb	conjunction	direct obj	preposition	indirect obj	conjunction	subject	auxillary	negative	verb	direct obj	preposition	indirect obj				
subject	verb	object	UNLESS	subject	verb	object	adverb	object	subject	verb	object						
IF	subject	verb	subject	verb	object	verb	direct obj	preposition	indirect obj	preposition	indirect obj						
adverb	subject	verb	object														

78 Pooley and Stevens, 1999

subject	verb	object															
subject	verb	conjunction	direct obj	preposition	Indirect obj	conjunction	subject	auxillary	negative	verb	direct obj	preposition	Indirect obj				
subject	verb	object	UNLESS	subject	verb	object	adverb	object	subject	verb	object						
IF	subject	verb	subject	verb	WHETHER	adverb	verb	direct obj	preposition	indirect obj	<<include>> use case		adverb	subject	verb	infinitive	object
IF	subject	passive	subject	verb	infinitive	object											
adverb	subject	verb	object	verb	Indirect obj	preposition	indirect obj	preposition	direct obj	conjunction	verb	preposition	object	passive infinitive		preposition	object

79 Pooley and Stevens, 1999

subject	verb	Infinitive	direct obj	preposition	indirect obj				
subject	*verb*	*WHETHER*	*adverb*	*verb*	*direct obj*	*preposition*	*indirect obj*	*<<include>> use case*	
IF	*object*	*subject*	*verb*	*Infinitive*	*object*				

adverb	subject	verb	direct obj	preposition	indirect obj	present perfect passive		gerrund	direct obj	preposition	indirect obj								
subject	verb	Preposition	object	passive infinitive		preposition	object												
adverb	passive	Preposition	gerrund	object	IF	subject	verb	direct obj	preposition	indirect obj		(continued on next line)							
	(continued from previous line)			adverb	IF	subject		present continuous		passive	preposition	object	subject	passive infinitive	object				

80 Pooley and Stevens, 1999

past participle	indirect obj	Preposition	direct obj	subject	verb	indirect obj	preposition	direct obj	preposition	direct obj	preposition	indirect obj							
IF	subject	verb	object	subject	verb	object	preposition	object	past	infinitive	preposition	object							
IF	subject	verb	object	THEN	subject	verb	object	passive	adverb	subject	verb	negative							

81 Eriksson and Penker, 1998

IF	subject	verb	negative	object															
subject	passive																		
subject	preposition	object	passive																
subject	passive																		
subject	verb	object																	
subject	passive																		
IF	subject	verb	object																
subject	passive																		
subject	passive																		
subject	preposition	object	passive																
subject	verb	object																	
subject	passive																		
subject	passive																		

82 Eriksson and Penker, 1998

subject	verb	object																	
subject	passive																		
passive	object	subject	passive																
subject	passive	Preposition	object	conjunction	object														
subject	passive																		

83 Eriksson and Penker, 1998

subject	passive																		

subject	passive								
subject	passive	Preposition	object	conjunctio n	object				

84 Eriksson and Penker, 1998

subject	passive infinitive		passive						
subject	prepositio n	object	passive	adverb	preposition	subject	past	prepositi on	object
subject	passive								
subject	passive infinitive		passive						
subject	(example list)	passive	conjunction	modal	passive infinitive				
subject	passive								
verb	direct obj	Preposition	indirect obj	indirect obj	conjunction	indirect obj	prepositi on	direct obj	

85 Eriksson and Penker, 1998

subject	prepositio n	object	passive						
subject	passive	IF	subject	prepositio n	object	passive	prepositi on	object	
IF	negative	subject	modal	passive infinitive		prepositio n	subject	modal	passive infinitive
subject	prepositio n	object	passive	(example given)					
subject	passive	Preposition	object						
subject	modal	verb	infinitive	conjunctio n	verb	indirect obj	indirect obj	conjuncti on	indirect obj	prepositi on	direct obj
subject	verb	(object list)							

86 Eriksson and Penker, 1998

subject	prepositio n	object	passive								
subject	prepositio n	object	passive								
subject	passive	conjunction	passive	prepositio n	object	adverb	prepositi on	subject	past	prepositi on	object

87 Eriksson and Penker, 1998

subject	prepositio n	object	passive
subject	passive	Preposition	object

88 Eriksson and Penker, 1998

subject	*prepositio n*	*object*	*passive*

subject	passive	Preposition	object	adverb	preposition	object	OR	object	conjunction	subject	present perfect
subject	passive infinitive		passive								
subject	preposition	object	passive	conjunction	subject	modal	passive infinitive				
subject	preposition	object	passive								

89 Eriksson and Penker, 1998

subject	passive						
verb	object	adverb	negative	subject	modal	passive infinitive	
subject	passive						

90 Leite et al 2000

subject	verb	Indirect obj	Preposition	direct obj				
verb	preposition	object						
verb	object							
verb	object							
verb	preposition	object						
IF	subject	past passive	Preposition	object	THEN	verb	object	
subject	verb	pronoun	subject	verb	Indirect obj	preposition	direct obj	
subject	OR	subject	modal	negative	verb	object		
verb	object							
subject	modal	passive infinitive						
verb	object							
object	preposition	subject	verb					
verb	object							

91 Leite et al 2000

subject	verb	Indirect obj	Preposition	direct obj	Infinitive	preposition	object		
IF	subject	Preposition	object	negative	passive	THEN	verb	preposition	object
subject	verb	Preposition	object	subject	object				
verb	object								
subject	verb	object							

subject	OR	subject	verb	prepositio n	object	(object list shown)
subject	verb	object				
verb	object					
subject	OR	subject	verb	direct obj	preposition	indirect obj
subject	OR	subject	verb	object		
subject	OR	subject	verb	object		
subject	OR	subject	verb	direct obj	preposition	indirect obj
verb	prepositio n	indirect obj	Preposition	direct obj		
verb	prepositio n	object				
verb	prepositio n	object				

92 Leite et al 2000

subject	verb	indirect obj	Preposition	direct obj	preposition	indirect obj					
IF	subject	Preposition	object	negative	passive	THEN	verb	prepositi on	object		
subject	verb	indirect obj	Preposition	indirect obj	preposition	indirect obj	conjuncti on	direct obj			
IF	subject	verb	object	THEN	subject	verb	object	conjuncti on	object	past	prepositi on	object
subject	verb	object	past	prepositio n	object	conjunctio n	object				
subject	verb	object	Infinitive								
subject	OR	subject	verb	prepositio n	object	(object list shown)					

93 Leite et al 2000

IF	subject	Preposition	object	negative	passive	THEN	verb	prepositi on	object		
subject	verb	Preposition	object	subject	object						
subject	verb	indirect obj	Preposition	indirect obj	preposition	indirect obj	conjuncti on	direct obj			
IF	subject	verb	object	THEN	subject	verb	object	conjuncti on	object	past	prepositi on	object
subject	verb	object	past	prepositio n	object	conjunctio n	object				
subject	*verb*	*direct obj*	*Preposition*	*indirect obj*							
subject	*verb*	*direct obj*	*Preposition*	*indirect obj*							

94 Leite et al 2000

subject	*verb*	*indirect obj*	*Preposition*	*indirect obj*	*preposition*	*indirect*	*conjuncti*	*direct obj*			

IF	subject	verb	object	THEN	subject	obj verb	on object	conjuncti on	object	past	prepositi on	object
subject	verb	object	past	prepositio n	object	conjunctio n	object					

95 Leite et al 2000

subject	verb	Indirect obj	Preposition	Indirect obj	preposition	direct obj						
IF	subject	Preposition	object	negative	passive	THEN	verb	prepositi on	object			
subject	verb	Preposition	object	subject	object							
verb	object											
subject	verb	object										
subject	OR	subject	verb	prepositio n	object	(object list shown)						

96 Leite et al 2000

IF	subject	Preposition	object	negative	passive	THEN	verb	prepositi on	object			
subject	verb	Preposition	object	subject	object							
verb	object											
subject	verb	direct obj	Preposition	Indirect obj								
subject	verb	direct obj	Preposition	indirect obj								

97 Leite et al 2000

subject	OR	subject	verb	direct obj	preposition	Indirect obj						
verb	subject	modal	passive infinitive	adjective	preposition	object						
subject	OR	subject	verb	direct obj	preposition	Indirect obj						
verb	subject	modal	passive infinitive	adjective	preposition	object						
subject	verb	Indirect obj	Preposition	direct obj								
subject	verb	indirect obj	Preposition	direct obj								

98 Leite et al 2000

subject	OR	subject	verb	direct obj	preposition	Indirect obj						
verb	subject	modal	passive infinitive	adjective	preposition	object						
subject	OR	subject	verb	direct obj	preposition	Indirect obj						
verb	subject	modal	passive infinitive	adjective	preposition	object						
subject	verb	direct obj	Preposition	Indirect obj	preposition	subject	gerrund	object				
subject	verb	direct obj	Preposition	Indirect obj								

subject	verb	indirect obj	Preposition	direct obj										
subject	verb	indirect obj	Preposition	direct obj										

99 Leite et al 2000

subject	OR	subject	verb	direct obj	preposition	indirect obj								
verb	subject	modal	passive infinitive	adjective	preposition	object								
subject	OR	subject	verb	direct obj	preposition	indirect obj								
verb	subject	modal	passive infinitive	adjective	preposition	object								
IF	subject	present perfect passive		THEN	subject	verb	direct obj	preposition	indirect obj	preposition	indirect obj	gerrund	object	
IF	subject	present perfect passive		THEN	subject	verb	direct obj	preposition	indirect obj					
subject	verb	indirect obj	Preposition	direct obj										
subject	verb	indirect obj	Preposition	direct obj										

100 Leite et al 2000

verb	object								
verb	object								
verb	object								
IF	subject	OR	subject	negative modal	verb	object	THEN	verb	object
IF	subject	modal	passive infinitive	THEN	verb	object			
IF	subject	Preposition	object	verb	THEN	verb	object		
IF	subject	negative	past passive	THEN	verb	object			

101 Rosenberg. 1999

subject	verb	object										
adverb	subject	verb	object	passive	preposition	indirect obj	preposition	direct obj				
subject	adverb	verb	object	pronoun	verb	preposition	direct obj	preposition	indirect obj			
adverb	determiner 'all'		subject	verb	preposition	object	subject	verb	object	infinitive	preposition	object

102 Rosenberg. 1999

subject	verb	object	infinitive	direct obj	preposition	indirect obj		
subject	verb	object	Preposition	gerrund	object			
subject	verb	object	Preposition	gerrund	verb	preposition	object	adverb

103 Rosenberg. 1999

subject	*verb*	*object*	*infinitive*	*direct obj*	*preposition*	*indirect obj*

subject	verb	object	conjunction	subject	past	verb	object								
adverb	subject	verb	object	verb	gerrund	passive infinitive		conjunction	verb	preposition	direct obj	preposition	indirect obj		
subject	verb	object	infinitive	direct obj	preposition	indirect obj									
adverb	subject	verb	gerrund	object	subject	verb	object	verb	indirect obj	preposition	direct obj				
subject	verb	object	(example given)	conjunction	adverb	verb	object	adverb							

104 Rosenberg. 1999

subject	verb	object	infinitive	subject	preposition	object	adverb	verb	object	OR	object				
adverb	subject	verb	object												
adverb	subject	verb	object	past	preposition	direct obj	preposition	indirect obj							
subject	verb	object	pronoun	verb	preposition	direct obj	preposition	indirect obj							
adverb	determiner 'all'		subject	verb	object	subject	verb	object	infinitive	subject	infinitive	object	conjunction	verb	preposition object
subject	verb	object	infinitive	direct obj	preposition	indirect obj	past	preposition	object						
IF	subject	adverb	verb	(comma)	subject	verb	object	infinitive	object						
IF	subject	past	Preposition	object	auxillary	negative	verb	preposition	object	subject	verb	preposition	object		
subject	verb	indirect obj	Preposition	direct obj	conjunction	subject	verb	object	(example given)						
adverb	subject	present perfect	object	subject	verb	indirect obj	preposition	direct obj							
IF	subject	verb	(comma)	verb	object	verb	object	conjunction	verb	direct obj	preposition	indirect obj			
IF	subject	past	Preposition	object	present perfect passive (negative with adverb)				subject	verb	object	(invoke use case name - OPEN use case approach)			

105 Rosenberg. 1999

subject	verb	object	infinitive	direct obj	preposition	indirect obj					
subject	verb	object	conjunction	object	(example given)	preposition	gerrund	object			
IF	subject	verb	(comma)	verb	object	verb	object	conjunction	verb	direct obj	preposition	indirect obj

106 Rosenberg. 1999

subject	verb	object	infinitive	direct obj	preposition	indirect obj									
adverb	object	subject	verb	preposition	object	pronoun	subject	verb	infinitive	object	OR	object	infinitive	preposition	object
subject	verb	object	conjunction	object	(example given)	preposition	gerrund	object							
IF	subject	verb	(comma)	verb	object	verb	object	conjunction	verb	direct obj	preposition	indirect obj			

107 Rosenberg. 1999

subject	verb	direct obj	Preposition	Indirect obj								
subject	verb	Indirect obj	Preposition	direct obj	conjunction	verb	Indirect obj	prepositi on	direct obj			
subject	auxillary	verb	object	adverb	object	negative	object	adverb	object	conjuncti on	object	
prepositio n	subject	present perfect continuous		object	subject	verb	object	prepositi on	gerrund			
subject	verb	object	conjunction	adverb	verb	object	adverb					
IF	subject	negative auxillary		verb	indirect obj	direct obj	past	(comma)	subject	verb	object	infinitive object
IF	subject	verb	(comma)	verb	object	verb	object	conjuncti on	verb	direct obj	prepositi on	Indirect obj

108 Schneider and Winters, 1998

subject	verb	Preposition	subject	verb	object							
subject	verb	object	conjunction	object								
IF	subject	verb	adverb	object	(comma)	subject	future	object	conjuncti on	object		
subject	future	direct obj	Preposition	Indirect obj								
subject	future	Indirect obj	conjunction	Indirect obj	preposition	direct obj						
subject	future	gerrund	object	past	preposition	subject	passive					
subject	future	object										
subject	future	object										
subject	future	object	future	object	adverb	gerrund	conjuncti on	future	indirect obj	prepositi on	direct obj	
prepositio n	subject	passive	subject	passive	object	subject	passive	prepositi on	object	conjuncti on	subject	verb
IF	subject	verb	object	(comma)	subject	future	object	infinitive	object			

109 Schneider and Winters, 1998

subject	verb	object	passive	prepositio n	object							
subject	verb	direct obj	Preposition	Indirect obj								

110 Schneider and Winters, 1998

subject	modal	verb	object	object	OR	object						
subject	future	object										
IF	subject	past	object									
subject	future	object										
IF	subject	past	object	OR	object							
subject	future	direct obj	Preposition	Indirect obj								
subject	future	direct obj	Preposition	Indirect obj								

subject future object

111 Schneider and Winters, 1998

subject	verb	Preposition	subject	verb	object
subject	verb	object	conjunction	object	
subject	future	direct obj	Preposition	indirect obj	passive infinitive
FOR	subject	passive			
subject	future	object	conjunction	object	
subject	future	direct obj	Preposition	indirect obj	
END					
subject	future	object			
subject	future	object			

subject	future	object	future	object	adverb	gerrund	conjunction	future	indirect obj	preposition	direct obj	
preposition	subject	passive	subject	passive	object	subject	passive	preposition	object	conjunction	subject	verb

112 Schneider and Winters, 1998

subject	verb	Preposition	subject	verb	object							
subject	verb	object	conjunction	object								
IF	subject	verb	adverb	object	(comma)	subject	future	object	conjunction	object		
WHILE	subject	verb	object									
subject	future	object	conjunction	object								
subject	future	direct obj	Preposition	indirect obj								
END												
subject	future	object										
subject	future	object										

subject	future	object	future	object	adverb	gerrund	conjunction	future	indirect obj	preposition	direct obj	
preposition	subject	passive	subject	passive	object	subject	passive	preposition	object	conjunction	subject	verb

adverb	object	Preposition	gerrund	object	subject	modal	verb	object
subject	negative	passive	conjunction	subject	verb			
IF	subject	verb	object	(comma)	subject	future	object	infinitive	object

113 Schneider and Winters, 1998

preposition	subject	verb	object	infinitive	object	subject	verb	direct obj	preposition	indirect obj	conjunction	verb	subject	object
adverb	subject	passive	Preposition	object	infinitive	object								

114 Schneider and Winters, 1998

subject	verb	Preposition	subject	verb	object	infinitive	object
subject	verb	object					
subject	verb	object					
subject	future	object					
subject	verb	object					
subject	passive	infinitive	object	conjunction	subject	verb	

115 Schneider and Winters, 1998

subject	future	direct obj	Preposition	indirect obj			
subject	future	direct obj	Preposition	indirect obj			
subject	future	object	Preposition	gerrund	indirect obj	preposition	direct obj
subject	future	direct obj	Preposition	indirect obj			

116 Schneider and Winters, 1998

subject	future	object					
subject	future	direct obj	Preposition	indirect obj			
subject	future	object	Preposition	gerrund	indirect obj	prepositio	direct obj
						n	
subject	future	direct obj	Preposition	indirect obj			

117 Schneider and Winters, 1998

subject	future	direct obj	Preposition	indirect obj			
subject	future	direct obj	Preposition	indirect obj			
subject	future	object	Preposition	gerrund	indirect obj	prepositio	direct obj
						n	
subject	future	direct obj	Preposition	indirect obj			
verb	object						
verb	object						

118 Schneider and Winters, 1998

subject	modal	verb	object	object	OR	object
subject	future	Preposition	object			
IF	subject	past	object			
subject	future	object				
IF	subject	past	object	conjunction	object	

subject	future	direct obj	Preposition	indirect obj	
subject	future	direct obj	Preposition	indirect obj	
subject	future	object			

119 Schneider and Winters, 1998

subject verb Infinitive object
<<use use case name>>
IF subject passive
subject passive object
subject passive infinitive object
IF subject passive
subject passive Preposition object

120 Schneider and Winters, 1998

subject verb Preposition subject verb object
subject verb object conjunction object
subject verb object conjunction object
subject verb direct obj Preposition indirect obj
subject verb direct obj Preposition indirect obj
subject passive Preposition object
subject verb

121 Schneider and Winters, 1998

subject verb Preposition subject verb object
subject future object
subject verb object conjunction object
subject future object
subject future object
subject future object
subject future object
WHILE subject auxillary negative object
IF subject verb object THEN
<<use use case name>>
ELSE IF subject verb object THEN
<<use use case name>>
ELSE IF subject verb object THEN
<<use use case name>>
ELSE IF subject verb object THEN
<<use use case name>>
ELSE IF subject verb object THEN
<<use use case name>>
ELSE IF subject verb object THEN
<<use use case name>>

ELSE IF subject verb object conjunction subject verb object THEN
<<use use case name>>
end if
subject future object
end loop
subject verb

122 Schneider and Winters, 1998

subject verb Preposition subject verb object
subject verb object
subject verb object conjunction object
subject verb direct obj Preposition indirect obj passive infinitive
FOR subject passive loop
<<use use case name>>
subject future indirect obj Preposition indirect obj preposition direct obj
end loop
subject verb object
subject verb object
subject verb object conjunction verb indirect obj preposition direct obj
<<use use case name>>
subject passive object subject passive preposition object conjunction subject verb
subject modal verb object adverb preposition gerrund object
subject negative passive conjunction subject verb

123 Schneider and Winters, 1998

subject verb Preposition subject passive
subject verb indirect obj Preposition indirect obj preposition direct obj past preposition object
subject verb object pronoun modal verb adverb indirect obj indirect obj conjunction direct obj
subject verb

124 Schneider and Winters, 1998

subject verb Preposition subject infinitive object passive
subject future object conjunction object OR object preposition object
subject future object
subject verb

125 Schneider and Winters, 1998

subject verb Preposition subject verb object
<<use use case name>>
subject future direct obj Preposition Indirect obj
subject future object Infinitive
subject future object
<<use use case name>>
<<use use case name>>
subject future object conjunction subject verb

126 Schneider and Winters, 1998

subject verb Preposition subject passive
subject verb object OR object conjunction object
subject verb object
IF subject past object conjunctio object
n
subject future direct obj Preposition Indirect obj gerrund adverb object conjuncti object
on
subject future object
end if
subject verb object conjunction subject verb

127 Schneider and Winters, 1998

subject verb Preposition subject Infinitive object passive
IF subject verb object THEN
subject future direct obj Preposition Indirect obj Infinitive Indirect prepositi Indirect prepositi direct obj
obj on obj on
OTHERWISE
subject future direct obj Preposition Indirect obj Infinitive Indirect prepositi Indirect prepositi direct obj
obj on obj on
end if
subject future object conjunction subject verb

128 Schneider and Winters, 1998

subject verb Preposition subject verb object
<<use use case name>>
subject future direct obj Preposition Indirect obj
subject future object
IF subject negative present perfect passive THEN
<<use use case name>>
OTHERWISE

verb object
end if
subject verb

129 Schneider and Winters, 1998

subject verb Preposition subject verb object
<<use use case name>>
subject future object conjunction subject verb

130 Schneider and Winters, 1998

subject verb Preposition subject verb object
subject passive
subject verb object conjunction object
subject verb object
subject passive adverb Indirect obj prepositio n negative Indirect obj prepositi on direct obj
subject verb

131 Schneider and Winters, 1998

subject verb Preposition subject verb object
subject passive
subject verb object
subject verb object
subject verb object past prepositio n Indirect obj prepositio n direct obj conjuncti on subject verb

132 Schneider and Winters, 1998

subject verb Preposition subject verb object
subject passive
subject verb object
subject verb object
subject verb object conjunction subject verb

133 Schneider and Winters, 1998

subject verb Preposition subject verb object
subject verb Indirect obj Preposition direct obj
WHILE subject verb object
subject verb object
FOR determiner 'all' direct obj preposition Indirect obj loop
<<use use case name>>
subject verb direct obj Preposition Indirect obj

IF	subject	verb	object							
subject	verb	object								
subject	verb	direct obj	Preposition	Indirect obj						
subject	passive	object								
end if										
IF	subject	verb	object	pronoun	negative	passive	conjunction	negative	passive	THEN
FOR	object	loop								
<<use use case name>>										
verb	object	conjunction	object	preposition	object					
verb	object									
end loop										
IF	subject	passive	object							
verb	object									
end if										
subject	verb	object								
subject	verb	direct obj	Preposition	Indirect obj	conjunction	Indirect obj				
subject	verb	direct obj	Preposition	Indirect obj	conjunction	subject	verb	object	passive infinitive	preposition
end if										
end loop										
subject	verb									

134 Schneider and Winters, 1998

subject	verb	Preposition	subject	verb	object	conjunction	subject	present perfect passive	
subject	verb	object	past	object	conjunction	verb	direct obj	preposition	Indirect obj
FOR	subject	past	object	(comma)	verb	subject	object		
subject	verb								

135 Schneider and Winters, 1998

subject	verb	Preposition	subject	verb	object			
subject	verb	object						
subject	verb	object	conjunction	object				
subject	verb	Indirect obj	Preposition	Indirect obj	preposition	direct obj		
subject	verb	object	past	preposition	direct obj	preposition	indirect obj	
subject	verb	direct obj	Preposition	Indirect obj				
subject	verb	object	Infinitive	object				

subject verb

136 Schneider and Winters, 1998

subject verb Preposition subject verb object

subject verb indirect obj Preposition indirect obj preposition direct obj past preposition object

subject verb object pronoun modal verb adverb object object conjunction object

subject verb

137 Schneider and Winters, 1998

subject verb Preposition subject preposition object passive

IF subject verb object THEN

subject future indirect obj Preposition direct obj infinitive indirect obj preposition indirect obj preposition direct obj

OTHERWISE

subject future indirect obj Preposition direct obj infinitive indirect obj preposition indirect obj preposition direct obj

end if

subject verb object conjunction subject verb

138 Schneider and Winters, 1998

subject verb Preposition subject preposition object passive

subject verb object infinitive indirect obj preposition direct obj

subject verb object conjunction subject verb

139 Schneider and Winters, 1998

subject verb Preposition subject verb object conjunction subject present perfect passive

subject verb object infinitive object past object

FOR subject past

subject verb object infinitive

FOR subject past object

subject future object

end loop

subject verb object infinitive object

end loop

subject verb

140 Schneider and Winters, 1998

subject verb Preposition subject verb object passive infinitive

subject future object
subject future object
WHILE subject modal negative verb object loop
IF subject verb object THEN
<<use use case name>>
ELSE IF subject verb object THEN
<<use use case name>>
ELSE IF subject verb object THEN
<<use use case name>>
ELSE IF subject verb object THEN
<<use use case name>>
ELSE IF subject verb object THEN
<<use use case name>>
ELSE IF subject verb object THEN
<<use use case name>>
ELSE IF subject verb object conjunction subject verb object THEN
<<use use case name>>
end if
subject future object
end loop
subject verb

141 Schneider and Winters, 1998

subject verb Preposition subject passive
subject verb object
subject verb object conjunction object
subject verb direct obj Preposition indirect obj passive infinitive
FOR subject past
subject verb direct obj Preposition indirect obj
subject verb direct obj Preposition indirect obj
end
subject verb object
subject verb object
subject verb object
subject verb object
subject verb object infinitive object
subject verb object infinitive object

subject verb object
subject verb object infinitive object
subject verb direct obj Preposition Indirect obj conjunction subject verb
adverb preposition gerrund object subject modal verb object
subject negative passive conjunction subject verb

142 Schneider and Winters, 1998

subject verb Preposition subject passive
<<use use case name>>
subject future direct obj Preposition Indirect obj
subject future object infinitive
subject future object
subject future object infinitive object
subject future object infinitive object
subject future object infinitive object
subject future object conjunction subject verb

143 Schneider and Winters, 1998

subject verb Preposition subject passive
subject verb object
subject verb object OR object conjunction object
subject verb object
IF subject past object conjunction object
subject future Indirect obj Preposition Indirect obj preposition direct obj
subject future direct obj Preposition Indirect obj gerrund adverb object conjunction object
subject future object
end if
subject verb Indirect obj Preposition direct obj
subject verb object conjunction subject verb

144 Schneider and Winters, 1998

subject verb Preposition subject passive
<<use use case name>>
subject future direct obj Preposition Indirect obj
subject future object
IF subject negative present perfect passive
subject future object infinitive object

subject future object
subject future object infinitive object
OTHERWISE
verb object
end if
subject verb

145 Schneider and Winters, 1998

subject verb Preposition subject passive
<<use use case name>>
subject future object conjunction subject verb

146 Schneider and Winters, 1998

subject verb Preposition subject passive
subject passive
subject verb object conjunction object
subject verb object
subject verb Indirect obj Preposition direct obj conjunction negative verb
subject verb object infinitive object
subject verb

147 Schneider and Winters, 1998

subject verb Preposition subject infinitive object passive
subject passive
subject verb object
subject verb object
subject verb object past preposition Indirect obj preposition direct obj conjunction subject verb

148 Schneider and Winters, 1998

subject verb Preposition subject passive
subject passive
subject verb object
subject passive
subject verb object
subject verb direct obj Preposition Indirect obj
subject verb object conjunction subject verb

149 Schneider and Winters, 1998

subject verb Preposition subject verb object
subject verb object

subject	verb	indirect obj	Preposition	direct obj	preposition	indirect obj					
WHILE	subject	verb	object								
subject	verb	direct obj	Preposition	indirect obj							
subject	verb	object									
FOR	direct obj	Preposition	indirect obj	loop							
subject	verb	direct obj	Preposition	indirect obj							
subject	verb	direct obj	Preposition	indirect obj							
IF	subject	verb	object								
subject	verb	object									
subject	verb	object	Infinitive	object							
subject	passive	object									
end if											
end loop											
IF	subject	verb	object	pronoun	negative	passive	conjuncti on	negative	passive	THEN	
FOR	object	loop									
subject	verb	direct obj	Preposition	indirect obj							
subject	future	indirect obj	conjunction	indirect obj	preposition	direct obj					
subject	future	object									
end loop											
IF	subject	passive	object								
subject	future	object									
end if											
subject	verb	object									
subject	verb	direct obj	Preposition	indirect obj	conjunction	indirect obj					
subject	verb	direct obj	Preposition	indirect obj	conjunction	subject	verb	object	passive infinitive	preposition	
end if											
subject	verb	object	Infinitive	object							
end loop											
subject	verb										

150 Schneider and Winters, 1998

subject	verb	Preposition	subject	verb	conjunction	object	passive infinitive	
subject	future	object	conjunction	object	OR	object	prepositi on	object
subject	future	object						
subject	verb							

151 Schneider and Winters, 1998

subject	verb	Preposition	subject	verb	object
subject	verb	object	gerrund	object	
subject	verb	Indirect obj	Preposition	direct obj	
subject	verb				

152 Schneider and Winters, 1998

subject	verb	Preposition	subject	passive	
subject	verb	object			
subject	verb	object			
subject	verb	object			
subject	verb	Indirect obj	Preposition	direct obj	
subject	verb	object	conjunction	subject	verb

Appendix C

Pilot Study Materials

C1. Experience Questionnaire

Name:	Male/Female: Age:

Previous systems analysis experience:

Application / System	Industry, Academic or individual experience:	Number of Years:

Previous programming experience:

Application / System	Industry, Academic or individual experience:	Number of Years:

Please list any formal qualification in English Language or English Literature (e.g. GCSE, First Certificate, Degree):

Have you been to a supermarket to shop? (Yes / No)
Have you used an ATM? (Yes / No)

C2. Pre-Test

Imagine that you want to order "The UML User Guide" book over the Internet because you'll have to study the UML for your Masters degree. You are at a computer and have already started the "Omazon.co.uk" application.

Write a use case description for this interaction. You'll have to consider how the book will get to you and also how you'll pay for it. Please work on your own for this exercise.

C3. Phase 4: Reading Comprehension Questions

ATM Task

Please answer the following questions about the use case you read. Please only write on the paper provided and not on the use case itself. Add additional comments if you want to say anything further. There is a blank sheet at the end of the questions if you have further comments about the use case you have read.

Note that you might find some events in the use case description in an order different to what you might expect. Or you might find events that you didn't expect at all. However, please consider that they may still be valid.

1. Please give a brief description of what you think the use case is about. If you find that you don't understand a specific part of the use case then please note where in the use case the problem is.

2. Does the customer ask for receipt? Yes / No

Where does this occur in the use case?

3. Is there any illogical order to the use case? For example, the customer gets their cash before entering their PIN number. Yes / No

If yes, what is in the wrong order and where is this in the use case?

4. Do you have to stop and look back up the description more than one line to find out what the sentence you're reading is referring to? Yes / No

If yes, where in the use case is this?

5. Does the UC use pronouns if there is more than one actor (he, she, it, etc)? Yes / No

Is there any confusion over what the pronouns refer to? Yes / No

If yes, why and where in the use case does this occur?

6. Does there appear to be a logical flow to the order of events, that is, one event naturally follows another? Yes / No

If no, where in the use case does the logical flow get confused?

7. Is the writing style jarring; that is, there is not enough expression to justify each event? Yes / No

If yes, where in the use case does this occur?

8. Is the use case plausible? Yes / No

If no, why not and where in the use case are there problems?

9. Are there any alternative or exception paths described? Yes / No

Are these paths viable in your opinion? Yes / No

If no, why not?

10. If you were to grade the use case, what mark would you give it out of 10? Justify your grade.

Add any additional comments

Phase 4: Reading Comprehension Questions Retail Task

(All questions the same for Retail as for ATM task except)

2. Does the customer pay for their goods? Yes / No

Where does this occur in the use case?

3. Is there any illogical order to the use case? For example, the checkout operator receives payment before entering the price of products into the system. Yes / No

If yes, what is in the wrong order and where is this in the use case?

C4. Pilot Tasks

This section describes the tasks for writing the use case descriptions (phase three of the pilot).

The ATM

A supplier of ATMs is to produce a new cash point for a major bank. A requirements analysis has revealed a number of important functions. A customer must have a valid credit card with an acceptable PIN number. A customer can only withdraw up to £30 at a time. As well as withdrawing cash with or without a receipt, a customer can check their balance, order a statement, make a deposit, change their PIN number and also withdraw a further amount without having to re-enter their card. You, as a customer, decide that you want to check the balance of your account, change your PIN number and withdraw £25 with a receipt. You also have to contend with the problem of entering the wrong PIN number and asking for too much money.

From the contextual information given, you are asked to write a use case description of the *Accessing the ATM* use case.

Retail

A supplier of supermarket checkout machines is to produce a new checkout machine. A requirements analysis has revealed a number of important functions. A checkout operator must use each checkout machine to record purchases and receive payments from customers. All regular customers have a club card that allows them to save on their bill. Customers can pay by credit card, cash or cheque. You, as the checkout operator, have to record product purchases, record club card points and record payment by credit card, cheque or cash. You also have to contend with the problem that a product has no barcode and a payment problem that only the store manager can resolve.

From the contextual information given, you are asked to write a use case description of the *Purchase Products* use case.

End of pilot task

To keep participants occupied for the hour an end of pilot task was given. This is to act as a relief against the pressure of participants leaving the location early. The task:

Draw the use case diagram for the whole (ATM / Retail) system.

C5. Data Sets for Counts of Rule / Guideline Use

		Count of usage/misusage of rules/guidelines																									
CP Rules		A1		A2		A3		A4		A5		A6		B1		B2		B3		B4		B5		B6		**Total**	
		Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	**Ct**	**%**
#Events Incl. Alternatives		27	n/a	31	n/a	50	n/a	45	N/a	46	n/a	17	n/a	13	n/a	10	n/a	38	n/a	5	n/a	30	n/a	25	n/a	**337**	**28.08**
Style 1	Alts in main	0	0	0	0	0	0	0	0	1	2.17	0	0	6	46.15	1	10.00	3	7.89	1	20.00	0	0	2	8.00	**14**	**6.25**
	~ sep var*	1	n/a	1	n/a	1	n/a	1	N/a	1	n/a	1	n/a	0	n/a	1	n/a	1	n/a	0	n/a	1	n/a	0	n/a	**n/a**	**n/a**
2	Pronouns	0	0	4	12.90	3	6.00	9	20.00	2	4.35	0	0	1	7.69	4	40.00	1	2.63	0	0	0	0	2	8.00	**26**	**7.72**
3	Adverbs	0	0	1	3.23	0	0	0	0	0	0	0	0	0	0	1	10.00	2	5.26	0	0	1	3.33	4	16.00	**9**	**2.67**
3	Adjectives	4	14.81	1	3.23	2	4.00	9	20.00	9	19.57	0	0	0	0	1	10.00	0	0	1	20.00	2	6.67	0	0	**29**	**8.61**
4	Negatives	1	3.70	2	6.45	0	0	2	4.44	0	0	4	23.53	4	30.77	1	10.00	4	10.53	1	20.00	2	6.67	3	12.00	**24**	**7.12**
5	Explanations	0	n/a	4	12.90	1	2.00	4	8.89	2	4.35	1	5.88	0	0	0	0	3	7.89	0	0	0	0	12	48.00	**27**	**8.01**
6	~present	0	0	9	29.03	0	0	2	4.44	0	0	0	0	0	0	3	30.00	2	5.26	1	20.00	0	0	1	4.00	**18**	**5.34**
*7	Global	1	3.70	7	22.58	3	6.00	4	8.89	2	4.35	5	29.41	2	15.38	2	20.00	3	7.89	1	20.00	3	10.00	6	24.00	**39**	**11.57**
*7	Local	4	14.81	9	29.03	4	8.00	14	31.11	16	34.78	6	35.29	9	69.23	5	50.00	14	36.84	4	80.00	16	53.33	8	32.00	**109**	**32.34**
8	Responses	3	11.11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	20.00	0	0	0	0	**4**	**1.19**
^9	Underline UC names	0	0	0	0	7	14.00	0	0	0	0	3	17.65	0	0	0	0	0	0	2	40.00	0	0	2	8.00	**14**	**4.15**
Structure 1		20	74.07	13	41.94	42	84.00	26	57.78	36	56.52	5	29.41	0	0	2	20.00	11	28.95	0	0	13	43.33	3	12.00	**171**	**50.74**
2		2	7.41	2	6.45	3	6.00	0	0	4	8.70	2	11.76	2	15.38	1	10.00	2	5.26	0	0	11	36.67	3	12.00	**32**	**9.50**

~ sep var* means '1' there is a separate section for variations or '0' there is no separate section

* not including actors

^7 means that Style 7 can be used in structure 1 and 2

CREWS Guidelines		C1		C2		C3		C4		C5		C6		D1		D2		D3		D4		D5		D6		**Total**	
		Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	Ct	%	**Ct**	**%**
#Events Incl. Alternatives		30	n/a	47	n/a	66	n/a	25	N/a	23	n/a	17	n/a	20	n/a	19	n/a	47	n/a	22	n/a	26	n/a	44	n/a	**386**	**32.17**
SG1	Atomic	27	90.00	46	97.87	64	96.97	12	48.00	21	91.30	16	94.12	17	85.00	19	100	47	100	22	100	26	100	38	86.36	**355**	**91.97**
	Composite	3	10.00	1	2.13	2	3.03	13	52.00	2	8.70	1	5.88	3	15.00	0	0	0	0	0	0	0	0	6	9.09	**31**	**8.03**
SG2	Unique order	30	100	47	100	66	100	12	48.00	20	86.96	17	100	20	100	19	100	47	100	22	100	26	100	44	100	**370**	**95.85**
	~unique	0	0	0	0	0	0	0	0	3	13.04	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**3**	**0.78**
	~ sep var*	0	n/a	1	n/a	1	n/a	1	N/a	0	n/a	0	n/a	1	n/a	1	n/a	1	n/a	0	n/a	1	n/a	0	n/a	**n/a**	**n/a**
SG3	Iterations	3	10.00	1	2.13	3	4.55	13	52.00	2	8.70	0	0	1	5.00	1	5.26	2	4.26	1	4.55	1	3.85	1	2.27	**29**	**7.51**

	Alts in main	5	16.67	0	0	0	0	0	0	3	13.04	2	11.76	3	15.00	0	0	3	6.38	16	72.73	0	0	7	15.91	39	13.40
SG4	Homonyms	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Synonyms	0	0	0	0	0	0	4	16.00	1	4.35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	1.30
	*anaphors	1	3.33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	9.09	1	3.85	0	0	4	1.04
SG5	~present	2	6.67	1	2.13	1	1.52	1	4.00	0	0	0	0	1	5.00	1	5.26	1	2.13	0	0	1	3.85	1	2.27	10	2.59
	Passive	2	6.67	0	0	0	0	5	20.00	5	21.74	4	23.53	2	10.00	3	15.79	16	34.04	9	40.91	1	3.85	10	22.73	57	14.77
SG6	Negatives	2	6.67	1	2.13	0	0	2	8.00	2	8.70	0	0	4	20.00	1	5.26	5	10.64	6	27.27	1	3.85	4	9.09	28	7.25
	Adverbs	0	0	0	0	3	4.55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0.78
	Modals	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
CG1		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2		0	0	0	0	3	4.55	0	0	0	0	0	0	0	0	4	21.05	0	0	0	0	0	0	1	2.27	8	2.07
3		3	10.00	0	0	1	1.52	1	4.00	0	0	0	0	3	15.00	0	0	0	0	0	0	0	0	0	0	8	2.07
4		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	3.85	0	0	1	0.26
5		7	23.33	31	65.96	36	54.55	2	8.00	5	21.74	2	11.76	8	40.00	5	26.32	16	34.04	5	22.73	16	61.54	10	22.73	143	37.05
6		10	33.33	0	0	1	1.52	0	0	1	4.35	0	0	5	25.00	1	5.26	5	10.64	10	45.45	0	0	2	4.55	36	9.33
7		0	0	0	0	0	0	0	0	2	8.70	0	0	0	0	1	5.26	1	2.13	1	4.55	1	3.85	0	0	6	1.55
8		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

*pronouns only

~ sep var* means '1' there is a separate section for variations or '0' there is no separate section

Appendix D

Experiment 1: Writing the Description Experimental Material and Data Sets

D1. Experimental Tasks

ATM task

"A supplier of ATMs is to produce a new cash point for a major bank. A requirements analysis has revealed a number of important functions. A customer must have a valid credit card with an acceptable PIN number to access the ATM. A customer can only withdraw up to £30 at a time. As well as withdrawing cash with or without a receipt, a customer can check their balance, order a statement, make a deposit, change their PIN number and also withdraw a further amount without having to re-enter their card.

From the contextual information given, you are asked to write a use case description of a customer who wants to check the balance of their account and withdraw £25 with a receipt. You also have to contend with the problem of entering the wrong PIN number.

Write your use case description here. (If you have any assumptions that you need to make please write them after the description)"

Retail task

"A supplier of supermarket checkout machines is to produce a new checkout machine. A requirements analysis has revealed a number of important functions. A checkout operator must use each checkout machine to record purchases and receive payments from customers. All regular customers have a club card that allows them to save on their bill. Customers can pay by credit card, cash or cheque.

From the contextual information given, you are asked to write a use case description showing a checkout operator who records product purchases of a regular customer and then receives payment by cash. You also have to contend with the problem of a product with no barcode.

Write your use case description here. (If you have any assumptions that you need to make please write them after the description)"

D2. Full Breakdown of Marks Awarded for Hypothesis 2: Comprehension of Descriptions

Each C is marked out of 10. The score given is calculated by adding up all errors in attributes (all of which carry 1 mark per error) of that facet and subtracting from 10. For example, the Coverage facet has two attributes: Span and Scope. If there is one Scope error in the description, it scores 9 out of 10 for Coverage. If there are two Span and one Scope errors then the Coverage score is 7. The total score is out of 70 (because there are 7 C's worth 10 marks each). If there is no separate section for alternatives and exceptions, the description scores 0 for this C. This is why the Separation attribute carries a value of 10 marks (or 0 if there is a separate section!).

CP ATM (Group A)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Scope	1	0	0	4	0	0	0	2	0	0	3	0	1	2	3
Span	1	1	1	0	0	2	0	0	2	0	0	0	1	0	0
Coverage (10)	8	9	9	6	10	8	10	8	8	10	7	10	8	8	7
Text Order	0	0	1	0	1	0	1	1	0	1	3	0	0	0	1
Dependencies	0	4	1	4	4	4	0	0	4	4	1	0	4	0	0
Rational Answer	0	3	1	0	2	0	0	0	0	0	0	0	0	0	0
Cogent (10)	10	3	7	6	3	6	9	9	6	5	6	10	6	10	9
Coherent (10)	10	9	7	0	10	10	6	10	10	8	4	9	7	0	9
Consistent Abstraction (10)	4	10	6	10	10	0	10	8	10	10	10	10	8	6	6
Variations	0	0	3	4	2	0	1	1	1	2	0	0	0	1	1
Sequence	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
Consistent Structure (10)	10	10	7	6	8	10	9	9	9	8	10	10	10	8	9
Consistent Grammar (10)	3	6	9	5	5	9	0	4	9	6	7	5	5	5	1
Separation (10)	10	10	10	10	10	0	10	10	10	10	0	0	10	0	0
Viable	0	0	0	0	0	0	0	0	0	0	3	3	0	0	0
Numbering	0	0	0	0	0	0	0	0	0	0	1	1	0	2	0
Consideration of Alternatives (10)	0	0	0	0	0	10	0	0	0	0	6	6	0	8	10
Total (70)	45	47	45	33	46	53	44	48	52	47	50	60	44	45	51

Table D-1. Complete marks awarded, Group A CP ATM

CREWS ATM (Group C)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Scope	0	0	0	3	1	0	1	0	0	1	3	0	0	0	0
Span	2	0	2	1	1	0	0	1	2	2	1	1	3	0	1
Coverage (10)	8	10	8	6	8	10	9	9	8	7	6	9	7	10	9
Text Order	4	1	1	3	0	1	0	0	1	2	3	1	1	0	2
Dependencies	4	0	0	4	1	0	1	1	0	4	4	0	3	1	1
Rational Answer	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
Cogent (10)	2	9	9	3	9	9	9	9	8	4	3	9	6	8	7
Coherent (10)	8	7	10	9	10	9	8	10	10	7	0	8	8	10	10
Consistent Abstraction (10)	7	8	9	6	8	8	10	9	9	10	9	6	10	10	10
Variations	4	2	4	2	0	2	4	3	2	0	3	3	0	0	2
Sequence	0	0	0	1	0	0	0	6	0	0	3	0	0	2	1
Consistent Structure (10)	6	8	6	7	10	8	6	0	8	10	4	7	10	8	7
Consistent Grammar (10)	7	3	5	0	4	8	4	1	7	10	0	1	7	3	8
Separation (10)	10	0	0	10	10	0	10	10	10	10	0	10	0	0	10
Viable	0	0	0	0	0	0	0	0	0	0	1	0	2	1	0
Numbering	0	1	3	0	0	0	0	0	0	0	0	0	2	0	0
Consideration of Alternatives (10)	0	9	7	0	0	10	0	0	0	0	9	0	6	9	0
Total (70)	38	54	54	31	49	62	46	38	50	48	31	40	54	58	51

Table D-2. Complete marks awarded, Group C CREWS ATM

CP Retail (Group B)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Scope	0	2	0	6	2	1	0	0	1	1	0	0	0	0	0
Span	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
Coverage (10)	10	8	10	4	8	9	10	10	9	9	10	10	9	10	9
Text Order	0	1	0	0	0	0	0	1	0	0	3	0	0	0	0
Dependencies	2	0	0	0	0	0	0	1	0	0	0	0	0	0	0
Rational Answer	0	0	0	0	0	0	3	0	0	0	3	0	0	0	0
Cogent (10)	8	9	10	10	10	10	7	8	10	10	4	10	10	10	10
Coherent (10)	10	10	9	6	8	9	8	6	7	10	9	9	9	8	9
Consistent Abstraction (10)	6	2	8	0	8	8	9	10	8	10	6	9	6	9	8
Variations	2	0	4	0	0	0	5	0	2	2	4	0	3	0	3
Sequence	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Consistent Structure (10)	8	10	6	10	10	10	5	10	8	8	6	10	7	10	7
Consistent Grammar (10)	7	7	8	0	0	9	4	5	6	10	0	9	4	8	4
Separation (10)	0	0	10	10	0	0	0	10	10	0	10	10	10	0	10
Viable	1	0	0	0	3	0	0	0	0	0	0	0	0	0	0
Numbering	2	1	0	0	0	2	0	0	0	1	0	0	0	0	0
Consideration of Alternatives (10)	7	9	0	0	7	8	10	0	0	9	0	0	0	10	0
Total (70)	56	55	51	30	51	63	53	49	48	66	35	57	45	65	47

Table D-3. Complete marks awarded, Group B CP Retail

CREWS Retail (Group D)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Scope	0	0	0	2	1	0	1	0	0	1	0	1	0	3	0
Span	0	0	3	0	0	0	0	0	0	0	2	2	1	0	1
Coverage (10)	10	10	7	8	9	10	9	10	10	9	8	7	9	7	9
Text Order	0	1	2	1	1	2	1	0	2	0	0	1	0	0	1
Dependencies	0	0	0	0	0	0	0	0	0	2	4	0	0	0	1
Rational Answer	0	0	0	3	0	0	0	0	0	0	0	0	0	0	1
Cogent (10)	10	9	8	6	9	8	9	10	8	8	6	9	10	10	7
Coherent (10)	8	8	10	9	10	8	8	10	7	8	10	10	9	3	9
Consistent Abstraction (10)	7	10	8	10	9	5	7	10	10	3	9	10	8	10	7
Variations	1	3	1	6	1	0	2	0	1	1	2	0	1	3	1
Sequence	4	0	0	0	0	0	0	0	0	0	0	0	0	1	0
Consistent Structure (10)	5	7	9	4	9	10	8	10	9	9	8	10	9	6	9
Consistent Grammar (10)	7	9	9	4	7	0	8	7	0	1	6	8	5	8	0
Separation (10)	10	10	10	10	0	0	0	0	10	10	10	10	10	0	10
Viable	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0
Numbering	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
Consideration of Alternatives (10)	0	0	0	0	8	10	9	10	0	0	0	0	0	10	0
Total (70)	47	53	51	41	61	51	58	67	44	38	47	54	50	54	41

Table D-4. Complete marks awarded, Group D CREWS Retail

D3. Examples of Rationale for Allocation of Marks

Example description and marks justification for subject A3

Subject A3 (CP ATM) scores 45 out of 70.

"1. Insert card.

2. Input PIN and validate PIN.

2.1. If PIN correct, go to 3. If not, repeat 2. If 3rd time, go to 6.

3. Check balance.

4. If balance has enough money, then withdraw £25, else 6.

5. Take receipt.

6. Eject card."

Coverage

There are two attributes that make up this C, Span and Scope. There are no scoping problems but there is uncertainty as to whether the £25 is actually presented to the assumed Customer and that the Customer actually takes it. As such, there are difficulties with the Span of the description – there's not enough in it. The description is penalised for this and the description scores 9 out of 10 for Coverage.

Cogent

The description is incomplete in that the Customer does not ask for a receipt – this is a Dependency problem. The final part of event 2.1 cannot be reached ("If 3rd time, go to 6") because if the PIN is incorrect the reader is directed to "repeat 2." This is a Rational Answer problem. The re-entry of the PIN could be much clarified by making it an alternative or exceptional event after the main flow. It is also the case that ATMs eject cards before any cash or receipt is ejected because the bank would rather the Customer walk away with their card and forget their cash and their receipt, rather than the reverse – this was a genuine problem with the first ATMs. This is a Text Order problem. The use case scores 7 out of 10 for Cogent.

Coherent

There are three coherence problems: sentence 2 does not cohere to sentence 1 – although this is not a major problem; sentence 3 does not cohere to 2.1 (which contains 3 separate sentences). Sentence 5 does not cohere to 4 (or any other sentence). As an example of coherence, sentence 4 locally coheres to 3 because there is a repeat of "balance". Sentence 6 globally coheres to the first sentence: card->card. Coherent scores 7 out of 10.

Consistent Abstraction

Abstraction is a mix because the PIN is validated, which is an internal design issue. It only becomes external design if the description clearly states that the Customer is visually aware of a PIN validation message. Since this probably never happens on ATMs unless the PIN is incorrect, there's no point mentioning PIN validation at all. If a PIN is invalidated, then this can be noted in an alternative / exceptional flow of events outside of the main flow. The description scores 6 out of 10 for this C.

Consistent Structure

Consistent Structure is compromised by the Variation regarding PIN validity in the main flow. There are 3 errors. Variation is marked by counting the number of alternatives in the main flow. Event 2.1 contains two alternatives and event 4 has one. Sequencing is fine for this description. Consistent Structure scores 7 out of 10.

Consistent Grammar

There is only 1 grammar penalty (although there is an obvious lack of an actor in the entire description) because there is a negative in event 2.1. The *if...then...* structure is not part of the CP Rules because this structure allows alternatives, negatives and uncertainty. This is why many of the CREWS descriptions have unnecessary problems; CREWS allows this structure (CG6).

Consideration of Alternatives

The description scores zero for Consideration of Alternatives because there are none.

Example description and marks justification for subject C13

As a comparison to A3's use case description, the following, by subject C13 – CREWS ATM - scored 54 out of 70.

"1. The user inserts card.

2. 'Loop' until PIN is correct.

3. The user enters PIN.

4. The user presses balance button.

5. The balance is displayed on screen.

6. The user presses return to main menu button.

7. The user presses withdraw button.

8. The ATM prompts for amount.

9. The user selects an amount.

10. The ATM dispenses amount.

11. The ATM prompts receipt choice.

12. The user selects print receipt.

14. The user selects return card.

15. The ATM returns card.

Exceptions

2. 'Loop' until wrong PIN is entered 3 times.

4. The ATM denies access.

10. The user's balance is not sufficient.

11. The ATM displays insufficient funds.

Assumptions

The user can only enter the wrong PIN 3 times.

The first screen allows the option of balance.

The option of return card is on all user screens.

The prompt for receipt is given after cash is dispensed.

The ATM has enough money left to dispense correct amount.

The ATM has paper to print the receipts on."

Coverage

There are three Span problems. The reader is not told that the ATM prompts the User to select balance nor that the user takes the cash and receipt. The description scores 7 out of 10 for Coverage.

Cogent

There is a Text Order problem similar to that described for subject A3 in that the Customer's card is returned after cash and receipt have been dispensed. The three Span problems are also closely tied to Dependencies. Event 4 needs a pre-condition (a dependency) for it to occur. This is missing. Post-conditions after events help ensure dependencies to the next events. This is missing in terms of the user taking the receipt and taking the card. Sentence 2 might appear peculiar but it is CREWS Content Guideline 7 and the reader should assume that sentence 3 is attached to 2. The exceptions deal with the problem of an incorrect PIN. The use case scores 6 out of 10 for this C.

Coherent

There are only a couple of coherence problems: sentence 2 does not cohere to 1 and 8 does not cohere to 7; assumptions have to be made. The description scores 8 out of 10 for coherence.

Consistent Abstraction

Abstraction is fine. There is no mention of the machine having to validate the PIN. The use case scores 10 for Abstraction.

Consistent Structure

There are no Variations in the main flow nor is there a Sequencing problem. The description scores 10 out of 10 for Consistent Structure.

Consistent Grammar

There are three grammar errors, for example, the use of passive voice in event 5. Grammar scores 7 out of 10.

Consideration of Alternatives

The Alternatives facet has a few concerns. Firstly, there are Viability problems in that the reader is unsure what sentence 4 is referring to, although assumptions can be made. Also, there are two Numbering problems. Events 10 and 11 in the exceptions should be numbered probably at 8 and 9 in the main flow in that the machine should not allow a customer to attempt to withdraw money if there is not enough in their account. The description scores 6 out of 10 for this C.

Example description and marks justification for subject D4

An example use case description is provided by subject D4 (CREWS Retail). The subject scored 41 marks out of 70. Here is the description:

"1. The operator takes product from customer.

2. Operator checks product for barcode.

3. If 'no barcode' then <<include: Get Barcode>>.

4. Operator enters barcode into machine.

5. Operator records product information to the form.

6. Operator informs customer of price.

7. Operator records price to the form.

8. Customer gives money to the operator.

9. Operator checks money against price.

10. If 'not enough money' then 'operator informs customer'.

10.1 Customer gives money to operator.

10.2 Goto 9.

11. Operator enters amount of money into machine.

12. If 'customer requires change' then 'take change from machine'.

12.1 Operator gives customer change.

13. Operator records the money received and change given to the form.
14. Customer gives club card to the operator.
15. Operator signs form to complete manual record.
16. Operator swipes club card in machine reader to update card.
17. Operator informs customer of updates.
18. Operator gives club card to customer.
19. Operator closes the machine to end transaction.

Assumptions

Customer has enough cash so that transaction can be completed.

Operator has to record purchases manually to a form.

Machine calculates change and operator takes out required amount.

Customer wants to collect points to club card and not to use it to save money on transactions."

Coverage

There are Scope errors; the recording of information on a form, presumably because the machine has failed in some unspecified way. Span, though, is fine. The description scores 8 for Coverage.

Cogent

It is clear that the use case completes and that the transaction finishes (Dependencies). However, there are some concerns regarding the Rational Answers of the description. The sentences 5, 6 and 7, for example, state that although the barcode reader works, the operator has to write the product's price down on a 'form'. The Assumptions section tries to make this clearer but also states that the machine, which is assumed to be the Till, calculates change required. If it does this, then prices have been entered. If the machine is just a pocket calculator, then this should be stated. Also, since the barcodes are read into the machine, this should be able to calculate a running total. It is unclear why the operator need record anything on a form. Text Order has one potential error in that the club card is dealt with after payment has completed. This is unlikely. The description scores 6 out of 10.

Coherent

There is one coherence slip Event 10.2 does not refer to the event before it or infer the next event. Although it states "Goto 9", this code-oriented structure is not recommended. This C gets 9 out of 10.

Consistent Abstraction

There are no abstraction problems.

Consistent Structure

Structurally there are Variations in the description that should be placed in a separate section. For instance, the problem statement reads that the description should deal with a product without a barcode. This is done in event 3. Really this is an exception and should be placed in a separate section below the main flow. (The barcode missing from the product is listed in the experimental task statement as an, what is hoped, exceptional task, and this should suggest to the subjects that an alternatives / exceptions section is required.) The choice of giving change (events 12 and 12.1) offers Variations in the main flow that are better described in the Alternatives section, as should events 10, 10.1 and 10.2. The description scores 4 for this C.

Consistent Grammar

There are grammar mistakes such as usage of passive voice in event 13. This scores 4 out of 10.

Consideration of Alternatives

There is no score for Consideration of Alternatives because there is no Alternatives section after the main flow.

Example description and marks justification for subject B10

An example of a high scoring use case description is provided by subject B10 (CP Retail), gaining a mark of 66 out of 70 in total:

"1. Customer gives operator shopping.

1.1 Operator scans products.

1.2. Operator totals shopping bill.

2. Operator asks for customer's club card.

3. Operator scans club card.

3.1. Operator returns club card to customer.

3.2. Operator asks for payment from customer.

4. Operator enters payment amount, type (type: cash, cheque, credit card).

4.1. Operator gives receipt and change, if customer payment by cash.

Exceptions and alternatives

No barcode on product.

1.2. Operator calls supervisor for assistance. Supervisor gets barcode.

3. If customer does not have club card, operator offers club card.

3. If club card does not scan, enter club card number

else

Call supervisor.

4. If Customer has no money retrieve shopping and void transaction."

Coverage

There is one Scope problem in event 4 with the choice of payment types offered. The problem statement asked that the customer pay by cash only. There are no Span problems. Coverage scores 9 out of 10.

Cogent

There are no Text Order, Dependency or Answer problems. The description gains maximum marks of 10 out of 10.

Coherent

There are no coherence problems. The description gets 10 out of 10.

Consistent Abstraction

There is problem domain information (event 1, 2 etc) and specification information (event 1.1, 3 etc) mixed into the same description. However, the nature of this problem should take into consideration interactions between actors, since the Customer is the driver for this use case. As such, any such interactions between Customer and the Till or the Cashier are considered fine for both groups B and D. This description gets full marks.

Consistent Structure

Structurally there is nothing to fault the description except the Variation at event 4 and 4.1. As such, the description scores 8 for structure.

Consistent Grammar

There are no grammar errors; this scores 10 out of 10.

Consideration of Alternatives

There is a separate Alternatives section and all events are considered viable here. There is one Numbering concern. Exception 3 has an *if... else* statement that is very akin to program code. Although this is one statement, it might be wiser to number the else action. Consideration of Alternatives scores 9 out of 10.

D4. Hypothesis 1 Data Sets

Group A CP ATM	Subjects	A1		A2		A3		A4		A5		A6		A7		A8		A9		A10		A11		A12		A13		A14		A15		Total	
		Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean
Events in main		12	n/a	12	n/a	7	n/a	26	n/a	7	n/a	10	n/a	19	n/a	19	n/a	12	n/a	18	n/a	13	n/a	12	n/a	13	n/a	26	n/a	24	n/a	230	15.33
Events incl. Alternatives		12	n/a	12	n/a	7	n/a	26	n/a	7	n/a	12	n/a	19	n/a	19	n/a	12	n/a	18	n/a	20	n/a	21	n/a	13	n/a	30	n/a	26	n/a	254	16.93
Style																																	
1	Not Numbered	0	0.00	0	0.00	0	0.00	8	0.31	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	13	5.12
1	Alts section	0	n/a	0	n/a	0	n/a	0	n/a	0	n/a	1	n/a	0	n/a	0	n/a	0	n/a	0	n/a	1	n/a	1	n/a	0	n/a	1	n/a	1	n/a	5	0.33
1	Alts in main	0	0.00	0	0.00	3	0.43	9	0.35	2	0.29	0	0.00	1	0.05	1	0.05	1	0.08	2	0.11	0	0.00	0	0.00	0	0.00	1	0.08	1	0.04	27	11.74
2	Non-present tense	4	0.33	1	0.08	0	0.00	5	0.19	4	0.57	1	0.08	10	0.53	6	0.32	0	0.00	3	0.17	3	0.15	4	0.19	5	0.38	1	0.03	2	0.08	48	18.90
3	Adverbs	1	0.08	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.08	1	0.06	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	3	1.18
3	Adjectives	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
3	Negatives in main	0	0.00	3	0.25	1	0.14	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.04	0	0.00	5	2.17
3	Pronouns	2	0.17	0	0.00	0	0.00	0	0.00	1	0.14	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.05	0	0.00	3	0.10	7	0.27	14	5.51
4	Explanations	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	3	0.10	0	0.00	3	1.18
5	Local Coherence	7	0.64	7	0.64	2	0.33	6	0.24	4	0.67	7	0.78	7	0.39	16	0.89	6	0.55	5	0.29	4	0.33	8	0.73	4	0.33	13	0.52	14	0.61	112	52.09
5	Global Coherence	4	0.36	3	0.27	1	0.17	4	0.16	2	0.33	2	0.22	7	0.39	2	0.11	5	0.45	10	0.59	2	0.17	2	0.18	5	0.42	2	0.08	8	0.35	59	27.44
6	No action response	1	0.08	0	0.00	2	0.29	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	3	1.18
7	Underline UC names	0	0.00	0	0.00	1	0.14	0	0.00	2	0.29	2	0.17	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	5	1.97
Structure																																	
1		4	0.33	4	0.33	0	0.00	1	0.04	2	0.29	6	0.50	9	0.47	3	0.16	10	0.83	11	0.61	4	0.20	7	0.33	8	0.62	21	0.70	8	0.31	98	38.58
2		0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	4	0.33	1	0.05	7	0.37	0	0.00	0	0.00	0	0.00	1	0.05	0	0.00	7	0.23	2	0.08	22	8.66

Group B CP Retail	Subjects	B1		B2		B3		B4		B5		B6		B7		B8		B9		B10		B11		B12		B13		B14		B15		Total	
		Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean
Events in main		14	n/a	27	n/a	11	n/a	30	n/a	22	n/a	20	n/a	10	n/a	14	n/a	18	n/a	9	n/a	28	n/a	10	n/a	19	n/a	12	n/a	18	n/a	262	17.47
Events incl. Alternatives		16	n/a	36	n/a	11	n/a	30	n/a	32	n/a	35	n/a	21	n/a	14	n/a	18	n/a	14	n/a	28	n/a	10	n/a	19	n/a	17	n/a	18	n/a	319	21.27
Style																																	
1	Not Numbered	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.07	0	0.00	0	0.00	0	0.00	0	0.00	0	0	1	0.31
1	Alts section	1	n/a	1	n/a	0	n/a	0	n/a	1	n/a	1	n/a	1	n/a	0	n/a	0	n/a	1	n/a	0	n/a	0	n/a	0	n/a	1	n/a	0	n/a	7	0.47
1	Alts in main	2	0.14	0	0.00	4	0.36	0	0.00	0	0.00	0	0.00	5	0.50	1	0.07	2	0.11	0	0.00	4	0.14	0	0.00	3	0.16	0	0.00	3	0.17	24	9.16
2	Non-present tense	1	0.06	2	0.06	1	0.09	6	0.20	7	0.22	1	0.03	5	0.24	2	0.14	2	0.11	0	0.00	3	0.11	1	0.10	3	0.16	2	0.12	3	0.17	40	12.54
3	Adverbs	0	0.00	0	0.00	0	0.00	0	0.00	1	0.03	0	0.00	0	0.00	1	0.07	0	0.00	0	0.00	9	0.32	0	0.00	1	0.05	0	0.00	1	0.06	13	4.08
3	Adjectives	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
3	Negatives in main	2	0.14	0	0.00	1	0.09	0	0.00	0	0.00	0	0.00	1	0.10	0	0.00	2	0.11	0	0.00	4	0.14	0	0.00	2	0.11	0	0.00	2	0.11	14	5.34
3	Pronouns	0	0.00	1	0.03	0	0.00	4	0.13	8	0.25	0	0.00	0	0.00	2	0.14	0	0.00	0	0.00	1	0.04	0	0.00	0	0.00	0	0.00	0	0.00	16	5.02
4	Explanations	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.07	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.31
5	Local Coherence	11	0 85	21	0.81	6	0.60	16	0.55	16	0.76	15	0.79	2	0.22	5	0.38	10	0.59	8	1.00	17	0.63	7	0.78	11	0.61	9	0.82	11	0.65	165	66.80
5	Global Coherence	2	0.15	5	0.19	3	0.30	9	0.31	3	0.14	3	0.16	5	0.56	4	0.31	4	0.24	0	0.00	9	0.33	1	0.11	6	0.33	0	0.00	5	0.29	59	23.89

6	No action response	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.04	0	0.00	0	0.00	0	0.00	0	0.00	1	0.31
7	Underline UC names	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.06	0	0.00	0	0.00	2	0.20	3	0.16	0	0.00	4	0.22	10	3.13
Structure																																	
1		1	0.06	12	0.33	5	0.45	9	0.30	2	0.06	20	0.57	5	0.24	2	0.14	4	0.22	4	0.29	10	0.36	3	0.30	8	0.42	2	0.12	4	0.22	91	28.53
2		8	0.50	12	0.33	4	0.36	5	0.17	7	0.22	10	0.29	6	0.29	2	0.14	6	0.33	2	0.14	5	0.18	4	0.40	3	0.16	5	0.29	3	0.17	82	25.71

Group C CREWS ATM	Subjects	C1		C2		C3		C4		C5		C6		C7		C8		C9		C10		C11		C12		C13		C14		C15		Total	
		Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean
Events in main		23	n/a	13	n/a	14	n/a	30	n/a	23	n/a	21	n/a	22	n/a	27	n/a	16	n/a	13	n/a	32	n/a	24	n/a	15	n/a	19	n/a	15	n/a	307	20.47
Events incl. Alternatives		23	n/a	16	n/a	21	n/a	30	n/a	23	n/a	22	n/a	22	n/a	27	n/a	16	n/a	13	n/a	40	n/a	24	n/a	19	n/a	20	n/a	15	n/a	331	22.07
Style Guidelines																																	
SG1 atomic		21	0.91	16	1.00	20	0.95	25	0.83	22	0.96	21	0.95	16	0.73	26	0.96	16	1.00	12	0.92	39	0.98	23	0.96	19	1.00	19	0.95	10	0.67	305	92.15
1 composite		2	0.09	0	0.00	1	0.05	5	0.17	1	0.04	1	0.05	6	0.27	1	0.04	0	0.00	1	0.08	1	0.03	1	0.04	0	0.00	1	0.05	5	0.33	26	7.85
SG2 unique order		23	1.00	16	1.00	18	0.86	30	1.00	23	1.00	22	1.00	22	1.00	27	1.00	16	1.00	13	1.00	38	0.95	24	1.00	19	1.00	18	0.90	14	0.93	323	97.58
2 ~unique		0	0.00	0	0.00	3	0.14	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	2	0.05	0	0.00	0	0.00	2	0.10	1	0.07	8	2.42
2 Sep section		0	n/a	1	n/a	1	n/a	0	n/a	0	n/a	1	n/a	0	n/a	0	n/a	0	n/a	0	n/a	1	n/a	0	n/a	1	n/a	1	n/a	0	n/a	6	0.40
SG3 iterations		2	0.09	0	0.00	0	0.00	6	0.20	0	0.00	3	0.14	6	0.27	5	0.19	2	0.13	0	0.00	4	0.10	2	0.08	0	0.00	0	0.00	2	0.13	32	9.67
3 alts in main		3	0.13	2	0.15	1	0.07	2	0.07	0	0.00	2	0.10	4	0.18	3	0.11	1	0.06	0	0.00	3	0.09	3	0.13	0	0.00	0	0.00	2	0.13	26	8.47
SG4 homonyms		0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
4 synonyms		0	0.00	0	0.00	2	0.10	1	0.03	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	3	0.91
4 *anaphors		0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	3	0.08	4	0.17	0	0.00	0	0.00	0	0.00	2	0.60
SG5 ~present		1	0.04	4	0.25	2	0.10	9	0.30	2	0.09	0	0.00	2	0.09	4	0.15	1	0.06	0	0.00	25	0.63	3	0.13	2	0.11	6	0.30	10	0.67	71	21.45
5 passive		0	0.00	2	0.13	0	0.00	9	0.30	0	0.00	0	0.00	1	0.05	2	0.07	1	0.06	0	0.00	18	0.45	1	0.04	2	0.11	0	0.00	8	0.53	44	13.29
SG6 negatives		0	0.00	1	0.06	1	0.05	3	0.10	1	0.04	2	0.09	3	0.14	3	0.11	2	0.13	0	0.00	0	0.00	2	0.08	1	0.05	1	0.05	0	0.00	20	6.04
6 adverbs		0	0.00	0	0.00	0	0.00	0	0.00	3	0.13	0	0.00	1	0.05	2	0.07	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.07	7	2.11
6 modals		0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
Content Guidelines																																	
CG1		0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
2		1	0.04	0	0.00	0	0.00	1	0.03	1	0.04	2	0.09	0	0.00	2	0.07	1	0.06	0	0.00	0	0.00	1	0.04	0	0.00	0	0.00	0	0.00	9	2.72
3		1	0.04	2	0.13	1	0.05	0	0.00	0	0.00	3	0.14	0	0.00	0	0.00	0	0.00	2	0.15	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	9	2.72
4		0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
5		8	0.35	4	0.25	10	0.48	3	0.10	15	0.65	11	0.50	10	0.45	9	0.33	11	0.69	8	0.62	7	0.18	11	0.46	15	0.79	4	0.20	5	0.33	131	39.58
6		5	0.22	2	0.13	0	0.00	4	0.13	0	0.00	1	0.05	0	0.00	4	0.15	3	0.19	2	0.15	10	0.25	3	0.13	0	0.00	2	0.10	0	0.00	36	10.88
7		1	0.04	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	2	0.11	0	0.00	0	0.00	3	0.91
8		0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.04	0	0.00	0	0.00	0	0.00	1	0.30

Group D CREWS Retail	Subjects	D1		D2		D3		D4		D5		D6		D7		D8		D9		D10		D11		D12		D13		D14		D15		Total	
		Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean	Cnt	Mean
Events in main		24	n/a	13	n/a	13	n/a	22	n/a	17	n/a	19	n/a	15	n/a	11	n/a	13	n/a	20	n/a	11	n/a	11	n/a	11	n/a	13	n/a	15	n/a	228	15.20
Events incl. Alternatives		24	n/a	13	n/a	13	n/a	22	n/a	20	n/a	26	n/a	17	n/a	20	n/a	13	n/a	20	n/a	11	n/a	11	n/a	11	n/a	14	n/a	15	n/a	250	16.67
Style Guidelines																																	
SG1 atomic		24	1.00	13	1.00	13	1.00	22	1.00	20	1.00	24	0.92	13	0.76	20	1.00	13	1.00	18	0.90	10	0.91	10	0.91	11	1.00	12	0.86	14	0.93	237	94.80
1 composite		0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	2	0.08	4	0.24	0	0.00	0	0.00	2	0.10	1	0.09	1	0.09	0	0.00	2	0.14	1	0.07	13	5.20

SG2	unique order		20	0.83	13	1.00	13	1.00	22	1.00	20	1.00	26	1.00	17	1.00	20	1.00	13	1.00	20	1.00	11	1.00	11	1.00	11	1.00	13	0.93	15	1.00	245	98.00
2	~unique		4	0.17	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.07	0	0.00	5	2.00
2	Sep section		0	n/a	0	n/a	0	n/a	0	n/a	1	n/a	1	n/a	1	n/a	1	n/a	0	n/a	0	n/a	0	n/a	0	n/a	0	n/a	1	n/a	0	n/a	5	0.33
SG3	iterations		0	0.00	2	0.15	0	0.00	1	0.05	1	0.05	1	0.04	2	0.12	3	0.15	0	0.00	0	0.00	1	0.09	0	0.00	0	0.00	4	0.29	3	0.20	18	7.20
3	alts in main		1	0.04	3	0.23	1	0.08	6	0.27	1	0.06	0	0.00	2	0.13	0	0.00	1	0.08	1	0.05	2	0.18	0	0.00	1	0.09	3	0.23	2	0.13	24	10.53
SG4	homonyms		0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
4	synonyms		0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.04	0	0.00	0	0.00	0	0.00	1	0.05	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	2	0.80
4	*anaphors		0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.06	1	0.05	1	0.08	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	2	0.13	5	2.00
SG5	~present		1	0.04	1	0.08	0	0.00	5	0.23	1	0.05	7	0.27	0	0.00	1	0.05	12	0.92	3	0.15	2	0.18	0	0.00	2	0.18	1	0.07	4	0.27	40	16.00
5	passive		0	0.00	1	0.08	0	0.00	0	0.00	1	0.05	7	0.27	0	0.00	0	0.00	3	0.23	3	0.15	0	0.00	0	0.00	2	0.18	0	0.00	4	0.27	21	8.40
SG6	negatives		1	0.04	0	0.00	1	0.08	1	0.05	2	0.10	1	0.04	1	0.06	1	0.05	1	0.08	2	0.10	2	0.18	2	0.18	3	0.27	0	0.00	2	0.13	20	8.00
6	adverbs		1	0.04	0	0.00	0	0.00	0	0.00	0	0.00	1	0.04	0	0.00	0	0.00	0	0.00	2	0.10	0	0.00	0	0.00	0	0.00	1	0.07	4	0.27	9	3.60
6	modals		0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.05	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.40
Content Guidelines																																		
CG1			0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
2			5	0.21	2	0.15	5	0.38	4	0.18	3	0.15	0	0.00	1	0.06	7	0.35	1	0.08	4	0.20	2	0.18	0	0.00	0	0.00	0	0.00	0	0.00	34	13.60
3			0	0.00	0	0.00	0	0.00	1	0.05	0	0.00	0	0.00	0	0.00	0	0.00	1	0.08	1	0.05	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	3	1.20
4			0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00
5			1	0.04	0	0.00	3	0.23	0	0.00	5	0.25	10	0.38	6	0.35	1	0.05	2	0.15	0	0.00	2	0.18	5	0.45	2	0.18	0	0.00	2	0.13	39	15.60
6			3	0.13	3	0.23	2	0.15	2	0.09	2	0.10	1	0.04	2	0.12	3	0.15	3	0.23	1	0.05	0	0.00	0	0.00	2	0.18	2	0.14	2	0.13	28	11.20
7			1	0.04	1	0.08	1	0.08	0	0.00	0	0.00	1	0.04	2	0.12	0	0.00	1	0.08	0	0.00	1	0.09	1	0.09	0	0.00	1	0.07	0	0.00	10	4.00
8			0	0.00	1	0.08	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	0	0.00	1	0.40

Appendix E

Experiment 2: Questioning the Description Experimental Material and Data Sets

E1. Experimental Questions

Software Design: Use Cases to Class Diagrams

This exercise is intended to lead you through the questions that we should ask of a use case, in order to move towards design. Hence, please attempt to work through sections A, B and C before trying to produce your UML Class Diagram (section D).

You are expected to carry out this exercise with reference to the ATM Use Case Description.

SECTION A: Specific Questions

Please answer the following questions on the ATM use case

1. Does the ATM validate the Customer's card?

   Where does this happen?

   What actor validates the card?
2. What checks the PIN number?

   Who is the actor?
3. When is the Customer's balance displayed?

   Where does this information come from?

   How does the ATM get this information?
4. What happens if the balance cannot be displayed?
5. Where is the new PIN stored?

   How do we know this?
6. When a cash withdrawal is selected, how does the ATM know the Customer has funds?
7. Is the Customer's account updated before the cash is taken, after the cash is taken, or when the card is released?

   How is the account updated?
8. Where and how does the ATM know that the Customer has taken the cash, receipt and card?

SECTION B: General Questions

1. Moving from sentence to sentence (event by event) what dependencies do you have to assume?

   List these dependenciés, noting where they occur.

   List those dependencies that appear to be assumptions

2. Do you have to assume there are other actors involved in the system that are necessary to assure dependencies but are not stated in the use case (these actors could be other systems)?

   Where should these actors appear?

   Are they passive (just receive information) or active (do something on/for the system)?

SECTION C: List classes, their operations and attributes

Use the following points to guide you, in your production of the list.

- To identify potential classes look for nouns/names in the use case.
- Are there classes you have to "invent"?
  - That is, they are implied but not explicitly stated.
- What operations link classes to other classes?
- What operations do the classes perform on themselves or perform for other classes?
- What attributes can be identified for each class from the use case?
  - These attributes are usually nouns that belong to other classes.

SECTION D: Draw a class diagram for the ATM

- You will have to consider structures such as inheritance and whole/part
- Include services and attributes.

E2. Experimental Treatments: Use Case Descriptions

CP ATM

Use Case Name: Accessing the ATM

Actors: Customer, ATM

Context: The Customer wishes to get their balance, change their PIN and withdraw £25 cash with a receipt.

Pre-conditions: The Customer has approached a machine that is not in the middle of a transaction.

Post-conditions: The Customer knows their balance, has changed their PIN and has £25 cash and a receipt. The ATM is ready for the next customer.

Main Flow

1. The Customer inserts their card.
2. The ATM requests the Customer's PIN.
3. The Customer enters their PIN.
4. ATM displays the following options: Withdraw Cash; Withdraw Cash with Receipt; Check Balance; Order Statement; Make Deposit; Change PIN.
5. Customer selects "Check Balance".
6. ATM displays balance and asks if the Customer wants to perform another operation.
7. Customer selects "Yes".
8. ATM displays the options as in 4.
9. Customer selects "Change PIN".
10. ATM requests the new PIN (the current PIN has already been validated).
11. The Customer enters the required new PIN.
12. The ATM asks that the new PIN be repeated.
13. The Customer enters the PIN again.
14. The ATM confirms that the PIN has been changed and asks if another operation is required.
15. Customer selects "Yes".
16. ATM displays the options in 4 again.
17. Customer selects "Withdraw Cash with Receipt".
18. The ATM displays common amounts up to £30 and also "another amount".

19. The Customer selects "another amount" (£25 is not one of the options).
20. The ATM requests the amount.
21. The Customer enters £25 using the keypad.
22. The ATM displays a wait for cash message.
23. The ATM dispenses the cash.
24. The Customer takes the cash.
25. The ATM displays a wait for receipt message.
26. The ATM prints a receipt.
27. The Customer takes receipt.
28. The ATM asks if another option is required.
29. The Customer selects "No".
30. The ATM ejects card and bleeps (to alert the Customer).
31. The Customer takes the card.
32. The ATM thanks the Customer.
33. The ATM displays a wait message until it is ready to serve the next customer.
34. The ATM displays its readiness.

Exceptions

2. The card is invalid, the ATM ejects it and asks the Customer to seek assistance.

4. The PIN is wrong. The ATM goes back to 2 twice more (to allow the Customer to retry), on the fourth try it retains the card and asks the Customer to seek assistance.

14. The two PIN entries do not match. The ATM goes back to 10 after giving the reason why.

22. The amount is invalid (i.e. above £30). The ATM returns to 18, after explaining why.

Alternatives

At any point the Customer may press the "Cancel" button. If the ATM is in the middle of an operation it rolls back the operation and displays the list of options. If it isn't it goes to 30.

19. The Customer selects an amount that is on the list. ATM goes to 22.

At 4, 8, 16 the Customer could select the options in any order.

CREWS ATM

Use Case Name: Accessing the ATM

Actors: Customer, ATM

Context: The Customer wishes to get their balance, change their PIN and withdraw £25 cash with a receipt.

Pre-conditions: The Customer has approached a machine that is not in the middle of a transaction.

Post-conditions: The Customer knows their balance, has changed their PIN and has £25 cash and a receipt. The ATM is ready for the next customer.

Main flow of events:

1. The Customer inserts the credit card into the ATM card slot.
2. ATM validates card input.
3. ATM prompts Customer to enter PIN.
4. Customer enters PIN via keypad.
5. ATM validates Customer.
6. ATM accesses Customer bank details from database.
7. ATM prompts Customer with option menu.
8. Customer presses "Check Balance" button.
9. ATM outputs Customer balance.
10. ATM prompts Customer with "Another service?" option.
11. Customer presses "Yes" button.
12. ATM prompts Customer with option menu.
13. Customer presses "Change PIN" button.
14. ATM prompts user to enter new PIN.
15. Customer enters new PIN.
16. ATM prompts user to confirm new PIN.
17. Customer re-enters new PIN.

18. ATM writes new PIN to Customer credit card.
19. ATM confirms PIN change successful.
20. ATM prompts Customer with "another service?"
21. Customer presses "Yes" button.
22. ATM prompts Customer with option menu.
23. Customer presses "Withdraw Cash" button.
24. ATM prompts user with Cash Amount options.
25. Customer presses "£25" option.
26. ATM checks Customer has available funds.
27. ATM asks Customer if receipt is wanted.
28. Customer presses "Yes" button.
29. ATM prints receipt.
30. ATM counts cash to correct amount.
31. ATM writes new bank balance to Customer bank details in the database.
32. ATM prompts user with "Another service?"
33. Customer presses "No" button.
34. ATM ejects card.
35. ATM ejects cash.
36. Customer takes card and cash.
37. ATM resets for next card entry.

<u>Alternative flows</u>

8. Customer presses alternative "option" button.
13. Customer presses alternative "option" button.
23. Customer presses alternative "option" button.
11. Customer presses "No" button.
21. Customer presses "No" button.
28. Customer presses "No" button.
33. Customer presses "No" button.

<u>Exceptional flows</u>

1. Customer inserts credit card into slot incorrectly.
   1.1 ATM ejects card.

4. Customer enters PIN incorrectly.

4.1 ATM asks for PIN re-entry.

4.2 Loop back to 4.

4.3 If 4 has been exceptional 3 times then retain card.

4.4 Prompt Customer to consult bank.

15. Customer enters a new PIN already in use by another Customer.

15.1 ATM prompts for alternative new PIN.

15.2 Goto 15 in main flow.

17. Customer re-enters new PIN incorrectly.

17.1 ATM prompts for alternative new PIN.

17.2 Goto 15 in the main flow.

26. Customer does not have available funds.

26.1 ATM informs Customer "Insufficient funds".

26.2 ATM prompts "Another service?"

29. ATM has insufficient printing paper.

29.1 ATM informs Customer "No receipt".

29.2 ATM continues to 30.

30. ATM has insufficient cash.

30.1 ATM informs Customer.

30.2 ATM counts available cash and continues to 31.

E3. Dependencies: 'Expert' Answers

CP ATM

NB. Pre-conditions that state "as post" refer to the post-condition of the previous event

Use Case Name: Accessing the ATM

Actors: Customer, ATM

Context: The Customer wishes to get their balance, change their PIN and withdraw £25 cash with a receipt.

Pre-conditions: The Customer has approached a machine that is not in the middle of a transaction.

Post-conditions: The Customer knows their balance, has changed their PIN and has £25 cash and a receipt. The ATM is ready for the next customer.

Main Flow

1. The Customer inserts their card.

(pre: ATM functioning, Customer has card; Post: card accepted – validated – by ATM)

2. The ATM requests the Customer's PIN.

(Pre: card validated; Post: screen visible to Customer and correct message, keypad activated)

3. The Customer enters their PIN.

(Pre: as last post, Customer knows PIN; Post: input read, PIN validated by machine, system link to Customer Account established in some way)

4. ATM displays the following options: Withdraw Cash; Withdraw Cash with Receipt; Check Balance; Order Statement; Make Deposit; Change PIN.

(Pre: as post; Post: screen visible to Customer, keypad activated)

5. Customer selects "Check Balance".

(Pre: screen visible; Post: system responds correctly to button press, system goes to Customer Account to access balance)

6. ATM displays balance and asks if the Customer wants to perform another operation.

(Pre: System has retrieved balance from correct account; Post: correct information displayed on screen, keypad activated)

7. Customer selects "Yes".

(Pre: Further operations message displayed; Post: button press correctly recognised by system)

8. ATM displays the options as in 4.

(Pre: System responds to 'Yes' option; Post: list of options displayed, keypad activated)

9. Customer selects "Change PIN".

(Pre: options visible, correct buttons activated; Post: system recognises correct button selection)

10. ATM requests the new PIN (the current PIN has already been validated).

(Pre: the change PIN subroutine is activated; Post: message displayed, keypad activated)

11. The Customer enters the required new PIN.

(Pre: as last post; Post: system responds to key input, system validates input, stores number entered, show number of keys pressed with '')*

12. The ATM asks that the new PIN be repeated.

(Pre: as post; Post: message displayed, key pad activated)

13. The Customer enters the PIN again.

(Pre: as post; Post: system reads key press, system displays '' per number entered, system stores number)*

14. The ATM confirms that the PIN has been changed and asks if another operation is required.

(Pre: system compares entered numbers, validates numbers, reads number to card and to corresponding account; Post: message displayed, key pad activated)

15. Customer selects "Yes".

(Pre: as post; Post: key selection recognised, start display options subroutine)

16. ATM displays the options in 4 again.

(Pre: as post: Post: display options visible on screen, key pad activated)

17. Customer selects "Withdraw Cash with Receipt".

(Pre: as post; Post: recognise key input, link to Withdraw Cash with Receipt subroutine, automatically check Customer balance)

18. The ATM displays common amounts up to £30 and also "another amount".

(Pre: on receipt of positive balance (i.e. equal to or more than lowest withdrawal sum available); Post: display amounts, activate keypad)

19. The Customer selects "another amount" (£25 is not one of the options).

(Pre: options displayed; Post: "Another amount" key recognised)

20. The ATM requests the amount.

(Pre: as post; Post: display Enter Amount screen, activate key pad)

21. The Customer enters £25 using the keypad.

(Pre: screen displayed; Post: recognise key input, display on screen, recognise enter key press)

22. The ATM displays a wait for cash message.

(Pre: Enter key pressed, amount requested less than balance; Post: display message, calculate new balance, send balance to Customer account, start note dispensing process)

23. The ATM dispenses the cash.

(Pre: note dispenser selects correct notes and amount, notes raised to slot; Post: notes visible to Customer, note dispensed sensor set to off, display Take Money message)

24. The Customer takes the cash.

(Pre: notes dispensed and visible to Customer; Post: note dispenser sensor set to on, signal sent to receipt printer, signal to display Receipt message)

25. The ATM displays a wait for receipt message.

(Pre: as post; Post: message displayed)

26. The ATM prints a receipt.

(Pre: account number and amount dispensed sent to printer, paper available, (laser) printer functioning; Post: receipt printed, receipt moved to receipt slot, receipt visible to customer, receipt slot signal set to off, take receipt message displayed)

27. The Customer takes receipt.

(Pre: receipt visible to customer, message visible to customer; Post: receipt slot signal set to on)

28. The ATM asks if another option is required.

(Pre: as post, display message screen; Post: message screen displayed, activate keypad)

29. The Customer selects "No".

(Pre: as post; Post: read key input, send eject card message)

30. The ATM ejects card and bleeps (to alert the Customer).

(Pre: eject card signal; Post: card slot sensor set to off, sound bleeper)

31. The Customer takes the card.

(Pre: as post; Post: card slot sensor to on, bleeper off signal)

32. The ATM thanks the Customer.

(Pre: thank customer message signal; Post: display message)

33. The ATM displays a wait message until it is ready to serve the next customer.

(Pre: wait message signal on time out of previous message; Post: display message, reset current memory)

34. The ATM displays its readiness.

(Pre: wait message ends after ATM has reset; Post: display message)

Exceptions

2. The card is invalid, the ATM ejects it and asks the Customer to seek assistance.

(Pre: invalid card entry; Post: eject card signal, reject card message displayed)

4. The PIN is wrong. The ATM goes back to 2 twice more (to allow the Customer to retry), on the fourth try it retains the card and asks the Customer to seek assistance.

(Pre: validate PIN else on 4^{th} entry unable to validate PIN; Post: accept PIN else on 4^{th} try retain card, display message)

14. The two PIN entries do not match. The ATM goes back to 10 after giving the reason why.

(Pre: PIN entry registered; Post: reject PIN, display message)

22. The amount is invalid (i.e. above £30). The ATM returns to 18, after explaining why.

(Pre: amount read > 30; Post: display message, return to event 18)

Alternatives

At any point the Customer may press the "Cancel" button. If the ATM is in the middle of an operation it rolls back the operation and displays the list of options. If it isn't it goes to 30.

(Pre: Cancel button signal; Post: if in operation return to display options else eject card)

19. The Customer selects an amount that is on the list. ATM goes to 22.

(Pre: display list; Post: read input, goto event 22)

At 4, 8, 16 the Customer could select the options in any order.

(Pre and Post as main flow)

CREWS ATM

NB. Pre-conditions that state "as post" refer to the post-condition of the previous event

Use Case Name: Accessing the ATM
Actors: Customer, ATM
Context: The Customer wishes to get their balance, change their PIN and withdraw £25 cash with a receipt.
Pre-conditions: The Customer has approached a machine that is not in the middle of a transaction.
Post-conditions: The Customer knows their balance, has changed their PIN and has £25 cash and a receipt. The ATM is ready for the next customer.

Main flow of events:

1. The Customer inserts the credit card into the ATM card slot.

(Pre: ATM ready to receive card; Post: card slot signal off)

2. ATM validates card input.

(Pre: as post; Post: card recognised)

3. ATM prompts Customer to enter PIN.

(Pre: card validated; Post: display message, activate keypad)

4. Customer enters PIN via keypad.

(Pre: as post; Post: read keypad input, store PIN in memory)

5. ATM validates Customer.

(Pre: Retrieve PIN; Post: PIN validated)

6. ATM accesses Customer bank details from database.

(Pre: as post; Post: link to Customer Account, retrieve details to memory)

7. ATM prompts Customer with option menu.

(Pre: valid PIN; Post: display message on screen, activate keypad)

8. Customer presses "Check Balance" button.

(Pre: as post; Post: read keypad input)

9. ATM outputs Customer balance.

(Pre: system retrieves balance from memory; Post: display balance to screen)

10. ATM prompts Customer with "Another service?" option.

(Pre: as post; Post: display message, activate keypad)

11. Customer presses "Yes" button.

(Pre: as post; Post: read key input)

12. ATM prompts Customer with option menu.

(Pre: check input; Post: display options screen, activate keypad)

13. Customer presses "Change PIN" button.

(Pre: options visible, correct buttons activated; Post: system recognises correct button selection)

14. ATM prompts user to enter new PIN.

(Pre: set change PIN operation; Post: display message, activate keypad)

15. Customer enters new PIN.

(Pre: as last post; Post: system responds to key input, system validates input, stores number entered, show number of keys pressed with '')*

16. ATM prompts user to confirm new PIN.

(Pre: as post; Post: message displayed, key pad activated)

17. Customer re-enters new PIN.

(Pre: as post; Post: system reads key press, system displays '' per number entered, system stores number)*

18. ATM writes new PIN to Customer credit card.

(Pre: as post, system compares entered numbers, validates numbers, reads number to corresponding account; Post: card PIN rewritten)

19. ATM confirms PIN change successful.

(Pre: as post; Post: message displayed)

20. ATM prompts Customer with "another service?"

(Pre: as post; Post: message displayed, key pad activated)

21. Customer presses "Yes" button.

(Pre: as post; Post: read input)

22. ATM prompts Customer with option menu.

(Pre: check input; Post: display options screen, activate keypad)

23. Customer presses "Withdraw Cash" button.

(Pre: as post; Post: recognise key input, link to Withdraw Cash subroutine, automatically check Customer balance)

24. ATM prompts user with Cash Amount options.

(Pre: check input option; Post: display Cash Amounts screen, activate keypad)

25. Customer presses "£25" option.

(Pre: as post; Post: recognise key input)

26. ATM checks Customer has available funds.

(Pre: check amount selected; Post: amount selected < customer balance validated)

27. ATM asks Customer if receipt is wanted.

(Pre: as post; Post: display message, activate keypad)

28. Customer presses "Yes" button.

(Pre: as post; Post: read input)

29. ATM prints receipt.

(Pre: send print signal to receipt printer, send amount withdrawn, send account number, paper available, printer functions; Post: receipt printed)

30. ATM counts cash to correct amount.

(Pre: amount selected sent to cash dispenser; Post: correct notes chosen)

31. ATM writes new bank balance to Customer bank details in the database.

(Pre: retrieve amount selected; Post: customer account updated)

32. ATM prompts user with "Another service?"

(Pre: print receipt function completed?; Post: display message, activate keypad)

33. Customer presses "No" button.

(Pre: as post; Post: read input, start card eject function)

34. ATM ejects card.

(Pre: eject card signal received; Post: card sits in slot)

35. ATM ejects cash.

(Pre: cash counted; Post: cash slot sensor set to off, cash in slot)

36. Customer takes card and cash.

(Pre: card in slot, cash in slot; Post: card slot sensor set to on, cash slot sensor set to on, card slot empty, cash slot empty)

37. ATM resets for next card entry.

(Pre: cash and card slots empty; Post: system reset)

Alternative flows

8. Customer presses alternative "option" button.

(Pre: display Options Screen, activate keypad; Post: read signal)

13. Customer presses alternative "option" button.

(Pre: display Options Screen, activate keypad; Post: read signal)

23. Customer presses alternative "option" button.

(Pre: display Options Screen, activate keypad; Post: read signal)

11. Customer presses "No" button.

(Pre: display "Another Service?" message, activate keypad; Post: read signal)

21. Customer presses "No" button.

(Pre: display "Another Service?" message, activate keypad; Post: read signal)

28. Customer presses "No" button.

(Pre: display "Another Service?" message, activate keypad; Post: read signal)

33. Customer presses "No" button.

(Pre: display "Another Service?" message, activate keypad; Post: read signal)

Exceptional flows

2. Customer inserts credit card into slot incorrectly.

(Pre: card slot signal on; Post: card slot signal off, read card)

2.1 ATM ejects card.

(Pre: card unreadable; Post: card ejected, card slot sensor set to off)

4. Customer enters PIN incorrectly.

(Pre: enter pin message, activate key pad, read input; Post: PIN invalidated)

4.1 ATM asks for PIN re-entry.

(Pre: as post; Post: display message)

4.2 Loop back to 4.

(as 4)

4.3 If 4 has been exceptional 3 times then retain card.

(Pre: PIN inputted 3 times incorrectly; Post: eat card)

4.4 Prompt Customer to consult bank.

(Pre: card eaten; Post: display message)

15. Customer enters a new PIN already in use by another Customer.

(Pre: Enter New PIN message displayed, activate key pad, read input; Post: check PIN)

15.1 ATM prompts for alternative new PIN.

(Pre: PIN rejected; Post: display message)

15.2 Goto 15 in main flow.

(As 15 in main flow)

17. Customer re-enters new PIN incorrectly.

(Pre: Reenter PIN message, activate keypad, read input; Post: reject PIN)

17.1 ATM prompts for alternative new PIN.

(Pre: as post; Post: display message)

17.2 Goto 15 in the main flow.

(As 15 in main flow)

26. Customer does not have available funds.

(Pre: Customer selection for money withdrawal input, check balance; Post: predicted balance calculated)

26.1 ATM informs Customer "Insufficient funds".

(Pre: as post; Post: message displayed)

26.2 ATM prompts "Another service?"

(Pre: as post; Post: display message, activate keypad)

29. ATM has insufficient printing paper.

(Pre: print receipt signal received, check paper; Post: no paper)

29.1 ATM informs Customer "No receipt".

(Pre: as post; Post: display message)

29.2 ATM continues to 30.

(Pre: as post; Post: goto event 30 in main flow)

30. ATM has insufficient cash.

(Pre: cash amount for withdrawal selected by customer, check against amount in ATM; Post: result is too little cash in machine)

30.1 ATM informs Customer.

(Pre: as post; Post: display message)

30.2 ATM counts available cash and continues to 31.

(Pre: as post; Post: cash counted, goto event 31 in main flow)

PAGE

MISSING

IN

ORIGINAL

E4. Hypothesis 1: Answers to Specific Questions

CP ATM	Subjects	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	Total	Average
Specific Questions																											
Does the ATM check the Customer's card?		0	0	0	2	0	0	0	2	0	0	0	0	0	0	2	0	0	0	0	0	2	0	0	0	8	0.33
Where does this happen?		1	1	1	0	0	1	1	0	0	1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	21	0.88
What actor validates the card?		1	1	1	0	1	1	1	0	1	1	1	1	1	0	2	1	1	1	1	1	1	1	1	1	22	0.92
What checks the PIN number?		1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	25	1.04
Who is the actor?		1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	25	1.04
When is the Customer's balance displayed?		2	2	0	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	46	1.92
Where does this information come from?		1	1	0	2	1	2	1	2	1	1	1	1	2	1	1	1	1	1	0	2	2	2	2	1	30	1.25
How does the ATM get this information?		1	1	0	2	0	2	1	2	1	1	0	1	1	0	1	1	1	1	0	1	2	2	2	1	25	1.04
What happens if the balance cannot be displayed?		1	0	1	2	0	2	2	2	0	1	1	2	2	1	2	1	1	2	1	1	2	2	2	1	32	1.33
Where is the new PIN stored?		1	1	1	2	1	1	2	2	1	1	1	1	2	1	1	1	1	1	1	1	2	2	2	1	31	1.29
How do we know this?		0	0	1	2	0	2	2	2	1	0	1	2	1	1	1	1	2	1	1	2	2	2	2	1	30	1.25
How does the ATM know the Customer has funds?		1	1	1	2	2	2	1	2	1	1	1	1	2	1	1	1	1	2	1	1	2	1	0	1	30	1.25
When is the Customer's account updated?		1	1	1	2	2	2	1	2	1	1	1	2	2	1	1	1	1	2	1	1	2	2	2	1	34	1.42
How is the account updated?		1	1	0	2	1	2	0	0	1	1	1	1	1	1	1	1	0	1	1	2	2	0	2	0	23	0.96
Where does the ATM know the cash, card, receipt taken?		2	2	6	0	0	6	0	0	0	0	6	6	0	0	0	6	0	0	3	6	3	6	0	0	52	2.17
How does the ATM know this?		3	3	0	6	1	3	0	0	1	6	6	6	3	0	3	3	3	3	3	0	0	3	3	3	62	2.58
Total		18	17	15	30	13	30	16	20	13	19	25	29	22	12	22	23	17	20	18	23	27	28	23	16	496	20.67

Table E-1. Hypothesis 1 Results, Group A, CP Rules

CREWS ATM	Subjects	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	Total	Average
Specific Questions																											
Does the ATM check the Customer's card?		0	0	0	0	0	0	0	0	0	2	0	0	0	0	2	0	0	0	0	0	0	0	0	2	6	0.25
Where does this happen?		0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	21	0.88
What actor validates the card?		1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	22	0.92
What checks the PIN number?		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	2	1	1	26	1.08
Who is the actor?		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	25	1.04
When is the Customer's balance displayed?		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	0	2	2	2	2	2	46	1.92
Where does this information come from?		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	2	2	0	0	2	0	0	2	39	1.63
How does the ATM get this information?		1	1	1	1	1	1	1	1	1	0	1	1	1	1	0	0	1	0	2	0	1	2	1	0	20	0.83
What happens if the balance cannot be displayed?		1	1	1	1	1	1	1	1	1	2	0	2	1	1	2	0	1	0	2	2	1	2	1	1	27	1.13
Where is the new PIN stored?		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	0	2	2	2	2	2	46	1.92
How do we know this?		2	2	2	2	1	2	0	2	2	2	2	2	2	2	2	2	2	2	0	2	1	2	2	0	40	1.67
How does the ATM know the Customer has funds?		2	2	2	2	2	2	0	2	2	2	2	2	2	2	2	2	2	2	0	2	1	2	2	2	43	1.79
When is the Customer's account updated?		2	2	2	0	1	2	0	2	0	2	2	2	2	2	2	2	2	2	0	0	2	2	2	2	37	1.54
How is the account updated?		2	2	2	0	2	2	2	0	2	0	2	2	0	2	2	0	2	2	0	0	1	2	1	2	32	1.33
Where does the ATM know the cash, card, receipt taken?		5	5	5	5	5	0	0	0	0	5	5	5	6	6	3	5	5	0	0	6	0	4	0	5	80	3.33
How does the ATM know this?		0	0	0	0	0	6	0	0	0	0	0	6	6	6	3	0	0	0	0	6	6	0	3	3	45	1.88
Total		24	25	25	21	23	26	14	18	18	23	24	32	30	32	28	22	25	16	8	28	23	25	20	25	555	23.13

Table E-2. Hypothesis 1 Results, Group B, CREWS Guidelines

E5. Dependencies Identified

Group A (CP)

Depedencies		A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19	A20	A21	A22	A23	A24	Totals	
1	Pre	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	2	0	0	0	0	0	0	1	1	5	
1	Post	0	0	0	1	0	0	0	1	0	0	1	1	0	0	1	0	0	1	0	1	1	0	0	0	8	
2	Pre	1	1	0	0	0	1	1	1	0	0	1	0	0	0	1	0	0	1	0	1	0	0	0	0	9	card validated
2	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
3	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	2	
3	Post	0	0	0	0	0	2	1	1	2	1	1	1	2	1	0	1	0	1	1	1	1	0	0	0	17	PIN validated, link to Customer Account, input read
4	Pre	0	0	0	0	0	1	1	1	0	1	1	1	1	1	0	1	0	1	0	1	1	0	0	0	12	Valid PIN
4	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
5	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
5	Post	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	3	
6	Pre	1	0	0	0	0	1	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	6	
6	Post	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	3	
7	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
7	Post	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	3	
8	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
8	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
9	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
9	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	
10	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
10	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
11	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
11	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	
12	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	
12	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	
13	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
13	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

14	Pre	0	1	1	0	0	0	1	1	0	0	0	1	0	0	1	0	0	0	1	1	1	0	0	2	11	Compares PIN entries, new PIN saved
14	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
15	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
15	Post	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	3	
16	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
16	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
17	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
17	Post	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	1	0	0	0	0	6	
18	Pre	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	3	
18	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
19	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
19	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
20	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
20	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
21	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
21	Post	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	
22	Pre	1	1	0	0	0	0	1	1	1	0	1	1	1	0	0	0	0	0	1	2	0	0	0	1	12	withdrawal less than balance, check ATM has enough money*
22	Post	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	1	3	
23	Pre	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0	5	
23	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	3	
24	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	
24	Post	0	0	0	0	0	1	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	4	
25	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
25	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
26	Pre	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	2	4	
26	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	
27	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
27	Post	0	0	0	0	0	1	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	4	
28	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
28	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
29	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
29	Post	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	2	

30	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	
30	Post	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	3	
31	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
31	Post	0	0	0	0	0	1	1	0	0	0	0	1	0	1	1	0	0	0	1	0	0	0	0	0	6	
32	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	
32	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	2	
33	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	
33	Post	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	5	
34	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	
34	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	
E2	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
E2	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
E4	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
E4	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
E14	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
E14	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
E22	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
E22	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Aany	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Aany	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
A19	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
A19	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	Totals	4	4	2	3	2	12	11	9	4	2	9	10	6	4	6	5	0	6	24	11	5	1	1	16	157	

Group B (CREWS)

Depedencies		B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17	B18	B19	B20	B21	B22	B23	B24	Totals	
1	Pre	0	1	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	4	
1	Post	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1	0	0	0	1	4	
2	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

2	Post	0	1	0	0	0	0	0	1	0	0	0	1	0	1	1	1	0	1	0	0	1	1	1	0	10	card recognised / validated
3	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	3	
3	Post	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	3	
4	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	
4	Post	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	3	
5	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
5	Post	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0	1	0	0	0	6	
6	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
6	Post	0	1	0	0	1	1	0	0	0	0	0	1	0	0	1	0	0	0	0	1	1	1	0	1	9	link to Customer Account
7	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
7	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	
8	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	
8	Post	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	3	
9	Pre	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	3	
9	Post	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	3	
10	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
10	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	2	
11	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
11	Post	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	3	
12	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
12	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	
13	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
13	Post	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2	
14	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
14	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	
15	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
15	Post	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	1	5	
16	Pre	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	
16	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	
17	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
17	Post	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2	
18	Pre	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	1	1	0	0	0	0	0	6	
18	Post	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	3	

19	Pre	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
19	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	
20	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
20	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	2	
21	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
21	Post	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2	
22	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
22	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	
23	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
23	Post	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	3	
24	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
24	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	
25	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
25	Post	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	3	
26	Pre	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	2	
26	**Post**	0	1	0	0	0	1	1	0	1	0	0	1	1	0	0	0	0	0	1	1	0	1	1	0	**10**	withdrawal less than balance
27	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
27	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	
28	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
28	Post	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	3	
29	Pre	0	1	1	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	5	
29	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
30	Pre	0	1	0	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0	0	0	0	5	
30	Post	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	3	
31	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	
31	Post	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2	
32	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
32	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	2	
33	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
33	Post	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2	
34	Pre	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
34	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
35	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	

35	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	Post	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	1	1	6
37	Pre	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	3
37	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A8	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A8	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A13	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A13	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A23	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A23	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A11	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A11	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A21	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A21	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A28	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A28	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A33	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A33	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E1	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E1	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E1.1	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E1.1	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E4	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E4	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E4.1	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E4.1	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E4.2	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E4.2	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E4.3	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E4.3	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E4.4	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E4.4	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

E15	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E15	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e15.1	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e15.1	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e15.2	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e15.2	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E17	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E17	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e17.1	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e17.1	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e17.2	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e17.2	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E26	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E26	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e26.1	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e26.1	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e26.2	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e26.2	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E29	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E29	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e29.1	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e29.1	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e29.2	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e29.2	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E30	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E30	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e30.1	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e30.1	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e30.2	Pre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
e30.2	Post	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Totals	0	10	15	0	2	10	2	3	4	0	0	6	7	5	7	2	3	8	7	4	3	32	4	8	142

E6. Classes and Actors Identified

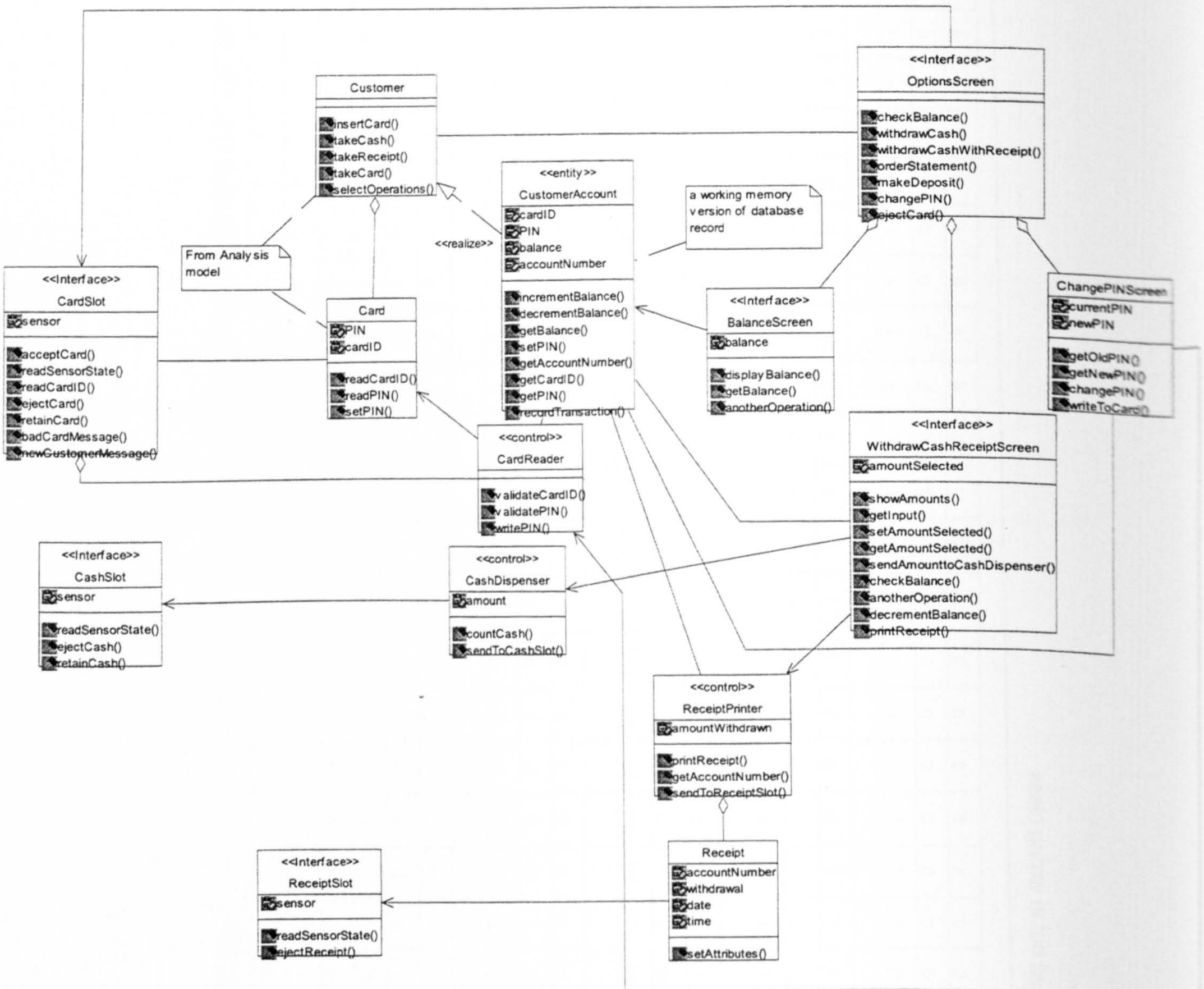
CP ATM	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19	A20	A21	A22	A23	A24	Totals
Correctly identified classes	1	3	1	2	3	3	2	0	1	0	6	0	2	3	1	1	3	0	1	1	3	2	2	0	41
Other classes	0	2	0	2	3	5	2	0	0	0	1	3	2	1	1	1	2	0	1	0	3	1	1	0	31
Correctly identified operations*	1	5	5	0	5	4	1	0	0	0	8	0	4	7	4	5	7	0	0	6	5	4	5	0	76
Other operations	6	26	0	11	7	0	21	5	0	2	3	9	11	2	18	5	13	0	0	1	3	7	10	0	160
Correctly identified attributes*	0	0	0	4	2	3	5	0	6	0	4	0	1	0	0	0	5	0	0	0	0	0	0	0	30
Other attributes	0	8	3	13	2	1	9	0	0	10	5	4	1	1	3	0	1	0	0	2	0	2	1	0	66
Missing Actors																									
Bank System Database	0	1	1	1	1	1	1	1	0	1	1	1	1	0	1	0	0	0	1	0	1	0	0	1	15
Connection Machine	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Other Actors	0	1	0	0	0	4	1	0	0	0	2	0	1	1	0	0	0	0	1	0	4	0	0	4	19
CREWS ATM	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17	B18	B19	B20	B21	B22	B23	B24	Totals
Correctly identified classes	2	2	2	4	3	8	4	3	3	2	1	6	4	3	4	1	1	3	3	3	4	4	4	3	77
Other classes	1	2	3	2	3	7	1	2	2	1	1	4	1	1	2	1	2	1	2	2	2	3	4	3	53
Correctly identified operations*	0	3	2	1	6	4	0	2	5	5	1	7	2	1	5	0	0	2	4	5	5	3	3	4	70
Other operations	0	3	3	3	4	6	7	3	5	10	12	8	0	5	12	1	2	5	10	9	7	1	13	8	137
Correctly identified attributes*	0	1	0	2	0	2	0	1	0	1	0	2	1	0	9	0	0	3	3	2	6	1	1	7	42
Other attributes	4	4	0	2	1	4	2	3	0	0	2	3	0	0	4	2	2	0	0	0	5	0	6	0	44
Missing Actors																									
Bank System Database	1	1	1	0	1	0	0	1	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	17
Connection Machine	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
Other Actors	0	1	1	0	0	0	0	1	2	0	0	2	1	2	1	0	0	1	0	1	0	6	1	4	24

** operations/attributes of correctly identified classes*

"Other" classes - incorrectly identified classes

"Other" operations and attributes correctly identified but in wrong class

E7. ATM Class Diagram



c7 ATM class diagram

Appendix F

Case Study Documentation

F1. Project F Documents presented to the Author

F1.1 Meeting notes 12/03/01 of Current Project F Process

These notes were presented to author as an outline of the Project F process and problem.

Current process

There are approximates 20 pack code types.

Customer applies

Account Opened with Transfer of stock or not –	<Front Office (FO) allocates pack code>
Data passed to Back Office (BO)	<does it need to go via BO?>

Ordinary account + Company X numbers generated at BO (Bank Of Scotland)

BO sends file to Print server (text file sent 3 times a day – 9am 12am and 3pm)	<	I drive>
Three types of files for Company X –	SM	application data
	SA	existing customer
	SI	ISA customer
Two type for V. (a company using Company X) -	VM	application data
	VA	existing customer

Manual copy of file from Print server to INBOX in Marketing drive.
Links to Ms Access
Import file into Database
Program will de-dupe on pack type and only generates a file with A1. Also sorts by pack type.
MS Access is currently in Marketing directory and needs to be moved
File should go straight into database.
Print off each pack type.

Select A1 if there are 20 applications it will open all the relevant Word files and merges the data into the file. Operator has to select Merge to Print. This will request the printing in the correct file format.

If C1 operator has to open up the relevant forms and print them (will be pre filled) but has to manually collate them into the envelope.

Volume handling

1 all applications go out same day
2 If over 1000 a day we have an arrangement with a Mailing house to off load some applications.
3 Arrangement with Mailing House (based in Sussex)
   a. we need to give them 1 weeks notice
   b. only send A1 and ISA packs (IN)
   c. have to supply them with Application forms, Envelopes and postal Docket book (as we get billed by the post office)
   d. Print Room Staff exports a text file, encrypts the file and emails it to Mailing House.

Arrangements

Printing of envelopes and applications are done at QPress who also store the supply for us. Based in East Sussex.

Print House report on how many packs have been sent out.
Print Room Staff has to reconcile the packs sent by Mailing House and the mailing dockets.

Contract with Mailing House is on a project basis.
We should be able to re-negotiate pricing based on A Company (multinational) mailing
They will handle white labels (companies that use Company X product in their own company - Company X act as the tool for these companies).

Issues

1. Updates do not flow through

When a client changes details of the application prior to the account being opened they have to phone Customer Services for the update to be actioned. The data is not passed on automatically down to F Project. The PAF override feature should have reduced the workload however, it still needs addressing.

2. Manual changes

Customer Administration department provides Print Room Staff with a list of Customer numbers with details. Print Room Staff will print out new application packs and send out. There is no communication to the client that the changes have been incorporated etc.

3. Outstanding items to progress with

   Bar codes – will we be using them in the future in conjunction with scanning?

Questions

1. Does the data have to go through Back Office?

2. How will Mailing House work?
3. How will we accommodate changes?

Recommendations / observations

1. Automated link from Front Office to production of pack
2. Ideal if Front Office should have a field where you can track if the pack has been sent out – date stamp and pack type to be recorded
3. Button on internal system to request new pack (having made changes) – update log
4. Ability to modify the pack on the fly
5. Cater for white label and Multinational A.
6. Reduce management of databases from 2 to 1

Process:

Customer applies

Account Opened Front Office allocates pack code

Data passed

Back Office

Send file

Print Server server

3 times a day

9am 12am and 3pm)

Company X – V.-

SM application data
SA existing customer
SI ISA customer

VM application data
VA existing customer

F1.2 Rapid Impact Assessment

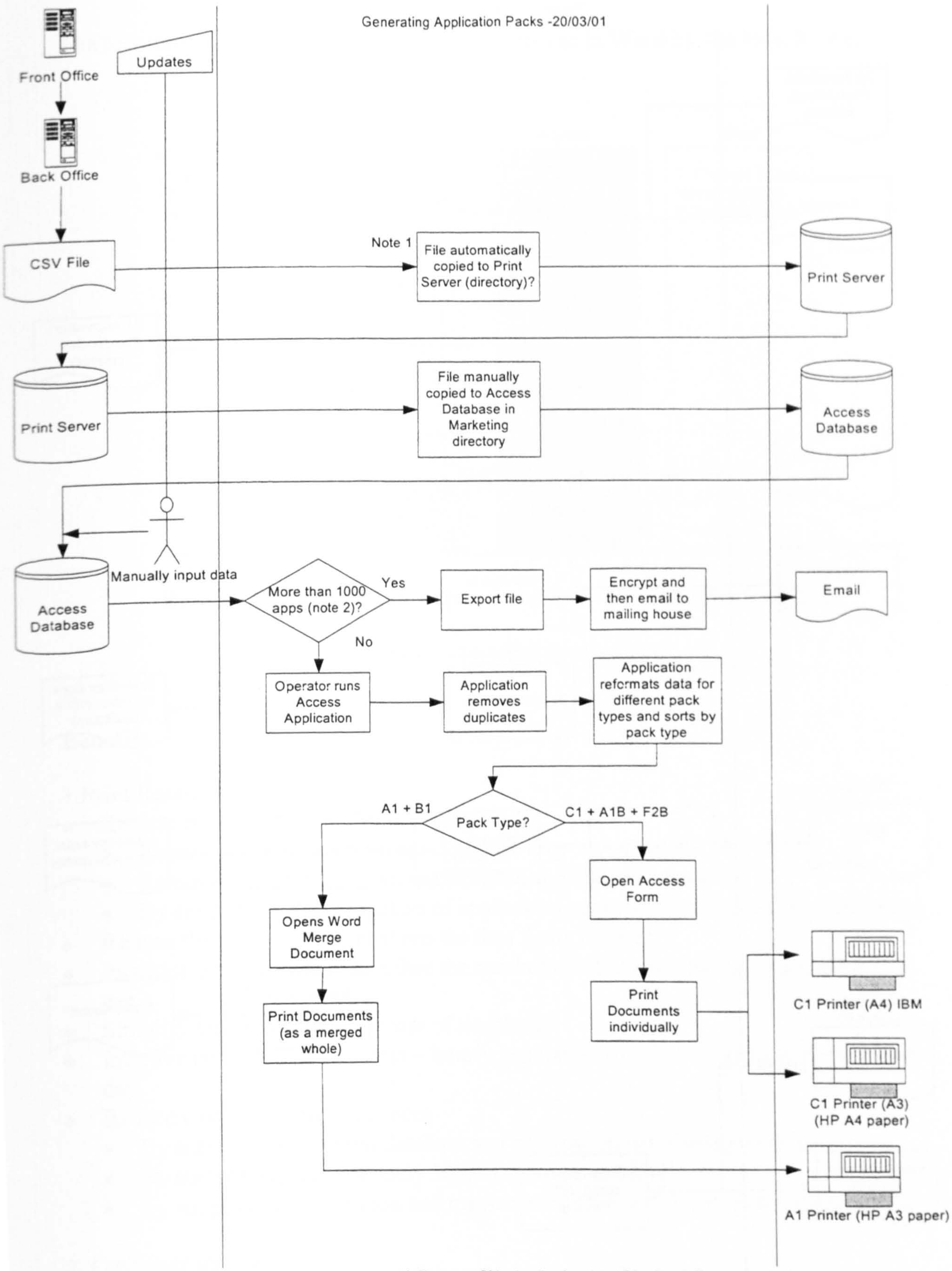
Current Process

Four applications are in use and the activities / tasks are listed for a better understandingof the procedures being followed today.

Front Office	Back Office	Access	Word
1. Provides all information	1.Provides Client No. 2. Provides formatting	1. Tanslates flat file 2. Removes Add 3. Dedupe 4. Format name 5. Export CSV file 6. Re-generation of file 7. Handle updates 8. Reporting	1. Change Word Merge location 2. Merge to Print

Figure F1-1. Overview of Current Process

The following flow diagram shows the workflow in practice today.



Note 1 - this process occurs three times a day and delivers 3 file types: SM - Application data, SA - Stock Transfer info, SI - ISA application
Note 2 - Mailing House needs more than a weeks notice for this process to be in place. All stationary must be supplied to the Maling House within this time (via QPress)

Figure F1-2. Generating Application Packs

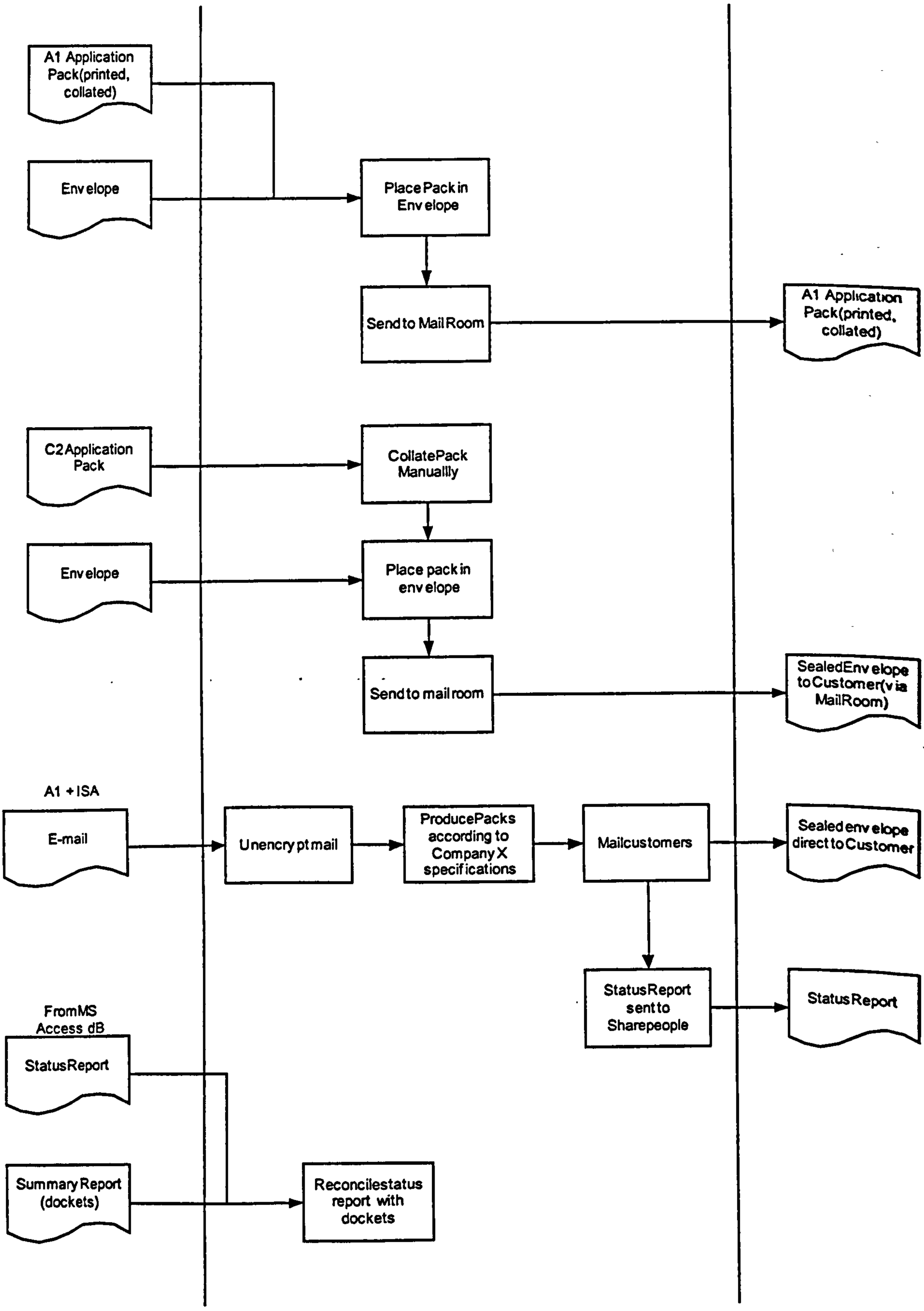
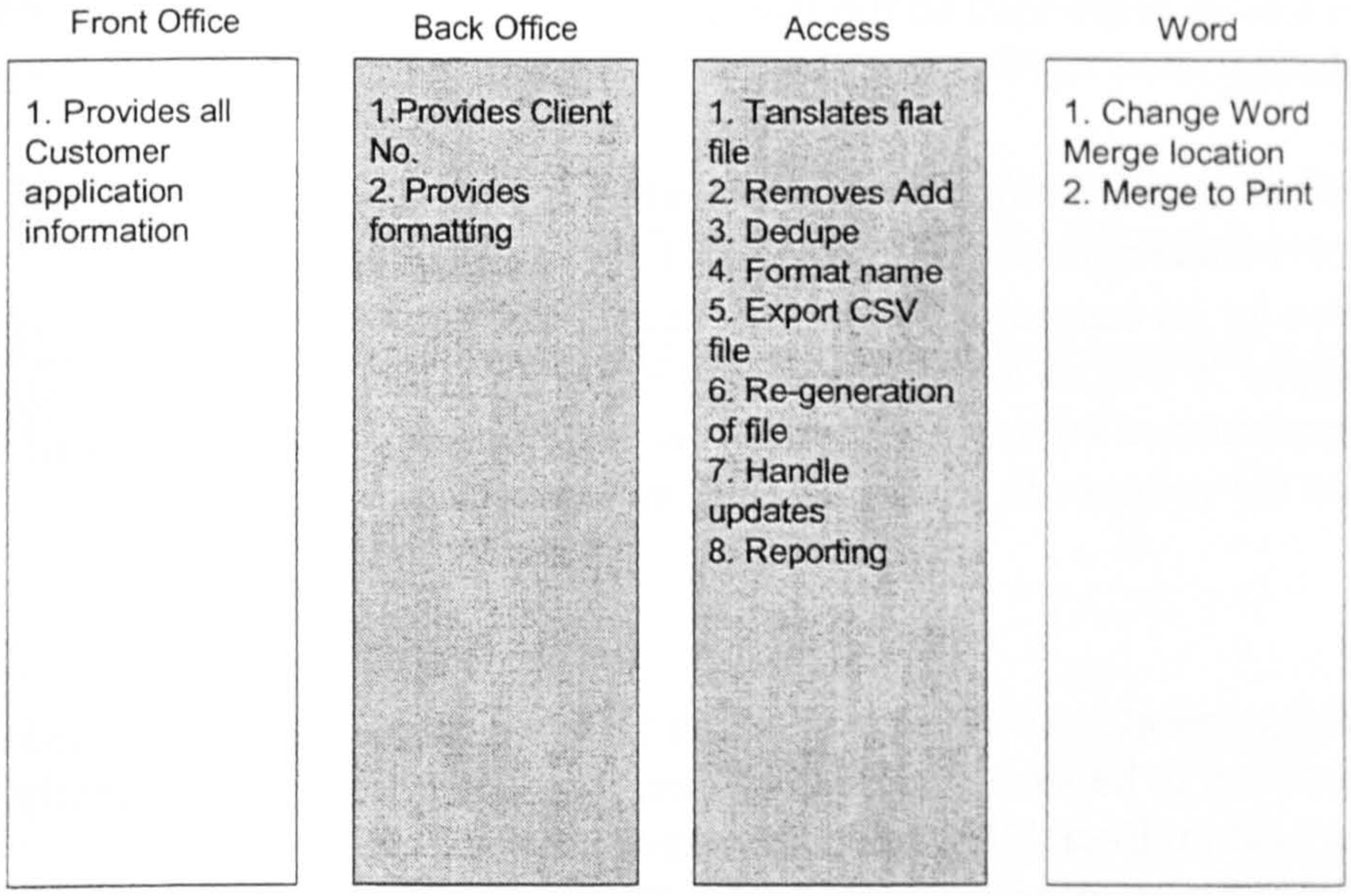


Figure F1-3. Packing and Mailing

Proposed solution

The recommendation is to produce a file for direct use in Word by the Print Room.



By-pass Back Office and Access Database – Produce file for Word

Figure F1-4. Proposed Solution Overview

Benefits

1.Print Room

- Reduce costs of development of print process
  - Remove Back Office from development loop
  - Remove standalone MS Access database application
  - By simplifying the production of application packs and customer communication.
- Reduce the number of applications the data flows through
- Provides opportunity to rationalise the number of letters and pack types currently in use
- Simplify the maintenance process of templates
- Improve on frequency of output – hourly generation of work as compared to 3 times a day.
- Reduce cost of staffing print room
  - By removing MS Access database and application processing workload
  - By automating approximately 50% of current workload
  - By simplifying the creation and maintenance of Word documents currently in use.

2. Customer services

- Will be able to respond to queries related to application status as they will be able to see the date when it was mailed in the audit log

- Updates will flow automatically and new pack sent out when customer advises change of details.
- Manual update of application pack no longer required e.g. change of address as this is now automated
- Automatic process for re-sending application packs

3. Marketing

- Improved data mining, as we will be able to track conversion times more accurately
- Improved management MIS on success of campaigns
- Allows for an easy and effective method of creating new pack types with a 10% effort
- Company X will be able to cope with expected increased number of customers and correspondence volumes. Currently we have an overflow arrangement with a Mailing House for volumes in excess of 1000 applications per day.

4. IT

- Provides opportunity to clean up Back Office database. Only activated customer information to be stored in Back Office. Applicant details are currently unnecessarily being held on Back Office due to current practice.
- The technology interface with Access Database will no longer be required. Reliance on external contractor to fix problems will be alleviated and maintenance costs reduced.
- Supports future plans of allowing customers to print applications locally
- Provides platform for implementing complete automated solution whereby Print Manager software will automatically print, collate and stuff envelopes without the need for human intervention.

5. Regulatory

- Reduced regulatory risk with a simplified structure. Complete audit trail is available for inspection.

Additional Revenues / Cost Savings

1. Awaiting data.

Risks

The following risks have been identified:

1. MS Access Database is not capable of supporting volume. Increasing the size of the hard disk is not a solution. Already the database is in danger of failing. An immediate solution is required.

2. If printing fails the only resolution is to re-run the entire batch costing wastage of time and materials.

3. Manual collation of pack content from different printers increases the risk of customers receiving wrong documentation e.g. another customer application form or stock transfer form is sent incorrectly.

4. With the Print Room chief leaving there is no expert on the current process, leaving it vulnerable to error or failure. The new process will not be dependent on one resource (Print Room chief). With the new solution there will be no single point of failure.

5. Customer data changes have to be manually printed and entered into the database increasing the risk of customer details not matching the Front Office data.

6. We need to ensure that adequate resources are available for the project to be completed successfully.

F2. Problem and Requirements Elicitation Notes and Feedback

These sets of notes were taken whilst in discussion with members of the IT team and also in observing the Print Room staff describe the process they go through to print a batch of applications. The notes are directly copied from the notebook into which they were jotted down, thus their appearance appears scruffy (Bray 2002, p.337). Where expansion is necessary, this is done in '[ ]'.

F2.1 Project F Problem Description (notes taken from informal interview with Project F Project Manager)

"[There is an] application form problem. [Project F is about the] set up of accounts (and funding accounts).

This is a re-engineering problem and essentially we want a design solution [I can provide only the requirements end and partial designs].

A fundamental business need (requirement) is to provide application [forms] to clients. Set up of system in place but it is feeble.

Print room issues a concern: used to send printing to an external company but marketing issues are a big problem. There are political concerns with the printing company. An ex-Director insisted on using them. [The name of the printing company is kept out of this thesis at the request of Company X.] There is a rival company. [The person in charge of the Print Room] wanted to set up own print room to deal with volume issues. In-house printing set up. Envelope stuffers and print packs a problem. The person in charge of the print room is leaving [within a week of this interview – there will be some print room politics because of this].

A goal is to move into the European market, to have a European product range and contend with EU regulations. 23 pack types a problem.

[Step back – pack types needed? Pack type link and client relationship????]

The Project F Process

Customer applies [online], a credit check is done. The credit check is not 100% automatic. Utilities bills are also asked for [2 current bills] (was this but no longer – this can be overridden). Problem with the credit check is that there is not record of the date of the credit check made – marketing would like to know the date of the credit check. .

A customer number is created (logon) and access given to the system but customer can't trade yet – not issued with a bank account.

Credit check done (instant reply) – there are different levels of response. Based on the response an account is issued. Flag Back Office to inform of account. Bank of Scotland then confirm credit status and account creation. Only then can the Customer be accepted. A trading account is then accepted.

Problems: client contacted and override problem [????].

On applying, options:

Transfer stock from other tradings previously done (because customers do not always have spare cash)

Enter stock details (this is vetted and only acceptable stock agreed)

Stock Transfer form sent back to client with application form for account (a legal requirement [at the time of the study])

If no stock transfer, the account application form is sent to the client.

Client (customer) signs form and returns it.

Customer profile:

Single person account

Joint account (only residents in the same country – the UK)

Products:

Trading Account

ISA (maxi-ISA) max of £7000 – investing stocks and shares. (Future product should allow the transfer of PEPs and mini-ISAs.)

Call centre [outside scope of this case study]:

Customers call in. A call rep initialises letter and system pro-fills form to a standard template (there are 45 standard templates!)

Extend out to letters (from pack types) ????

The aim is automate the whole application printing process [from the perspective of the project manager] and send letters by outsourcing. Should also be able to print from here in special circumstances. Outsourcing dependent on cost and in-house robustness.

The current out-sourcing printing company print with only 3 colours. The current Company X logo has 2 colours luckily – used to be 4 colours and the printing company said they couldn't print it.

Predicted customer volume 100,000 (19,000 at the moment [of the study]) and aiming for 600 applications a day.

Printers:

There is printer manufacturer politics! IBM unreliable machines but it is policy to use them! The aim is to automate the whole printing process but this needs a cost / benefit / risk analysis [outside the scope of this case study].

Terminology:

SM: application data / direct debit data

SA: application and stock transfer

SI: ISA

C2: transfer stock

These print jobs are generated 3 times a day.

*Goal is to rationalise pack types and reduce them down to 2 if possible.

*Future goal is to email customers whilst their application is being processed."

F2.2 Print Room Process Observation Notes

These notes were taken whilst observing a Print Room Staff (PRS) member do two print jobs and in discussion with other members of the print room team.

PRS transfers in a SA (Application and Stock Transfer) file from the Back Office server (via the Print Server – want to get rid of this server). The SA file is copied to the Access Database by dragging and dropping the file to the Access Inbox.

Options on Access dB:

Import Data selected on Access dB

Select Pack Type and number

Print mailing list – internal

Export mailing list – mailing house text file

Undo

Example internal printing of mailing list demonstrated:

Select pack type from drop down box list and number to be printed.

Option to print A1 pack type.

Print selected

Pack type selected imported into Word mail merge function.

Mail merge occurs.

Printer is manually selected and PRS checks correct paper is available (paper size and in correct tray).

Print function selected in Word and the printing occurs.

Different pack type is selected and the print process begins again. This time ISA applications are selected. PRS has to change the paper forms in the printer.

Tray selection problem identified. The correct tray is not always printed to (though the correct tray has been selected by PRS). Is there a printer driver problem here?

Document types should be associated to print trays and automatically collated – though they are currently not.

Date problem on some Word letters – the date is sometimes, apparently randomly, omitted.

The number of pages printed per document can also be a problem. Often a blank page is added to the print for unknown reasons (might be a problem with Word). This only occurs on one (popular) pack type so the PRS has to select to print only pages 1 and 2.

Once printed forms are collated and piled up with an identifying sticky label (Post It note) applied to the top form so that the envelope stuffer can select the correct envelope size from many and should also be aware of the number of pages to be stuffed per envelope. [The inattentive stuffer might easily make a mistake.]

It is possible to undo previous mailing – this just lists the run at merge.

- Applications from Jersey and Guernsey are not printed since they are not subject to the same tax regulations as the United Kingdom. A check on post codes is done and any from these islands are pulled from the list. The potential customer is emailed of their misfortune.
    - The file imported from the Print Server is placed in the In Box and the text from this file goes into a new table: tbln : Table
    - Name formatting automated – pulls out Channel Islands records automatically.
- The Access dB then checks for duplicate records. It is possible that the customer clicked 'apply' twice or that a record was added manually by the PRS – this does occur when applications come in by phone or by post.
    - Duplicates are appended to tblProcessed : Table

It is possible to list records that are used e.g. duplicates and any import errors, in an Excel spreadsheet. It is also possible to update the dB from the spreadsheet.

There is a follow up system but this is not used – there is a lapse in days caused by a synchronisation problem.

You can switch databases dependent on client.

Can create a new database.

A back-up database to a local machine is required because the database crashed (and this is why IT want to get rid of the Access dB).

It is possible to check pack variations through a Report option. E.g. Pack (total), Pack (%).

There is 1 post per day: 6.30pm.

F2.3 Notes of meeting with IT team working on Project F

The following figures are provided by the Systems Analyst working on the Project F project. Figure F2-1 provides an overview of the current Project F process. The proposal is to put everything in the Front Office i.e. to remove the Back Office and Access from the chain. Even Word is to be automatically started from the Front Office. [It is interesting to note that the Print Server has been missed out. When asked about this, the Systems Analyst said that the Print Server only stored a copy of the file containing applicant details and so was not worth mentioning; that is, the Print Server does not do anything so why include it?]

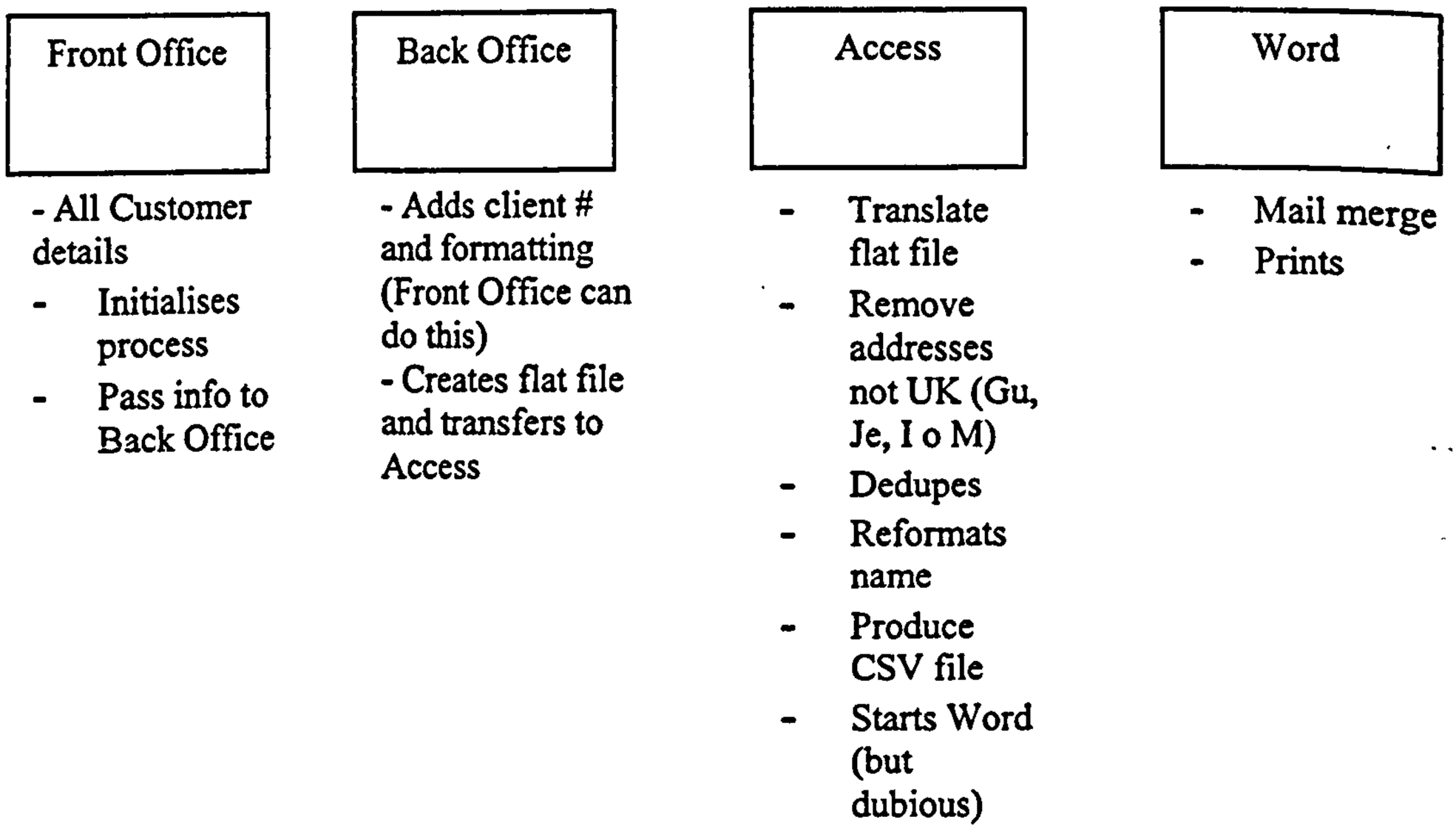


Figure F2-1. Overview of the current Project F process as presented by the systems analyst

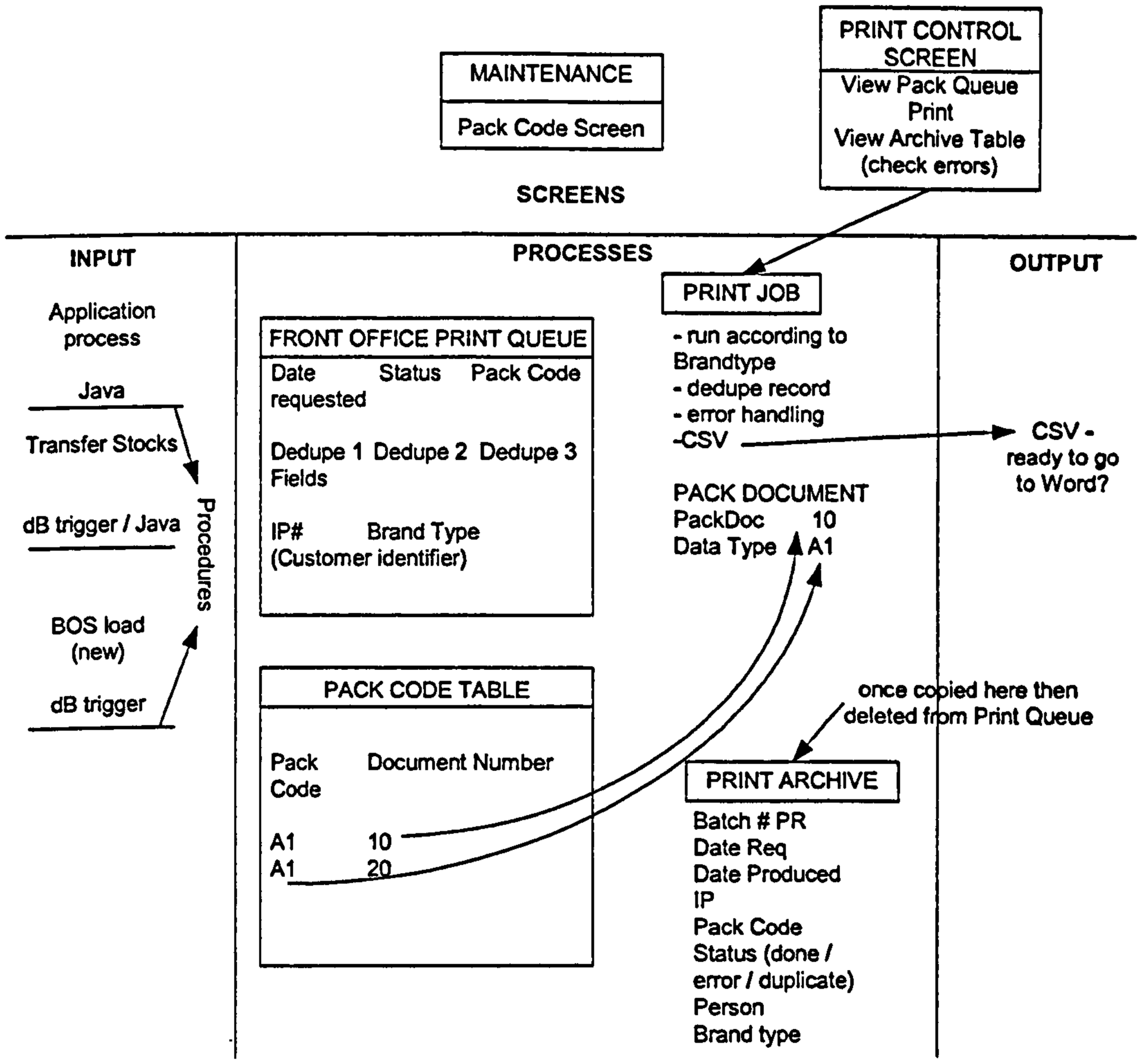


Figure F2-2. Detailed proposed design of Front Office as presented by the systems analyst

Me - How do you alter pack code types? Isn't it better in the procedures rather than in the database tables?

Systems analyst – unsure of this at present.

Me – What no business model i.e. what's the impact on the design, on the java between the interface and the database (see Figure F2-3)?

Systems analyst – we'll let the java boys worry about that.

Me – All deduping to be done manually from the database? Why not do it at run-time? It'll remove possible human error.

Systems analyst – Changes to applications come in over the phone so details get altered when they are already in the database (for example, if a Customer changes their address). Sometimes a new record is created and the old one is unaltered. It's simpler to remove duplicates only from the database.

[After this meeting, the author discussed the approach taken by the IT team working on Project F with the IT Development Manager. He said: "What we want to know is the impact of the database changes on the design? They [the Project F IT team] should be concerned with the Project F process from start to finish, its impact on the printing process, the application, on the underlying business model and then about the database. Altering the database tables is the last thing they should be worrying about. It's typical of the work here."]

Order CSV file by post code, name, surname – make sure enough space to fit in.

Front Office (Figure F2-2) new screen to be added: Print Packs for individual customer (record of the last 3 months).

The output is a CSV file. Word uses .hdr, .csv and .doc files so this will produce generic CSV files. It makes updating easy. The CSV file has 140 fieldnames so this is better to use that .doc files since they only have 30. The goal is to open a document, point to Word and mail merge. Also want to automate a bit more of the printing in terms of automatically selecting trays for printing. [Automatic tray selection had been achieved by the time this case study was completed.]

The CSV file will be written to a shared directory that the Print Room staff can access. That means there won't be any copying. Just print directly from the Front Office.

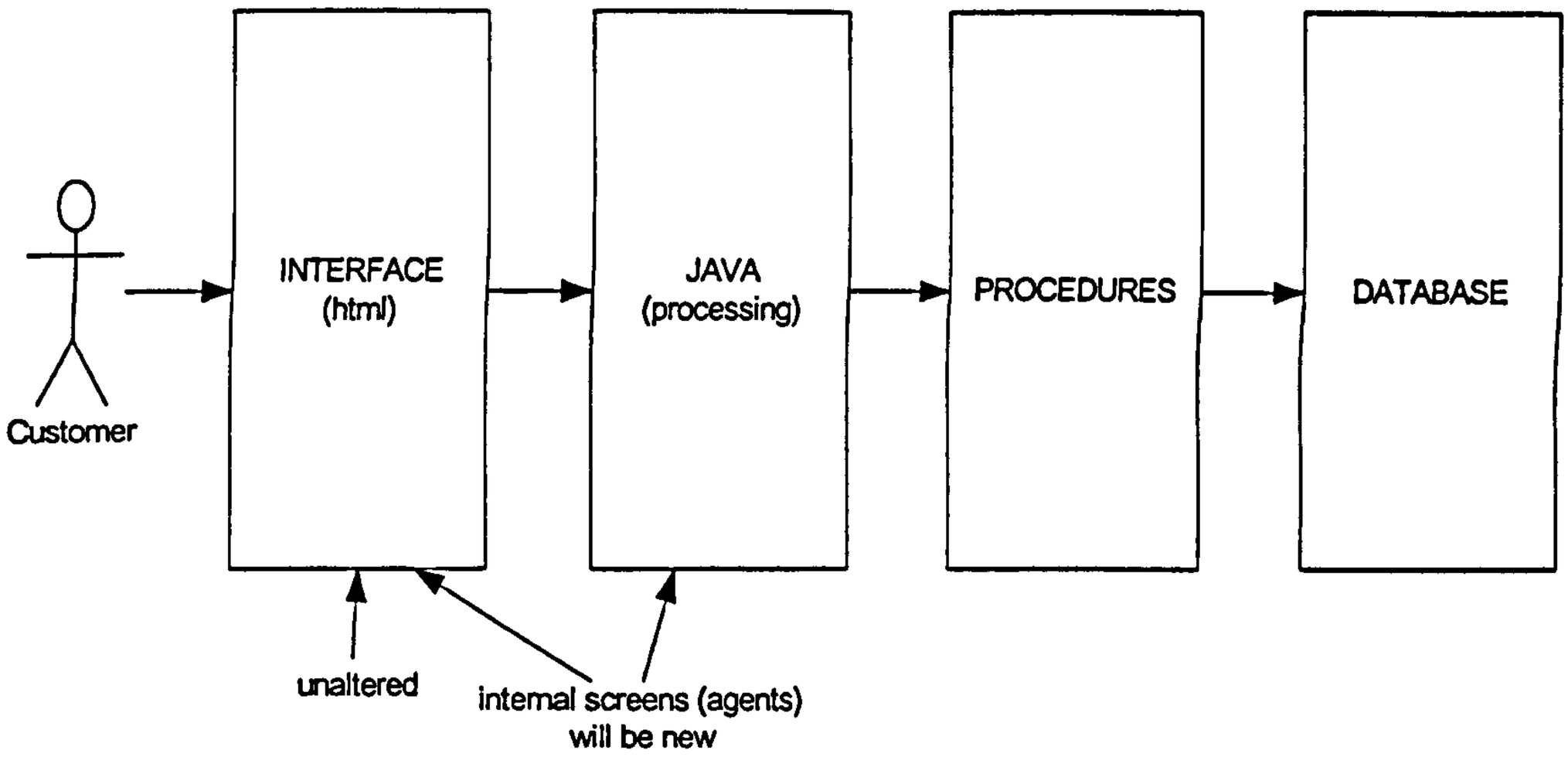


Figure F2-3. Overview of Front Office as presented by the systems analyst

The internal screens will be accessed only from within the company intranet. The screens are:

- Maintenance - a Pack Code screen (to make sure new Pack Codes are added or if a Customer record needs altering).
- Print Control Screen - with options to view pack queue, print and view archive table (which allows error checking)

F2.4 Feedback notes from interviews and presentations

Introduction

Two types of feedback have been obtained. There are a number informal comments provided by various employees at presentations given by the author. There have also been three formal interviews conducted. It would have been worthwhile conducting three interviews with others involved in the project but due to very heavy schedules and maternity leave, this might not be entirely possible.

Presentation Feedback

This will take the shape of groupings of comments that, though unrelated, all refer to various elements of the EARTH method. Feedback comments are left as plain text in this font. Comments added by the author, which include feedback directly to the project team members and discussion on issues that need further consideration, are enclosed in '[...]'.

Role Activity Diagrams

"RADs good in that they describe the process and what's involved very clearly. We've never had models like this and the business people [business managers, marketing department within the company who often represent the customers] need this kind of model (though they wouldn't understand it at all!)" - IT Development Manager

You forgot to model the states of the Customer (registrant, applicant and active). [Actually, I have modelled the states of the Customer in the envisioned process model – as pointed out to the stakeholder. These states are discussed in detail, with a state transition diagram, in the explanatory documentation about the RAD especially concerning the identification of another 4 states the Customer can be in – 2 of which are considered unknown, we don't know what happens to the Customer in these states - see Appendix F3.]" - IT Development Manager.

"The RADs are very useful. It would be good to get hold of RolEnact to use as a validation mechanism." - Project F Project Manager.

"Clearly see how everything is involved in the Project F process." - Software Developer.

Use Cases

"Use cases good for testing and for reshaping the application process at the interface. The online application process is too long [and gets longer with the ISA and PEP applications] and no one has ever produced this kind of document to see the exact steps required to complete an application. The trouble is time constraints on getting this done – we don't have time to do all this." - IT Development Manager.

"Do use cases help in the identification of attributes? [Classes are elicited from use cases that represents what happens at the graphical interface i.e. what the Customer types in. However, the EARTH method does not explicitly address this concern. See next section for more on this.]" - IT Development Manager [very good point made.]

"Use case diagrams give an overall useful picture of the main functions of the system. Quality Assurance would be interested in the descriptions [for validation and testing]." - IT Development Manager

"Use cases are very worthwhile for an overview. Need to get a tool such as Rational Rose." - Project F Project Manager.

"The use case diagram is an excellent diagram to present to the business as an overview of what's to be done [i.e. functionality]. The descriptions are very important for the finer details and the step-by-step operations. This is very useful for both IT and QA." - Company co-Founder and IT Vice-President.

Questioning Use Cases

"The Question Sets raise questions about the design and important issues of complexity such as how we manage the uniqueness of the Customer Number and is there sufficient check to avoid a duplicate record?" - IT Development Manager. [Refers to examples presented on a slide in a feedback presentation session conducted by the author. The exact

use cases are iUC1: Applying for Individual Trading Account, event line 36 and bUC1: Determine Type of Application, event 14. First, the generation of customer numbers can be corrupted – unsure how of this. Second, when should the dedupe take place? At run-time or in the database? The product is very data conscious; it is data-driven design so it's very important the correct data goes into the correct objects. The question sets helps this cause though there isn't any specific question to address this important point. We can't guarantee success with this approach but we can get the thinking processes working in the right direction. It also depends on the detail of the use case description. If we have a higher abstraction then there's going to be less focus on the attributes of each class. Our questions are more geared to finding relationships between classes – this maps more closely to the RADs. So here's a weakness of the approach but I'm not sure how to address this. Do we describe in so much detail that we can't see the wood for the trees and drown in analysis paralysis, or do we take a step back and observe at a slightly higher level. I think we should add a further question or set of questions to help us to not only tease the attributes out but also get us thinking where exactly which objects they should be placed in. Again, the devil is in the detail and the detail might be the devil to end a project prematurely if we really delve so deeply into our descriptions or always write the descriptions with so much detail as actually the case in this case study.]

"Questioning the use cases is very detailed and good to help in the design." - Company co-Founder and IT Vice-President.

Packages

"The package diagram reflects the current Front Office architecture, with the database package consisting of java connectors to the database itself. So you got your model is right!" - IT Development Manager.

Overall comments on method

"Good that everything is validated from business model through to design. We can see the clear progression from one phase to the next. We can specify what we want very clearly." - IT Software Developer.

"Concern with data identification. EARTH process seems to be process driven. The overriding object-oriented design approach is data-driven. Objects must have the correct attributes and this concern is not explicitly addressed by the EARTH method." - IT Development Manager [See comments on *Questioning use cases* for more on this.]

"Overall process is too long [for Company X] primarily because of time-to-market constraints. Slight concern that it's very process driven in appearance. The IT department need to look at the use cases onwards to help them design and code, and the business people ought to do some RAD and use case work (though they wouldn't have a clue about them)." - IT Development Manager

"The EARTH process is worth using but there will be cultural problems in its take-up. This [the author's work] is very close to the actual work carried out on the project [although the Company X approach was to work from the inside out and the author's the outside in]. We have very similar understandings of the problem and proposed designs [I'm not sure exactly where these design similarities are!] Want to integrate both sets of work carried out to present as a full documentation set to the business." - Project F Project Manager.

"Overall process well worth considering for the IT department but it's more likely that each department / area would take what's useful to them and ignore the rest. Everyone can take little bits of this method for themselves and use that." - Company co-Founder and Vice-President.

Interviews (Appendix F2.4)

The interviews were formally structured. That is, a specific set of questions were asked in the same order to all stakeholders.

Interview 1

Stakeholder Role: Software Developer

Action Research bit about the stakeholder: Degree in Computing. This is the stakeholder's first job in the software industry.

1. Can you describe your current requirements analysis and design process? What's good about it, bad about it?

It'd data driven in that we model the data to store and what happens to the data. [Why?] Keen to keep hits on the database down so it's necessary to store data in an accessible, non-intensive way.

On designing the graphical interfaces this is done by asking the users [Print Room Staff] what they want and also by representing how you would use the data. The Print Room Staff give usability requirements.

In general, there is a prototyping approach where prototypes are presented to Customers and they are asked if this is what they want. Requirements come from Customer Services, Marketing and Business and there is an iterative process to requirements validation in the form of the prototypes. The prototyping approach is good because it is a starting point for thinking about requirements.

Documents produced:

- Project Charter – gives a business reason for the project
- Technical Specification – describes what is to be built and this document's target audience is business as well as IT [this equates to a Requirements document].

- Technical Technical Specification – this is for IT and written by IT. It is an expansion on the Technical Specification.

Bad points:

- Business people need to specify the requirements – i.e. detail them. But because of busy schedules and the fact that the business don't know what they always want, this does not happen. It's impossible to built the required machine when it is not specified.
- Process problem: IT left to specify the requirements and this takes a lot of time. It's a headache especially as they aren't exactly sure of what they are building.

2. The EARTH method describes various steps from requirements to design. What are your thoughts on the method?

It's requirements driven rather than data driven and that's not often how it works here. It would be better to apply the EARTH method because specification is carried out [I suppose via use case descriptions!] and the IT department aren't wasting time trying to specify uncertain requirements. Not sure if this [EARTH] would fit here as a new job would have to be created between business and IT. [i.e. a systems analyst / requirements engineer – if there's a difference. Note that there are a couple of systems analysts in the IT department, one of which is the next stakeholder.]

3. Let's go through some of these steps and consider the pros and cons of each part.

3.1 What are the strengths of the Business Process Model (RADs)?

These tie people down to defined roles. It's written down so no one can complain about this. The scope of the project is more fixed so requirements don't keep creeping in. Stops people from adding new requirements as they see fit.

Especially for new members of the development team the RADs are really handy. They tell you where everything goes. I knew everything that was in the diagram anyway before seeing it but this [RAD] gives a new perspective. It is invaluable, especially to those new to the department.

3.2 What are the weaknesses of the Business Process Model?

Time constraints that are common place here won't allow this kind of analysis to take place. [Of course, to describe the current processes in each application would need only one effort. If this time were to be spared, the outcome would prove invaluable especially when it came to maintaining, updating or changing the applications. Another bonus point to this is that we would be able to see where there is overlap between the roles in different applications and their corresponding processes.]

3.3 What are the strengths (benefits) of the use case diagrams?

High-level understanding of what's going on where in terms of functionality. We can see where there is complexity. It would be useful to dump the use cases and descriptions onto the business people so they know what is going on.

3.4 What are the weaknesses of the use case diagrams?

The include and extend relationships add too much complexity and would cause a problem when presenting this to the business people and customers.

3.5 What are the strengths of the use case descriptions?

Very specific and the business understands the step by step process. From an IT perspective, descriptions are good for showing the correct order of events. It's sometimes very important to get the order exactly right.

3.6 What are the weaknesses of the use case descriptions?

Very time consuming if there are lots of possible paths.

4. Give your thoughts on the question sets that are used to interrogate the use case descriptions, including weaknesses as well as strengths.

4.1 Dependencies (pre- and post-conditions)

Good points: Pre- and post- very handy to define where you start and end and also helps you bound your system. So looping [programming constructs in general] are constrained by the pre- and post- conditions.

Bad points: Too precise at times. This might impose stricter conditions than really required i.e. we might have a "very bad scenario" as opposed to the "happy day scenario". It all depends on the robustness required for the system being built. Overly constrained at times. There's a risk of introducing more errors by dealing with lots of constraints that might not be necessary. For instance, there might be an overly constrained requirement that deals with system failure risks that might occur two times a year. To code these risks out of the system means a vast amount of programming effort and lots of lines of code. This introduces the risk of introducing many more errors that occur more frequently than the frequency of occurrence of the original risk. [The stakeholder also noted that when the system is safety critical it is worth the extra effort. The interviewer suggested that much of the questioning wouldn't raise a lot of issues but when there appears to be a major functional event then the questioning is very useful. The stakeholder agreed with this.]

4.2 Interfaces

Similar comments to the *dependencies* comments. Over-analysis could lead to performance problems [in the IT department, not with the speed of the application itself].

4.3 Actors

These questions give good perspective to look at interactions to other systems. It's hard to anthropomorphise and imagine yourself to be another system.

4.4 System

From a business perspective people don't care about this. But for development it is important. It's vital to know what the interfaces are and how to connect to the right systems. Technically this is valid.

5. Can you describe how the EARTH method might improve / detract from you current development method.

The process here is business driven. There's not enough specification given. There's constant feedback to business to ask if this / that requirement is right. It would be very hard to introduce the EARTH method. Beneficial to introduce it in parts.

The business people might not take to RADs as their method is to use matrices. The business needs to deliver specifications and not spreadsheets with ticks in.

From an IT perspective, a business take up of use cases would be very handy to see customer goals, not necessarily detailed down to description level. Time constraints (and boredom) with writing descriptions and then questioning them is a negative.

Interview 2

Stakeholder Role: Systems Analyst

Action Research (adding context). This stakeholder has working in many large companies in this industry as a systems analyst. The stakeholder is generally uneducated of current methodologies and paradigms (such as object-orientation) but is highly skilled and analytically gifted.

1. Can you describe your current requirements analysis and design process? What's good about it, bad about it?

[Laughs at this question]

What process?

Generally, business talks to IT about its ideas for a new product or project. There are subsequent meetings where IT say what they can and can't do. The business people provide a specification of sorts: often 2 lines in an email or a telephone conversation. We write a specification based on the requirements received and get Business to sign them off. This is a Business focused technical specification. There isn't a bespoke technical specification produced. The goal of signing of the specification is to avoid new requirements half way through – but this is never easy.

Project scope always changes due to specification failures [i.e. when a requirement is not stated and therefore not specified] and when the business people can't make up their minds about what they want. IT tends to do the work of the business.

2. The EARTH method describes various steps from requirements to design. What are your thoughts on the method?

See there is a clear path of actions but business will have a problem seeing the first step and they wouldn't follow a use case approach. Questioning approach sensible, in the search for classes.

3. Let's go through some of these steps and consider the pros and cons of each part.

3.1 What are the strengths of the Business Process Model (RADs)?

Distinction between roles very useful. They provide a good overview. Clearly define what the roles do. It's good to have our systems mapped out like this. It's easy to trace through, especially to see interfaces. Though there is a lack of detail, the RADs raise a lot of questions: how long does the system wait for the Credit Checker to return the credit status of the Customer? What happens if it waits too long? Interfaces? How do we connect up? This gets us thinking about questions that need to be answered.

3.2 What are the weaknesses of the Business Process Model?

Lack of detail [though this is a very personal perspective and the stakeholder realises that the RADs provide an overview of the process.]

3.3 What are the strengths (benefits) of the use case diagrams?

[none stated]

3.4 What are the weaknesses of the use case diagrams?

The diagrams are hard to look at and understand quickly. You have to know the notation. If these are intended for business it would be hard for them to interpret and understand, which is a bad thing.

3.5 What are the strengths of the use case descriptions?

These are fine and worth doing [stakeholder has seen them before in the guise of test scripts].

3.6 What are the weaknesses of the use case descriptions?

[none stated]

4. Give your thoughts on the question sets that are used to interrogate the use case descriptions, including weaknesses as well as strengths.

[Overall comments given only, as opposed to discussing set by set.]

[I] use these sort of questions. They are a good thing to do. If you read a load of use cases you would ask these kind of questions. I would use the questions in business analysis. Timing constraints and interfaces would be asked mainly subconsciously.

There's no downside to questioning. In the context of discussion with users these are very useful, especially in building a rapour with them and getting the necessary information about the system. The same approach applies to the use cases themselves.

5. Can you describe how the EARTH method might improve / detract from you current development method.

Would use RADs, UC diagrams (descriptions only when really necessary because it takes time to develop them and there are serious time constraints on each project). The question sets are important and should be used [Though if there are no descriptions, it is difficult to see how the questions can be applied. We need to develop question sets that allow for closer mapping between RADs and use cases.] The stakeholder can't comment on design elements because this is not within his expertise.

The context diagram is not very clear in terms of the actual work undertaken in the re-engineering. Some of the domains are not relevant to this problem (Bank Wizard, Address Searcher, Credit Checker, National Insurance Wizard). These domains are completely automated so don't need to be thought about. What are the primary, secondary and tertiary domains – or actors – for this specific project? [It was explained to the stakeholder that the context diagram captures all problem domain areas that have an impact on the Fulfiment process regardless of whether it has an impact on the actual project work being carried out at this time. It is also the case that parameters might change in the future or the process order might alter, in which case the project team has to be aware of these

domains. If they are not documented then problems will arise. One interesting point from this context diagram is that the author explained that the CREST domain would have no impact on or be impacted by the Project F process. As such it was a mistake to include it. However, the stakeholder thought that CREST can become embroiled in the Project F process, but the stakeholder wasn't quite sure how. The context diagram has raised a possible issue for the stakeholder.]

Interview 3

Stakeholder Role: IT Quality Assurance Manager

Action Research. The stakeholder has a background in software quality assurance and systems testing in defence industries. The polarisation of work process from a highly-structured to an often apparently disjoint approach to software development has taken some getting used to. The stakeholder wishes to bring more formalism to the development process but realises this will take time and must be done "by stealth." The QA team within the IT department is effectively the testing team. Very little quality assurance can be done due to the reasons described in the following interview.

1. Can you describe your current requirements analysis and design process? What's good about it, bad about it?

The IT department is severely lacking in any process. If the entire project is completely contained within IT then the result is good because we have the expertise here. If the user is intelligent [know exactly what they need] then the outcome is usually OK but we don't have this with bigger projects. There are groups of business users and each has their own needs. This means that business requirements are never put together with the consequence that nothing is signed off.

Andersen Consulting was the standard document production. There was a loose framework: project charter but then a jump more or less directly into coding. Some projects have no documentation. The new Project Office has adopted a RIA (Rapid Impact Assessment) document, which is the Project Charter by another name. Each project has a project sponsor and manager (which might be the same person). There is no further required documentation from this. IT usually has to turn the Project Charter / RIA into design. But this is rarely documented.

Every project is delivered behind schedule. Haven't ever delivered all required functions. Basically, no one is ever entirely happy because Business doesn't specify their requirements and usually only present high-level goals. IT doesn't build everything, only

that which it thinks it can deliver. The product is always a compromise. Luckily there is in-house domain expertise in the IT department.

There used to be very small teams so an overall process was not necessary. But now there are bigger projects with larger teams, processes are required (but are missing). The marketing department are constantly looking for new product ideas and their respective markets. As such, they are headstrong in attitude and put unrealistic demands on the IT department to deliver. IT is constantly fire fighting to accommodate marketing.

2. The EARTH method describes various steps from requirements to design. What are your thoughts on the method?

I like it. The whole modelling concept helps in designing tests [QA / Testing role]. From the outset of a project this is good to map out. QA is often left guessing about testing in the current environment. QA would like to test from the requirements onwards but as there is very little documentation this is hard to do. It is shame it [EARTH method] hasn't been used on the actual project.

Good idea of a more formal requirements capture and analysis because it has been done well in your work. The method is very in-depth and some time needs to be spent on it to do it justice. Unfortunately, people see it as complex and I'm not sure if they will take it up. There is a need to put formality into practice. The IT Directors are keen to add more formality to the whole IT process but the work culture needs to change a bit so this can occur. Not sure if they will take anything from EARTH but they are impressed by it. Many of the software developers are keen to learn more about it and actually do some design work but they need to lead in that direction by their managers.

3. Let's go through some of these steps and consider the pros and cons of each part.

3.1 What are the strengths of the Business Process Model (RADs)?

They are absolutely good to do. No one does this kind of thing at the moment. (If the Project F process had been described and designed along these lines then there wouldn't have been the explosion in pack types. There are now 128 different packs with the

introduction of the ISA and PEP accounts!) [One of the major problems with the Packs was their quantity. There were 23 packs before the addition of the new account types (Maxi ISA, Mini ISA, Transfer ISA, Transfer PEP – which all automatically generate new Trading Accounts.) The re-design of the database schema to simplify pack generation seems not to have taken place. The current pack code generator takes the sum of all attributes entered at the interface and returns a Cartesian Product of possible combinations of pack codes. Figure F2-4 describes this:

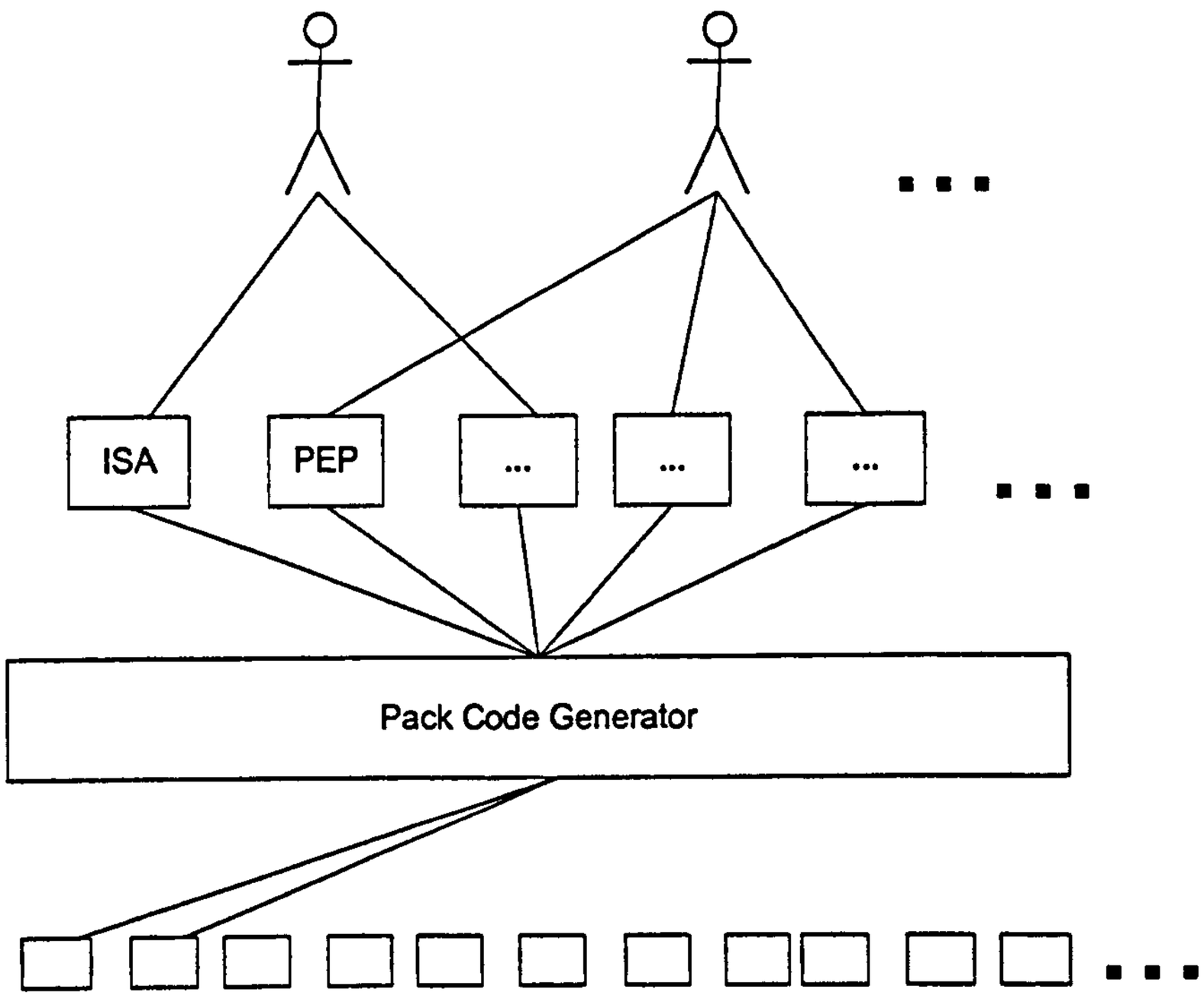


Figure F2-4. Current Pack Code Generator

The Pack Code Generator has a list of all possible combinations for pack types. When a Customer applies for whatever kind of account, a pack code is generated from the sum of the states of the relevant attributes of the application (selected, ¬selected) thus a Cartesian Product is required to know all possible states.

To reduce the number of pack codes, it has been suggested that a much simpler approach be taken: that each account type has its own pack code and if a Customer applies for a Mini ISA, a Joint Trading Account and to Tranfer Stock, then 3 pack codes are generated

that correspond to the account or function required, thus avoiding the proliferation of pack codes from the Cartesian product. Figure F2-5 describes this:

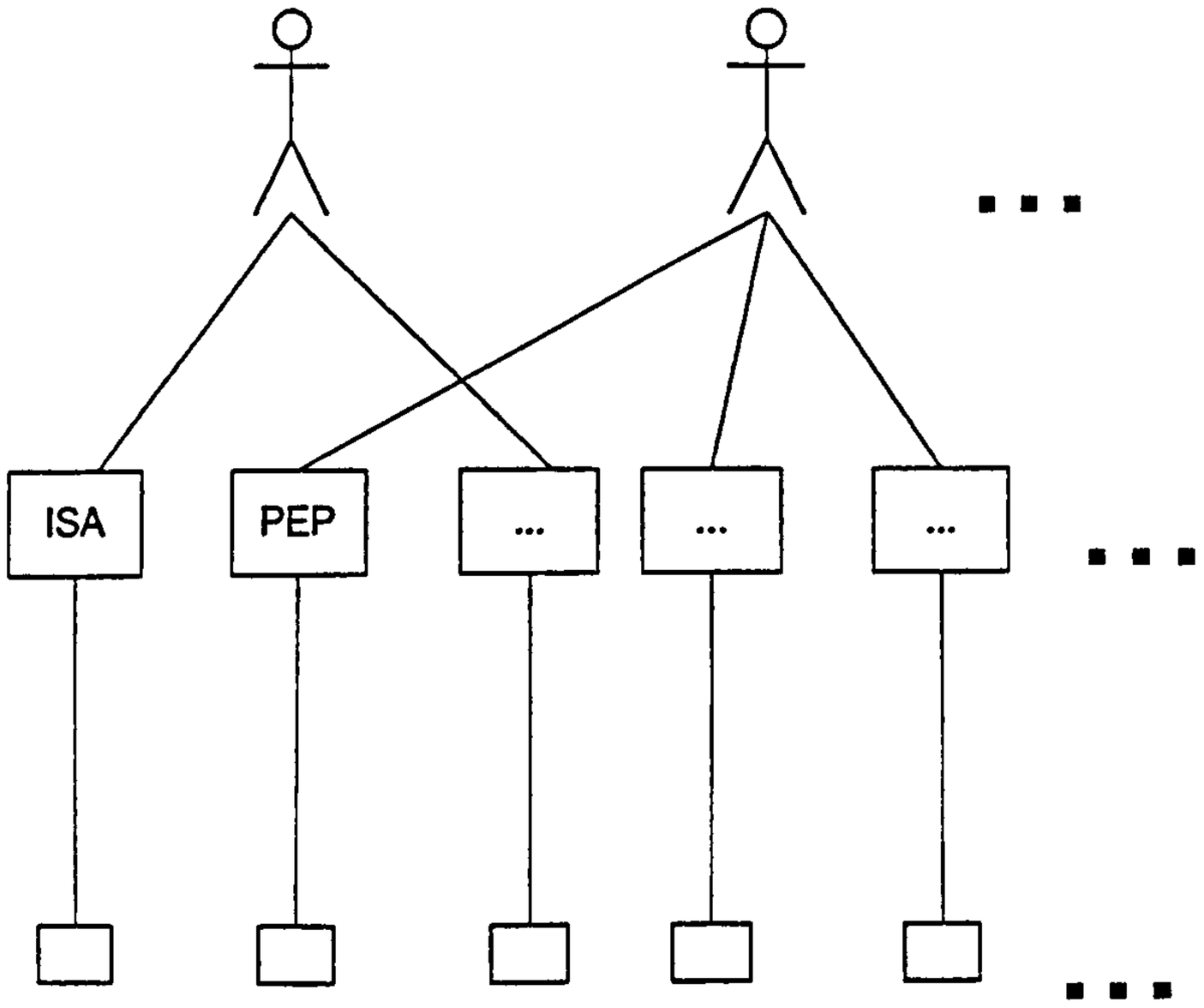


Figure F2-5. Proposed solution to reducing Pack Codes

Unfortunately, it appears that this simplifying proposal has not been implemented.]

Business people hopeless so a business analyst should do some RAD modelling. There are project managers doing business analysis – this is not a good idea. There are no real business analysts. Systems analysts are required.

The RAD is good for developers. Some are doing this kind of modelling but not in any detail and in an ad hoc way.

3.2 What are the weaknesses of the Business Process Model?

Not with the model but with time constraints on projects, this sort of work won't get done.

3.3 What are the strengths (benefits) of the use case diagrams?

[no comment]

3.4 What are the weaknesses of the use case diagrams?

[no comment]

3.5 What are the strengths of the use case descriptions?

Definitely a good thing for QA and testing as a basis for test development and user acceptance tests.

3.6 What are the weaknesses of the use case descriptions?

None.

4. Give your thoughts on the question sets that are used to interrogate the use case descriptions, including weaknesses as well as strengths.

[Overall comments given only, as opposed to discussing set by set.]

Essential activity. If you go down the road of doing some of this before coding, then coding would be a lot easier and quicker, instead of the usual approach of hacking prototypes. Changing the working mentality would be a drawback for their introduction. The approach is good and thorough and it reflects the depth of detail needed from a testing perspective.

5. Can you describe how the EARTH method might improve / detract from you current development method.

The method is long-winded so adoption will be difficult. It would be good to streamline the process so different departmental areas can take what is good for them without necessarily worrying about the rest. What can we take on to get a quick win? We would have to try this to demonstrate its effectiveness. Then introduce it by stealth.

Interview 4

Stakeholder Role: IT Development Manager

Action Research: Stakeholder has a good educational background (Masters level) and several years experience in various industries. Over the last 4 years, the stakeholder has developed software for futures and stock trading companies.

1. Can you describe your current requirements analysis and design process? What's good about it, bad about it?

We have a rapid development approach. We don't have any specific methods that we always use. What we do depends on the size of the project and the time we have until delivery. We start with a project charter and get a basic requirements set. We get some business requirements from the business departments. There are usually several meetings between business and IT and some sort of document to support those meetings. The document can be text or some kind of process flow diagram. This is deemed as a walkthrough for determining requirements. Then we code from here!

There's no design, no existing model to work from. This is poor really, especially for reuse. Our success has been based on the knowledge our developers have of the product. Usually the requirements are stored in someone's head and remain there until delivery of the product.

We sometimes taken a prototyping approach with HTML screens. We've done this a couple of times. This is then taken into servlet work and demos are given to business people. A major problem is that Marketing do their own prototypes and present them to business as walkthroughs. Then when the product is about to go live, business or marketing raise their ugly heads and say what they really want.

Prototypes are really useful but business doesn't pay them enough attention. They don't look at the detail. This leads to complaints of the product not meeting expectations. The whole process needs more rigidity. There should be a sign off on all stages but it only happens with the project charter at the moment. [To validate this, Project F is going live

this week but the Functional Requirements Document is still to be signed off – plus it doesn't talk much about functions or requirements, instead it provides an overview of the current system and gives a 1 and a half page account of the proposed architectural changes needed. The Technical Specification is only half complete. There are huge sections left blank – all pertaining to the requirements. The only reasonably completed sections is that which discusses the database schema and provides examples of SQL. This is design and code and shouldn't be in the document! I suspect that the documents will never be completed nor signed off.]

Good point: We have an open development service. This means we are happy to take and share new ideas with the rest of the business.

Bad point: Business is too blasé because it sees that IT is getting on with things and so doesn't take responsibility for products. It only panics when things go live [and go wrong]. Business needs to take earlier ownership, commitment of projects [so that they can get what they want]. There are always the same people in the project loop irrespective of whom the customer is. This means that they don't care about the project if it isn't their area. For example, if a product is not customer-focussed i.e. it is an internal product, why should Compliance have to sign off the project? [Compliance is the regulatory department that makes sure the product does not breach trading and banking laws.] Only those involved in the project should sign it off; this would ensure more business interest and commitment to the project.

2. The EARTH method describes various steps from requirements to design. What are your thoughts on the method?

Overall, [the EARTH method is] good, valid methodology for getting what we want out of projects.

Good points: RADs useful for us and business; use cases good for test scripts. There's lots of attention to detail and you're forced to capture this early on. Classes and interaction diagrams necessary for reuse and review of current systems and design. We are forced to do work up front before code. EARTH follows a number of clear stages. We could talk

and do interviews with business to get requirements details rather than present a bunch of prototypes.

Downside: Too much documentation and there's a long lead-time before design. Time constraints won't allow us to do everything.

It would be good to model every current system in IT with EARTH so we have details of everything – if we took this as a project in itself – and then every new project could expand on the model, rather than straight on the actual system.

Would use EARTH in a 3-6 month project but most of the projects are shorter than this so we might have to take bits of EARTH and use what we can. Not sure if this invalidates EARTH.

3. Let's go through some of these steps and consider the pros and cons of each part.

3.1 What are the strengths and weaknesses of the Business Process Model (RADs)?

RADs – not very readable. They look complicated. Would have to train business to use them. They wouldn't have a clue as is.

3.2 What are the strengths and weaknesses of the UC Diagram?

The diagram is quick and useful. It's fairly similar to walkthroughs currently done in that it tries to identify the entities in the system and their functionality. [No, it doesn't. It identifies entities that do something outside the system and what they want to do on the system.]

3.3 What are the strengths and weaknesses of the UC Description?

The descriptions are good for test scripts. They capture lots of detail. Actually similar to how a servlet is written! This is dangerous because it might encourage jumping from a use

case straight into code. It's also frightening to see that there's loads to do! Descriptions are good for showing what's going on in the system.

4. What are the strengths and weaknesses of the Question sets?

Pretty good [but too hung over to remember the details!]

5. Can you describe how the EARTH method might improve / detract from you current development method.

As said before.

Process model useful as a tool to walkthrough with business but we would have to write them another way so business understands it [!]

UC diagram good for reviewing with business. Descriptions good for everyone (testing, understanding the system).

Packages, classes etc good for generating code.

The EARTH method would fit into an overall improvement process. Force a step-by-step approach to starting a project and this hasn't been the case. The City way is for someone to say, "Why don't you do so and so?" and as a developer you say, "OK, that's a good idea." You are then expected to deliver (or get sacked!).

There should be a set number of tasks to be part of the business cycle. EARTH should be part of that as it gives the rigidity needed. There would still be the usual meetings with business but IT would also be carrying out the EARTH method to integrate with the typical approach. Everything would have to be signed off by business and IT.

We'd use EARTH to assess and improve what we have.

F3. Document presented to Company X

Modelling Project F

Contents

1. Document Overview

Report Focus

This report is a revision of the previous document Modelling Filfilment 4 (dated 26/04/01)

This document describes the modeling undertaken by the author on the Project F. The aim of the document is to describe the process from front to back of how a design has been created for the project. This means that we begin by examining the problem context (section 2) and the business processes (section 3) that take place to achieve successful delivery of application packs to Customers. From here we can begin to model the interfaces of the system and the internal design by applying the EARTH use case oriented process method.

Company X

Company X provides, amongst other products, an online brokerage system that allows its Customers to buy stocks and shares on various markets worldwide. Company X has identified a need to expand into further markets and also has realized that the current account application procedure could potentially cause problems in the near future with the expected (desired) increase in applications for accounts.

Project X Overview

Project F is the process by which a prospective Customer applies for and gets a trading account (individual trading account, joint trading account, Maxi ISA account, Mini ISA account, ISA transfer, PEP transfer). As stated, the application procedure is being overhauled but not at the interface (although new products are regularly added and some team members consider the online application process to be too long-winded). For the Customer, application will remain the same. The changes that are fundamental to improved service all take place in the system itself (Front Office) and the process by which application packs are printed within the company.

This project is fundamental to the success of the business because it is known that the current application and printing process can manage up to 600 applications per day before it becomes unwieldy. When this amount of applications has been reached in the past, outsourcing to a printing company has taken place. Projected growth of the company for the next year means that the application and printing process must be able to cope with 1,000 applications a day. All printing of application packs is to take place in-house. As such, Project F has to be re-engineered to cope with the expected growth in number of applications. To make the situation more complex, more products are being offered. This means that the variety of types of application pack available for printing will increase, putting more pressure on the printing process (section 4 describes the current printing process).

Please note that the issue of security is not included in this document.

Project Goals

G1. The print process has the capacity to print 1,000 Customer applications per day.

This is a business and operations goal. There is no proposed increase in hardware. As such, the software design will have to ease the print goal by making the generation of pack types and printing packs easier.

G2. The selection of printing jobs is more efficient than the current system.

It is very difficult to determine how efficient is more efficient. There has been no timing measures taken, or complexity measures taken. However, the current process is very labour intensive and open to much human error. It has been suggested that the generation of print jobs and the selection of print trays should be automated. The complete removal of human interaction is impossible – someone has to fill the printers with paper! Automation is essentially a long-term goal and not part of the suggested solution in this document (although this is closer to realisation than expected).

G2.1. The design of the application process is to facilitate more effective generation of pack types to allow easier printing of applications.

To achieve this re-design and to clarify how this re-design impacts on the system, it was decided to model the system from various angles. There is the business process view, which shows how various domains in the problem interact. This roughly maps to a use case model, which describes the application procedures at the interface. The use case model goes beyond this in helping determine the internal design scenarios at the Business Model Layer and at the Database Layer. Identification of use cases at the external and internal levels is not a matter of top-down decomposition but a recognition from the business process model that there are interactions that are at the graphical user interface level (Customer application), interactions between connecting systems that are driven by the machine (although dependent upon completion of the application form are not decompositions of these use cases). There are sub-system level use cases that connect various sub-domains where necessary.

Use cases are interrogated by question sets that help identify classes and the class model as a whole.

The re-engineering develops a class model from the architectural structure identified (package diagrams). From these class diagrams, it is possible to model with sequence and collaboration diagrams to identify object interactions that correspond to the use case.

There are no Customer-elicited goals because this is a re-engineering problem. The stakeholders are the developers and the employees that work in the print room. However, improving the print process should hopefully increase the speed in which Customers receive their application packs.

2. Context Diagram of Envisioned System

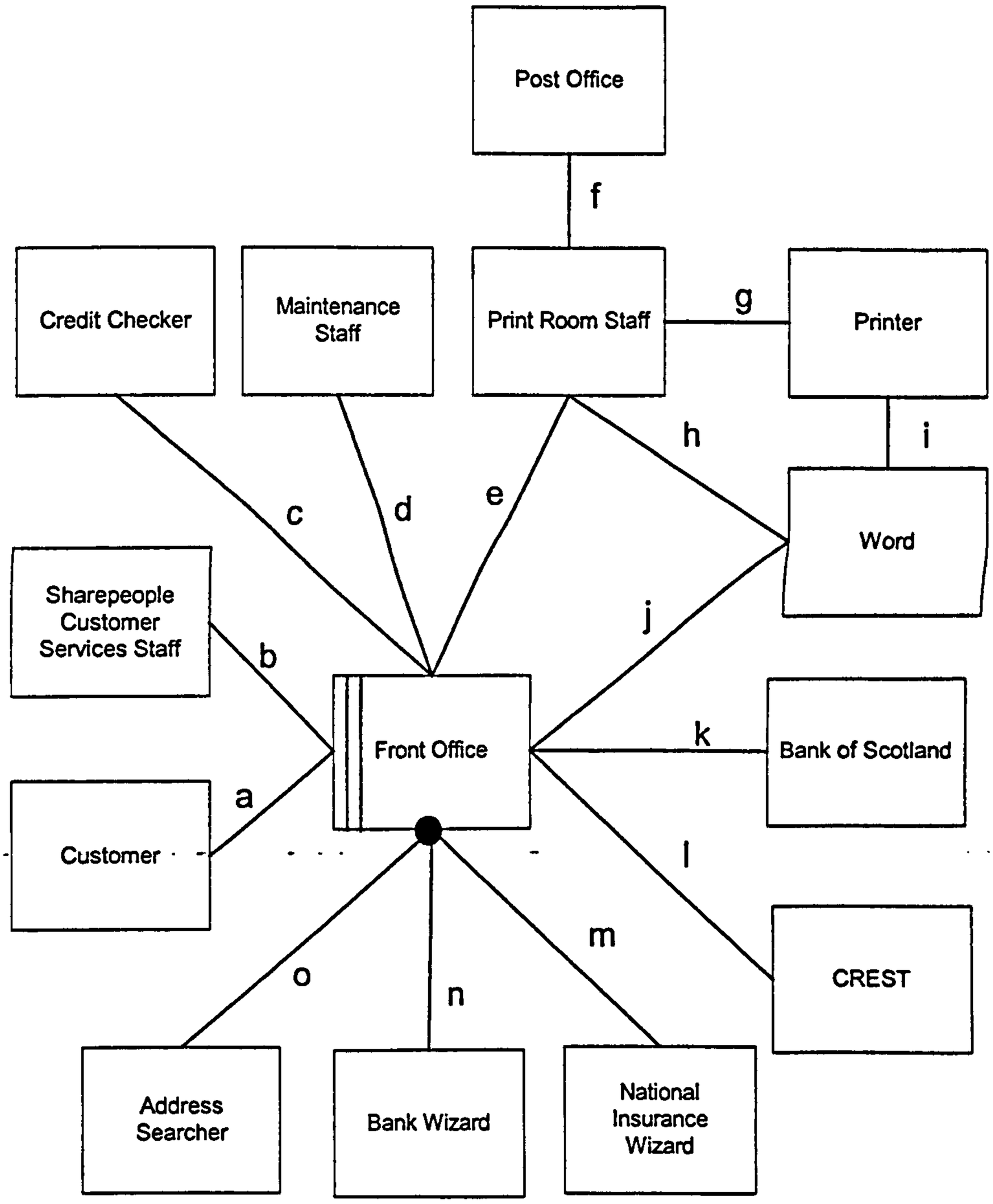


Figure 2: Context Diagram for Envisioned Project F

Context Diagram Description

Note that the context diagram describes the domains in the entire Project F process and includes domains that may not be part of the proposed re-engineering project. The context diagram (fig 2) describes the domains of interest within the problem and describes how these domains interact. The Front Office domain is the machine to be built – it is denoted by the double vertical stripe. This context diagram reflects the envisioned system and not the existing system. The black dot that connects the Front Office to the Address Searcher domain, the Bank Wizard domain and the National Insurance Wizard domain indicates that these three domains are contained within the Front Office machine.

Note that the context diagram is not a typical structured analysis context diagram. It is a Jackson context diagram (Michael Jackson, *Software Requirements and Specifications*,

1995 and *Problem Frames*, 2001). This is not solely a description of data flow to and from the machine but a description of the common interfaces that represent shared phenomena between the domains of interest. For instance, the Bank of Scotland and the Front Office are connected because the Front Office needs to inform the BOS that new accounts have been created with the account details. The shared phenomena, events and interfaces that connect the domains of interest are:

a. Online Application
b. Notify of completed application
c. Customer Bank Details
d. Pack Codes and Types; Customer application details
e. Print Directory to Customer Application Packs
f. Application forms to mail
g. Printing materials (paper, documents, forms)
h. Pack Type to print; Printer information
i. Print job
j. CSV file for merging
k. Bank account details
l. Trade account and trade details
m. National Insurance number
n. Bank account details (address, A/C number, sort code)
o. Postcode; address

The shared phenomena are expanded upon later in the business process and use case models (see section 3 for more on this).

The domains of interest:

Front Office

This is the machine that has to be built (modified, in this case) to achieve the desired effects in the problem domains.

Customer

The *Customer* domain is significant because it is the *Customer* who applies for an account via the internet with the *Front Office.* Perhaps we could have an *Account* domain that connects the *Customer* and the *Front Office?*

Credit Checker

This domain refers to another system that connects to the Front Office to check the credit status of the Customer. We might have shown a connection between *Customer* and *Credit Checker* but since there is no direct contact this is deemed unnecessary.
Address Searcher

Address Searcher connects to the *Front Office* internally. That is, *Address Searcher* is housed within *Front Office.* It is considered a separate domain of interest because its sole role is to identify and present addresses to the *Customer* from the postcode they enter in

the application procedure. Does this mean there should be a connection between *Customer* and *Address Searcher?*

Bank Wizard

This domain plays a similar role to that of *Address Searcher. Bank Wizard* is housed within the *Front Office* and determines the location of the *Customer's* bank based upon bank details entered. Does this suggest a connection between *Customer* and *Bank Wizard?*

National Insurance Wizard

This domain is similar to the *Address Searcher* and the *Bank Wizard.* It determines the validity of the *Customer's* national insurance number, which is a legal requirement for purchasing/transferring an ISA or PEP.

CREST

CREST is a company that deals in Stocks and Shares on the market. A *CREST* account is created when the *Customer* makes their first trade. However, this domain should probably not be considered part of Project F, except that when an account is created it generates a Pack Code. Is Pack Code a domain of interest? It is, afterall what much of this re-engineering project is about – the simplification of the Pack Code so that printing applications is much easier to do.

Bank of Scotland (BOS)

The *Bank of Scotland* are informed of account creation by the *Front Office* once the *Customer* has returned completed application forms. *BOS* can reject the account if they see fit. *BOS* send account details to the *Customer* such as cheque book and account details. Should we show this connection? The danger is that we are describing packets of data moving between the domains rather than describing a set of common interfaces and where the domains share phenomena.

Company X Customer Services Staff

Company X Customer Services Staff connect to the *Front Office.* This *Company X Customer Services Staff* domain did, in its first phase, have a connection to *Customer* but upon reflection there is no direct interface between them although the *Customer* sends their completed application form to the *Company X Customer Services Staff.* The mail system (*Post Office*) acts as a connection domain between the two. We do not have to go into this detail on the context diagram.

Maintenance Staff

This domain is not part of the application procedure but might have an effect on the data in *Front Office* where all the *Customer* accounts are stored. The *Maintenance Staff* will make changes to *Customer* accounts as and when necessary.

Print Room Staff

The *Print Room Staff* are responsible for printing the application packs once the *Customer* has completed the application process. The *Print Room Staff* domain interfaces with *Word* and *Printers* to print the applications.

Word

Word is an important domain because it performs the necessary mail merge ready for printing to occur. Application files are transferred from *Front Office* to *Word* by the *Print Room Staff.*

Printer

There are 3 separate *Printers.* Because the printing of application forms is a highly significant part of the application process, the *Printer* domain is included here. *Print Room Staff* sets the *Printer* properties (via *Word*) and stock paper trays, dependent on Pack Types, ready to print the applications.

Post Office

This domain is significant in that the Post Office is responsible for the safe delivery of application packs to *Customers.* Although this might be considered beyond the scope of the problem, it is not. Company X are concerned that the *Post Office* perform more than one collection per day – but this is dependent upon the number of applications printed per day. The *Post* Office liaises with the *Print Room Staff* members. This domain cannot be influenced by the machine but how it works is determined by the number of applications received and printed by *Front Office.*

2.1 Domain Requirements

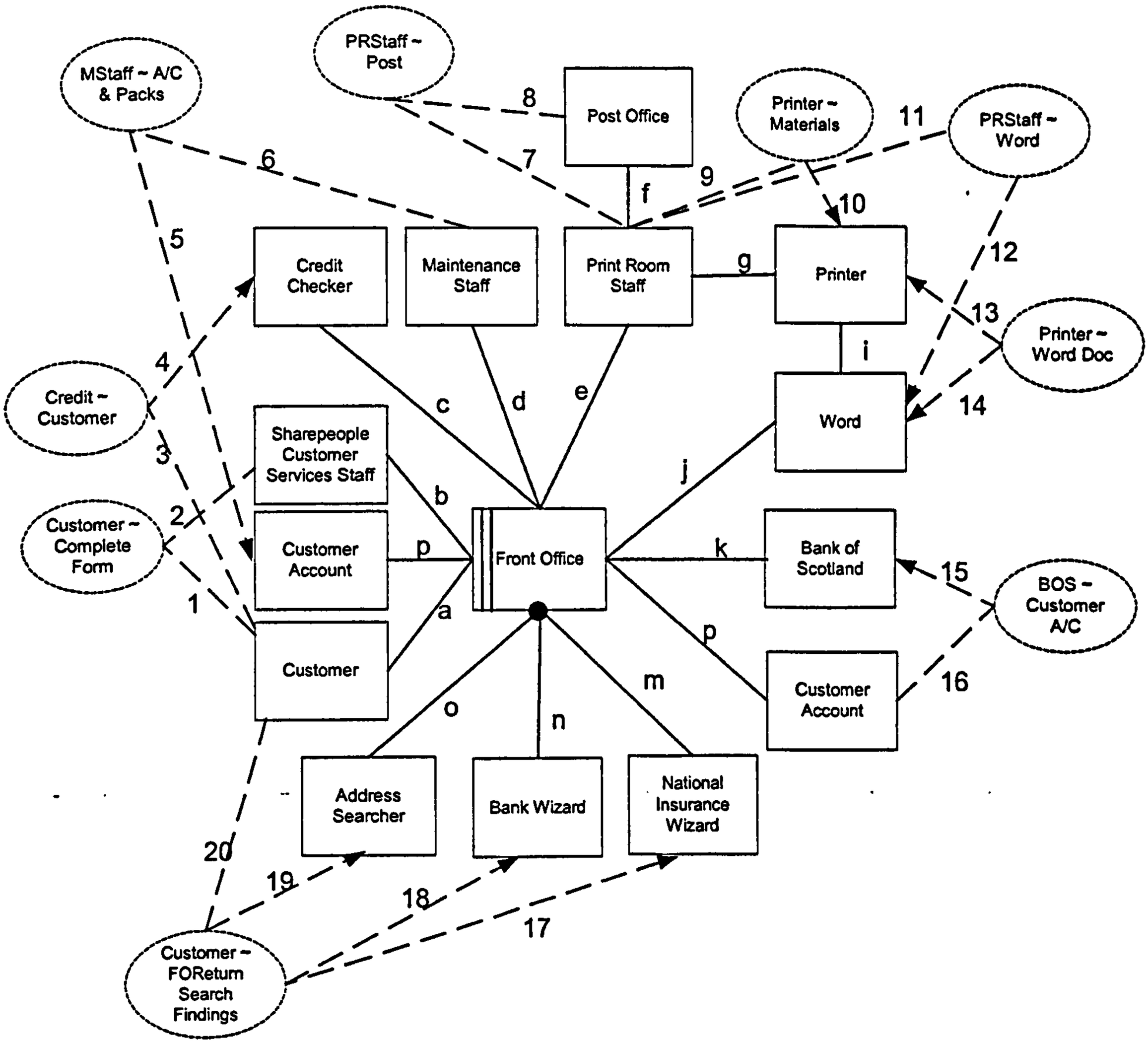


Figure 2.1: Domain Requirements Diagram

To go beyond the context diagram (fig. 2) and consider how the requirements affect the domains of interest, we have added requirements sets (fig. 2.1). These are represented by the ovals. The text in the ovals represents the domains that are affected by or affect the requirements. For example, the requirement set *Customer ~ FOReturn Search Findings* connects the domain Customer to the domains Address Searcher, Bank Wizard and National Insurance Number with various requirements. The Customer requirement (20) is that they enter a postcode (for Address Searcher – which Address Searcher must use to locate the streets with this exact postcode and return these to the interface of the Front Office (a, o)). The same idea holds for the Customer and the Bank Wizard: the Bank Wizard must locate and return the address of the bank (18) this is represented by the account details entered by the Customer (20). The Front Office has to represent this at the interface (a, n). The same goes for the National Insurance Wizard. This domain must return a validation (17) of the number entered by the Customer (20). The Front Office represents this at the interface visible to the Customer (a, m).

The arrows from the requirements sets to the domains indicate that this is referenced requirement, in that the domain pointed to must fulfill the requirement under the constraints specified. Those that have no arrow indicate that the requirement represents what this domain can do but the domain cannot be controlled or constrained by the requirement. For example, the Customer domain has the requirement (20) which states that the Customer has to enter their bank account details. The Customer can enter any number or name they feel like, although most often they enter the correct numbers. Or the Customer could enter the numbers but with a mistake in them and not realise their error. As such the Customer is not constrained by the requirement. The Bank Wizard, however, must return the address of the bank that corresponds to the real life bank as represented by the Customer's entry. The constraint is that this is a valid bank account or is not a valid bank account. If the bank account is valid then this is the address of the bank where the account is stored. This address must be correct. There's no possibility of inaccuracy.

A new domain is introduced: **Customer Account**

This is a lexical representation of the account created by the Front Office on completion of the application. Its interface to the Front Office is *p*: new bank account details. It could be considered a design domain, as it is created entirely in the machine.

Domain Characteristics

Biddable Domains [B]

Biddable domains are those that cannot entirely control or guarantee behaviours and are usually humans.

The biddable domains in this diagram are: Customer, Company X Customer Services Staff, Maintenance Staff, Print Room Staff and Post Office.

Causal Domains [C]

Causal domains are those have a control or behaviour on the machine domain (Front Office) that can be exactly described and constrained.

The causal domains in this diagram are: Credit Checker, Printer, National Insurance Wizard, Bank Wizard, Address Searcher, Word and Bank of Scotland.

Lexical Domains [X]

Lexical domains are those that represent data or information about the World and do not affect behaviours but are representations of what is being controlled.

The lexical domain in this diagram is: Customer Account.

The Requirements Sets

The requirements describe the desired effects and behaviours that the machine is to achieve in the environment. There are nine requirement sets in the diagram, represented by the ovals with dotted lines. Each requirement set connects a number of domains:

Customer ~ Complete Form

This requirement set connects two biddable domains: Customer [B] and Company X Customer Services Staff [B]. The requirement says that the Company X Customer Services Staff must inform the Front Office of the return of the Customer's application form, provided it is correctly completed.

1. CU ! { Sign form; return form }
2. SH ! { Process form }

CU represents the *Customer* domain. It is responsibility for interfacing with the *Company X Customer Services Staff (SH)* domain is to sign and return the application form.

Credit ~ Customer

This requirement set connects a biddable domain and a causal domain: Customer [B] and Credit Checker [C]. The requirement for Credit Checker stipulates that it return correct information regarding the Customer credit status *and* in a format acceptable to the Front Office.

3. CU ! { Valid bank account }
4. CC ! { Credit status information }

MStaff ~ A/C & Packs

This requirement set connects a lexical domain and a biddable domain: Customer Account [X] and Maintenance Staff [B]. The requirement states that the Maintenance Staff can access and update Customer Account records, which should also include the application pack information.

5. CA ! { Customer Account }
6. MS ! { Update Account }

PRStaff ~ Post

This requirement set connects two biddable domains: Print Room Staff [B] and Post Office [B]. This requirement is entirely in the problem domain; it has no direct relationship to the machine. The requirement states that the Print Room Staff must pass the printed application packs to the Post Office for delivery to the Customer.

7. PS ! { Deliver packs }
8. PO ! { Post packs }

Printer ~ Materials

This requirement set connects a biddable domain and a causal domain: Print Room Staff [B] and Printer [C]. The requirement, entirely in the problem domain, is that the Print Room Staff fill the Printer with the correct materials ready for a print run.

9. PS ! { Paper, ink }
10. PR ! { Paper, ink } ?

Print Room Staff ~ Word

This requirement set connects a biddable domain and a causal domain: Print Room Staff [B] and Word [C]. The requirement states that the Print Room Staff get CSV files to merge to Word documents and to select printers for printing.

11. PS ! { Put CSV file; print selection }
12. WD ! { Mail merge }

Printer ~ Word Doc

This requirement set connects two causal domains: Printer [C] and Word [C]. The requirement, in the problem domain, is that Word documents must be printed on the Printer. The Word documents must be suitable for printing.

13. PR ! { Print file }
14. WD ! { File to print }

BOS ~ Customer A/C

This requirement set connects a causal domain and a lexical domain: Bank of Scotland [C] and Customer Account [X]. The requirement states that the Bank of Scotland must either accept or reject the new account for the Customer created by the Front Office.

15. BS ! { A/C validation; notification }
16. CA ! { Account }

Customer ~ FOReturn Search Findings

This requirement set connects a biddable domain and three causal domains contained within the Front Office machine domain: Customer [B], Address Searcher [C], Bank Wizard [C] and National Insurance Wizard [C]. The requirement states:

- that the National Insurance Wizard validates the National Insurance number entered by the Customer (this is for ISA accounts only);
- that the Bank Wizard must return the exact address of the bank from the bank details (account holder, account number, branch sort code, date account opened) entered by the Customer;
- that the Address Searcher must return a list of street addresses that correspond to the postcode entered by the Customer.

17. NI ! { Validate }
18. BW ! { Return bank }
19. AS ! { Return streets }
20. CU ! { National Insurance number; bank details; post code }

Because this project is process driven it is wise to describe the business process for the application procedure. This can be found in the following section.

3. Current Process Models

Process Model Introduction

The chosen method of modelling the business process is Role Activity Diagrams primarily because they are a proven technology and because Roles and Activities are neatly backwards-traceable to context diagrams and forwards-traceable to use case diagrams (see table 1).

Context Diagram	Role Activity Diagram	Use Case Diagram
Domain of Interest	Role	Actor
Interface (domain connector)	Interaction	Use Case
Machine Domain	Role	System

Table 1: Traceability between models ("is a" relationship)

Note that this traceability is not guaranteed when moving from the Role Activity Diagram to the Use Case Diagram. This is because the use case diagram only describes actors that directly interact with the system. As such, domains of interest and roles that may have an important part to play in the problem domain or are indirectly affected by the machine are not described in the use case diagram although it is for/against these domains that the system is built.

We can map the domains of interest in the context diagram to roles in the Role Activity Diagram. There is no concept of process in the context diagram – all we show is the domains of interest and describe their shared phenomena (common interfaces). We are scoping the problem. This is a start, as stated, but we need to carry this further. Clearly, Project X is process driven. The driving process is the application procedure and the generation of an application pack. The interfaces between the domains of interest are explored further in the process model and we can define the order of interaction between the domains thereby increasing our knowledge and understanding of the problem, the process and its impact.

The business process models are in two sections. To further comprehend the existing problem and to enable thinking, understanding and description of a possible solution – to the process difficulties – not to building the machine, this document describes business process models for the current system. Then there is a description of the process for an envisioned application printing process (section 5).

This section depicts a process model of the **current** system. It is incomplete in that it deals only with online account applications. It does not consider Call Centre account applications. The model helps us understand the business domain and also the architecture of the system.

There are 3 figures:

Fig. 3.1 The Front Office

This shows the roles and activities involved – at a high-level – of the Front Office, from the Customer applying for a Trading Account to the Back Office receiving generated Pack

Codes. There are 4 roles: Customer, Front Office, Credit Checker System and Back Office. These are briefly described in turn:

Customer

The *Customer* role is key to the application procedure because the *Customer* begins it. The *Customer* "Applies for account" to the *Front Office* via the Company X website. This interaction begins the application process. The *Customer* awaits a response from the application.

Front Office

This is the machine and deals with the *Customer* application by checking the *Customer's* credit status, generating a pack code and a new account and then informing the *Back Office.*

Credit Checker System

This role represents another system that checks the credit worthiness of the applicant *Customer.* The *Credit Checker System* informs the *Front Office* of the credit status of the *Customer.* This is all done in real time in a matter of seconds (exact times unknown).

Back Office

This is the server that sits behind the *Front Office.* This is where accounts and packs are stored.

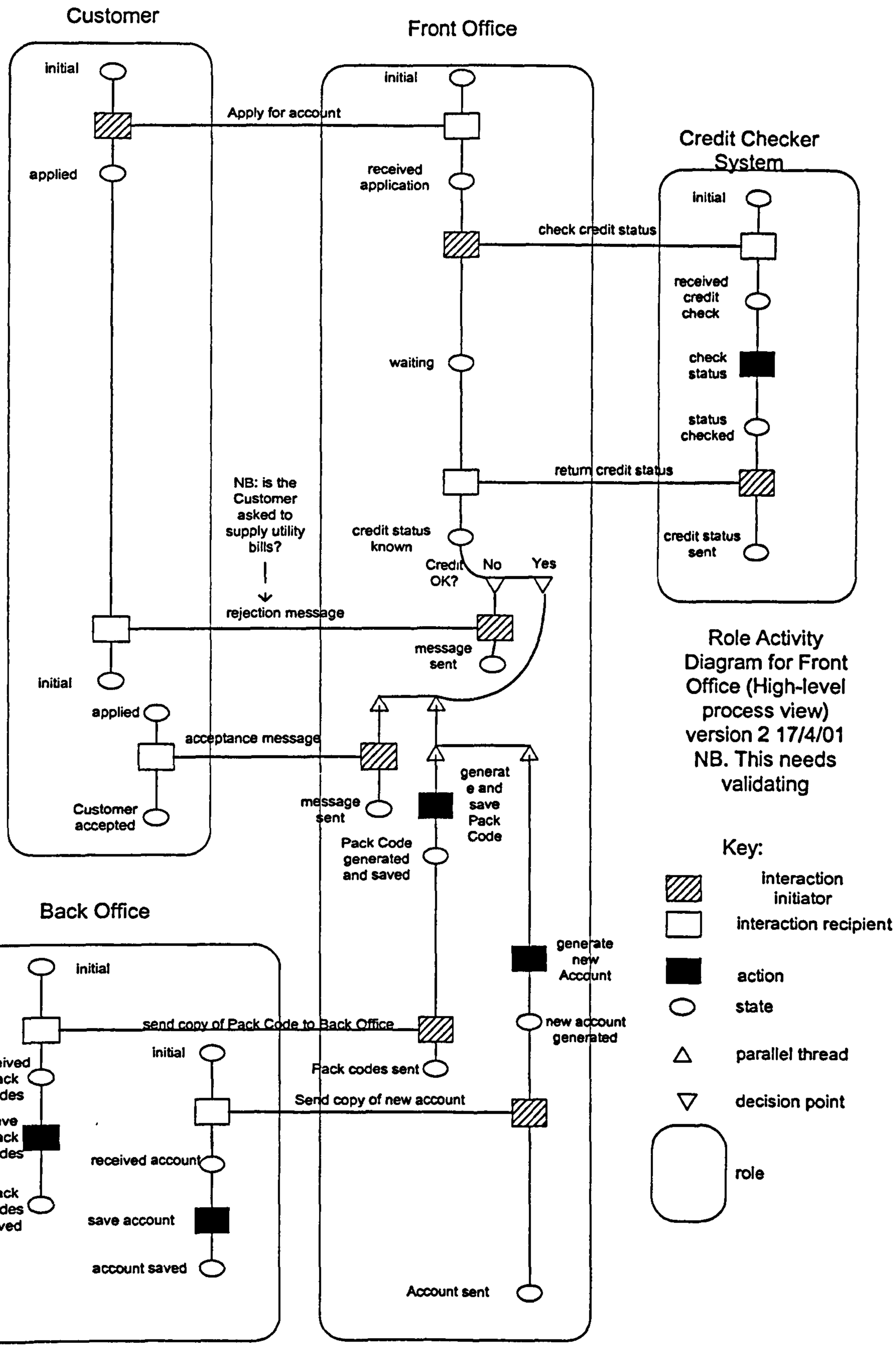


Figure 3.1 Role Activity Diagram of Front Office

Fig. 3.2 The Back Office

This shows how the Bank Office links to other servers and systems, for instance, to the Bank of Scotland for account verification. There are 5 roles in this diagram: Back Office, Front Office, Printer Server, Customer, Bank of Scotland. The Customer role appears three times. This means that the Customer interacts with other roles at three separate occasions, that the Customer is an instance of a role.

Back Office

The *Back Office* has many separate interactions with other roles. This indicates that there is not one continual thread from start to finish but many instances of interaction. The *Back Office* periodically updates the *Front Office* with possible changes to records. The *Back Office* copies *Customer* details and Pack Codes to the *Print Server*. When the *Back Office* has received a signed application from the *Customer* it then, so long as the parameters are correct, activates the *Customer's* account and informs them of this action. The *Back Office* informs the *Bank of Scotland* that an account has been created and activated. If the result of the *BOS* credit check is account rejection, then the *Back Office* has to inform the *Customer*.

Front Office

The *Front Office* does little in this RAD except receive updated records from the *Back Office*.

Print Server

The *Print Server* is another link in the printing chain. It receives and stores *Customer* details and application Pack Codes.

Customer

The *Customer* role appears in three instances. Its first is to interact with the *Back Office* by sending a completed, signed application form. When Company X receive the signed application, the *Back Office* emails or text messages the *Customer* to state that the account has been successfully set up. The *Customer's* second instance is to receive a rejection of their application from *Back Office*. The *Customer's* third instance is to receive account goods (cheque book etc) from the *Bank of Scotland.*

Bank of Scotland (BOS)

The *BOS* role receives details of a new account and runs its own credit validity check on the account owner. The *BOS* informs the *Back Office* of the results of the credit check it has run. If the credit checks are unfavourable, then the *BOS* can refuse to accept the *Customer's* account. If the *BOS* accepts the account then it will send bank details to the *Customer* direct.

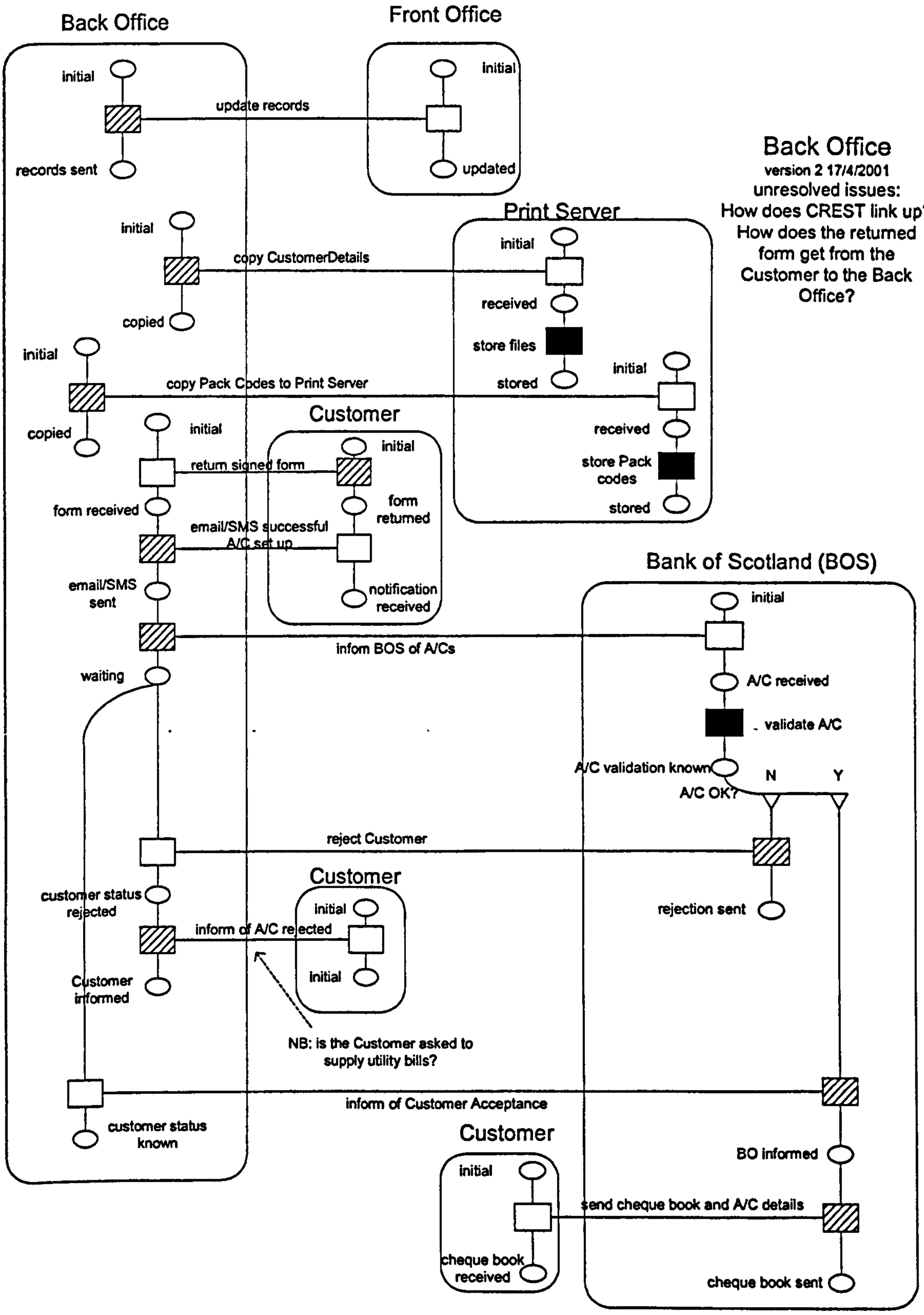


Figure 3.2 Role Activity Diagram for Back Office

Fig. 3.3 The Print Room

This shows the activities that the Print Room Staff carry out and with what other roles they interact. The printing process is quite complex and has been elaborated further with a use case model (section 4). The roles in this diagram are: Print Room Staff, Print Server, Access Database, Word, Printer, Post Worker and Print Company. There are two processes defined in this diagram – that of in-house printing and that of outsourcing printing to a print company.

Print Room Staff

The *Print Room Staff* have to access the *Print Server*, locate the files for printing and copy these files from the *Print Server* to an *Access Database.* This procedure is haphazard because it is a drag and drop operation. Once the files are on the *Access Database,* the *Print Room Staff* sends the files to *Word* so that they can be merged with the correct documents ready for printing. The *Print Room Staff* prepare the *Printer* by loading the correct forms into the right trays of a specific *Printer* dependent upon the pack type to be printed. They then prepare to print by selecting printer and tray in *Word* before they print. The *Print Room Staff* then have to collect the printed applications and stuff them into envelopes, ready to be taken off by a *Post Worker* for posting to Customers.

When there are more than 1,000 applications to print in one day, the print run is out sourced to a *Print Company*. The *Print Room Staff* place the file for outsourcing onto the *Print Server* and inform the *Print Company* that the file is there.

Print Server

This is simply a repository where the files for printing applications are stored. It has no self-determined interaction because the *Print Room Staff* manipulates it to transfer files across to the *Access Database.* The *Print Server* has two more instances of usage in this RAD. It acts as a store for outsourced printing files that are copied across by the *Print Room Staff* and accessed by the *Print Company.*

Access Database

This is a purpose-built database that takes files copied to it, populates its tables with the data from the files and performs a dedupe operation. The dedupe eliminates all duplicate applications and also pulls out applications that have a postcode outside the UK. *Print Room Staff* then select the pack type to print and mail merge to *Word.* The Access database is considered unstable and unreliable.

Word

This is Microsoft *Word*. It is used because its mail merge application works well – although there is the occasional glitch where an extra, blank page is printed on certain pack types. *Word* sends the print job to the *Printer* to print.

Printer

There are 3 printers: C1 (A4) IBM, C1 (A3/A4) HP, A1 (A3) IBM. Politically, IBM is favoured, though HP is a more reliable. The Print Room Staff prepares the specific *Printer* for printing. When the *Printer* receives a print job, it prints out the packs.

Post Worker

The *Post Worker* takes the mail from the *Print Room Staff* to send off to *Customers*.

Print Company

This role comes into play when there are 600 or more applications in one day. (The in-house printers and print room staff cannot cope with more than this number of applications per day.) They are informed of a print run and go collect the file from the *Print Server*. They then inform the *Print Room Staff* that they have collected the file for printing. The print job is then executed.
(The option of outsourcing print jobs has been discontinued.)

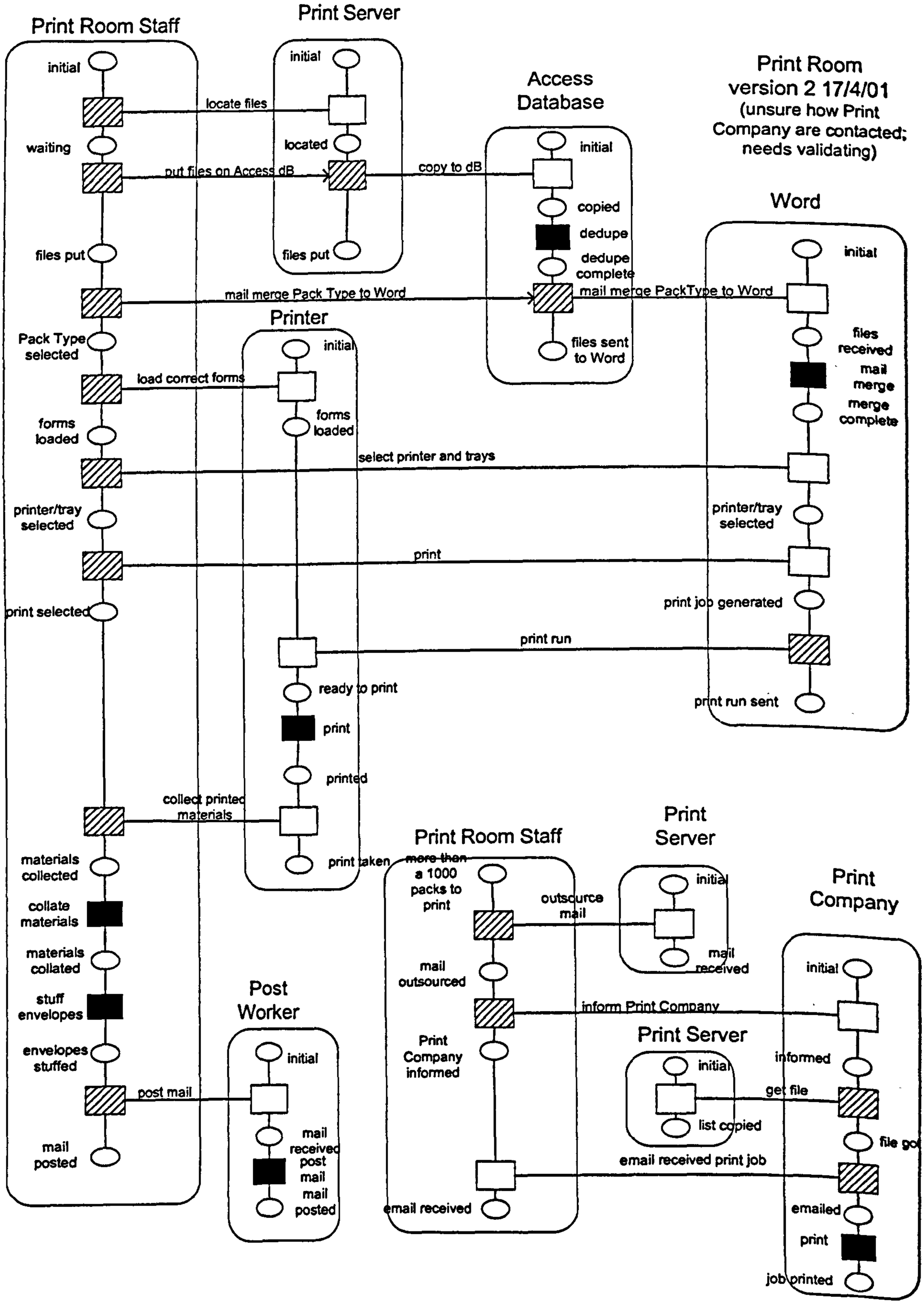


Figure 3.3 Role Activity Diagram for Print Room

4. Current Use Case Model of the Print Room Process

Overview

The role of the Post Room Staff is to print and then put the correct "Application and Welcome to Company X" documents into envelopes dependent upon the type of account opened and the individual requirements within each account. The enveloped packs are then posted. Figure 4.1 depicts the current use case process.

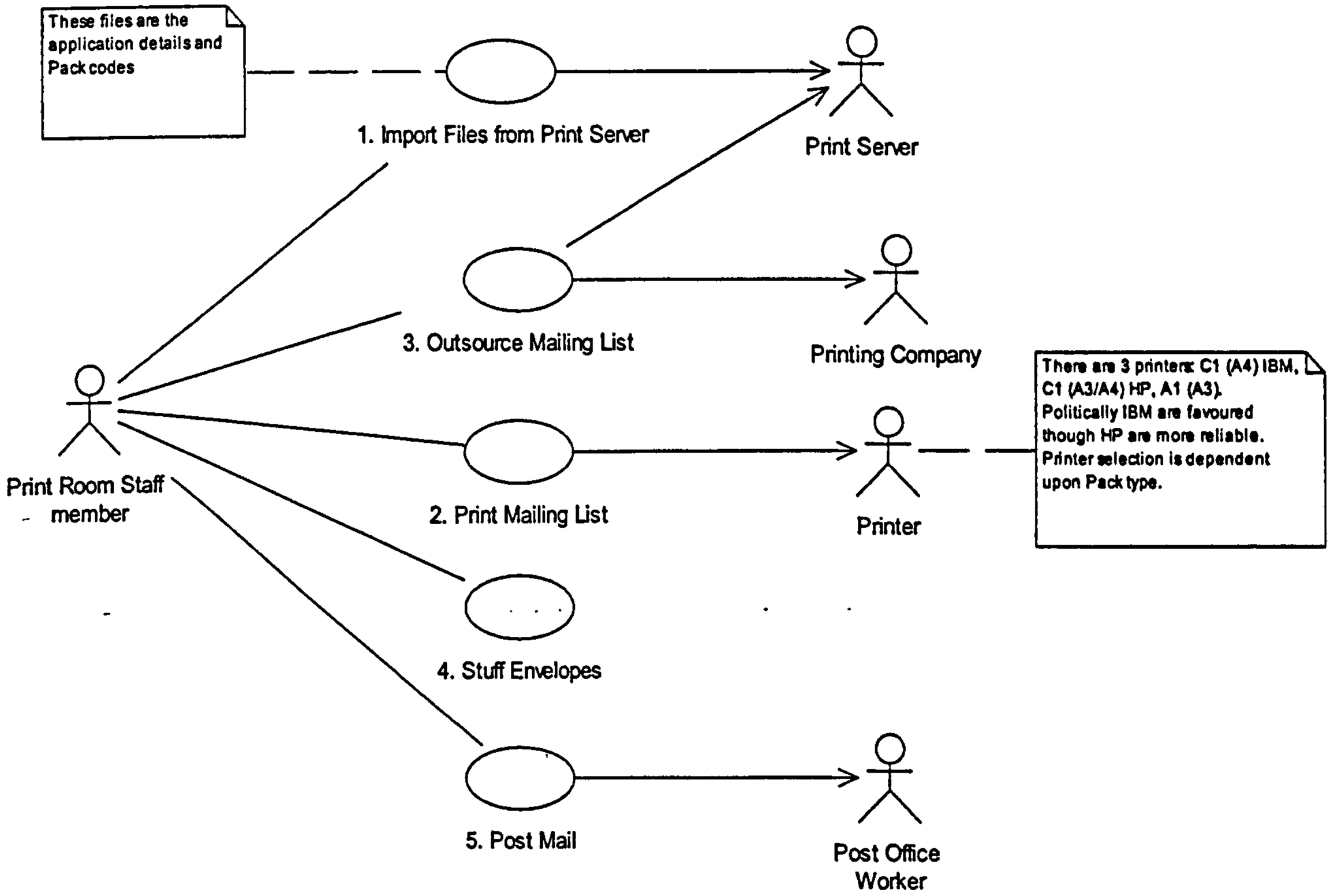


Figure 4.1 Print Room Use Case Diagram

Actor Specification

Actor Name: Print Room Staff Member
Type: Primary
Description: The *Print Room Staff Member* is responsible for file transfer from the *Print Server* to the Access Database, for selecting the Pack Type to do a print run, for conducting the mail merge from Access to Word, and for preparing the Printers to receive the correct documents and print job. Once the print run is over the printed applications are then collated and stuffed into envelopes ready for posting. There are instances of the *Print Room Staff Member* as described in the use case descriptions.

Actor Name: Print Server

Type: Secondary
Description: The *Print Server* stores the files required for printing applications. The applications are accessed by a *Print Room Staff Member* for copying to the Access Database.

Actor Name: Printer
Type: Secondary
Description: Although the *Printer* does not begin an interaction and only responds to events played upon it, it has a political role in Project F. The use case diagram (fig. 4.1) shows there are 3 printers: 2 are IBM and 1 HP. IBM are politically favoured but HP is a more reliable machine.

Actor Name: Printing Company
Type: Secondary
Description: The *Printing Company* is given print jobs of over 1,000 applications. They get the pack lists to print from the *Print Server.*

Actor Name: Post Office Worker
Type: Secondary
Description: This actor collects enveloped application packs deposited by the *Print Room Staff Member* ready for posting to the Customer.

Use Case Descriptions of the Print Room Processes

prUC1: Import files from Print Server

Actors: Print Room Staff member(s), Print Server, Access Database

Context: The post room is ready for another print run. To do this, a member of staff has to get files from Back Office to the Access database.

Pre-conditions: Back Office is functioning and Access database is functioning

Main flow of events

1. The Print Room Staff member accesses the Print Server.
2. The Print Room Staff member locates the files for transfer.
3. The Print Room Staff member opens the 'In' folder in the Access database.
4. The Print Room Staff member imports files to the 'In' folder (explanation: files are copied by 'drag and drop' with mouse)
5. The Print Room Staff member sees the copied file icon in the 'In' folder.

Post conditions

The files are successfully copied to the 'In' folder in the Access database.

Exceptional flows

e1. The Print Room Staff member cannot access the Print Server. What happens here?
e2. The Print Room Staff member cannot locate files for transfer. Why not? What happens here?
e3. The 'In' folder cannot open. Where does the Print Room Staff member import the files to (in main flow 4)?
e4. The files do not copy. Is there a back up path?
e5. No icon is visible. The Print Room Staff member checks the file has been copied. How?

Assumptions

The Print Room Staff member knows the directory where the files are stored in Print Server and can access it.

prUC2: Print Mailing List (Pack Types)

Actors: Print Room Staff member(s), Printers

Context: It is either 9am, 12pm or 3pm and the Print Room staff are readying for a new print run to commence.

Pre-conditions: Files have been imported (copied) to the Access database from the Back Office; the Access Database is functioning normally.

Main flow of events:

1. The Print Room Staff member selects "Pack Type" to print from Pack Type list.
2. The Print Room Staff member selects "Print Mailing List".
3. The Print Room Staff member sees instructions for opening a Company X account in Microsoft Word.
4. The Print Room Staff member sees a mail merge in Microsoft Word.
5. The Print Room Staff member selects "Print" from the File menu.
6. The Print Room Staff member selects the printer to fit the Pack Type.
7. (Optional) The Print Room Staff member selects the number of document pages to print.
8. (Optional) The Print Room Staff member checks the correct forms are in the printer.
9. The Print Room Staff member starts the print run.
10. The Printer prints the documents.

Exceptional Flows

e6. The Print Room Staff member selects the wrong printer.
    e6.1 The Print Room Staff member cancels the print run. (Here or 10?)
    e6.2 The Print Room Staff member selects the correct printer.
    e6.3 The Post Room Staff member restarts the print run.

e9. The print run does not complete.
    e9.1 The Print Room Staff member cancels the current print run.

e9.2 The Print Room Staff member locates the cause of failure. (?)
e9.3 The Print Room Staff member rectifies the problem. How?
e9.4 The Print Room Staff member restarts the print run.

e10. The printer feeds papers from the wrong tray.
e10.1 The Print Room Staff member does what?

Post-conditions: The packs are printed.

Assumptions/Notes: There are presently 23 pack types. The Print Room Staff member has to select the printer (which prints A3 and A4 documents dependent on the Pack Type). The Print Room Staff member has also to select number of pages to print, dependent upon Pack Type, because of a possible fault with Word. There are several different documents that need to be loaded into the specific printer and its corresponding paper tray before the print run can commence. The times of print runs might be altered so that there is a real time printing and mail out? This would cause problems with selecting Pack Types, printers and the correct forms.

Problems:

As highlighted by the Assumptions section above:
- High number of Pack Types forcing a
- high number of different documents to mail out causing
- different printers to do different types of print run

Options:
- Reduce the number of pack types, which will
- reduce the number of documents, which should
- make printer selection easier.

prUC3: Outsource Mailing List

Actors: Print Room Staff member, Print Company, Print Server

Context: More than 1000 packs need to be printed per day. This is more than the Print Room printers can deal with.

Pre-condition: Above 1000 applications are made in a day. Files have been imported (copied) to the Access database from the Back Office; the Access Database is functioning normally.

Main flow of events:

1. The Print Room Staff member selects "Export Mailing List".
2. The Print Room Staff member informs the Printing Company of impending job.
3. The Printing Company access the Print Server to get the print job.
4. The Printing Company inform the Print Room Staff of completion of print job.

Post-condition: -

prUC4: Stuff Envelopes

Actors: Print Room Staff member 1, Print Room Staff member 2

Context: A print run is in progress and there are envelopes to be filled with documents.

Pre-condition: The documents have printed correctly.

Main flow of events:

1. The Print Room Staff member 1 takes the printed material from the printer.
2. The Print Room Staff member 1 folds the printed material.
3. The Print Room Staff member 1 places the material on a desk.
4. The Print Room Staff member 1 writes on a post-it note the details of the print (details?)
5. The Print Room Staff member 1 puts the post-it note onto the printed material.
6. A Print Room Staff member 2 places the printed materials into the envelopes.

Post condition: Envelopes are stuffed, ready for Print Room Staff member to Post Mail

Assumptions/Notes: It is unknown what details are put on the post-it note. It appears that envelopes could be incorrectly stuffed very easily due to the number of envelopes to fill.

*More observation of this important process needs to occur.

prUC5: Post Mail

Actors: Print Room Staff member, Post Office Worker

Context: The print run has been successful and envelopes have been stuffed.

Pre-condition: -

Main flow of events:

1. Print Room Staff member places stuffed envelopes in the correct "To Post" location before 6pm (this is the only postal collection time from Sceptre Court.)
2. A Post Office Worker collects the post.

Post-condition: -

Assumptions/Notes: It might be possible to arrange other mail collections during the day.

5. Envisioned Process Model

It has been decided that the process for application and printing of pack types should be thoroughly remodeled. The goal is to work the entire process through the Front Office alone, removing connections to servers such as the Back Office, the Print Server and also the Access Database.

It is expected that the files for printing will now become available via an interface to the Front Office.

This Role Activity Diagram (figs. 5a and 5b) describes how the envisioned process for Customer Application will work.

Overview of the Process

The complex process of application begins with the Customer applying. A credit check is performed and the Customer informed of the success of the application (state: if successful, an applicant). A new account is created. The relevant Pack code is generated and the Print Room Staff proceed to print the application forms and documents. The Customer (applicant) must return signed documents for the trading account to be activated. The Customer is informed of account activation and the Customer's state now moves to active. At this point the Bank of Scotland is informed of the account details. The BOS validates the account and can reject it if necessary. What happens to the state of the Customer if rejected? If accepted, the BOS will send account materials (cheque book etc) to the Customer.

Roles Played

Customer

The *Customer* applies for an account. It has been stated that a *Customer* has only 3 states: registrant, applicant and active. However, upon closer examination of the application process, we can see that the *Customer* has 7 states (initial, registrant, transient (possible applicant), applicant, unknown (possible applicant?), active and unknown (alternative active?)). The *Customer* states are described in the state diagram below (figure 5).

State 1. Initial.

Before the *Customer* applies he is in an *initial* state. The *Customer* can apply for an account or to register to use the website.

State 2. Registrant.

If a *Customer* registers she changes state to a *registrant*. The transition only shows from the initial state. If the *Customer* successfully opens an account, then she has all the privileges and access that the *registrant* does.

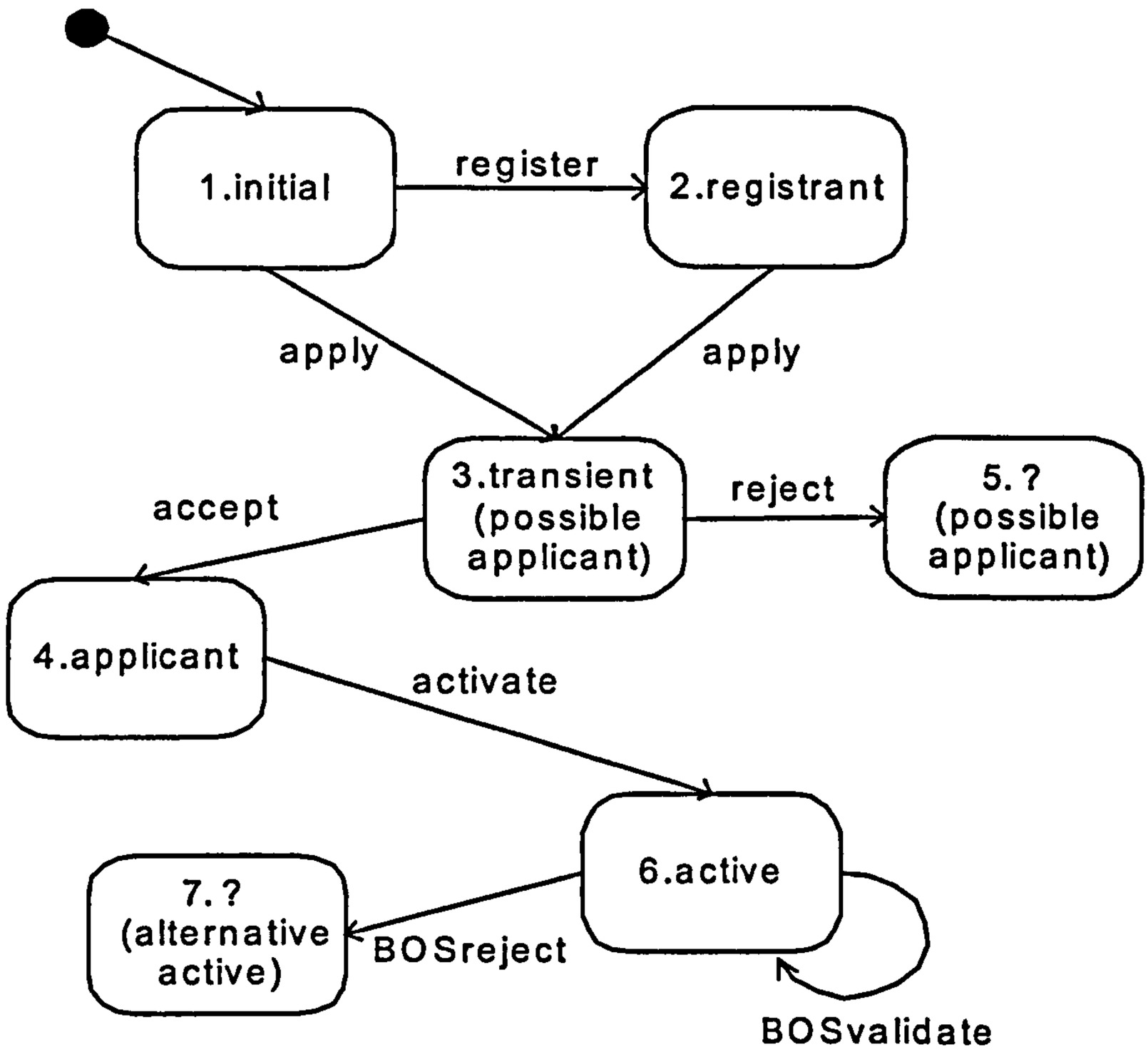


Figure 5. State Transition Diagram for Customer

State 3. Transient (possible applicant).

The *Customer* is held in a waiting state once the application has begun. The *Customer* must wait on news of whether he is credit worthy or not. This means they are an *applicand* or *possible applicant.* The *Customer* can apply from either state 1 (initial) or state 2 (registrant).

State 4. Applicant.

If the *Customer's* credit status is accepted, then the *Customer* transitions to the *applicant* state.

State 5. ? (possible applicant)

The *Customer* transitions to this "unknown" state when he has his credit worthiness rejected. It is not clearly defined what happens here. How does the *Customer* become an *applicant* from here? One possible solution that Company X suggest and do is to ask the *Customer* to send copies of recent utility bills as evidence of credit worthiness. There's no transition from this state to elsewhere because it is unclear whether this practice still continues.

State 6. Active.

When the *Customer* completes and returns application forms, so long as he has done this correctly, his account will be activated, transitioning him from state 4 *applicant* to state 6 *active.* The *active Customer* (state 6) must have their account validated by the *Bank of Scotland.* If the account is validated then there is no change of state in state 6. However, if the *Bank of Scotland* reject the account then the *Customer* state transitions to an "unknown" state 7.

State 7. ? (alternative active).

This "unknown" state is labelled *alternative active.* We do not know what happens here. Can the *Customer* supply utility bills and have the account accepted that way? How does the *Customer* return to state 6?

The *Customer*, if they wish to proceed beyond state 4 *applicant* to state 6 *active,* must sign and return the application forms sent to them. All the *Customer* then does is wait for their new account details to arrive. Of course, they may fall into state 7 if the *Bank of Scotland* rejects their application even after the account is activated by Company X.

Front Office

This is the machine to be built. It provides the interface for the *Customer* to register and to apply for an account. Once the *Front Office* is informed of the *Customer's* credit status, it generates a new account, dedupes records to avoid duplication of the account and generates a pack code. A CSV file is the result of a transformation from the database records to something that *Word* can easily deal with to do a mail merge ready for printing.

The *Front Office* is updated by the *Company X Customer Admin* staff upon receipt of the *Customer's* returned and correctly completed application pack. This updating activates the *Customer's* account. The *Customer* is informed of this via email or text message. The *Front Office* informs the *Bank of Scotland* of the *Customer's* account details. Upon receipt of acceptance notification from the *Bank of Scotland* the *Front Office* informs the *Customer* (acceptance notification can be a rejection).

Credit Checker

This role represents another system that checks the credit worthiness of the applicant *Customer.* The *Credit Checker* informs the *Front Office* of the credit status of the *Customer.* This is all done in real time in a matter of seconds (exact times unknown).

Print Room Staff

The *Print Room Staff* directly access the *Front Office* to get the CSV ready for merging in *Word. Print Room Staff* perform the mail merge on *Word* and perform a print run, preparing the *Printer.* Once the printed materials are collated and stuffed into envelopes, they are given to the *Post Office* for postage to the *Customer.*

Word

This is Microsoft *Word*. It is used because its mail merge application works well – although there is the occasional glitch where an extra, blank page is printed on certain pack types. *Word* sends the print job to the *Printer* to print.

Printer

There are 3 printers: C1 (A4) IBM, C1 (A3/A4) HP, A1 (A3) IBM. Politically, IBM is favoured, though HP is a more reliable. The *Print Room Staff* prepares the specific *Printer* for printing. When the *Printer* receives a print job, it prints out the packs.

It is foreseen that the manual selection of printing trays on the different *Printers* to print the various types and size of packs will be automated. It is undetermined when this automation will take place, or even if it is feasible (though trials show this to be possible – by programming macros into Word).

Post Office

The *Post Office* post the packs once a day. It is hoped that this will increase to at least twice a day. However, this depends upon the amount of packs printed and whether the *Post Office* can be persuaded to collect more than once. If a pack is printed in the morning, it will not be posted until 6pm in the evening, meaning almost a day's delay before the applicant receives their pack. Company X view this as a day wasted.

Bank of Scotland (BOS)

The *BOS* role receives details of a new account and runs its own credit validity check on the account owner. The *BOS* informs the *Front Office* of the results of the credit check it has run. If the credit checks are unfavourable, then the *BOS* can refuse to accept the *Customer's* account. If the *BOS* accepts the account then it will send bank details to the *Customer* direct.

Database Maintenance Staff (DMS)

The *DMS* can access and update account information stored on the *Front Office* when they want. If a *Customer's* address details have changed, for instance, then the *DMS* can update the *Customer's* record accordingly.

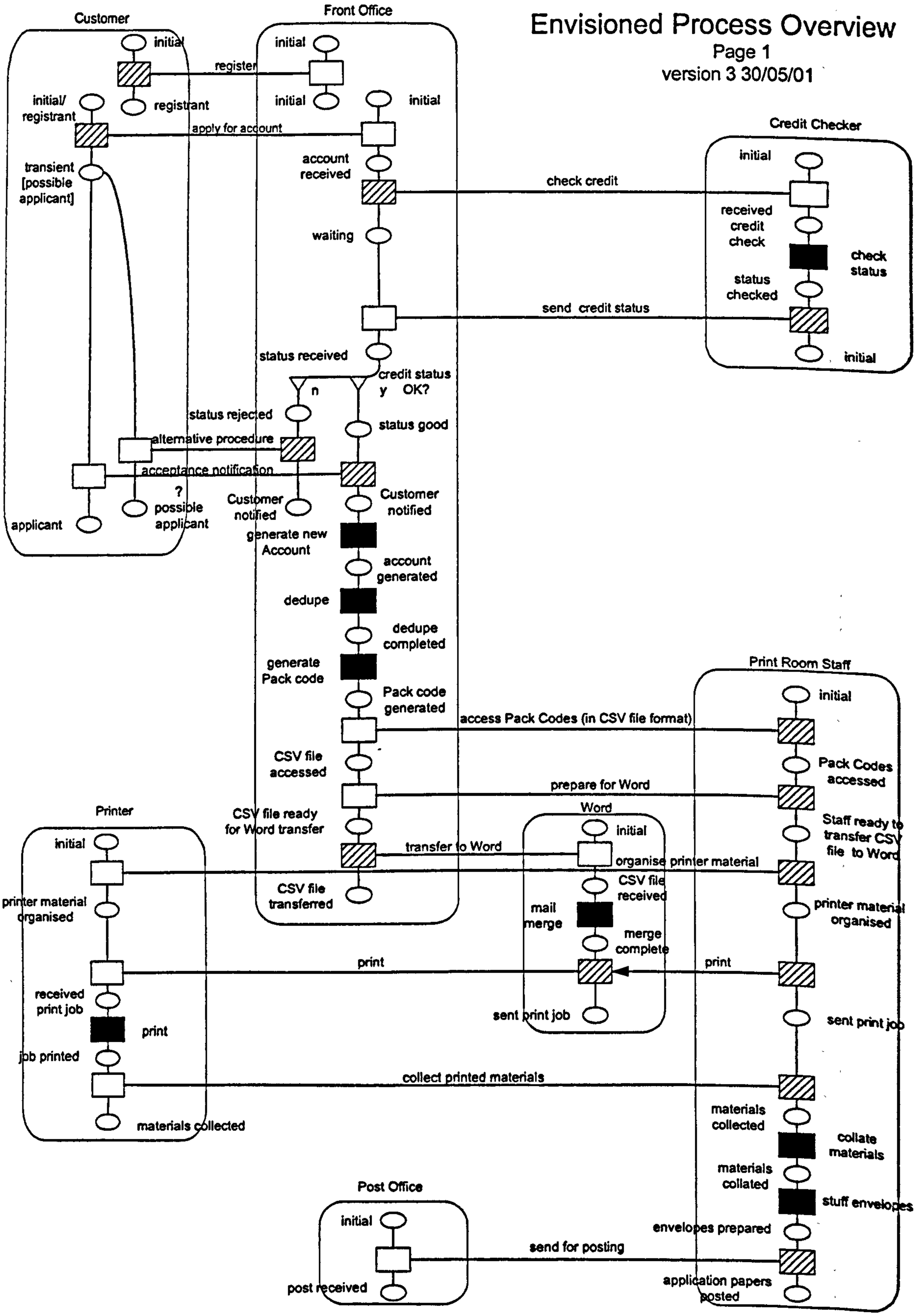


Figure 5a. Page 1 of Role Activity Diagram for Envisioned Process

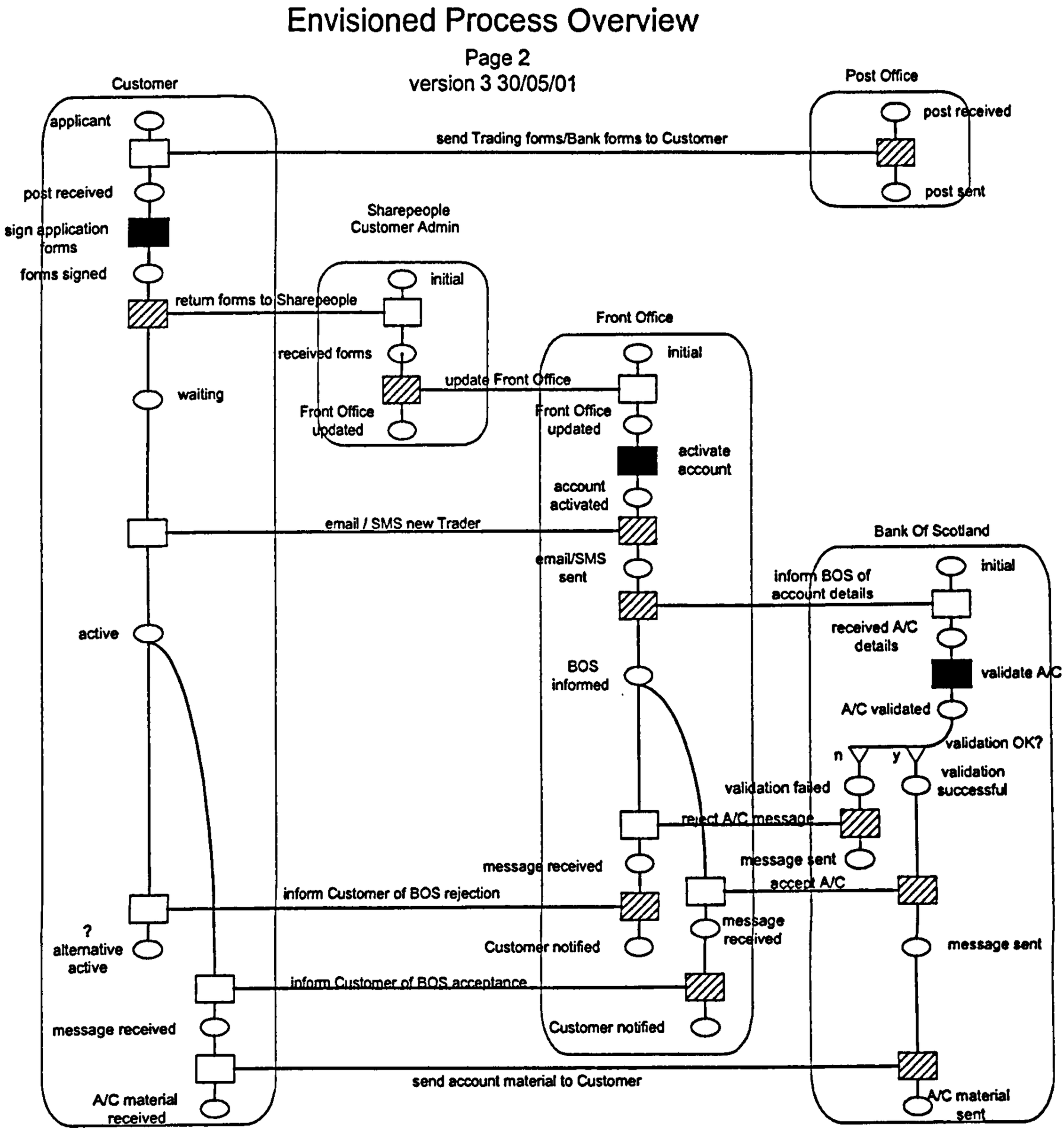


Figure 5b. Page 2 of Role Activity Diagram for Envisioned Process

Note that the envisioned process has reduced the number of roles from 12 to 10. This small reduction of 2 does not reflect the simplification of the current process to the envisioned. The high risk procedure of dragging and dropping files from the *Printer Server* to the *Access Database* of the current process has been replaced by directly accessing the *Front Office.* The *Back Office* has been removed as has the *Print Server.* Two new roles are introduced to the envisioned process: *Company X Customer Admin* and *Database Maintenance Staff.* The *Company X Customer Admin* role should appear in the current process as well but is left out to not over-complicate the diagram. As such, this role is not new in the envisioned process. The *Database Maintenance Staff* is also an existing role in the current process that was not described because it is unclear which role performed the tasks of updating the records at the *Back Office* before they were copied to the *Front Office.* The fundamental simplification of the process is the reduction in complexity of the printing process, in terms of where the CSV files are accessed and how they are transferred to *Word.*

6. Envisioned Package Diagram

The rest of the document turns towards the design of the system. Section 7 describes the use cases at the interface (external design that details the *Customer's* role in completing the application) and also within the system (internal design driven by the actions and interactions of the *Front Office* role to other roles in the envisioned process model). To give a high-level overview of the software architecture of the system, and to guide our understanding of the use case model, this section introduces a UML Package diagram of the proposed architecture driven by the use case model.

Figure 6.1 depicts a UML (Unified Modelling Language) package diagram for the **proposed** system that contains 3 packages and these represent the sub-system structure for the Front Office. The packages impose not design constraints as such but are simply a "filing" mechanism to help organize diagrams, and as a consequence, the design.

1. Interface

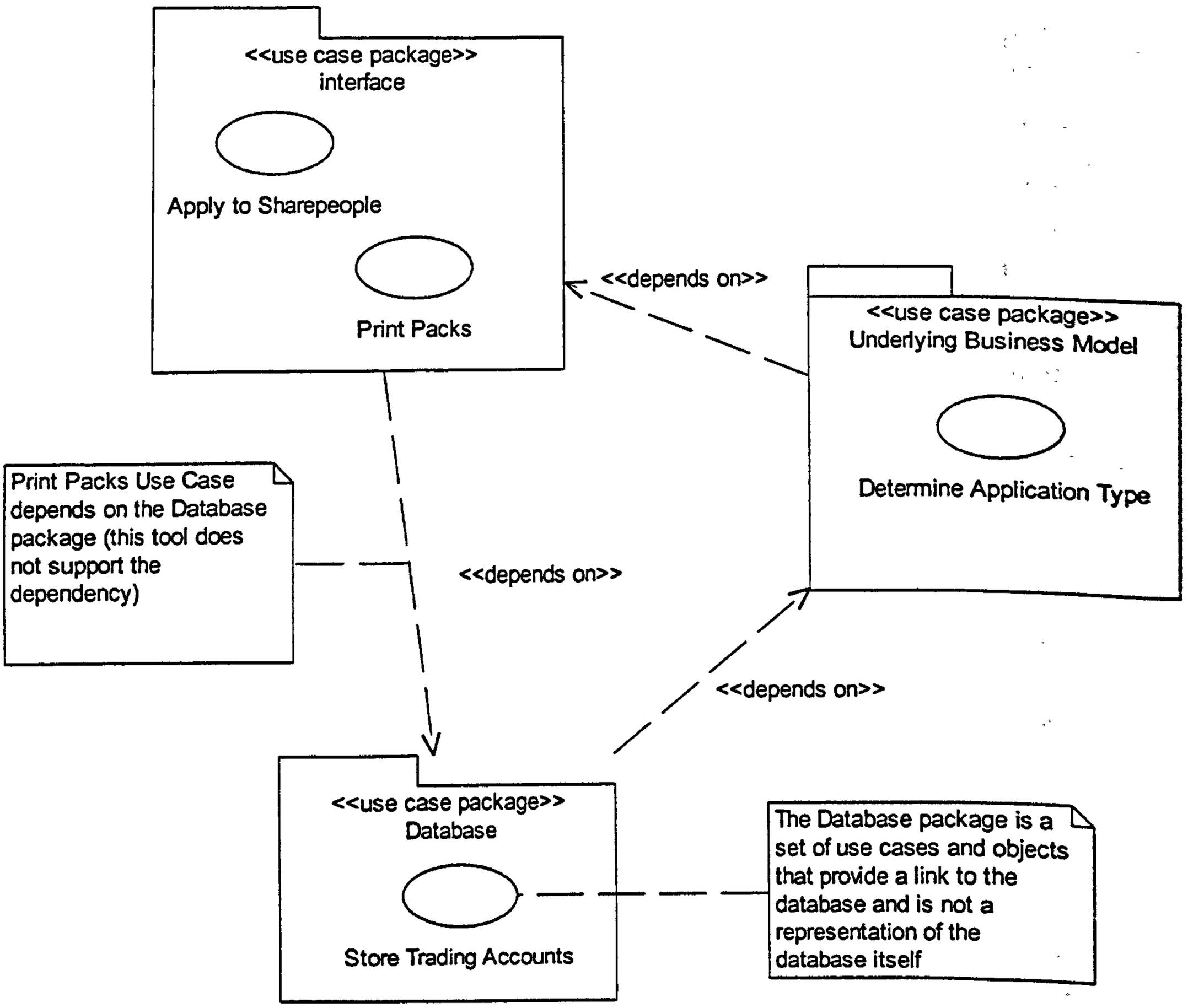
The Interface Package contains all the use cases for accessing the system at a graphical interface level. That is, if a Customer wishes to apply for an account, then their interactions with the Front Office are captured here. These "Apply to Company X" use cases model the system as-is; the use cases are taken directly from the Company X website. When Print Room Staff members wish to do a print run, then they need to interact with the Front Office too.

2. Underlying Business Model

This is where all the processing happens. This package captures the business rules and processes, taking Customer information from the interface package and determining a new Trading Account for the Customer. Note that this package is dependent on the Interface package. This dependency relationship is natural because no new Trading Account processing can occur unless a Customer Application is first received.

3. Database

This package is not the physical database but a layer above it in working memory. The task of the use cases in this package is to prepare and store Trading Accounts to the Oracle database itself. Note that this package is dependent upon the Underlying Business Model Package.



Use Case Package Diagram of the Front Office version 3 11/4/2001

Fig. 6.1 Package Diagram

7. Use Case Model

There are 4 use case diagrams that realize the package diagram (section 6). These are:

Fig. 7.1.1 Interface Use Cases for Apply to Company X
Fig. 7.2.1 Interface Use Cases for Printing and Maintaining Packs
Fig. 7.3.1 Business Layer Use Cases
Fig. 7.4.1 Database Layer Use Cases

Each diagram has a corresponding set of use case descriptions that show the processes involved that fulfill the use case model.

7.1 Interface (Apply to Company X – Register and Trading Accounts)

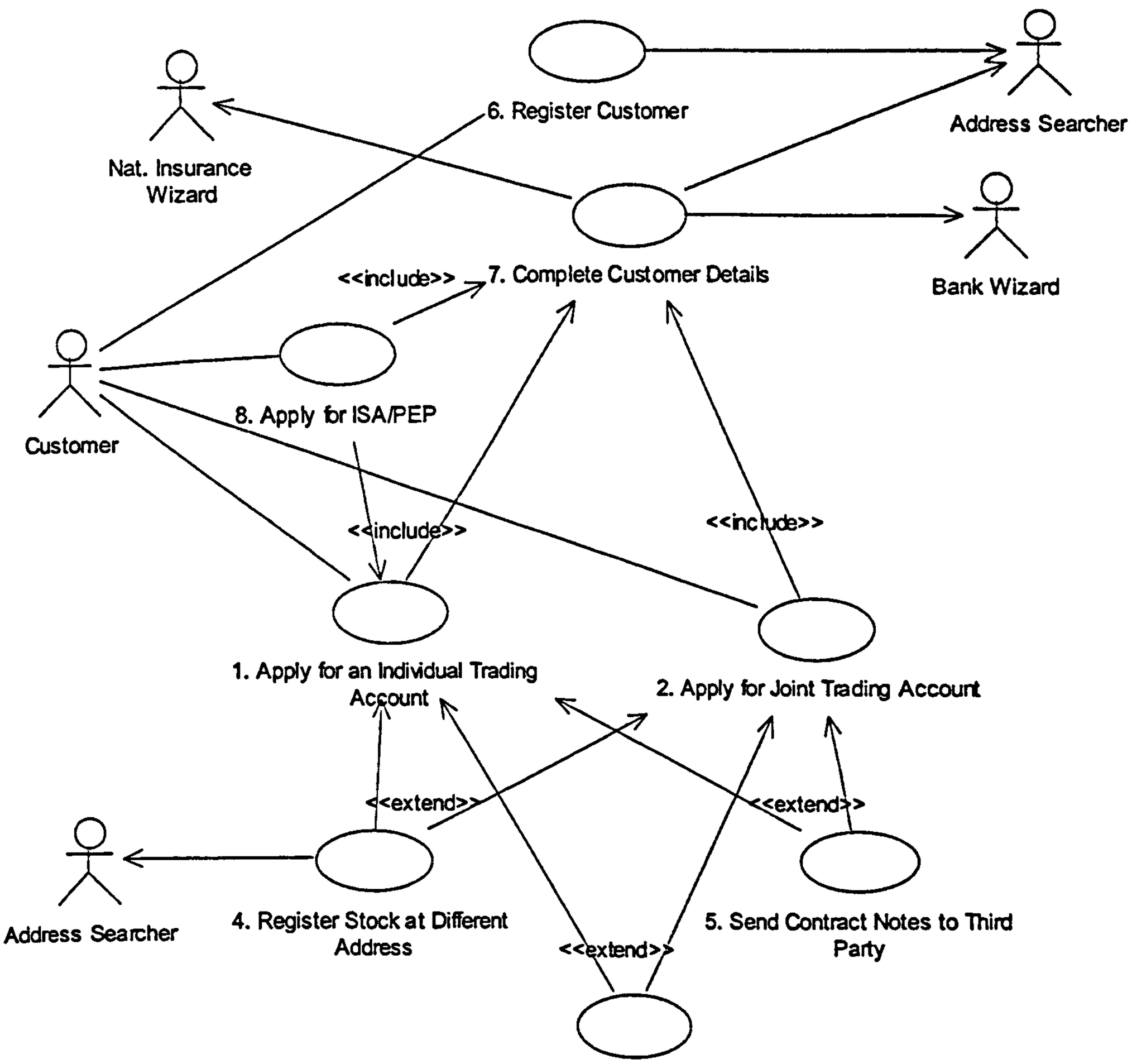


Fig. 7.1.1 Interface Use Case Diagram for Apply to Company X

Figure 7.1.1 realizes the "Apply to Company X" use case from the interface package.

The use case diagram uses the stereotype relationships <<include>> and <<extend>>. The included use case "Complete Customer Details" contains information from the second page of the application form. Use Case 6 "Register Customer" repeats most of this information but not all. As such it does not include use case 7. The Address Searcher actor is a server that locates Customer addresses from the postcode entered.

The <<extend>> relationships describe connections between use cases that might occur dependent upon specific conditions. For instance, if a Customer decides to register stock at a different address he will indicate this on the application at the interface. However, if this option is not selected then the use case is not instantiated.

Actor Specification

Actor Name: Customer
Type: Primary
Description: The Customer begins the whole application process by applying for an account – this could be an ISA or a Trading Account. The Customer interacts with the Front Office by completing an application form on the Company X website.

Actor Name: Address Searcher
Type: Secondary sub-system
Description: The Address Searcher is a sub-system actor. It is an application housed within the Front Office that returns addresses that match postcodes entered by the Customer when applying.

Actor Name: Bank Wizard
Type: Secondary sub-system
Description: The Bank Wizard is a sub-system actor. It is an application housed within the Front Office that returns the bank branch address that houses the Customer's bank account (as entered by the Customer when applying).

Actor Name: National Insurance Wizard
Type: Secondary sub-system
Description: The National Insurance Wizard is a sub-system actor. It is an application housed within the Front Office that validates the Customer's national insurance number, which the Customer must enter to apply for an ISA (if the Customer has ever been issued a national insurance number).

The package use case "Apply to Company X" is described and lower-level use cases (iUC1-iUC7) flow from this description.

Package Diagram level Interface Use Case:
Apply to Company X

Actors: Customer

Context: The Customer wishes to open an account with Company X to start trading in shares.

Pre-condition: The Customer has located the Company X website on the internet. The website is functioning.

Main flow of events:

1. The Customer applies for an individual trading account.

Alternative flow of events:

a1. The Customer applies for a joint trading account.

Post-condition: Front Office (Company X system) sets up a trading account for the Customer.

Alternative flow of events:

a1. The Customer applies for an ISA/PEP

Alternative flow of events:

a1. The Customer registers on the website.

Post-condition: Customer is a registrant.

iUC1: Apply for an Individual Trading Account

Actors: Customer

Context: The Customer wants to open a trading account on the Company X website because the Customer wants to start trading in stocks and shares on the stock market.

Pre-conditions: The Customer logs on to the Company X web site; the Company X website is accessible.

Main Flow of Events

1. The Customer types www.Company X.com into the address bar of the web browser.
2. The Company X website appears on the screen.
3. The Customer selects "Apply Now".
4. The website takes the Customer to the Apply Screen.
5. The Customer reads the guide on how to apply.
6. The Customer sees the choice of Accounts (details: Trading Account or ISA).
7. The Customer selects "Trading Account"
8. The website takes the Customer to the Application Form (details: page 1 of 3)
9. The Customer selects the Country of Residence.
10. The customer selects "Individual Trading Account".
11. The website asks "Where possible do you prefer to deal in certificates?"
12. The Customer ticks the option.
13. The website asks how the Customer wishes to have interest and dividend paid.

14. The Customer selects "Cash".
15. The website asks which currency the Customer wishes any income paid in.
16. The Customer selects "GB Pound".
17. The website asks if the Customer would like to deal in UK registered warrants.
18. The Customer ignores the option.
19. The website asks if the Customer wants stock to be registered at a different address.
20. The Customer ignores the option.
21. The website informs Customer that a "contract note" will be sent on every trade.
22. The website asks the Customer if the Customer requires a contract note to be sent to a third party.
23. The Customer ignores the option.
24. Customer selects "continue".
25. The website takes the Customer to the second page of the Application form.
26. Website presents the Customer with "About the Primary Holder" screen.
27. Customer completes details.
28. Customer selects "Continue".
29. The website shows the completed page 2.
30. Customer selects "Continue".
31. The website goes to page 3 of the Application form.
32. The website presents the Customer with the Customer's details on the application.
33. The Customer selects "Apply".
34. The website presents the Customer with "Application" screen.
35. The Customer sees "Successful Submission" message.
36. The website displays the "Customer Number".
37. The Customer enters a numeric PIN twice.
38. The website presents the Customer with "Change Dealing Password".
39. The Customer enters a password.
40. The Customer clicks "Change" button.
41. The website presents the Customer with a Welcome Information screen.

Alternative Flows of Events
a20 Customer selects Register Stock at Different Address.

a23 Customer selects Send Contract Note to Third Party.

a33 Customer selects "Back".
- a33.1 Website displays page 2 of application form.
- a33.2 Customer makes changes to application.
- a33.3 Customer clicks "Continue" (use case returns to event 33 in the main flow).

a33 Customer selects Transfer Stock

Post-conditions:
The Customer has successfully opened a Trading Account with Company X. The system has generated a unique Customer Number (event 36).

Note: Need to consider this exceptional flow in detail.
Exceptional Flow of events

e34. The website presents the Customer with "Failure" screen.
e34.1 The website informs the Customer of alternative application procedures.
<The Use Case Ends>

iUC2: Apply for Joint Trading Account

Actors: Customer (Primary Holder, Joint Holder)

Context: Two Customers want to open a joint trading account on the Company X website because they want to start trading in stocks and shares on the stock market.

Pre-conditions: The Customers log on to the Company X web site; the Company X website is accessible.

Main Flow of Events

1. Customer types www.Company X.com into the address bar of the web browser.
2. Company X website appears on the screen.
3. The Customer selects "Apply Now".
4. The website takes the Customer to the Apply Screen.
5. The Customer reads the guide on how to apply.
6. The Customer sees the choice of Account (details: Trading Account or ISA).
7. The Customer selects "Trading Account"
8. The website takes the Customer to the Application Form (details: page 1 of 3)
9. The Customer selects the Country of Residence.
10. The customer selects "Joint Trading Account".
11. The website asks "Where possible do you prefer to deal in certificates?"
12. The Customer ticks the option.
13. The website asks how the Customer wishes to have interest and dividend paid.
14. The Customer selects "Cash".
15. The website asks which currency the User wishes any income paid in.
16. The Customer selects "GB Pound".
17. The website asks if the Customer would like to deal in UK registered warrants.
18. The Customer ignores the option.
19. The website asks if the Customer wants stock to be registered at a different address.
20. The Customer ignores the option.
21. The website informs Customer that a "contract note" will be sent on every trade.
22. The website asks the Customer if the Customer requires a contract note to be sent to a third party.
23. The Customer ignores the option.
24. Customer selects "continue".
25. The website takes the Customer to the second page of the Application form.
26. Website presents the Customer with "About the Primary Holder" screen.
27. Customer completes details.
28. Website presents the Customer with "About the Joint Holder".
29. Customer completes details.
30. Customer selects "Continue".
31. The website shows the completed page 2.
32. The Customer clicks "Continue".

33. The website goes to page 3 of the Application form.
34. The website presents the Customer with the Customer's details on the application.
35. The Customer selects "Apply".
36. The website presents the Customer with "Application" screen.
37. The Customer sees "Successful Submission" message.
38. The website displays the "Customer Number".
39. The Customer enters a numeric PIN twice.
40. The website presents the Customer with "Change Dealing Password".
41. The Customer enters a password.
42. The Customer clicks "Change" button.
43. The website presents the Customer with a Welcome Information screen.

Alternative Flows of Events
a20 Customer selects Register Stock at Different Address.

a23 Customer selects Send Contract Note to Third Party

a35 Customer selects "Back".
a35.1 Website displays page 2 of application form.
a35.2 Customer makes changes to application.
a35.3 Customer clicks "Continue" (use case returns to event 35 in the main flow).

a35 Customer selects Transfer Stock

Post-conditions: The Customer has successfully opens a Joint Trading Account with Company X. The system has generated a unique Customer Number (event 38).

Note: Need to consider this exceptional flow in detail.
Exceptional Flow of events

e36. The website presents the Customer with "Failure" screen.
e36.1 The website informs the Customer of alternative application procedures.
<The Use Case Ends>

iUC3: Transfer Stock <<extend>>

Note: this use case is extended from:
iUC1. Apply for Individual Trading Account at alternative flow a33 and
iUC2. Apply for Joint Trading Account at alternative flow a35)

Actors: Customer

Context: Customer is ready to apply for a trading account and has completed the forms. Before clicking the Apply button, the Customer decides to Transfer Stock.

Pre-condition: The Customer has reached the appropriate state in iUC1 and iUC2 ready to Transfer Stock.

Main flow of events

1. Website displays "Transfer Stock" screen.
2. Customer searches for stock for transfer.
3. Customer selects "Add Transfer" to transfer stock.
4. Customer selects "Skip Transfer".
5. If extension from iUC1, Customer returns to event 33 in the main flow, else Customer returns to event 35 in the main flow of iUC2.

Alternative flow of events

At any time Customer can select "Skip Transfer". Customer returns to event 33 in the main flow of iUC1 or if extended from iUC2 returns to event 35 in the main flow.

iUC 4: Register Stock at Different Address <<extend>>

Note:
This use case is extended from:
iUC1. Apply for Individual Trading Account at alternative flow a20 and
iUC2. Apply for Joint Trading Account at alternative flow a20.

The flow of events in this use case takes place in iUC7: Complete Customer Details between events 16 and 17.

Actors: Customer, Address Searcher

Context: Customer has elected to register stock at different address and so must enter the different address details.

Pre-condition: Register Stock at Different Address has been pre-selected.

Main flow of events

1. Customer enters "Postcode".
2. Customer clicks "Address Search".
3. Customer selects street from list.
4. Customer selects "confirm".
5. Website shows Stock Address.

Alternative flow of events

a1. Customer ignores entire use case (explanation: the Stock Address is the same as the Customer's permanent address. Customer moves to next event in respective main flow)

iUC5: Send Contract Notes to Third Party <<extend>>

Note:
This use case is extended from:
iUC1. Apply for Individual Trading Account at alternative flow a23 and
iUC2. Apply for Joint Trading Account at alternative flow a23.

The flow of events in this use case takes place in iUC7: Complete Customer Details between events 16 and 17.

Actors: Customer

Context: Customer has pre-selected the option to send a contract note to a third party and now has to complete the third party's particulars on the application form.

Pre-condition: Contract note to third party selected.

Main flow of events

1. The Customer enters Surname of Third Party.
2. The Customer enters First name of Third Party.
3. The Customer enters Postcode of Third Party.
4. The Customer enters Number/House Name of Third Party.
5. Customer enters Address of Third Party (details: street, town/city, county etc)
6. Customer selects Country of Third Party.

Alternative flow of events

a1. Customer ignores entire use case (explanation: the Third Party Address is the same as the Customer's permanent address. Customer moves to next event in respective main flow)

iUC6: Register Customer

NB This use case is included because a pack code was generated from this. However, it is not certain that this is still the case.

Actor: Customer, Address Searcher.

Context: A Customer wishes to use the Company X website but without opening an account.

Pre-conditions: Customer has web access; Company X site functioning

Main flow of events:

1. Customer enters www.Company X.com into the location box.
2. The Customer accesses the Company X website.
3. Customer selects "Register FREE" option.

4. The Customer sees the register screen.
5. The Customer reads the information on the screen.
6. Customer clicks the "Agree" to terms button.
7. The website displays the Registered User Application form page.
8. The Customer selects "Title" from list.
9. The Customer selects Gender.
10. Customer enters Name (details: first name, middle name(s), surname).
11. Customer enters DoB.
12. Customer enters Email address.
13. Customer selects Country of Residence from list.
14. Customer enters Postcode.
15. Customer clicks "Address Search" button.
16. Website displays list of addresses.
17. Customer selects correct address.
18. Customer selects to not receive junk mail.
19. Customer selects to not receive marketing phone calls.
20. Customer clicks "Continue" button.
21. Website displays the Registration screen.
22. Customer reads information on the screen.
23. Customer reads Customer Number.
24. Customer enters new PIN.
25. Customer re-enters new PIN.
26. Customer clicks "Change" button.
27. Website displays "Welcome to Company X" screen.

Alternative flow:
a6. Customer clicks "Back" button.
    a6.1 Website returns to the home page.

Post-conditions:
Customer is a registered user. NB Pack code generated for this? Or has this generation stopped?

iUC7: Complete Customer Details <<include>>

This use case is included from:
iUC1: Apply for Individual Account at event 27 and returns to event 28;
iUC2: Apply for Joint Account at event 27 and returns to event 28 and then from event 29 and returns at event 30
iUC8: Apply for an ISA/PEP at event 19 and returns at event 20.

Actor: Customer (Individual and Joint Holder), (Address Searcher, Bank Searcher, National Insurance Wizard)

Context: Customer has completed the first page of the Trading Account application form and now wants to complete the second page

Pre-condition: Page 1 of the form completed.

1. Customer selects "Title".
2. Customer selects "Gender"
3. Customer types in name (details: "First Name", "Middle Name(s)", "Surname").
4. Customer enters "Date of Birth".
5. Customer selects "Nationality".
6. Customer types in mother's maiden name.
7. Customer ignores Credit Card Number option.
8. Customer reads Credit Card information.
   8.1 **(Only on ISA/PEP applications:** Customer enters National Insurance Number).
9. Customer enters email.
10. Customer enters telephone number (details: Customer enters one or more from "evening number", "daytime number", "mobile number).
11. Customer enters "Postcode".
12. Customer enters the date Customer entered current residence.
13. Customer clicks "Address Search".
14. Customer selects street from list.
15. Customer selects "confirm".
16. Website shows Customer address.
17. Customer enters Bank Account information (details are: "Name of Account Holder", "Account Number", "Branch Sort Code", "Date Opened").
18. Customer selects the No Junk Mail option.
19. Customer selects the No Marketing Phone Calls option.

Alternative flow of events
a15. Customer selects "Amend".
   a15.1 Website returns Customer to event 13 in the main flow.

a18. Customer returns to iUC10 (if included from iUC8).

Exceptional Flows of Events
e8.1 Customer ticks "Never been issued an NI#.

e14 Customer cannot find number and street.
   e14.1 Customer re-enters postcode.
   e14.2 Customer clicks Address Search.
   e14.3 Customer returns to use case main flow event 14.

Post-condition: Page 2 of the application form has been completed.

iUC8: Apply for a ISA/PEP

Actor: Customer

Context: Customer wishes to purchase an ISA (Maxi, Mini, Transfer ISA) before the legal cut off date and so uses the Company X Website to do this.

Pre-condition: Website is up and running, Customer logs on to internet OK

Main Flow of Events

1. The Customer types www.Company X.com into the address bar of the web browser.
2. The Company X website appears on the screen.
3. The Customer selects "Apply Now".
4. The website takes the Customer to the Apply Screen.
5. The Customer reads the guide on how to apply.
6. The Customer sees the choice of account (details: Maxi ISA, Mini ISA, Transfer ISA – which creates an new ISA account, PEP Transfer).
7. The Customer selects "Maxi ISA"
8. Website shows application for a Maxi ISA (application page 1 of 7).
9. Website presents option. (Login as Current Customer / Not a Current Customer).
10. Customer selects "Not a Current Customer".
11. Website takes Customer to page 2 of application (Authorisation and Declaration)
12. Customer reads details.
13. Customer selects "Accept".
14. Website takes Customer to page 3 of application (Management Fees Payment).
15. Customer selects "ISA Account" (alt: Trading Account).
16. Customer selects "Continue".
17. Website takes Customer to page 4 of application (Trading Account Details).
18. Customer completes applying for trading account details (includes iUC1 events 9-24).
19. Customer completes details.
20. Customer selects "Continue".
21. Website takes Customer to page 6 of the application (Personal Information).
22. Customer reads information.
23. Customer chooses not to receive marketing mailings.
24. Customer chooses not to allow Company X to pass on information to other Multinational A companies.
25. Customer chooses not allow other companies to use personal information.
26. Customer selects not to allow Company X to telephone Customer about marketing.
27. Customer selects "Continue".
28. Website takes Customer to page 7 of the application (completed form).
29. Customer selects "Apply".
30. Customer completes applying for trading account details (includes iUC1 events 34-41)

Note that event 41 is not reached – website fails at event 40 of iUC1 in iUC8.

Alternative Flows of Events

a7. Customer selects "Mini ISA".

a7. Customer selects "Transfer ISA". (Note: cannot get beyond page 5 of application form on the website.)

a7. Customer selects "PEP Transfer". (Note: PIN entry failing on "application" screen on website.)

a10. Customer selects Login In as Current Customer

a13. Customer selects "Reject"

a13.1 Website returns Customer to Website Home Page

a27. Customer selects "Back".
a27.1 Website returns the Customer to completed page 5 of the application.
a27.2 Customer makes necessary changes.
a27.3 Customer selects "Continue".
a27.4 Website takes Customer to event 21 in the main flow.

a29. Customer selects "Back". (Website fails here).

Post-condition: New Maxi ISA account (should be) created for Customer – although the use case does not terminate in this specific instance.

7.2 Interface (Maintain and Print Packs)

The interface package use case "Print Packs" is realized by the interface use cases iUC9: Maintain Pack Codes and iUC10: Print Pack Types.

The use cases Print Pack Types and Maintain Pack Codes deal with the process of printing customer application packs and also maintaining the records in the database. The interface screens for this are only for company internal use.

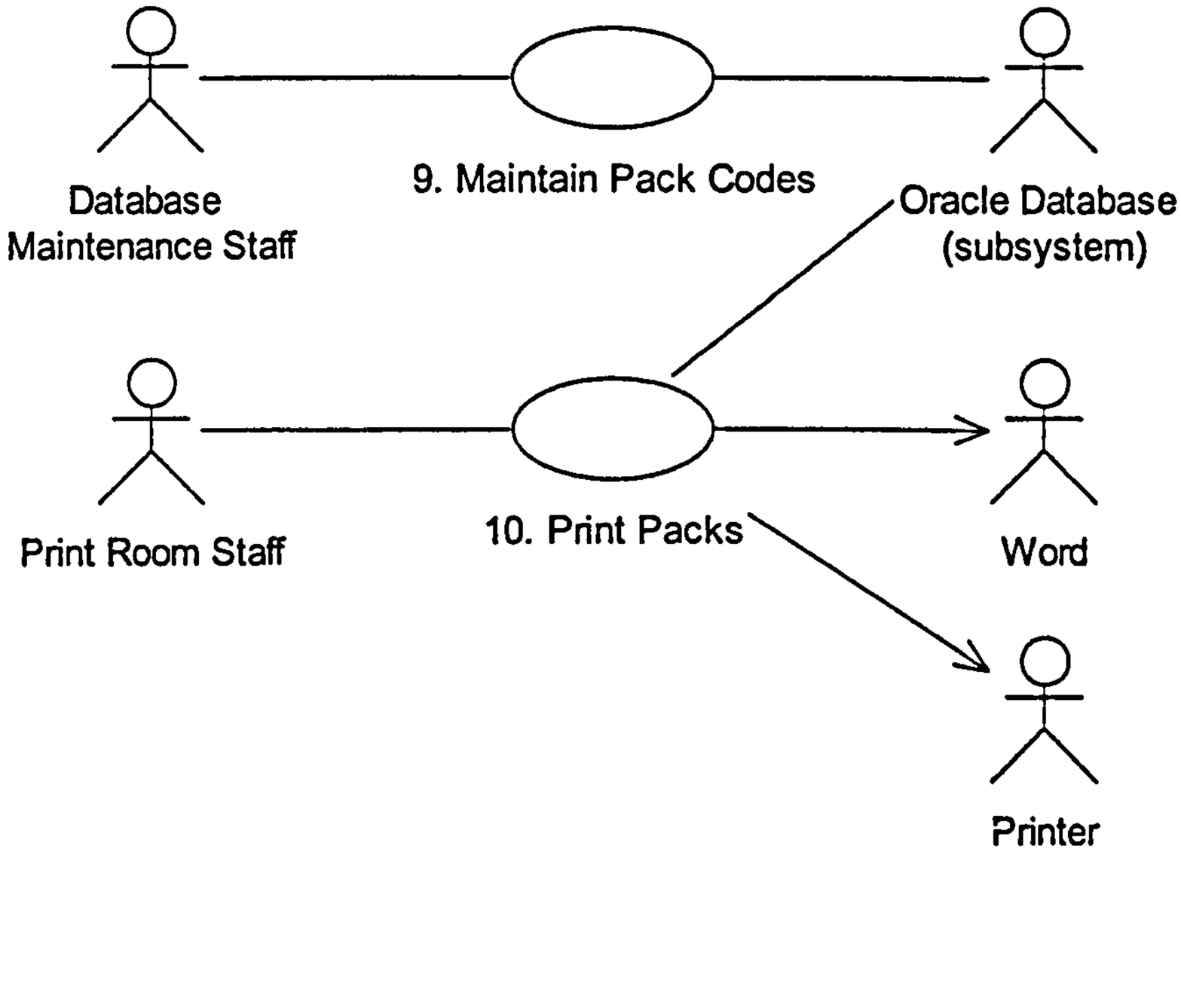


Figure 7.2.1. Use Case Diagram for Interface UC Package "Print Packs"

Actor Specification

Actor Name: Print Room Staff
Type: Primary
Description: The Print Room Staff are responsible for printing application packs that have been completed by Customers on the website.

Actor Name: Database Maintenance Staff.
Type: Primary
Description: The Database Maintenance Staff are responsible for updating any information about the Customer's account that might have changed outside of the scope of automated updates. That is, the DMS might alter the address of the Customer or add a telephone number. Automated updates might include money in and out concerning financial trades.

Actor Name: Printer
Type: Secondary

Description: There are 3 printers: C1 (A4) IBM, C1 (A3/A4) HP, A1 (A3) IBM. Politically, IBM is favoured, though HP is a more reliable. The Print Room Staff prepares the specific Printer for printing. When the Printer receives a print job, it prints out the packs. It is foreseen that the manual selection of printing trays on the different Printers to print the various types and size of packs will be automated. It is undetermined when this automation will take place, or even if it is feasible.

Actor Name: Oracle Database
Type: Secondary subsystem
Description: The Oracle Database is the data store of the Front Office (and so represents the Front Office here). All Customer records are kept here and all pack codes are stored here.

Actor Name: Word
Type: Secondary Application
Description: Word is used to mail merge documents ready for printing and then is manipulated to select a printer for the print run.

Package Diagram Level Interface Use Case: Print Packs

Actor: Print Room Staff

Context: The Print Room Staff are preparing to do another print run. They may need to update the database first.

Pre-conditions: There are packs to print (system state?).

Main flow of events:

1. Print Room Staff accesses the Front Office.
2. Print Room Staff maintains Pack Codes.

Post-condition: Pack Codes updated.

Alternative flow of events:

a.2 (Optional) Print Room Staff maintains Pack Codes.
    a2.1 Print Room Staff prints pack type.

Post-condition: Packs printed.

iUC9: Maintain Pack Codes

Actor: Database Maintenance Staff, Oracle Database

Context: Database Maintenance Staff wish to alter some of the pack codes and corresponding records; for instance, the information supplied by a Customer might be incorrect.

Pre-condition: Oracle accessible?

Main flow of events:

1. The Database Maintenance Staff access the Front Office through the Maintenance screen.
2. The Database Maintenance Staff access copies of records in the Oracle Database.
3. The Database Maintenance Staff make changes to the records.
4. (Optional) The Database Maintenance Staff dedupes the records.
5. The Database Maintenance Staff saves changed records back to the Oracle Database.

Post-condition: Records saved in database

Assumption: Event 4 – deduping of records – should have occurred prior to here. It should be automated and not left to human judgement.

iUC10: Print Pack Types

Actor: Print Room Staff, Printer, Word, Oracle Database

- Context: There are packs that need to be printed to send off to registrants (?), applicants and to active account holders. The Print Room Staff regularly print off runs of various pack types.

Pre-condition: There are packs waiting to be printed.

Main flow of events:

1. Print Room Staff access the Front Office through the Print Control Screen.
2. Print Room Staff ask Front Office to retrieve CSV file from Oracle Database.
3. Print Room Staff access CSV file from Front Office directory.
4. Print Room Staff starts Word.
5. Print Room Staff transfers CSV file from Front Office to Word.
6. Print Room Staff begins Mail Merge on Word.
7. (Optional) Print Room Staff organize Printer material (details: Print Room Staff put forms into the correct trays on the Printer).
8. Word finishes the Mail Merge.
9. Print Room Staff selects Print.
10. Printer prints the material.

Alternative flows of events

Events 7 and 8 can occur in any order.

Post-conditions: Print job complete.

7.3 Business Layer Use Case Model

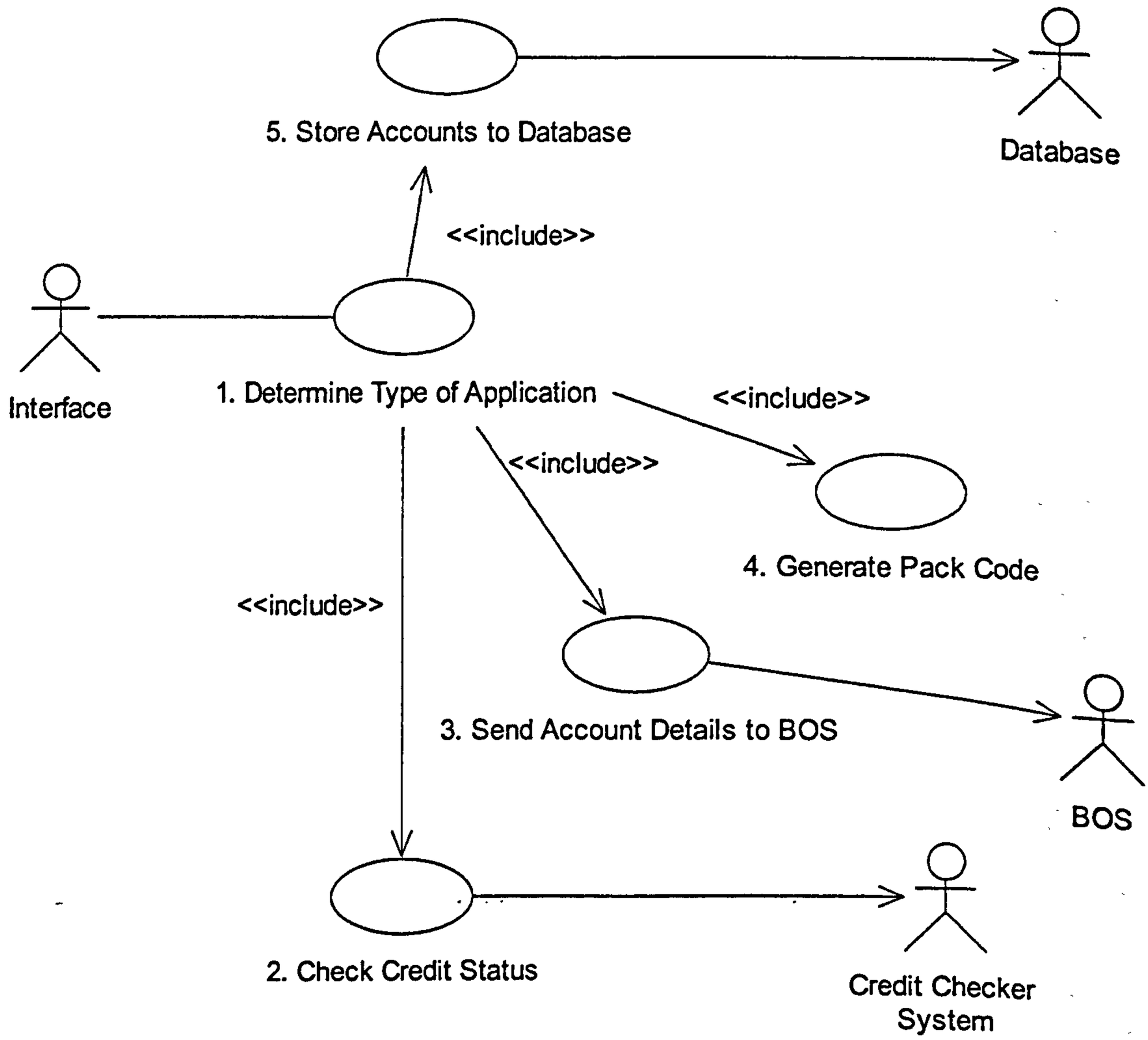


Fig. 7.3.1 Use Case Diagram of Business Layer Package

The use case diagram (fig. 7.3.1) models the underlying processes that the Front Office performs to convert (or otherwise) a Customer application at the interface into a new Trading Account.

Figure 7.3.1 is driven by the process Determine Type of Application. All other use cases in the diagram have an <<include>> relationship to this use case. The <<include>> simply denotes that for the Determine Type of Application to complete successfully, it needs to employ the included use cases.

Actor Specification

Actor Name: Interface
Type: Primary subsystem
Description: The Interface subsystem informs the Business Model Layer that an application is being completed at the GUI by a Customer.

Actor Name: Credit Checker System
Type: Secondary
Description: This actor represents another system that checks the credit worthiness of the applicant Customer. The Business Layer subsystem drives this process. The Credit Checker System returns the credit worthiness of the Customer.

Actor Name: Bank of Scotland
Type: Secondary
Description: The Bank of Scotland is supplied with the Customer's new account information. The Bank of Scotland can accept or reject the account.

Actor Name: Database
Type: Secondary subsystem
Description: This is the Database Layer package whose role is to prepare information formats so that they match the structure of the Oracle Database itself.

Use Case Descriptions for the Underlying Business Model Layer Package

The package use case "Determine Application Type" is described and lower-level use cases flow from this description.

Package Diagram level Underlying Business Model Use Case:
Determine Application Type

Actors: Credit Checker System, BOS, Database (sub-system)

Context: The Customer has completed the application form on the website. The Business Layer now has to get that application and determine the type of account required, do all necessary checks, generate a Pack Code and send the account off to the database for storage.

Pre-condition: Customer has clicked "Apply" at the GUI.

Main flow of events:

1. The Business Layer determines the application type.
2. The Business Layer checks Customer credit status.
3. The Business Layer stores account in Database.
4. The Business Layer sends account details to BOS.
5. The Business Layer generates a pack code.

Post-condition: The Customer's trading account is set up.

bUC1: Determine Type of Application

Actors: Interface (sub-system), Database (sub-system)

Context: The Business Layer receives an application and has to process it.

Pre-condition: The application has been completed. Customer clicks "Apply".

Main flow of events:

1. Interface informs Business Layer of completed Application.
2. The Business Layer retrieves general application details.
3. The Business Layer stores the general application details.
4. The Business Layer retrieves the Customer's contact details.
5. The Business Layer stores the Customer's contact details.
6. The Business Layer retrieves the Customer's Bank Details.
7. The Business Layer stores the Customer's bank details.
8. The Business Layer checks the Customer's credit status.
9. The Business Layer creates a new trading account for the Customer.
10. The Business Layer stores the new account details in the Database.
11. The Business Layer sends new account details to BOS.
12. The Business Layer retrieves the application details.
13. The Business Layer generates a pack code for the application.
14. The Business Layer increments the pack code print list.

Post-conditions: The Business Layer knows the application type, has created an account and validated it.

Exceptional flow of events:
e5 The Customer's postcode matches the Isle of Man, Guernsey or Jersey.
e5.1 The Business Layer refuses the Customer's application. The application process ends here.

bUC2: Check Credit Status <<include>>

Note:
This use case is included from:
bUC1 Determine Type of Application at event 8.

Actors: Credit Checker System, Customer (secondary), Company X (company)

Context: A Customer has completed an application and Company X want to check that the Customer has sufficient credit status to open an account.

Pre-conditions: Customer has completed application form. Bank details have been retrieved. The connection to the Credit Checker System is operational.

Main flow of events:

1. The Business Layer connects to the Credit Checker System.
2. The Business Layer sends the Customer's bank details to the Credit Checker System.
3. The Credit Checker System returns Customer credit worthy.

Post-conditions: Customer is credit worthy. Business Layer proceeds with application.

Exceptional flow of events

a3 The Credit Checker System returns Customer not credit worthy.
a3.1 Company X informs Customer of application rejection.
a3.1.1 Company X asks Customer to send Utility Bills for proof of credit worthiness.

bUC3: Inform BOS (Bank of Scotland) <<include>>

Note:
This use case is included from
bUC1 Determine Type of Application at event 11.

Actors: Bank of Scotland (BOS), Administrator to change state of Front Office on receipt of account forms from Customer (?)

Context: The Front Office has set up a bank account for the new applicant, has received returned signed forms from the Customer and now has to inform the BOS about the account.

Pre-condition: Front Office is informed that Applicant has returned completed forms.

Main flow of events:

1. The Front Office sends the account details to the BOS.
2. The BOS confirms the validity of the account.

Post-condition: The BOS validates the new account.

Exceptional flow of events:

a2. The BOS refuses to validate the account.
a2.1 The Front Office informs the Customer of rejection.

bUC4. Generate Pack Code <<include>>

Note:
This use case is included from:
bUC1. Determine Type of Application at event 13.

Actor: None. The use case is driven by the Business Layer sub-system.

Context: A Customer's application has been successful and a new account created. To make sure the Customer receives the correct information pack, which should include documentation specific to the application, the Business Layer has to examine the application to determine the Pack Type.

Pre-condition: The Customer account has been created and validated.

Main flow of events:

1. The Business Layer retrieves the Customer's application form.
2. The Business Layer determines the country of residence.
3. The Business Layer determines the account is "individual."
4. The Business Layer determines the Customer prefers to deal in certificates.
5. The Business Layer determines the Customer wants cash.
6. The Business Layer determines the currency of payment.
7. The Business Layer determines the Customer prefers to avoid UK warrants.
8. The Business Layer determines the Customer does not want stock registered at a different address.
9. The Business Layer determines the Customer does not want a contract note sent to a third party.
10. The Business Layer determines the Customer wants to transfer stock.
    10.1 The Business Layer determines the number of stock transfers.
11. The Business Layer correlates all attribute values from the Customer's application.
12. The Business Layer checks correlation against pack codes for match.
13. The Business Layer locates the correct Pack Code.
14. The Business Layer stores the found Pack Code.

Alternative flows of events

a3. The Business Layer determines the account is "joint".

a4. The Business Layer determines the Customer prefers not to deal in certificates.

a5. The Business Layer determines the Customer wants shares.

a7. The Business Layer determines the Customer prefers to deal in UK warrants.

a8. The Business Layer determines the Customer wants stock registered at a different address.

a9. The Business Layer determines the Customer wants a contract note sent to a third party

a10 The Business Layer determines the Customer does not want to transfer stock.

Post-condition:
Pack code has been generated and stored.

bUC5: Store Account in Database <<include>>

NB. This use case is included from:
bUC1 Determine Type of Application at event 10.

Actors: Database (sub-system)

Context: A new account has been created and fully validated. A permanent record of the account has to be written to the system's database.

Pre-condition: The Business Layer is ready to store the new account. Access to the database is available.

Main flow of events:

1. The Business Layer locates the new account.
2. The Business Layer transfers the account to the database.

Post-condition: the account is stored in the database.

7.4 Database Use Case Package

Figure 7.4.1 shows the database layer processes. Note that this is not modeling the Database itself but simply prepares new accounts for storage in a table in the database.

Much of this model has still to be determined.

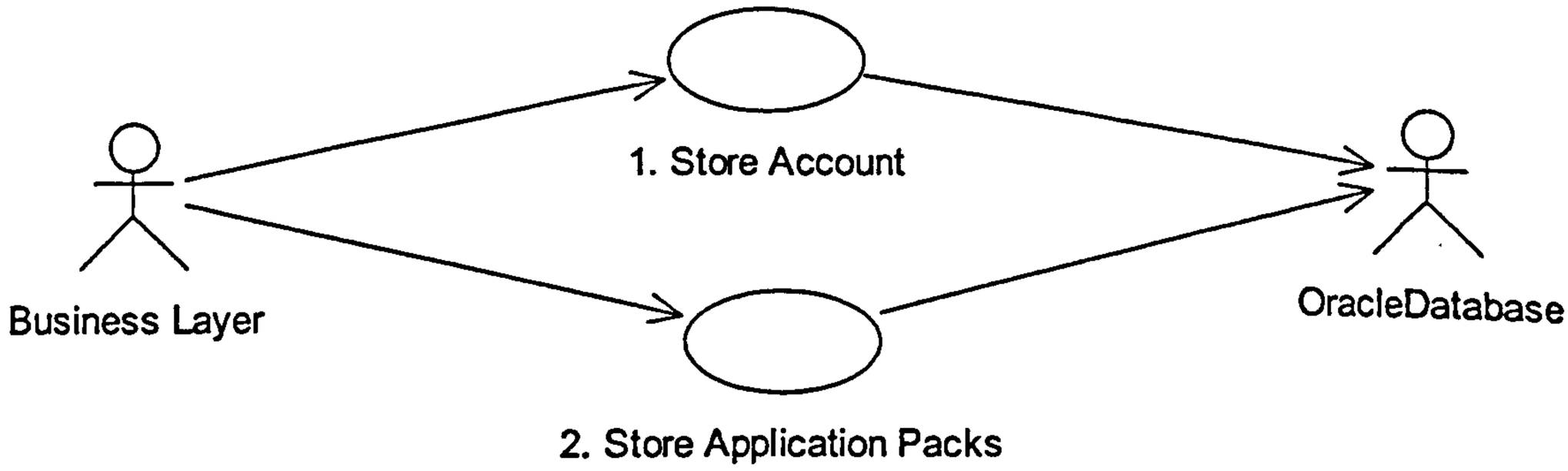


Figure 7.4.1 Use Case Diagram for Database Package

Actor Specification

Actor Name: Business Layer
Type: subsystem (primary?)
Description: The Business Layer actor represents the package "Underlying Business Model" – it is a subsystem of the Front Office. Its role is to provide the records required for transformation to be put into the Oracle Database.

Actor Name: Oracle Database
Type: Secondary subsystem
Description: The Oracle Database is the data store of the Front Office. All Customer records are kept here and all pack codes are stored here.

Use Case Descriptions for the Database Package

Package Diagram Level Database Use Case:
Store Trading Accounts

Actors: Business Layer (sub-system), Oracle (Database)

Context: There are records that need to be saved in the Oracle Database. The records consist of application packs and customer account information.

Pre-conditions: There are application packs and trading accounts to be stored.

Main flow of events:

1. The Database Layer receives application pack types from the Business Layer.
2. The Database Layer stores application packs on the Oracle Database.
3. The Database Layer receives Trading Accounts from the Business Layer.
4. The Database Layer stores Trading Accounts on the Oracle Database.

Post-conditions: Application packs are stored; trading accounts are stored.

dUC1: Store Account

Actors: Business Layer, Oracle (database)

Context: A customer Trading Account has been created and validated. It now needs a permanent record created for it in the database itself.

Pre-condition: New Trading Account created.

Main Flow of Events

1. The Business Layer informs the Database layer (system) of a new Account.
2. The Database layer (system) accesses the Account attributes
3. The Database Layer organises the details of the new account to fit the Oracle database table.
4. The Database stores the (re-)arranged Account details
5. The Database sends the account details to the Oracle dBase.

Post-condition:

Trading Account record created in the Oracle database and populated with corresponding data.

dUC2: Store Application Packs

Actors: Business Layer, Oracle (Database)

Context: The Business Layer has a (set of) Pack Codes in run-time memory and wishes to store them in the database so that application packs can be printed and sent out to Customers.

Pre-conditions: A PackCode object has been created and populated in the Business Layer.

Main flow of events:

1. The Business Layer informs the Database layer (system) of a new Pack.
2. The Database layer accesses the Pack attributes.

3. The Database Layer organises the details of the new Pack to fit the Oracle database table.
4. The Database (system) stores the (re-)arranged Pack details.
5. The Database sends the Pack details to the Oracle dBase.

Post-condition:

Pack Code record created in the Oracle database and populated with corresponding data.

Note: The Business Layer class model describes a linked list of each pack type. It could be that once every 10 or 100 or whatever number of objects or at whatever designated time interval the objects are sent to the database, as opposed to unique occurrences.

8.Class Model

The process of moving from use cases to class diagrams is not shown in this document to save a lot of space! The class models defined so far are:

Fig. 8.1.1 Interface class diagram for Package UC Apply to Company X
Fig .8.1.2 Interface class diagram for Package UC Print Packs
Fig. 8.2.1 Business Layer class diagram
Fig. 8.3.1 Database Layer class diagram

8.1 Interface Class Diagrams

The Interface class diagram for "Apply to Company X" Package level UC (fig 8.1.1) shows two types of class, denoted by the sterotypes <<GUI>> and <<interface>>. The <<GUI>> represents a class taken directly from the website graphical interface. The <<interface>> stereotype describes classes that provide links to other systems and are not seen in any graphical format.

The classes are drawn from use cases and the GUI classes have notes adorned to them stating which event in the use case description the class was identified (just to show the link – normally this is left to some other document).

Figure 8.1.2 is a class diagram for the Package UC "Print Packs". This class diagram (for now) includes the Interface Package <<GUI>> classes "PrintRoomStaff" and "MaintainRecords" plus Database package class "RecordAccess" as these add clarity without overcrowding the diagram. (Interaction models for this class diagram can be found in section 9).

Class Description for Figure 8.1.1

Class Name: Individual Account
Type: GUI
Description: This class at the graphical interface of the system provides a temporary store for the personal information entered by the Customer for Single Account Holders.
From use case: iUC1 events 10, 26; iUC2 events 10, 26; iUC7

Class Name: Joint Account
Type: GUI
Description: This class at the graphical interface of the system provides a temporary store for the personal information entered by the Customer for Joint Account Holders.
From use case: iUC2 event 28; iUC7

Class Name: Application
Type: GUI
Description: This class stores the information that each trading account needs to know and from where pack codes are mainly generated – the sum of its attributes should be used to generate a pack code when the information is held in the Business Layer.
From use case: iUC1 event 8; iUC2 event 8.

Class Name: Third Party

Type: GUI
Description: If a Third Party address is selected in the application process then the details of the third party are stored here.
From use case: iUC1 event a23; iUC2 event a23; iUC5

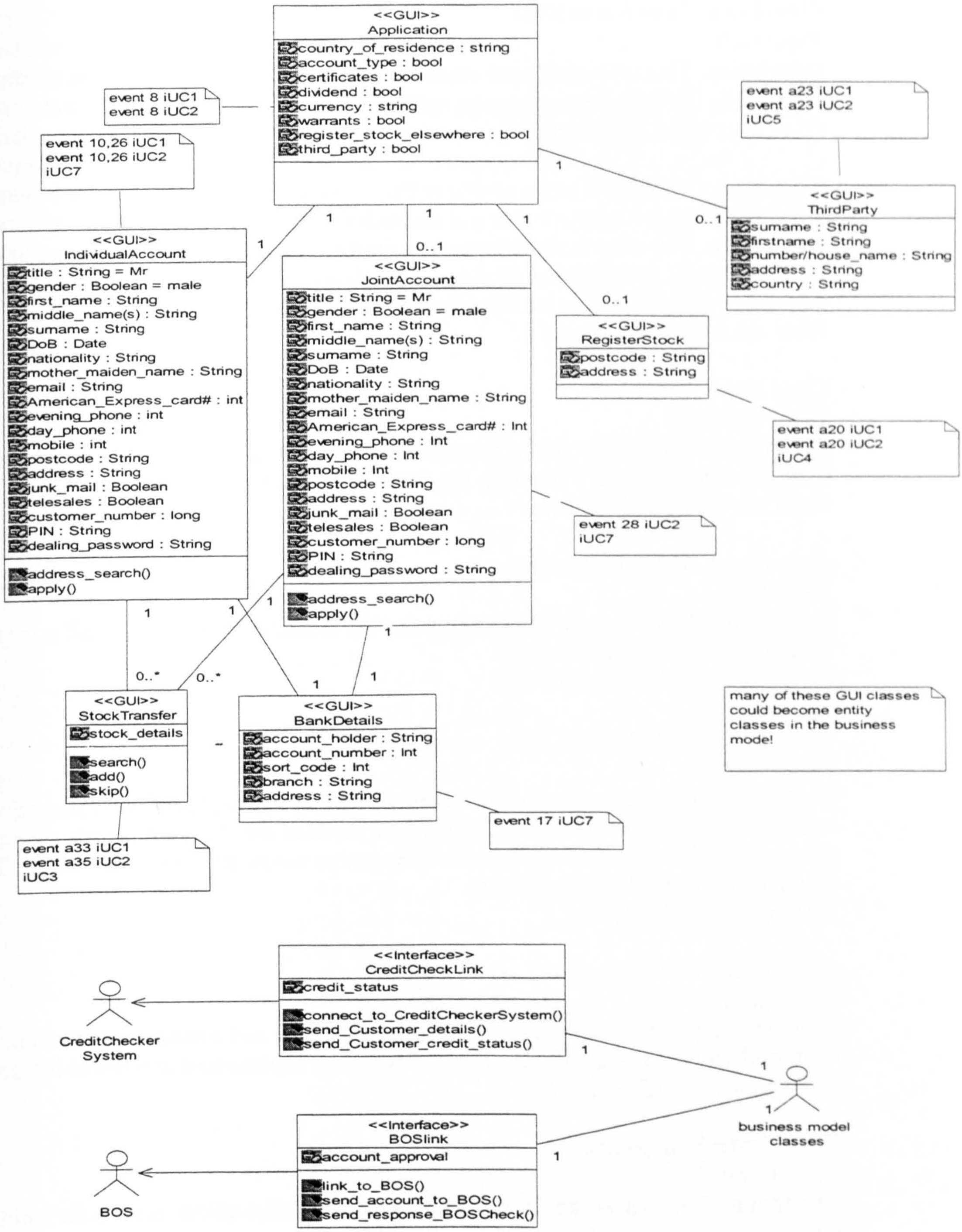
Class Name: Register Stock
Type: GUI
Description: Stores the postcode and address for stock registered at a different address.
From use case: iUC1 event a20; iUC2 event a20; iUC4

Class Name: Stock Transfer
Type: GUI
Description: This class has operations that allows the Customer to locate the stock they wish to transfer from another location to their Company X account.
From use case: iUC1 event a23; iUC2 event a35; iUC3

Class Name: Bank Details
Type: GUI
Description: Stores the Customer's bank account details (account in use by Customer, not the one created by Company X).
From use case: iUC7 event 17

Class Name: Credit Check Link
Type: Interface
Description: A class that contains the necessary protocols to link to the Credit Checker System, passing the Customer's bank details and receiving their credit worthiness in return. This class links the underlying business model and the Check Checker System.
From use case: bUC1 event 12; bUC2

Class Name: BOS Link
Type: Interface
Description: A class that contains the necessary protocols to link to the Bank of Scotland, passing the Customer's new account details and receiving their acceptance / rejection in return. This class links the underlying business model and the Bank of Scotland system.
From use case: bUC1 event 16; bUC3



Class Model for GUI Use Case Package: Apply to Sharepeople (showing which use cases these were identified) version 3 11/4/2001

Fig. 8.1.1 Class Diagram for Interface Package UC "Apply to Company X"

Class Description for Figure 8.1.2

Class Name: Print Room Staff
Type: GUI
Description: This is the graphical interface class that allows the Print Room Staff to retrieve the CSV file for mail merge to Word and also to get the audit trail.
From use case: iUC10 event 10

Class Name: Print Pack
Type: Database
Description: This class is stereotyped as a database class. It acts as a link between the Business Model and physical database. This class retrieves application pack data from the database and presents it to the Print Room Staff actor as a CSV file.
From use case: iUC10 event 2

Class Name: Word Link
Type: Interface
Description: This class contains the necessary protocols to connect to Microsoft Word and to all passage of the CSV file from the database to Word.
From use case: iUC10 event 5.

Class Name: Maintain Records
Type: GUI
Description: This is a graphical window that the Database Maintenance Staff access to update and edit Customer records.
From use case: iUC9 event 1 or 2

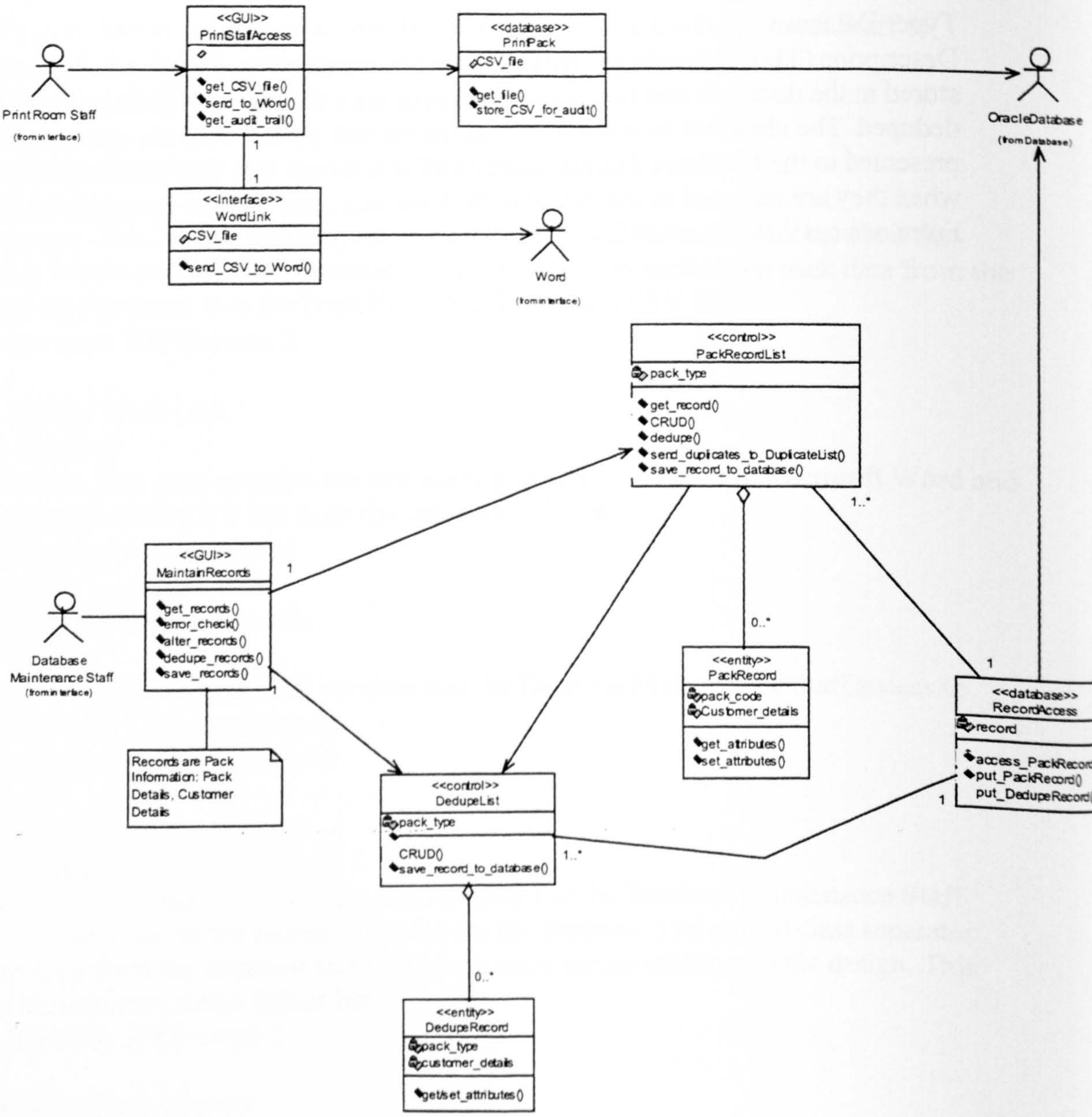
Class Name: Pack Record List
Type: Control
Description: This class executes the functionality that the Database Maintenance Staff wishes to carry out on the records pulled from the database. The control class separates the interface from the database and provides a more secure structure to the design. This would be implemented as linked list.
From use case: iUC9 event 2

Class Name: Pack Record
Type: Entity
Description: This is a store class that contains pack code and customer information. It is contained within the Pack Record List and would be implemented as a linked list node.
From use case: iUC9 event 2

Class Name: Dedupe List
Type: Control
Description: This is a linked list that keeps a record of all duplicate records for audit reasons.
From use case: iUC9 event 4

Class Name: Dedupe Record
Type: Entity
Description: This is the node in the linked list that keeps duplicate records when a dedupe take places.
From use case: iUC9 event 4

Class Name: Record Access
Type: Database
Description: This class allows the Database Maintenance Staff to access the records stored in the database and to return the records once they have been altered and / or deduped. The class has to make sure that the records pulled from the database are presented to the Database Maintenance Staff in a format that they can understand and that when they are returned to the database they are in a state that the database can understand.
From use case: iUC9 event 2.



Class Diagram for Use Case Package "Print Packs"
Version 3 26/4/2001

Figure 8.1.2 Class Diagram for Interface Package UC "Print Packs"

8.2 Business Layer Class Diagram

Figure 8.2.1 depicts the class diagram for the underlying business model. Class stereotypes defined here are <<entity>> and <<control>>. The <<entity>> class defines a store in memory that reflects the <<GUI>> classes at the interface. The <<control>> classes act as the processing go-betweens that keep the interface and the working memory apart. The <<control>> classes link to interface classes – defined as actors in the diagram. Some <<entity>> classes also link to database classes, defined as actors in the class diagram.

An <<instantiates>> stereotype is depicted between various control and entity classes. This means that the control classes are creating new objects so that they can store data within them.

The <<abstract>> class "Pack List" is the head of a linked list structure that allows for different PackType lists. Each list contains nodes ("Pack Code") that hold the information necessary to print a pack from an application. Each Pack Type list inherits the *dedupe()* operation that will traverse the appropriate list and delete any duplications. These will later be sent to the Oracle Database for storage.

Class Description for figure 8.2.1

Class Name: New Application
Type: Control
Description: This class starts the process of creating a new Customer Application by pulling the required Customer information from the GUI classes and storing them in entity classes that can be more securely manipulated.
From use case: bUC1 event 1.

Class Name: Customer Application
Type: Entity
Description: This is a store class that holds the Customer's account type information – it is instantiated by the New Application class.
From use case: bUC1 event 3.

Class Name: Customer Contact
Type: Control
Description: The Customer Contact retrieves the Customer's personal details from the GUI and places them in a Customer Details class.
From use case: bUC1 event 4.

Class Name: Customer Details
Type: Entity
Description: This is a store class that holds the Customer's personal contact details – it is instantiated by the Customer Contact class.
From use case: bUC1 event 5.

Class Name: Bank Process
Type: Control

Description: This class retrieves the Customer's bank details from the GUI class and passes the details to an entity class that it creates.
From use case: bUC1 event 6.

Class Name: Customer Bank Details
Type: Entity
Description: This is a store class that holds the Customer's bank details – it is instantiated by the Bank Process control class.
From use case: bUC1 event 7.

Class Name: Credit Check
Type: Control
Description: This class takes the Customer's Bank details and sends them towards the Credit Checker System via a Credit Check Link interface class.
From use case: bUC1 event 8, bUC2.

Class Name: Trading Account
Type: Entity
Description: This account class is created once the Customer's credit check status is returned and is acceptable. This class is created by the Credit Check class.
From use case: bUC1 event 9.

Class Name: BOS Check
Type: Control
Description: This control class takes the Trading Account details and informs the Bank of Scotland via an interface class BOS Link. If the BOS does not validate the Trading Account, what happens to its status?
From use case: bUC1 event 11, bUC3.

Class Name: Pack Coder
Type: Control
Description: This class takes the Customer Application details and decides the pack code that needs to be generated.
From use case: bUC1 event 12.

Class Name: Pack List
Type: Abstract
Description: This abstract superclass has all the operations that the linked lists of Pack Types will need. It has a dedupe operation that allows the Pack Type linked lists to pull out all duplicates of pack codes.
From use case: bUC1 event 13; bUC4.

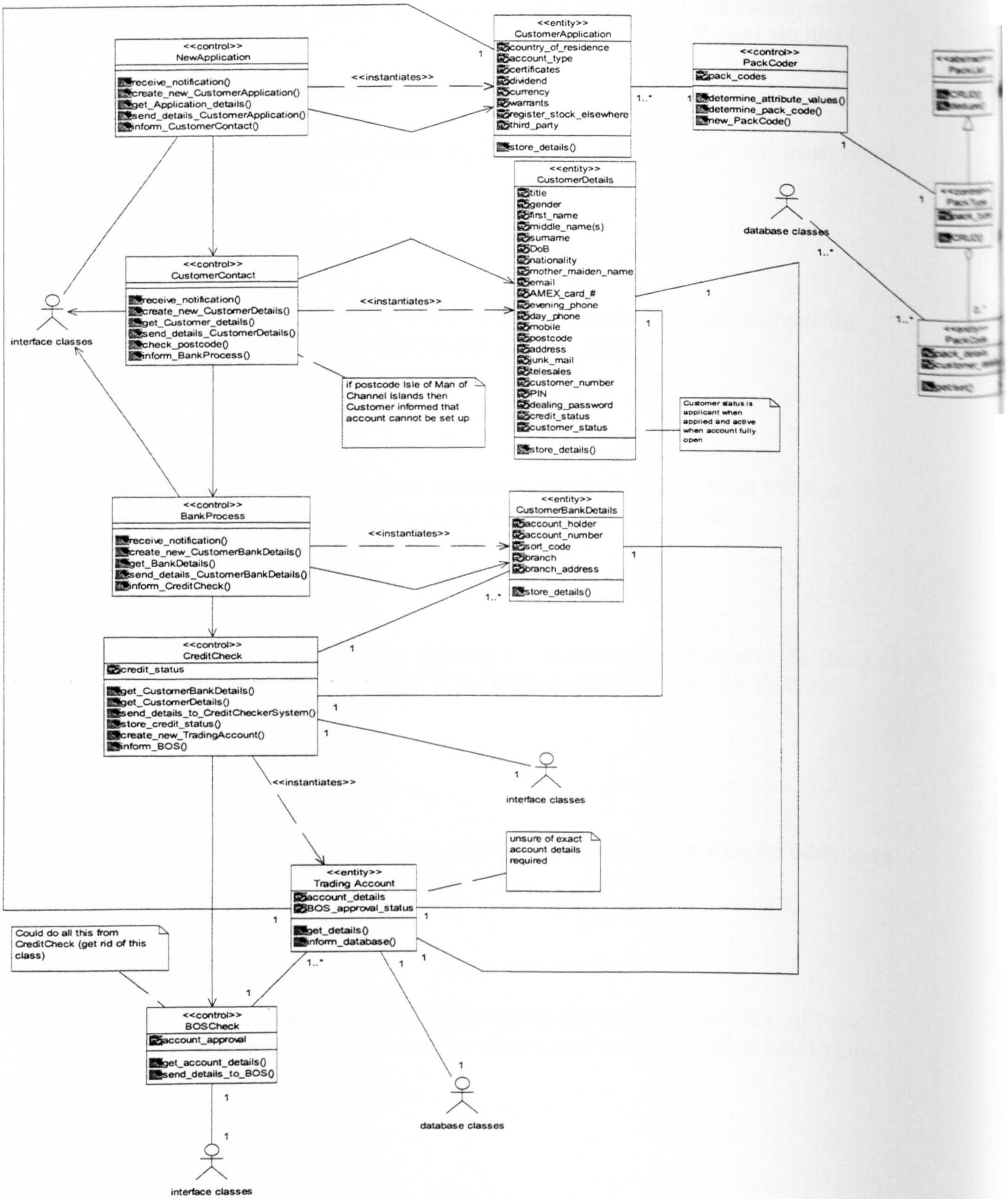
Class Name: Pack Type
Type: Control
Description: This is a linked list that inherits from the abstract Pack List. It stores the Pack Code nodes created by the Pack Coder class in a linked list.
From use case: bUC4.

Class Name: Pack Code
Type: Entity

Description: This is a node class that is created by the Pack Coder and stored in the appropriate Pack Type linked list.
From use case: bUC4.

TEXT BOUND INTO

THE SPINE



Class Diagram for Use Case Package "Underlying Business Model", describing the Determine Application Type use case v1 11/4/01

Fig. 8.2.1 Class diagram for business model package

8.3 Database Layer Class Diagram

The classes prepare the data to fit the database schema (and the opposite) and act as the link between the actual database and the working memory. Note that this is far from complete.

Class Description for figure 8.3.1

Class Name: New Pack Record
Type: Database
Description: This class organises pack details to and from the database.
From use case: dUC2 event 1.

Class Name: New Trading Record
Type: Database
Description: This class organises Trading Account details to and from the database.
From use case: dUC1 event 1.

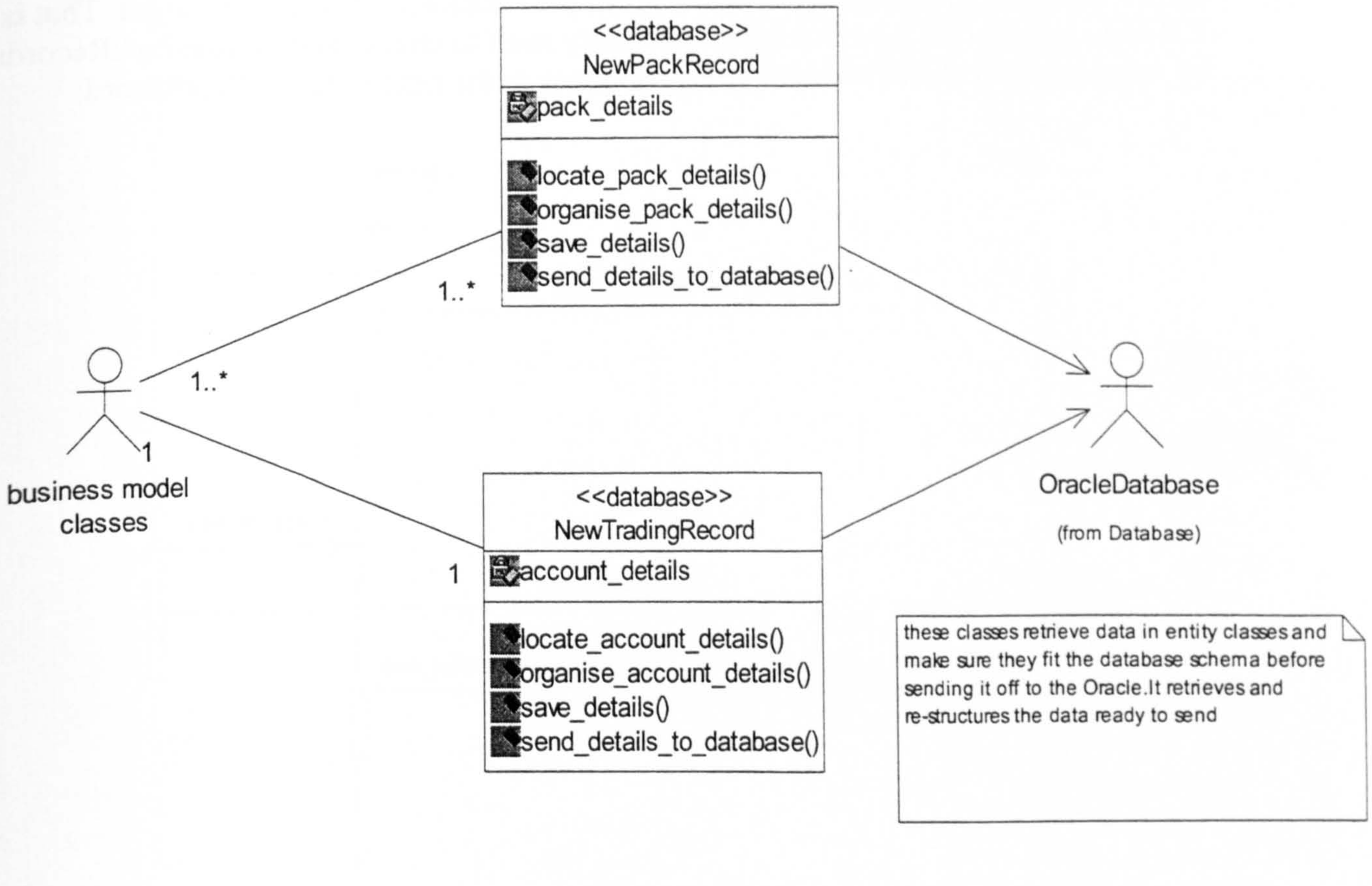


Figure 8.3.1 Class Diagram for Database Layer

9. Interaction Model

The interaction model depicts two interaction diagrams for the Class Diagram of Interface Package UC: Print Packs (fig. 8.1.2). The two interaction diagrams help validate the class diagram and determine the object interactions in terms of message order (fig 9.1) and object connection (fig 9.2) that are driven by the use case iUC9: Maintain Pack Codes.

Figure 9.1 Sequence Diagram for Maintain Records Class Diagram

Note that deduping can occur before altering records and is only an option because deduping takes place in the application procedure itself (see section 7.2). However, there might be some later changes made to records that cause a duplication to occur so this operation is added here.

Figure 9.2 Collaboration Diagram for Maintain Records Class Diagram

Note that the dedupe() operation links to self on the PackRecordList object (both figs 9.1 and 9.2). This is because the linked list of PackRecord objects is only an entity or store and all manipulation of this object occurs from the PackRecordList control object. That is, it is beginning an operation that works internally itself to check for duplicate PackRecords by retrieving each record and comparing its values to the next retrieved PackRecord.

9. Interaction Model

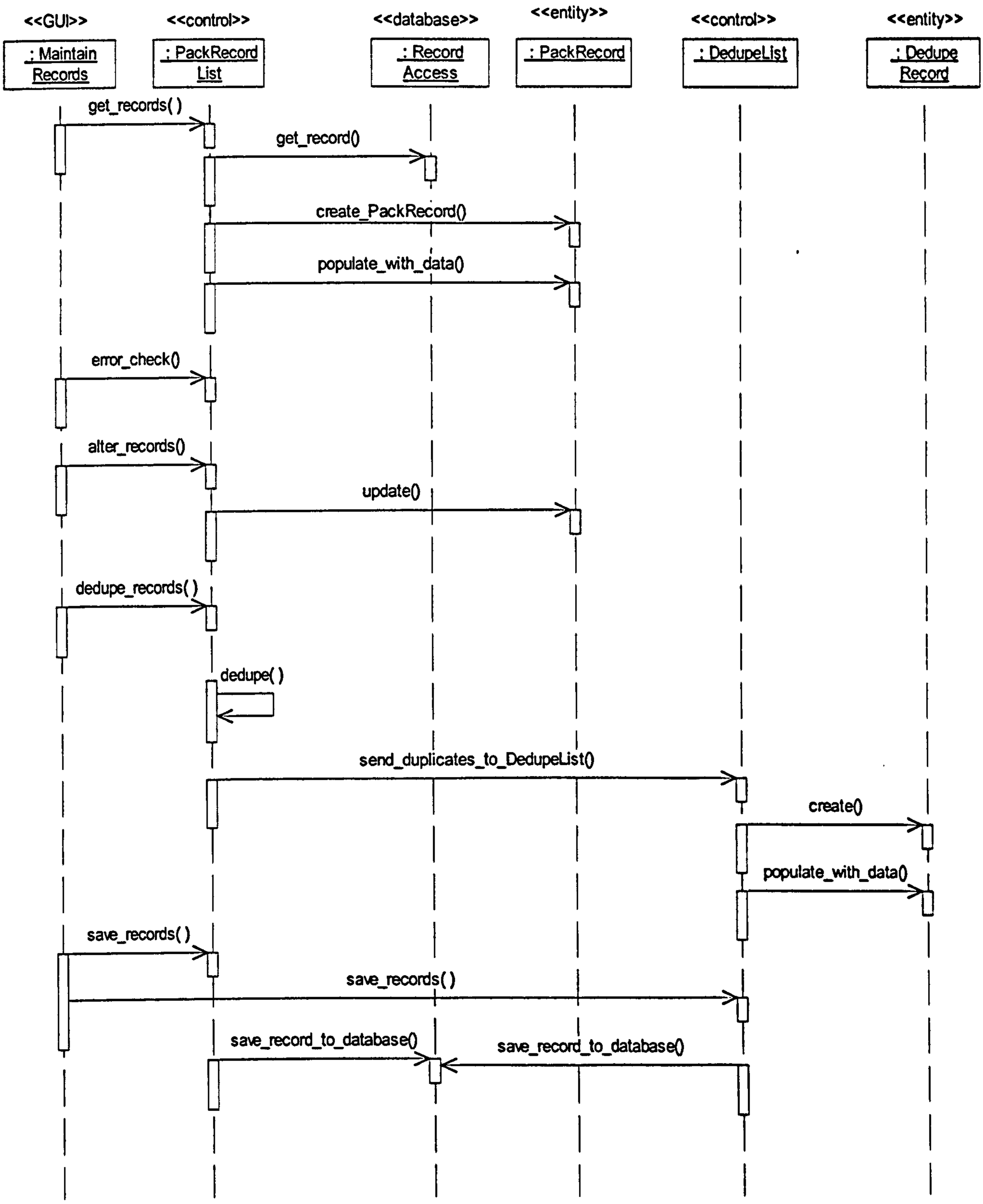
The interaction model depicts two interaction diagrams for the Class Diagram of Interface Package UC: Print Packs (fig. 8.1.2). The two interaction diagrams help validate the class diagram and determine the object interactions in terms of message order (fig 9.1) and object connection (fig 9.2) that are driven by the use case iUC9: Maintain Pack Codes.

Figure 9.1 Sequence Diagram for Maintain Records Class Diagram

Note that deduping can occur before altering records and is only an option because deduping takes place in the application procedure itself (see section 7.2). However, there might be some later changes made to records that cause a duplication to occur so this operation is added here.

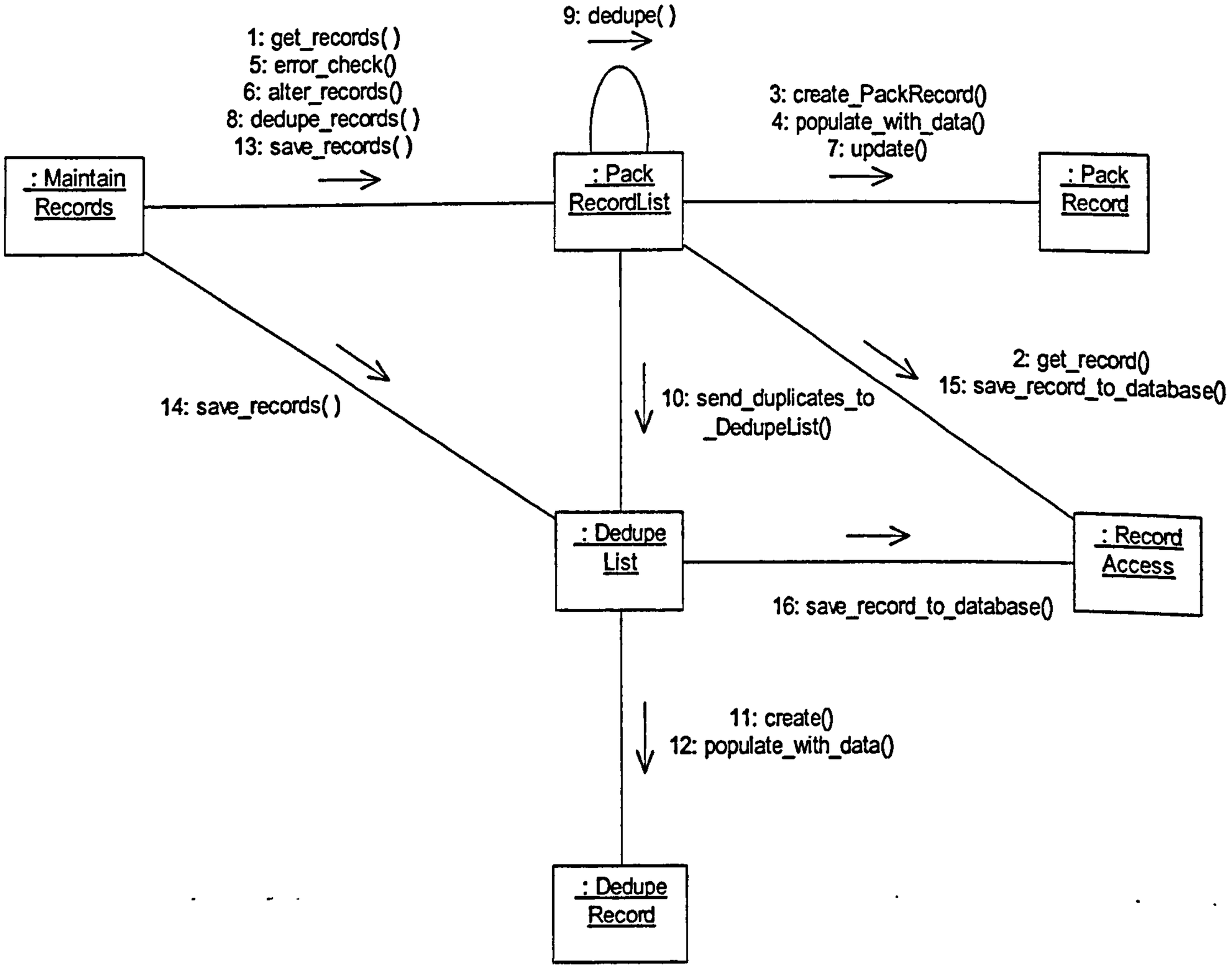
Figure 9.2 Collaboration Diagram for Maintain Records Class Diagram

Note that the dedupe() operation links to self on the PackRecordList object (both figs 9.1 and 9.2). This is because the linked list of PackRecord objects is only an entity or store and all manipulation of this object occurs from the PackRecordList control object. That is, it is beginning an operation that works internally itself to check for duplicate PackRecords by retrieving each record and comparing its values to the next retrieved PackRecord.



Sequence Diagram for Interface UC + Class Diagram: Maintain Records

Figure 9.2. Sequence Diagram for UC/Class Diagram: Maintain Records



Collaboration Diagram for Interface UC + Class Diagram: Maintain Records

Figure 9.2 Collaboration Diagram for UC/Class Diagram: Maintain Records

It would be wise to enact these interaction models to fully validate them.

10. Summary

The diagrams and descriptions in the document show how the current process works (figs 3.1, 3.2, 3.3, 4.1) and then how they might be remodeled (section 5 onwards).

This is preliminary work and there is still much to do. What we have so far is an overview and the task of figuring out the process for determining the pack codes and then printing them still needs resolving although the use cases bUC4: Generate Pack Code and dUC2: Store Application Packs class diagrams figs. 8.1.2, 8.2.1, 8.3.1, 9.1, 9.2 give an outline for this.

It is not this author's purpose to model the underlying database schema nor to restructure the database itself. The author's goal is model the process of applying for a trading account from the outside in. The diagrams and descriptions shown in this document are a starting point.

Questioning the use cases to identify classes is described in the following document.

F4. Use Case Question Sets (As presented to Company X)

1. Results of Questioning Interface Use Case Descriptions

Questioning iUC1:
Apply for an Individual Trading Account

Event 1
Pre-condition: Customer has Internet access. Connection OK. Customer knows the URL for Company X.
Post-condition: Customer has typed in URL and awaits a response.
Interface: Browser (typical)
System: interface level use case.
Actors: Customer.
Classes: Customer.

Event 2
Pre-condition: URL typed in.
Post-condition: Website visible to Customer
Interface: Company X Website
System: GUI objects (these could turn into boundary classes)
Actors: Customer, Front Office GUI (assume that Customer can see all components i.e. all web images, files etc, assume website is on first page).
Classes: Customer.

Event 3
Pre-condition: Customer has read screen and information and wants to apply.
Post-condition: Website takes Customer to the correct location.
Interface: Move to new form, GUI objects.
System: Navigates to next form.
Actors: Customer, Application form (?)
Classes: Customer, Application (?)

Event 4
Pre-condition: Secure screen message displayed (what are the security issues here?), Customer clicked Apply, apply screen available.
Post-condition: Apply screen visible and functioning.
Interface: Apply screen objects (boundary classes).
System: Interface.
Actors: Customer
Classes: Customer.

Event 5
Pre-condition: Customer at Application screen, it is visible.
Post-condition: Customer informed of how to apply.
Interface: text
System:
Actors: Customer
Classes: Customer

Event 6
Pre-condition: Customer wants to apply.
Post-condition: Customer aware of types of account and must select one.
Interface:
System:
Actors: Customer
Classes: Customer, Trading Account, ISA

Event 7
Pre-condition: Customer has decided to apply for Trading Account
Post-condition: System aware that Trading Account selected (web navigation purposes only)
Interface: mouse click on link, GUI button
System:
Actors: Customer
Classes: Customer, Trading Account

Event 8
Pre-condition: Website has received Trading Account selection.
Post-condition: Application Form page 1 visible.
Interface: New Screen (form)
System:
Actors: Customer
Classes: Customer, **Application form**

Event 9
Pre-condition: Customer can see Country of Residence, can select correct country
Post-condition: Country of Residence selected.
Interface: Drop down list box
System:
Actors: Customer
Classes: Customer

Event 10
Pre-condition: Customer understands Account types (Individual is the default selection)
Post-condition: Individual Trading Account ticked (as opposed to Joint)
Interface: check box
System:
Actors: Customer
Classes: Customer, Individual Trading Account is possible class later?

Event 11
Pre-condition: Customer can read text.
Post-condition: Text has been read.
Interface: Text
System:
Actors: Customer
Classes: Customer

Event 12

Pre-condition: Customer read and understood text
Post-condition: Option ticked
Interface: check box
System:
Actors: Customer
Classes: Customer

Event 13
Pre-condition: Customer can read text
Post-condition: Text has been read
Interface: Text
System:
Actors: Customer
Classes: Customer

Event 14
Pre-condition: Customer read and understood text
Post-condition: "Cash" ticked
Interface: check box
System:
Actors: Customer
Classes: Customer

Event 15
Pre-condition: Customer can read text
Post-condition: Text has been read
Interface: Text
System:
Actors: Customer
Classes: Customer

Event 16
Pre-condition: Customer read and understood text
Post-condition: Currency selected from Dropdown list
Interface: Dropdown list
System:
Actors: Customer
Classes: Customer

Event 17
Pre-condition: Customer can read text
Post-condition: Text has been read
Interface: Text
System:
Actors: Customer
Classes: Customer

Event 18
Pre-condition: Information read by Customer
Post-condition: No action taken (no state change)

Interface: Check box
System:
Actors: Customer
Classes: Customer

Event 19
Pre-condition: Customer can read text
Post-condition: Text has been read
Interface: Text
System:
Actors: Customer
Classes: Customer

Event 20
Pre-condition: Information read by Customer
Post-condition: No action taken (no state change)
Interface: Check box
System:
Actors: Customer
Classes: Customer

Event 21
Pre-condition: Customer can read text
Post-condition: Text has been read
Interface: Text
System:
Actors: Customer
Classes: Customer

Event 22
Pre-condition: Customer can read text
Post-condition: Text has been read
Interface: Text
System:
Actors: Customer
Classes: Customer

Event 23
Pre-condition: Information read by Customer
Post-condition: No action taken (no state change)
Interface: check box
System:
Actors: Customer
Classes: Customer

Event 24
Pre-condition: Customer has completed page 1 of Application form and is ready to continue.
Post-condition: Continue button clicked. Website moves Customer to next page of Application form (page 2)

Interface: button
System: checks that all necessary fields are complete, registers movement to next screen (would a different screen be visible in a joint account? How does the system determine account type? Is a Customer object (entity) already created in memory? Are Pack Codes already being defined?)
Actors: Customer
Classes: Customer

Event 25
Pre-condition: "Continue" selected, all in page 1 is OK
Post-condition: 2nd page of application form is visible to Customer
Interface: New form
System:
Actors: Customer
Classes: Customer, Application form

Event 26
Pre-condition: "Continue" selected, all in page 1 is OK
Post-condition: Customer sees the "About The Primary Holder" screen (assumption that Customer realizes that he/she is the primary holder)
Interface:
System:
Actors: Customer
Classes: Customer, **Individual Account (selected from event 10) – Primary Holder entity**

Event 27
Pre-condition: The Primary Holder screen is visible (go to iUC7: Complete Customer Details)
Post-condition: The Customer's details are completed
Interface: GUI
System: -
Actors: Customer
Classes: Customer

Event 28
Pre-condition: Customer has completed page 2 of application and is ready to continue
Post-condition: Continue button clicked
Interface: button
System: checks bank details are OK
Actors: Customer, Credit Checker?
Classes: Customer, Credit Checker?

Event 29
Pre-condition: Continue button clicked
Post-condition: Website displays completed page 2
Interface: form
System: retrieves Customer details from where?
Actors: Customer
Classes: Customer

Event 30
Pre-condition: Customer has completed page 2 of application and is ready to continue
Post-condition: Continue button clicked
Interface: button
System: checks bank details are OK
Actors: Customer, Credit Checker?
Classes: Customer, Credit Checker?

Event 31
Pre-condition: "Continue" has been selected; all on page 2 OK
Post-condition: 3rd page of application form is visible to Customer
Interface: new form
System:
Actors: Customer
Classes: Customer

Event 32
Pre-condition: Screen appears and page 2 completed OK
Post-condition: Customer read their details
Interface: text
System: Pulls details from page 2 – is this stored in memory (i.e. entity object) or pulled just from the interface
Actors: Customer
Classes: Customer, New Customer Application (?)

Event 33
Pre-condition: Customer accepts that details are OK
Post-condition: "Apply" selected (option "Back" to page 2, "Transfer Stock")
Interface: button
System: Saves details, where, writes to dB yet or just stored in memory for now?
Actors: Customer
Classes: Customer, New Customer Account(?)

Event 34
Pre-condition: Apply button clicked, all information viewed correct
Post-condition: Application screen appears
Interface: new form
System: ?
Actors: Customer
Classes: Customer, Application screen (boundary class)?

Event 35
Pre-condition: Page is visible. Submission successful
Post-condition: Customer can see page, has read it
Interface: text
System:
Actors: Customer
Classes: Customer

Event 36
Pre-condition: Customer number generated how? Does this mean that an account has been set up in the system? Stored on database or in run-time memory or both? Is a Pack Code generated here? Where is this stored?
Post-condition: Customer reads number
Interface: text
System: Number generated. Must be drawn from a list of Customer numbers to avoid repetition. Pack Code generated as well – is the Pack Code reflected in the Customer Number? Should it be? It needs to be on the Customer Record somewhere.
Actors: Customer, subsystem actor e.g. customer number list generator?
Classes: Customer (rename as Primary Holder? Entity / GUI) – stores application details of the Customer; Customer Number (database, control – to put Customer number to the GUI).

Event 37
Pre-condition: Customer sees PIN entry boxes
Post-condition: Customer entered PIN twice
Interface: edit boxes
System:
Actors: Customer
Classes: Customer

Event 38
Pre-condition: Customer presented with "Change Dealing Password" option.
Post-condition: Customer aware of what this is.
Interface: edit box
System:
Actors: Customer
Classes: Customer

Event 39
Pre-condition: Customer knows what "Change Dealing Password" means
Post-condition: (New) password entered
Interface: edit box, button
System: -
Actors: Customer
Classes: Customer

Event 40
Pre-condition: password entered
Post-condition: "change" button clicked
Interface: button
System: on click of button where does this information go?
Actor: Customer
Classes: Customer, entity class that links to database

Event 41
Pre-condition: Customer details accepted, needs checking and storing in system
Post-condition: Welcome screen appears with offers and information
Interface: new form

System: underlying functionality occurs here i.e. writing Customer details to database
Actors: Customer
Classes: Customer (actor), interface objects, Customer Account (?) entity, data passing from interface to entity (control class), entity interacting with database via database access classes

Alternative Flow 1

Event a20
Pre-condition: Information read by Customer
Post-condition: Stock to be registered at different address option selected.
Interface: check box
System: forces page 2 of the application to alter so address can be selected via Address Search.
Actors: Customer
Classes: Customer, CustomerApplicationDetails, StockRegistrationAddress, Address Search?

Alternative flow 2

Event a23
Pre-condition: Information read by Customer
Post-condition: Contract note to third party option selected.
Interface: check box
System: forces page 2 of the application to alter so address can be types in.
Actors: Customer
Classes: Customer, CustomerApplicationDetails, ThirdPartyAddress

Alternative flow 3

Event a33
Pre-condition: Customer decides details are not OK
Post-condition: Customer selects "Back"
Interface: button
System: Where are details retrieved from? Entity memory class?
Actors: Customer
Classes: Customer, Customer Account entity

Event a33.1
Pre-condition: Back button selected, Customer information has been retrieved
Post-condition: Page 2 displayed
Interface: GUI components
System: Where is information retrieved from?
Actors: Customer
Classes: Customer, Customer Account entity

Event a33.2
Pre-condition: Customer returned to page 2
Post-condition: Customer has altered details
Interface: GUI objects

System:
Actors: Customer
Classes: Customer

Event a33.3
Pre-condition: Customer has altered details
Post-condition: "Continue" button clicked (use case returns to event 49 in main flow)
Interface: button
System: Saves details back to Customer Account object or creates a new one?
Actors: Customer
Classes: Customer, Customer Account entity

Questioning iUC2:
Apply for a Joint Trading Account

Event 1
Pre-condition: Customer has Internet access. Connection OK. Customer knows the URL for Company X.
Post-condition: Customer has typed in URL and awaits a response.
Interface: Browser (typical)
System: interface level use case.
Actors: Customer.
Classes: Customer.

Event 2
Pre-condition: URL typed in.
Post-condition: Website visible to Customer
Interface: Company X Website
System: GUI objects (these could turn into boundary classes)
Actors: Customer, Front Office GUI (assume that Customer can see all components i.e. all web images, files etc, assume website is on first page).
Classes: Customer.

Event 3
Pre-condition: Customer has read screen and information and wants to apply.
Post-condition: Website takes Customer to the correct location.
Interface: Move to new form, GUI objects.
System: Navigates to next form.
Actors: Customer, Application form (?)
Classes: Customer, Application (?)

Event 4
Pre-condition: Secure screen message displayed (what are the security issues here?), Customer clicked Apply, apply screen available.
Post-condition: Apply screen visible and functioning.
Interface: Apply screen objects (boundary classes).
System: Interface.
Actors: Customer
Classes: Customer.

Event 5
Pre-condition: Customer at Application screen, it is visible.
Post-condition: Customer informed of how to apply.
Interface: text
System:
Actors: Customer
Classes: Customer

Event 6
Pre-condition: Customer wants to apply.
Post-condition: Customer aware of types of account and must select one.
Interface:
System:
Actors: Customer
Classes: Customer, Trading Account, ISA

Event 7
Pre-condition: Customer has decided to apply for Trading Account
Post-condition: System aware that Trading Account selected (web navigation purposes only)
Interface: mouse click on link, GUI button
System:
Actors: Customer
Classes: Customer, Trading Account

Event 8
Pre-condition: Website has received Trading Account selection.
Post-condition: Application Form page 1 visible.
Interface: New Screen (form)
System:
Actors: Customer
Classes: Customer, Application form

Event 9
Pre-condition: Customer can see Country of Residence, can select correct country
Post-condition: Country of Residence selected.
Interface: Drop down list box
System:
Actors: Customer
Classes: Customer

Event 10
Pre-condition: Customer understands Account types (Individual is the default selection)
Post-condition: Individual Trading Account ticked (as opposed to Joint)
Interface: check box
System:
Actors: Customer
Classes: Customer

Event 11
Pre-condition: Customer can read text.
Post-condition: Text has been read.
Interface: Text
System:
Actors: Customer
Classes: Customer

Event 12
Pre-condition: Customer read and understood text
Post-condition: Option ticked
Interface: check box
System:
Actors: Customer
Classes: Customer

Event 13
Pre-condition: Customer can read text
Post-condition: Text has been read
Interface: Text
System:
Actors: Customer
Classes: Customer

Event 14
Pre-condition: Customer read and understood text
Post-condition: "Cash" ticked
Interface: check box
System:
Actors: Customer
Classes: Customer

Event 15
Pre-condition: Customer can read text
Post-condition: Text has been read
Interface: Text
System:
Actors: Customer
Classes: Customer

Event 16
Pre-condition: Customer read and understood text
Post-condition: Currency selected from Dropdown list
Interface: Dropdown list
System:
Actors: Customer
Classes: Customer

Event 17
Pre-condition: Customer can read text

Post-condition: Text has been read
Interface: Text
System:
Actors: Customer
Classes: Customer

Event 18
Pre-condition: Information read by Customer
Post-condition: No action taken (no state change)
Interface: Check box
System:
Actors: Customer
Classes: Customer

Event 19
Pre-condition: Customer can read text
Post-condition: Text has been read
Interface: Text
System:
Actors: Customer
Classes: Customer

Event 20
Pre-condition: Information read by Customer
Post-condition: No action taken (no state change)
Interface: Check box
System:
Actors: Customer
Classes: Customer

Event 21
Pre-condition: Customer can read text
Post-condition: Text has been read
Interface: Text
System:
Actors: Customer
Classes: Customer

Event 22
Pre-condition: Customer can read text
Post-condition: Text has been read
Interface: Text
System:
Actors: Customer
Classes: Customer

Event 23
Pre-condition: Information read by Customer
Post-condition: No action taken (no state change)
Interface: check box

System:
Actors: Customer
Classes: Customer

Event 24
Pre-condition: Customer has completed page 1 of Application form and is ready to continue.
Post-condition: Continue button clicked. Website moves Customer to next page of Application form (page 2)
Interface: button
System: checks that all necessary fields are complete, registers movement to next screen (would a different screen be visible in a joint account? How does the system determine account type? Is a Customer object (entity) already created in memory? Are Pack Codes already being defined?)
Actors: Customer
Classes: Customer

Event 25
Pre-condition: "Continue" selected, all in page 1 is OK
Post-condition: 2nd page of application form is visible to Customer
Interface: New form
System:
Actors: Customer
Classes: Customer, Application form

Event 26
Pre-condition: "Continue" selected, all in page 1 is OK
Post-condition: Customer sees the "About The Primary Holder" screen (assumption that Customer realizes that he/she is the primary holder)
Interface:
System:
Actors: Customer
Classes: Primary Holder

Event 27
Pre-condition: The Primary Holder screen is visible (go to iUC7: Complete Customer Details)
Post-condition: The Customer's details are completed
Interface: GUI
System: -
Actors: Customer
Classes: Primary Holder

Event 28
Pre-condition: Customer has completed Primary Holder application.
Post-condition: Customer sees the Joint Holder application.
Interface:
System:
Actor: Customer
Class: **Joint Holder (GUI)**

Event 29
Pre-condition: The Primary Holder screen is visible (go to iUC7: Complete Customer Details)
Post-condition: The Customer's details are completed
Interface: GUI
System: -
Actors: Customer
Classes: Joint Holder

Event 30
Pre-condition: Customer has completed page 2 of application and is ready to continue
Post-condition: Continue button clicked
Interface: button
System: checks bank details are OK
Actors: Customer, Credit Checker?
Classes: Primary Holder, Joint Holder, Credit Checker?

Event 31
Pre-condition: Continue button clicked
Post-condition: Website displays completed page 2
Interface: form
System: retrieves Customer details from where?
Actors: Customer
Classes: Primary or Joint Holder

Event 32
Pre-condition: Customer has completed page 2 of application and is ready to continue
Post-condition: Continue button clicked
Interface: button
System: checks bank details are OK
Actors: Customer, Credit Checker?
Classes: Customer, Credit Checker?

Event 33
Pre-condition: "Continue" has been selected; all on page 2 OK
Post-condition: 3rd page of application form is visible to Customer
Interface: new form
System:
Actors: Customer
Classes: Customer

Event 34
Pre-condition: Screen appears and page 2 completed OK
Post-condition: Customer read their details
Interface: text
System: Pulls details from page 2 – is this stored in memory (i.e. entity object) or pulled just from the interface
Actors: Customer

Classes: Customer, New Customer Application (?)

Event 35
Pre-condition: Customer accepts that details are OK
Post-condition: "Apply" selected (option "Back" to page 2, "Transfer Stock")
Interface: button
System: Saves details, where, writes to dB yet or just stored in memory for now?
Actors: Customer
Classes: Customer, New Customer Account(?)

Event 36
Pre-condition: Apply button clicked, all information viewed correct
Post-condition: Application screen appears
Interface: new form
System: ?
Actors: Customer
Classes: Customer, Application screen (boundary class)?

Event 37
Pre-condition: Page is visible. Submission successful
Post-condition: Customer can see page, has read it
Interface: text
System:
Actors: Customer
Classes: Customer

Event 38
Pre-condition: Customer number generated how? Does this mean that an account has been set up in the system? Stored on database or in memory? Is a Pack Type generated here? Where is this stored?
Post-condition: Customer reads number
Interface: text
System: Number generated. Must be drawn from a list of Customer numbers to avoid repetition. Pack Type generated as well – is the Pack Type reflected in the Customer Number? Should it be? It needs to be on the Customer Record somewhere.
Actors: Customer, subsystem actor i.e. customer number list generator ?
Classes: Customer, Customer Number (entity, database). Control class involved somewhere to move number to screen.

Event 39
Pre-condition: Customer sees PIN entry boxes
Post-condition: Customer entered PIN twice
Interface: edit boxes
System:
Actors: Customer
Classes: Customer

Event 40
Pre-condition: Customer presented with "Change Dealing Password" option.
Post-condition: Customer aware of what this is.

Interface: edit box
System:
Actors: Customer
Classes: Customer

Event 41
Pre-condition: Customer knows what "Change Dealing Password" means
Post-condition: (New) password entered
Interface: edit box
System: -
Actors: Customer
Classes: Customer

Event 42
Pre-condition: password entered
Post-condition: "change" button clicked
Interface: button
System: on click of button where does this information go?
Actor: Customer
Classes: Customer, entity class that links to database

Event 43
Pre-condition: Customer details accepted, needs checking and storing in system
Post-condition: Welcome screen appears with offers and information
Interface: new form
System: underlying functionality occurs here i.e. writing Customer details to database
Actors: Customer
Classes: Customer (actor), interface objects, Customer Account (?) entity, data passing from interface to entity (control class), entity interacting with database via database access classes

Alternative Flow 1

Event a20
Pre-condition: Information read by Customer
Post-condition: Stock to be registered at different address option selected.
Interface: check box
System: forces page 2 of the application to alter so address can be selected via Address Search.
Actors: Customer
Classes: Customer, CustomerApplicationDetails, StockRegistrationAddress, Address Search?

Alternative flow 2

Event a23
Pre-condition: Information read by Customer
Post-condition: Contract note to third party option selected.
Interface: check box
System: forces page 2 of the application to alter so address can be types in.

Actors: Customer
Classes: Customer, CustomerApplicationDetails, ThirdPartyAddress

Alternative flow 3

Event a35
Pre-condition: Customer decides details are not OK
Post-condition: Customer selects "Back"
Interface: button
System: Where are details retrieved from? Entity memory class?
Actors: Customer
Classes: Primary Holder, Joint Holder Account entities

Event a35.1
Pre-condition: Back button selected, Customer information has been retrieved
Post-condition: Page 2 displayed
Interface: GUI components
System: Where is information retrieved from?
Actors: Customer
Classes: Primary Holder, Joint Holder Account entities

Event a35.2
Pre-condition: Customer returned to page 2
Post-condition: Customer has altered details
Interface: GUI objects
System:
Actors: Customer
Classes: Primary Holder, Joint Holder

Event a35.3
Pre-condition: Customer has altered details
Post-condition: "Continue" button clicked (use case returns to event 47 in main flow)
Interface: button
System: Saves details back to Customer Account object or creates a new one?
Actors: Customer
Classes: Primary Holder, Joint Holder Account entities

Questioning iUC3:
Transfer Stocks

Event 1
Pre-condition: Transfer selected from alternative flow a49.2 in UC1 or from a69.2 in UC2
Post-condition: Transfer Stock screen displayed
Interface: new form
System: This screen links to what? To CREST?
Actors: Customer, Stock System elsewhere?
Classes: Customer, Stock Transfer?

Event 2

Pre-condition: Transfer Stock screen displayed and Customer has entered Stock to search for
Post-condition: Stock located [assumed by author]
Interface: drop down lists and button
System: Links to Stock System somewhere
Actors: Customer, Stock System
Classes: Customer, Stock Transferer

Event 3
Pre-condition: Stock has been located
Post-condition: Stock is added to Customer portfolio
Interface: button
System: Transfers stock to Customer Account via what mechanisms?
Actors: Customer, Stock System
Classes: Customer, Stock Transferer (control?), Customer Account

Event 4
Pre-condition: Stock has been added to portfolio
Post-condition: Customer is returned to event 34 in main flow of iUC1 or event 36 of main flow in iUC2
Interface: return to Application form
System: ignores Stocks that have been located but not added ?
Actors: Customer, Stock System
Classes: Customer, Stock System?

Alternative flow
Pre-condition: At any time in the Stock Transfer screen
Post-condition: Customer is returned to event 34 in main flow of iUC1 or event 36 of main flow in iUC2
Interface: return to Application form
System: ignores Stocks that have been located but not added ?
Actors: Customer, Stock System
Classes: Customer, Stock System?

Questioning iUC4:
Register Stock at Different Address

Event 1
Pre-condition: Register Stock at Different Address has been pre-selected in page 1 of application form. Customer knows postcode of the different address.
Post-condition: Postcode entered.
Interface: edit box
System:
Actor: Customer
Classes: Primary Holder / Secondary Holder, RegisterStockAddress

Event 2
Pre-condition: Postcode has been entered
Post-condition: Address Search button clicked
Interface: button

System: search through postcode lists some how
Actors: Customer, Address Search (?)
Classes: Primary Holder / Secondary Holder, RegisterStockAddress, Address Search (?)

Event 3
Pre-condition: Website has returned a selection of streets
Post-condition: Customer's street (with correct number) is selected
Interface: Drop down list
System: Populates drop down list with streets taken from postcode locator (Address Search button) of some sort
Actors: Customer, Address Search ?
Classes: Primary Holder / Secondary Holder, RegisterStockAddress, Address Search (?)

Event 4
Pre-condition: Customer has located street from list
Post-condition: Confirm button clicked
Interface: button
System:
Actors: Customer
Classes: Primary Holder / Secondary Holder, RegisterStockAddress, Address Search (?)

Event 5
Pre-condition: Address confirmed
Post-condition: website displays Customer's address
Interface: text
System: takes address selected (events 4, 5) and sends to screen in text format
Actors: Customer, Address Search?
Classes: Primary Holder / Secondary Holder, RegisterStockAddress, Address Search (?)

Alternative flow

Event a1
Pre-condition: "Register Stock at Different address" same as Permanent Address
Post-condition: ignore this section and move on to next part of application
Interface:
System:
Actor: Customer
Classes: Primary Holder / Secondary Holder, RegisterStockAddress

Questioning iUC5:
Send Contract Notes to Third Party

Event 1
Pre-condition: The Customer has pre-selected to Send Contract Note to Third Party in page 1 of application. Customer knows name and address of third party.
Post-condition: Surname of third party entered.
Interface: edit box
System:
Actor: Customer
Classes: Primary Holder / Secondary Holder, Third Party

Event 2
Pre-condition: -
Post-condition: Third party first name entered
Interface: edit box
System:
Actor: Customer
Classes: Primary Holder / Secondary Holder, Third Party

Event 3
Pre-condition: -
Post-condition: Third party Postcode entered
Interface: edit box
System: enable edit box
Actor: Customer
Classes: Primary Holder / Secondary Holder, Third Party

Event 4
Pre-condition: -
Post-condition: Third party Number/House Name entered
Interface: edit box
System:
Actor: Customer
Classes: Primary Holder / Secondary Holder, Third Party

Event 5
Pre-condition: -
Post-condition: Third party address entered
Interface: edit box
System:
Actor: Customer
Classes: Primary Holder / Secondary Holder, Third Party

Event 6
Pre-condition: -
Post-condition: Third party Country selected
Interface: drop down list box
System:
Actor: Customer
Classes: Primary Holder / Secondary Holder, Third Party

Alternative flow

Event a1
Pre-condition: Third Party address same as Permanent Address
Post-condition: ignore this section and move on to next part of application
Interface:
System:
Actor: Customer
Classes: Primary Holder / Secondary Holder, Third Party

Questioning iUC6:
Register Customer

Event 1
Pre-condition: Customer has Internet access. Connection OK. Customer knows the URL for Company X.
Post-condition: Customer has typed in URL and awaits a response.
Interface: Browser (typical)
System: interface level use case.
Actors: Customer.
Classes: Customer.

Event 2
Pre-condition: URL typed in.
Post-condition: Website visible to Customer
Interface: Company X Website
System: GUI objects (these could turn into boundary classes)
Actors: Customer, Front Office GUI (assume that Customer can see all components i.e. all web images, files etc, assume website is on first page).
Classes: Customer.

Event 3
Pre-condition: Customer has read screen and information and wants to register.
Post-condition: Register option selected.
Interface: link.
System: Register link selected.
Actors: Customer
Classes: Customer

Event 4
Pre-condition: Register option selected.
Post-condition: Register screen visible.
Interface: register form
System: move to new form
Actor: Customer
Classes: Customer

Event 5
Pre-condition: Register screen visible
Post-condition: Text information read
Interface: text
System: -
Actor: Customer
Classes: Customer

Event 6
Pre-condition: Text read. Button enabled.
Post-condition: Button clicked.
Interface: Button

System: response to click
Actor: Customer
Classes: Customer

Event 7
Pre-condition: Agree button clicked
Post-condition: Registered User Application visible
Interface: new form
System: retrieve and display form
Actor: Customer
Classes: Customer, Register Application <<GUI>>

Event 8
Pre-condition: Registered User Application visible. Title list accessible.
Post-condition: Title selected
Interface: Drop down list box
System: -
Actor: Customer
Classes: Customer, Register Application <<GUI>>

Event 9
Pre-condition: Gender option visible
Post-condition: Gender selected
Interface: radio button
System: -
Actor: Customer
Classes: Customer, Register Application <<GUI>>

Event 10
Pre-condition: Name fields accessible
Post-condition: Names completed
Interface: edit fields
System: -
Actor: Customer
Classes: Customer, Register Application <<GUI>>

Event 11
Pre-condition: Date of Birth options accessible
Post-condition: DoB completed
Interface: 2 edit fields, 1 drop down list box (month)
System: -
Actor: Customer
Classes: Customer, Register Application <<GUI>>

Event 12
Pre-condition: Customer has email
Post-condition: email entered
Interface: edit field
System: -
Actor: Customer

Classes: Customer, Register Application <<GUI>>

Event 13
Pre-condition: Country of Residence list accessible
Post-condition: Country selected
Interface: drop down list box
System: -
Actor: Customer
Classes: Customer, Register Application <<GUI>>

Event 14
Pre-condition: Customer knows postcode
Post-condition: Postcode entered
Interface: edit field
System: -
Actor: Customer
Classes: Customer, Register Application <<GUI>>

Event 15
Pre-condition: Postcode entered
Post-condition: Address Search button clicked
Interface: button
System: signal to address searcher system?
Actor: Customer, Address Search System
Classes: Customer, Register Application <<GUI>>, Address Finder <<control>>, Address Retriever <<interface>>

Event 16
Pre-condition: Addresses have been located and returned from the Address Search System that match the postcode entered
Post-condition: Addresses displayed on screen
Interface: radio button list
System: sends postcode to Address Searcher System via interfacing objects and returns with listings of addresses. There must be enough memory space to retrieve corresponding addresses.
Actor: Customer, Address Searcher System
Classes: Customer, Register Application <<GUI>>, Address Finder <<control>>, Address Retriever <<interface>>, Address Storer <<entity>> - to store and load a number of addresses to the GUI.

Event 17
Pre-condition: List of addresses visible on screen. Correct one listed.
Post-condition: Correct address selected.
Interface: Radio button, text
System: -
Actor: Customer
Classes: Customer, Register Application <<GUI>>

Event 18
Pre-condition: Customer has read text

Post-condition: Option selected
Interface: check box
System: -
Actor: Customer
Classes: Customer, Register Application <<GUI>>

Event 19
Pre-condition: Customer has read text
Post-condition: Option selected
Interface: check box
System: -
Actor: Customer
Classes: Customer, Register Application <<GUI>>

Event 20
Pre-condition: Form completed.
Post-condition: Continue button clicked.
Interface: Button
System: responds to event click
Actor: customer
Classes: Customer, Register Application <<GUI>>,

Event 21
Pre-condition: Form completed correctly and submitted
Post-condition: Registration screen visible
Interface: form
System: -
Actor: Customer
Classes: Customer

Event 22
Pre-condition: Registration screen visible
Post-condition: Text read
Interface: text
System: -
Actor: Customer
Classes: Customer

Event 23
Pre-condition: Customer Number visible
Post-condition: Customer Number read
Interface: text
System: must generate a unique Customer Number from a list somewhere. This will probably come from the Registration form and is stored somewhere in the database
Actor: Customer
Classes: Customer, Registration objects <<control>> <<entity>> <<database>>

Event 24
Pre-condition: Customer read instructions
Post-condition: PIN entered

Interface: edit field
System: -
Actor: Customer
Classes: Customer

Event 25
Pre-condition: PIN entered
Post-condition: PIN re-entered
Interface: Edit field
System: -
Actor: Customer
Classes: Customer

Event 26
Pre-condition: Identical and correct PIN entered twice
Post-condition: Change button clicked
Interface: Button
System: PIN sent to Registrant record in database
Actor: Customer
Classes: Customer, Registration objects <<control>> <<entity>> <<database>>

Event 27
Pre-condition: Change clicked – all details correct
Post-condition: Welcome to Company X screen displayed
Interface: new form
System: responds to button click
Actor: Customer
Classes: Customer

Alternative

Event a6
Pre-condition: Customer read information
Post-condition: Back button clicked
Interface: Button
System: responds to event selected
Actor: Customer
Classes: Customer

Event a6.1
Pre-condition: Back button clicked
Post-condition: Home page of website is displayed
Interface: form
System: responds to event selected
Actor: Customer
Classes: Customer

Questioning iUC7:
Complete Customer Details

Event 1
Pre-condition: Customer read instructions and is ready to complete field and can find title from the list
Post-condition: Title has been selected from long list of options
Interface: Drop down list box
System:
Actors: Customer
Classes: Customer

Event 2
Pre-condition: Customer understands meaning, can see it
Post-condition: Gender selected
Interface: radio button
System:
Actors: Customer
Classes: Customer

Event 3
Pre-condition: Customer ready to type names
Post-condition: Customer typed in names
Interface: text, edit boxes
System:
Actors: Customer
Classes: Customer

Event 4
Pre-condition: Customer understands fields
Post-condition: DoB entered
Interface: edit field (day, year), drop down list (month)
System:
Actors: Customer
Classes: Customer

Event 5
Pre-condition: Customer can find nationality (note there is no option for British, but United Kingdom)
Post-condition: Nationality entered
Interface: drop down list box
System:
Actors: Customer
Classes: Customer

Event 6
Pre-condition: Customer knows mother's maiden name
Post-condition: Mother's maiden name entered
Interface: edit box
System:
Actors: Customer
Classes: Customer

Event 7
Pre-condition: Customer sees option to Enter Credit Card number (but doesn't have one)
Post-condition: No action taken (no state change)
Interface: edit box
System:
Actors: Customer
Classes: Customer

Event 8
Pre-condition: Customer sees text
Post-condition: Text read
Interface: text
System:
Actors: Customer
Classes: Customer

Event 9
Pre-condition: Customer has an email address
Post-condition: Email address entered
Interface: edit box
System:
Actors: Customer
Classes: Customer

Event 10
Pre-condition: Customer has (access to non-public) telephone
Post-condition: Telephone number entered (evening, day, mobile option – only 1 required)
Interface: 2 edit boxes per number: STD code and local number
System:
Actors: Customer
Classes: Customer

Event 11
Pre-condition: Customer reads residential information
Post-condition: Postcode entered
Interface: text, edit box
System:
Actors: Customer
Classes: Customer

Event 12
Pre-condition: Customer remembers moving to current address!
Post-condition: years at current residence entered; month, year selected
Interface: edit box; drop down list boxes
System: constraint: must be resident for 3 years – what if not?
Actors: Customer
Classes: Customer

Event 13
Pre-condition: Postcode has been entered
Post-condition: Address Search button clicked
Interface: button
System: search through postcode lists some how
Actors: Customer, Address Search (?)
Classes: Customer, Address Search (?)

Event 14
Pre-condition: Website has returned a selection of streets
Post-condition: Customer's street (with correct number) is selected
Interface: Drop down list
System: Populates drop down list with streets taken from postcode locator (Address Search button) of some sort
Actors: Customer, Address Search ?
Classes: Customer, Address Search

Event 15
Pre-condition: Customer has located street from list
Post-condition: Confirm button clicked
Interface: button
System:
Actors: Customer
Classes: Customer

Event 16
Pre-condition: Address confirmed
Post-condition: website displays Customer's address
Interface: text
System: takes address selected (events 40, 41) and sends to screen in text format
Actors: Customer, Address Search?
Classes: Customer, Address Search?

Event 17
Pre-condition: Customer has bank details available
Post-condition: Details entered
Interface: text, edit fields
System:
Actors: Customer, Credit Checker?
Classes: Customer, Credit Checker (control), BankAccount (GUI & entity)?

Event 18
Pre-condition: Customer has read text and understood it
Post-condition: Box ticked
Interface: check box
System:
Actors: Customer
Classes: Customer

Event 19

Pre-condition: Customer has read text and understood it
Post-condition: Box ticked
Interface: check box
System:
Actors: Customer
Classes: Customer

Exceptional Flow 1

Event e14
Pre-condition: Customer has clicked on Address Search after entering their postcode
Post-condition: Correct number on street not located
Interface: drop down box
System: links to Address Search subsystem or procedure or system
Actors: Customer, Address Search
Classes: Customer, Address Search?

Event e14.1
Pre-condition: Customer has failed to locate number and street
Post-condition: Postcode retyped
Interface: drop down list box
System:
Actors: Customer
Classes: Customer

Event e14.2
Pre-condition: Postcode entered
Post-condition: Address Search clicked
Interface: button
System: links to Address Search application or subsystem or system etc
Actors: Customer, Address Search
Classes: Customer, Address Search?

Event e14.3
Pre-condition: Correct address appears in list
Post-condition: event 40 in main flow
Interface: drop down list box
System: links to Address Search
Actors: Customer, Address Search
Classes: Customer, Address Search?

Questioning iUC8:
Apply for an ISA/PEP

Event 1
Pre-condition: Customer has Internet access. Connection OK. Customer knows the URL for Company X.
Post-condition: Customer has typed in URL and awaits a response.

Interface: Browser (typical)
System: interface level use case.
Actors: Customer.
Classes: Customer.

Event 2
Pre-condition: URL typed in.
Post-condition: Website visible to Customer
Interface: Company X Website
System: GUI objects (these could turn into boundary classes)
Actors: Customer, Front Office GUI (assume that Customer can see all components i.e. all web images, files etc, assume website is on first page).
Classes: Customer.

Event 3
Pre-condition: Customer has read screen and information and wants to apply.
Post-condition: Website takes Customer to the correct location.
Interface: Move to new form, GUI objects.
System: Navigates to next form.
Actors: Customer, Application form (?)
Classes: Customer, Application (?)

Event 4
Pre-condition: Secure screen message displayed (what are the security issues here?), Customer clicked Apply, apply screen available.
Post-condition: Apply screen visible and functioning.
Interface: Apply screen objects (boundary classes).
System: Interface.
Actors: Customer
Classes: Customer.

Event 5
Pre-condition: Customer at Application screen, it is visible.
Post-condition: Customer informed of how to apply.
Interface: text
System:
Actors: Customer
Classes: Customer

Event 6
Pre-condition: Customer wants to apply.
Post-condition: Customer aware of types of account and must select one.
Interface:
System:
Actors: Customer
Classes: Customer, Trading Account, ISA (Mini & Maxi), PEP

Event 7
Pre-condition: Customer has decided to apply for a Maxi ISA
Post-condition: System aware that Maxi ISA selected (web navigation purposes only)

Interface: mouse click on link, GUI button
System:
Actors: Customer
Classes: Customer, Maxi ISA Account

Event 8
Pre-condition: Website has received Maxi ISA selection.
Post-condition: Application Form page 1 visible.
Interface: New Screen (form)
System:
Actors: Customer
Classes: Customer, **Application form**

Event 9
Pre-condition: Maxi ISA selected
Post-condition: Log in option screen visible
Interface: GUI objects: text, edit boxes, buttons
Actor: Customer
System: -
Classes: Customer

Event 10
Pre-condition: Options visible
Post-condition: Continue selected
Interface: Button etc
Actor: Customer
System: -
Classes: New Customer Account objects (unlikely to be created here but on application submittance.

Event 11
Pre-condition: Continue selected
Post-condition: Page 2 of application visible
Interface:
Actor: Customer
System:
Classes: Customer

Event 12
Pre-condition: Details visible
Post-condition: -
Interface: text
Actor: Customer
System: -
Classes: Customer

Event 13
Pre-condition: Details read
Post-condition: Button clicked
Interface: button

Actor: Customer
System: -
Classes: Customer

Event 14
Pre-condition: Accept button clicked
Post-condition: 3rd page of application visible
Interface: -
Actor: Customer
System: -
Classes: Customer

Event 15
Pre-condition: option visible
Post-condition: option selected
Interface: combo box
Actor: Customer
System: -
Classes: Customer

Event 16
Pre-condition: Option selected (default selection if not chosen by Customer)
Post-condition: Button clicked
Interface: Button
Actor: Customer
System: -
Classes: Customer

Event 17
Pre-condition: Continue clicked
Post-condition: Page 4 of application visible
Interface:
Actor: Customer
System: -
Classes: Customer

Event 18
Pre-condition: Application options visible (iUC1: Apply for an Individual Trading Account events 9-24).
Post-condition: Application options completed
Interface: various GUI objects
Actor: Customer
System: -
Classes: Customer

Event 19
Pre-condition: The Primary Holder screen is visible (go to iUC7: Complete Customer Details)
Post-condition: The Customer's details are completed
Interface: GUI

System: -
Actors: Customer
Classes: Customer

Event 20
Pre-condition: Page 5 of application completed.
Post-condition: "Continue" button clicked.
Interface: button
Actor: Customer
System: -
Classes: Customer

Event 21
Pre-condition: Button clicked
Post-condition: Page 6 of application visible
Interface: -
Actor: Customer
System: -
Classes: Customer

Event 22
Pre-condition: Page visible
Post-condition:
Interface:
Actor: Customer
System:
Classes: Customer

Event 23
Pre-condition: Option visible
Post-condition: Option selected
Interface: check box
Actor: Customer
System: -
Classes: Customer

Event 24
Pre-condition: Option visible
Post-condition: Option selected
Interface: check box
Actor: Customer
System: -
Classes: Customer

Event 25
Pre-condition: Option visible
Post-condition: Option selected
Interface: check box
Actor: Customer
System: -

Classes: Customer

Event 26
Pre-condition: Option visible
Post-condition: Option selected
Interface: check box
Actor: Customer
System: -
Classes: Customer

Event 27
Pre-condition: Continue visible
Post-condition: Continue selected
Interface: Button
Actor: Customer
System: -
Classes: Customer

Event 28
Pre-condition: Continue selected
Post-condition: Page 7 of application visible
Interface: -
Actor: Customer
System: -
Classes: Customer

Event 29
Pre-condition: Page visible; apply option visible
Post-condition: Apply selected
Interface: button
Actor: Customer
System: Generates new Maxi ISA account
Classes: Customer, Maxi ISA account

Event 30
Pre-condition: Apply button selected (include iUC1 events 34-41)
Post-condition: Final application completed; Customer presented with Welcome information.
Interface:
Actor: Customer
System: Generates Customer number; sets up an account
Classes: Customer, Maxi ISA Account

Alternative flows:

Event a7
Pre-condition: Customer has decided to apply for a Mini ISA
Post-condition: System aware that Mini ISA selected (web navigation purposes only)
Interface: mouse click on link, GUI button
System:

Actors: Customer
Classes: Customer, Mini ISA Account

(all subsequent events should refer to Mini ISA only)

Event a7
Pre-condition: Customer has decided to apply to Transfer ISA
Post-condition: System aware that Transfer ISA selected (web navigation purposes only)
Interface: mouse click on link, GUI button
System:
Actors: Customer
Classes: Customer, Transfer ISA Account

(all subsequent events should refer to Transfer ISA only)

Event a7
Pre-condition: Customer has decided to apply to Transfer PEP
Post-condition: System aware that Transfer PEP selected (web navigation purposes only)
Interface: mouse click on link, GUI button
System:
Actors: Customer
Classes: Customer, Transfer PEP Account

(all subsequent events should refer to Transfer PEP only)

Event a10
Pre-condition: Options visible
Post-condition: Login in as a current user selected (TBD use case)
Interface: Button etc
Actor: Customer
System: -
Classes: Customer

Event a13
Pre-condition: Details read
Post-condition: Button clicked
Interface: button
Actor: Customer
System: -
Classes: Customer

Event a13.1
Pre-condition: Reject button clicked
Post-condition: Home Page visible
Interface: button
Actor: Customer
System: -
Classes: Customer

Event a27

Pre-condition: Back visible
Post-condition: Back selected
Interface: Button
Actor: Customer
System: -
Classes: Customer

Event a27.1
Pre-condition: Back button clicked
Post-condition: Page 5 of application visible
Interface: -
Actor: Customer
System: -
Classes: Customer

Event a27.2
Pre-condition: Page 5 of application visible
Post-condition: Page 5 of application visible
Interface: -
Actor: Customer
System: -
Classes: Customer

Event a27.3
Pre-condition: Continue visible
Post-condition: Continue selected
Interface: Button
Actor: Customer
System: -
Classes: Customer

Event a27.4
Pre-condition: Continue selected
Post-condition: Page 6 of application visible
Interface: Button
Actor: Customer
System: -
Classes: Customer

Event a29
Pre-condition: Back visible
Post-condition: Back selected (website fails here)
Interface: Button
Actor: Customer
System: -
Classes: Customer

Questioning iUC9:
Maintain Pack Codes

Event 1
Pre-condition: Connection to Front Office OK
Post-condition: Connected to Front Office
Interface: GUI (Maintenance Screen)
System: Allows connection
Actor: Database Maintainer
Class: Maintenance screen <<GUI>> class?

Event 2
Pre-condition: Oracle Database accessible
Post-condition: Access given
Interface: GUI (Maintenance Screen)
System: Classes must link to allow access.
Actor: Database Maintainer
Class: dB & control classes but access is via interface: maintenance GUI class? Linked list to store records in?

Event 3
Pre-condition: Records accessed and are in an alterable state
Post-condition: Records changed
Interface: GUI
System: provides access route
Actor: Database Maintainer
Class: database classes with operations of read etc; records are held as nodes in a linked list e.g. Record node in a Customer Record List.

Event 4
Pre-condition: Records accessed
Post-condition: Records deduped
Interface: GUI
System: Process to dedupe. Does the system record all deduped records? Yes, but are these stored directly to the database or are they put into linked lists?
Actor: Database Maintainer
Class: linked lists to store duplicate records for the audit trail.

Event 5
Pre-condition: Copies of dB records have been accessed
Post-condition: Records returned to database and tables updated
Interface: GUI
System: Updates records in the dB
Actor: Database Maintainer
Class: control / driver dB classes with update functions?

Questioning iUC10:
Print Packs

Event 1
Pre-conditions: Connection to Front Office OK
Post-conditions: Connected to Front Office.

Interface: Through an interface or a GUI – visible to the Print Room Staff (Print Control Screen).
System: Print Room Staff must have access to directory. Do they require security access?
Actors: Print Room Staff.
Classes: GUI objects

Event 2
Pre-conditions: Connected to Front Office; Oracle accessible
Post-conditions: CSV file pulled from Oracle and placed in directory visible to Print Room Staff.
Interface: GUI
System: Goes to Oracle Database and makes it generate CSV file. The file is then placed in a position visible to the Print Room Staff – stored in a database class?
Actor: Print Room Staff
Classes: GUI, Print Pack/CSV <<database>> - more than 1 database class?

Event 3
Pre-conditions: Front Office connection made.
Post-conditions: Directory located and files are accessible.
Interface: -
System: Access to files allowed (permissions). Is the CSV automatically generated by the Front Office in its own time or do the Print Room Staff tell the Front Office to generate the CSV file when they want to print?
Actors: Print Room Staff
Classes: link from GUI to database

Event 4
Pre-conditions: Word is available
Post-conditions: Word booted up
Interface: -
System: -
Actors: Print Room staff, Word
Classes: -

Event 5
Pre-conditions: Word open, Front Office accessible
Post-conditions: CSV file in Word
Interface: -
System: Permits file transfer to Word.
Actors: Print Room Staff, Word
Classes: File Transfer Protocol – Word interface class that contains these protocols.

Event 6
Pre-conditions: Word has CSV file
Post-conditions: Mail merge underway
Interface: -
System:
Actors: Print Room Staff, Word
Classes: -

Event 7
Pre-conditions: Printer needs organizing (not enough forms to cover print run)
Post-conditions: Forms put in place, printer drawers closed
Interface: -
System: -
Actors: Print Room Staff, Printer
Classes: -

Event 8
Pre-conditions: Mail merge started
Post-conditions: Mail merge complete
Interface: -
System: Word generates a merged file
Actors: Word
Classes: -

Event 9
Pre-conditions: Mail merge complete
Post-conditions: Print selected
Interface: Word interfaces with the Printer automatically. However, Word's printer driver might need to be manipulated to allow automated printer selection and collation of printed materials.
System: Batch print job (Word file) and send to Printer. The Front Office needs a record of what has been printed for its audit trail. This means that records must be stored in the database for all pack types and jobs that are sent for printing. Where does this audit record come from? Is it the CSV file that is transferred to Word? Is it the batched print job that is saved? It might be better to just have a text file in the database rather than creating audit objects. I suspect that the CSV file is the place to start.
Actors: Print Room Staff, Word
Classes: <<database>> Print Pack should contain a function to save CSV for audit; CSV file as an attribute?

Event 10
Pre-conditions: Printer received job; job translated into Printer speak
Post-conditions: Material printed
Interface: -
System: -
Actors: Printer
Classes: -

2. Results of Questioning Business Model Use Case Descriptions

Questioning bUC1:
Determine Type of Application

Event 1
Pre-condition: Customer clicks "apply". Message sent to control object that new application is ready for processing.
Post-condition: Control object informed of new application.
Interface: Button clicked from GUI
System: links objects from GUI to control object
Actors: GUI (Customer)
Classes: Application <<GUI>> object (apply() operation); NewApplication <<control>>

Event 2
Pre-condition: Business control object informed of new application.
Post-condition: Business control object retrieves new application details from GUI objects.
Interface: -
System: "Get" operation linking business model to interface
Actors: GUI
Classes: Application <<GUI>> object (apply() operation); NewApplication <<control>>

Event 3
Pre-condition: Business control has retrieved new application details. Control object instantiates new entity object.
Post-condition: New application details stored in entity object.
Interface: -
System: Links GUI object data to business entity object.
Actors: -
Classes: NewApplication <<control>> and Customer Application <<entity>>

Event 4
Pre-condition: Business control object informed that new application details have been stored. Triggers the control object to retrieve customer contact details.
Post-condition: Business control object retrieves customer contact details from GUI objects.
Interface: -
System: "Get" operation linking business model to interface.
Actors: GUI
Classes:Individual/Joint Account <<GUI>>, CustomerContact <<control>>

Event 5
Pre-condition: Business object has retrieved customer contact details. Control object instantiates new entity object.
Post-condition: Customer Contact details stored in new entity object.
Interface: -
System: Links GUI object data to business entity object.
Actors: -
Classes: Customer Contact <<control>> to Customer Details <<entity>>

Event 6
Pre-condition: Business control object has stored Customer Application
Post-condition: Business control object has connected to GUI object and retrieved the Customer Bank Details.
Interface: -
System: "Get" operation linking business model control object to interface object
Actors: GUI
Classes: BankProcess <<control>> object and BankDetails<<GUI>>

Event 7
Pre-condition: Business control object has retrieved Customer Bank Details. Business control object creartes new bank details entity object.
Post-condition: Bank Details stored in entity object.
Interface: -
System: Links GUI object data to business entity object.
Actors: -
Classes: BankProcess <<control>> and Customer Bank Details <<entity>>

Event 8
Pre-condition: Customer Bank Details have been stored in memory <<entity>> object.
Post-condition: Credit status has been returned.
Interface:
System: process that links the Bank entity object to the Credit Checker system actor
Actors: Credit Checker System
Classes: see bUC2

Event 9
Pre-condition: Credit status is fine.
Post-condition: New account is generated.
Interface: -
System: New Customer Account generated in memory.
Actors: -
Classes: Credit Check object; new entity Customer Account object, populated by various bits of Customer details, bank details, application details

????Does event 10 go here??? – forces an include in the UC diagram
Event 10
Pre-condition: New Account created.
Post-condition: Database has stored new account
Interface: -
System: New Account details sent to database
Actors: database
Classes: NewAccount <<entity>>, DatabaseLinker<<database>>

Event 11
Pre-condition: New Account created. BOS connection working.
Post-condition: Account acceptance reply from BOS.
Interface: -
System: objects that interface to the BOS (see bUC3)

Actors: BOS
Classes: NewAccount <<entity>>, BOS<<control>>, BOSlink<<interface>>

Event 12
Pre-condition: Application details still in memory
Post-condition: ApplicationDetails <<entity>> located and details taken.
Interface: -
System: -
Actors: -
Classes: ApplicationDetails <<entity>>, PackTypeProcess <<control>> ??any other entity objects contacted??

Event 13
Pre-condition: Application details retrieved
Post-condition: Pack Code generated.
Interface: -
System: see bUC4
Actors: -
Classes: ApplicationDetails <<entity>>, PackTypeProcess <<control>>, other entity objects?

Event 14
Pre-condition: Pack Code generated
Post-condition: Pack code stored in database.
Interface: Link class(es) to Oracle Database.
System: Must check that pack code and details sent to correct file in the database or added to correct list Pack List and Pack Type
Actors: Database
Classes: PackCoder <<control>>, Pack Code <<entity>> ?? (Or is the Pack Code a list of attributes drawn from objects that at run-time get bundled into an object of its own?), Pack Code <<entity>> - a linked list that would deal with duplication etc. at run-time before storing in the database.

Exceptional Flow
Event e5
Pre-condition: Customer contact details have been retrieved from the GUI. Postcode address is Isle of Man / Channel Islands.
Post-condition: Application rejected because postcode failed check.
Interface: ?
System: Process runs in control object that checks for Isle of Man or Channel Island postcodes.
Actors: -
Classes: CustomerDetails <<entity>>, CustomerContact <<control>> NB who is informed – Customer must be notified. Email generated?

Questioning bUC2:
Check Credit Status

Event 1
Pre-Condition: Business Layer is ready to check credit status of Customer. Business Layer has bank details of Customer.
Post-Condition: Credit Checker System is alerted to incoming check.
Interface: ISDN? What connection?
System: Generates procedure to connect to Credit Checker, this should involve interacting between entity objects Customer Details and Customer Bank Details, via a control object
Actor: Credit Checker
Class: Credit Check <<control>>, CustomerDetails <<entity>>, CustomerBankDetails <<entity>>, CreditCheckLink <<interface>>

Event 2
Pre-Condition: Connection made with Credit Checker System. This should involve an interface connector object that might need some middleware. A Credit Check <<control>> object has retrieved the necessary information to send Customer bank details (from Customer bank Details <<entity>>) and Credit Checker System in state to receive them.
Post-Condition: Customer bank details sent; received by Credit Checker.
Interface: There should be a dedicated connection to the Credit Checking Agency system since this check should take only a couple of seconds.
System: Business model locates Customer records – with Credit Check control object - from memory (Customer Application <<entity>> class and Bank Details <<entity>> class) and has sent information to Credit Checker System.
Actor: Credit Checker System
Class: Customer Application (entity), Credit Check (control), CreditCheckLink <<interface>>.

Event 3
Pre-Condition: The Credit Checker system has received the Customer bank information
Post-Condition: Customer credit worthy returned to Business Layer.
Interface: ?
System: credit worthiness status is checked on Customer record
Actor: Credit Checker System
Class: Customer Application (entity), Credit Check (process), Credit Interface (boundary)

Alternative flow

Event a3.1
Pre-Condition: The Credit Checker system has received the Customer bank information
Post-Condition: Customer credit unworthy returned to Front Office.
Interface: ?
System: credit worthiness status is checked on Customer record
Actor: Credit Checker System, Business Layer
Class: Customer Application (entity), Credit Check (process), Credit Interface (boundary)

Event a3.1.1
Pre-condition: Customer credit unworthy.

Post-condition: Customer informed of application rejection
Interface: GUI or email
System: Front Office generates email?
Actors: Customer
Classes: Customer Record

Event a3.1.1.1
Pre-condition: Customer has been notified of credit rejection (or told nothing?)
Post-condition: Customer asked to re-apply with utility bills as proof of credit status
Interface: GUI, email, letter (pack type?)
System: email generated?
Actors: Customer
Classes: Customer record.

Questioning bUC3:
Inform Bank of Scotland (BOS)

Event 1
Pre-condition: Customer Account created and stored in database. Link to BOS works.
Post-condition: Account details sent and received by BOS.
Interface: connection protocol to BOS
System: Business model must collate correct information into a file or whatever format and prepare to beam it to BOS via an interface connection class.
Actor: BOS
Class: BOS driver <<control>>, Trading Account <<entity>>, Bank Details <<entity>>, BOSlink <<interface>>

Event 2
Pre-condition: BOS has processed account information.
Post-condition: Front Office has positive reply from BOS.
Interface: connection protocol from BOS to Front Office
System: Front Office must link from interface back to Business model to determine result of BOS.
Actor: BOS
Class: BOS driver <<control>>, Trading Account <<entity>>, Bank Details <<entity>>, BOSlink <<interface>>

Exceptions

Event a2
Pre-condition: BOS has processed account information.
Post-condition: Front Office has negative received reply from BOS.
Interface: connection protocol from BOS to Front Office
System: Front Office must link from interface back to Business model to determine result of BOS.
Actor: BOS
Class: BOS driver <<control>>, Trading Account <<entity>>, Bank Details <<entity>>, BOSlink <<interface>>

Event a2.1

Pre-condition: Negative reply from BOS.
Post-Condition: Status of Customer Account altered. Customer informed that they can submit utility bills to get a valid account.
Interface: link to Customer (email, letter – new pack code?)
System: must generate correct means of communicating with customer
Actor: Customer (passive)
Class: Customer Trading Account <<database>>, BOS driver <<control>>, Customer Details <<entity>> ?

Questioning bUC4:
Generate Pack Code

Event 1
Pre-condition: Customer Application details still available i.e. object not destroyed
Post-condition: Pack Code control object has located and retrieved the Customer Application Details. Pack Type <<entity>> object created to determine the exact Pack Type required (by the end of the use case all attributes will be set whose combined value amounts to the exact pack code necessary)
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event 2
Pre-condition: PackType object instantiated. Can read country of residence
Post-condition: Country of residence attribute set
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event 3
Pre-condition: Can read type of account
Post-condition: Type of account determined
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event 4
Pre-condition: Can read certificate option.
Post-condition: Certificate option read.
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event 5
Pre-condition: Can read payment option.
Post-condition: Payment option read.

Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event 6
Pre-condition: Can read currency of payment.
Post-condition: Currency of payment determined.
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event 7
Pre-condition: Can read UK warrants option.
Post-condition: state of UK warrants option determined
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event 8
Pre-condition: Can read stock registration option
Post-condition: Stock registration determined
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event 9
Pre-condition: Can read contract note to third party option
Post-condition: Contract note to third party determined
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event 10
Pre-condition: Can read stock transfer option
Post-condition: Stock transfer option read
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event 10.1
Pre-condition: Stock transfer option selected. Number of stock transfers available.
Post-condition: Number of stock transfers determined.
Interface: -
System: -

Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event 11
Pre-condition: All attribute values have been determined. PackCoder <<control>> checks all fields have valid values.
Post-condition: The attribute values have been correlated.
Interface: -
System: Process (from PackCoder) that steps through and checks all values (in PackType)
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event 12
Pre-condition: Attribute correlation complete. Pack Code list available and correct.
Post-condition: Correlated attributes identified to specific pack code.
Interface: -
System: Parser process?
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event 13
Pre-condition: Correlated attributes checked against list.
Post-condition: Pack Code identified.
Interface: -
System: Parser process from Pack Coder <<control>>?
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event 14
Pre-condition: Pack code identified.
Post-condition: Pack code (includes Customer Details, Account Details (?) so that a mail merge can occur) added to pack type list.
Interface: -
System: Linked lists of each pack type that contain nodes for each new code package. These will be converted into CSV file sometime to make mail merge easier.
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType (node) <<entity>>, PackType List <<entity>>

Alternatives

Event a3
Pre-condition: Can read type of account
Post-condition: Type of account determined
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event a4

Pre-condition: Can read certificate option.
Post-condition: Certificate option read.
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event a5
Pre-condition: Can read payment option.
Post-condition: Payment option read.
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event a7
Pre-condition: Can read UK warrants option.
Post-condition: state of UK warrants option determined
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event a8
Pre-condition: Can read stock registration option
Post-condition: Stock registration determined
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event a9
Pre-condition: Can read contract note to third party option
Post-condition: Contract note to third party determined
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Event a10
Pre-condition: Can read stock transfer option
Post-condition: Stock transfer option read
Interface: -
System: -
Actor: -
Class: PackCoder<<control>>, CustomerApplication <<entity>>, PackType <<entity>>

Questioning bUC5:
Store Account in Database

Event 1
Pre-condition: A new TradingAccount <<entity>> object has been created by CreditCheck <<control>>.
Post-condition: TradingAccount is ready to transfer to database.
Interface: -
System: -
Actors: -
Class: TradingAccount <<entity>>, CreditCheck <<control>>, database object

Event 2
Pre-condition: TradingAccount is ready to transfer to database.
Post-condition: TradingAccount sends its own details to a database class
Interface: -
System: -
Actors: -
Class: TradingAccount <<entity>>, NewTradingAccount<<database>> object

3. Results of Questioning Database Use Case Descriptions

Questioning dUC1:
Store Trading Account

Event 1
Pre-condition: A trading account has been created in the Business Layer and is ready for storage.
Post-condition: The trading account has informed the database layer about readiness for storage.
Interface: sub-system level
System: Has to inform the database layer that there is an entity ready for storage. So does the entity have such an operation?
Actors: Business Layer
Classes: TradingAccount <<entity>>, NewTradingRecord <<database>>

Event 2
Pre-condition: The database class New Trading Account is informed of the new entity
Post-condition: The NewTradingRecord <<database>> has accessed the TradingAccount <<entity>> attributes
Interface: sub-system
System: -
Actors: Business Layer
Classes: NewTradingRecord <<database>>, TradingAccount<<entity>>

Event 3
Pre-condition: The TradingAccount has been accessed
Post-condition: Attributes are re-structure to fit database table
Interface: -
System: parser?
Actors: Business Layer
Classes: TradingAccount <<entity>> NewTradingRecord <<database>>

Event 4
Pre-condition: The attributes have been structured ready to go to the database.
Post-condition: The new structure is saved in the NewTradingRecord <<database>> object. This means that the re-structuring must occur in the <<database>> object as well as the storage. So the data must be located, retrieved, re-structured, saved.
Interface: -
System: -
Actors: Business Layer
Classes: NewTradingRecord <<database>>

Event 5
Pre-condition: NewTradingRecord ready to access Oracle database
Post-condition: New table created and populated in Oracle database
Interface: -
System: Operations required to populate the Database
Actors: Oracle Database

Classes: NewTradingRecord<<database>> ?? controller to send it to dBase?

Questioning dUC2:
Store Application Packs

Event 1
Pre-condition: A Pack Code has been created in the Business Layer and is ready for storage.
Post-condition: The Pack Code has informed the database layer about readiness for storage.
Interface: sub-system level
System: Has to inform the database layer that there is an entity ready for storage. So does the entity have such an operation?
Actors: Business Layer
Classes: PackCode <<entity>>, NewPackRecord <<database>>

Event 2
Pre-condition: The database class New PackRecord is informed of the new entity
Post-condition: The NewPackRecord <<database>> has accessed the PackCode <<entity>> attributes
Interface: sub-system
System: -
Actors: Business Layer
Classes: PackCode <<entity>>, NewPackRecord <<database>>

Event 3
Pre-condition: The PackCode has been accessed
Post-condition: Attributes are re-structure to fit database table
Interface: -
System: parser?
Actors: Business Layer
Classes: PackCode <<entity>>, NewPackRecord <<database>>

Event 4
Pre-condition: The attributes have been structured ready to go to the database.
Post-condition: The new structure is saved in the NewPackRecord <<database>> object. This means that the re-structuring must occur in the <<database>> object as well as the storage. So the data must be located, retrieved, re-structured, saved.
Interface: -
System: -
Actors: Business Layer
Classes: NewPackRecord <<database>>

Event 5
Pre-condition: NewPackRecord ready to access Oracle database
Post-condition: New table created and populated in Oracle database
Interface: -
System: Operations required to populate the Database

Actors: Oracle Database
Classes: NewPackRecord<<database>> ?? controller to send it to dBase?

References

Aaronson, D. and Ferres, S., 1983. Lexical categories and reading tasks. *Journal of Experimental Psychology: Human Perception and Performance*, 9, pp675-699.

Abbott, R., 1983. Program design by informal english descriptions. *Communications of the ACM*, 26 (11), pp882-894.

Abbott, R., 1987. Knowledge abstraction. *Communications of the ACM*, 30 (8), pp664-671.

Achour, C., 1997. Linguistic instruments for the integration of scenarios in requirements engineering. In: Dubois, E., Opdahl, A. and Pohl, K. (editors), *3rd International Workshop on Requirements Engineering: Foundation for Software Quality*, Barcelona, 16-17 June 1997, Namur University Press, pp93-106.

Achour, C., 1998a. Guiding scenario authoring. In: *8th European-Japanese Conference on Information Modelling and Knowledge Bases*, Vammala, Finland, 26-29 May 1998, pp181-200.

Achour, C., 1998b. Writing and correcting textual scenarios for systems design. *Natural Language and Information Systems Workshop*, Vienna, 28 August 1998. Version taken from: http://sunsite.informatik.rwth-aachen.de/CREWS/reports.htm, Report Series 98-26.

Achour, C., Rolland, C., Maiden, N. and Souveyet, C., 1999. Guiding use case authouring: results from an empirical study. *4th IEEE International Symposium on Requirements Engineering*, Limerick, 7-11 June 1999. Version taken from: http://sunsite.informatik.rwth-aachen.de/CREWS/reports.htm, Report Series 98-31.

Achour, C. and Maiden, N., 1999. Empirical study of use case authoring: experimental material. Internal Report, Centre de Recherche en Informatique, Univerisitie de Paris 1 – Sorbonne, http://panoramix.univ-paris1.fr/CRINFO/users/benachour/ESEM/, 15/11/99.

Alexander, I., 1998. Engineering as a co-operative inquiry: a framework. *Requirements Engineering Journal*, 3, pp130-137.

Alexander, I., 2000a. Scenarios in systems engineering – a range of techniques for engineering better systems, In: *IEE Seminar on Scenarios in the System Lifecycle* (Ref 00/138), London, 7 December 2000, IEE, pp1/1-1/6.

Alexander, I., 2000b. Editorial on REP'2000. *Requirenautics Quarterly*, 22, pp4-6.

Alexander, I. 2000c. Scenario-driven search finds more exceptions. 2nd International Workshop on the Requirements Engineering Process: Innovative Techniques, Models, Tools to support the RE Process, London, 6-8 September 2000. In: Tjoa, A., Wagner, R. and Al-Zobaidie, A. (editors), *Proceedings of DEXA'2000, 11th International Workshop on Database and Expert Systems Applications*, Los Alamitos, CA, IEEE Computer Society Press, pp991-994.

Alexander, I., 2002a. On abstraction in scenarios. *Requirements Engineering Journal*, 6, pp252-255.

Alexander, I., 2002b. Use / misuse case analysis in industrial practice. Presented at: Requirements Engineering Specialist Group of the British Computer Society Workshop: Scenarios Work! Improving Requirements Engineering with Use Cases and Scenarios, July 10 2002, U.C. London, http://mcs.open.ac.uk/computing/resg2/documents/Aleksander.pdf

Allen, R. (Ed.), 1984. *The Oxford Dictionary of Current English – 7th Edition*, Oxford.

Anda, B. and Jorgensen, M., 2000. Understanding use case models. *Beg, Borrow or Steal: Using Multidisciplinary Approaches to Software Engineering Research, Proceedings ICSE 2000 Workshop*, Limerick, June 5 2000.

Anda, B., Sjoberg, D. and Jorgensen, M., 2001. Quality and understanding of use case models. In: Lindskov Knudsen, J. (editor), *15th European Conference on Object-Oriented Programming*, Budapest, June 18-22 2001, Berlin, LNCS, Springer-Verlag, pp402-428.

Andrews, A. and Pradhan, A., 2001. Ethical issues in software engineering: the limits of policy. *Empirical Software Engineering Journal*, 6 (2), pp105-110.

Anton, A., 1996. Goal-based requirements analysis. In: *2nd International Conference on Requirements Engineering*, Colorado Springs, 15-18 April 1996, IEEE Computing, pp136-144.

Anton, A. and Potts, C., 1998. A representational framework for scenarios of system use. *Requirements Engineering Journal*. 3 (3/4), pp219-241.

Anton, A., Dempster, J. and Siege, F., 2000. Deriving goals from a use case based requirements specification for an electronic commerce system. In: Opdahl, A., Pohl, K. and Rossi, M. (editors), *6th International Workshop on Requirements Engineering: Foundation for Software Quality*, Stockholm, 5-6 June 2000, Essen, Essener Informatik Beitrage, pp10-19.

Anton, A., Carter, R., Dagnino, A., Dempster, J. and Siege, D., 2001. Deriving goals from a use case based requirements specification. *Requirements Engineering Journal*, 6 (1), pp63-73.

Arlow, J., 1998. Use cases, UML visual modelling and the trivialisation of business requirements. *Requirements Engineering Journal*, 3, pp150-152.

Arlow, J. and Neustadt, I., 2002. *UML and the Unified Process: Practical Object-Oriented Analysis and Design*, London, Addison-Wesley.

Armour, F. and Miller, G., 2001, *Advanced Use Case Modelling*, Addison-Wesley.

Barton, S. and Sanford, A., 1993. A case study of anomoly detection: Shallow semantic processing and cohesion establishment. *Memory and Cognition*, 21, pp477-487.

Beck, K. and Cunningham, W., 1989. A laboratory for teaching object-oriented thinking. In: Proceedings of OOPSLA'89, New Orleans, 1-6 October 1989. Version taken from: http://c2.com/doc/oopsla89/paper.html

Benner, K., Feather, M., Lewis Johnson, W. and Zorman, L., 1993. Utilizing scenarios in the software development process. In: Prakash, N., Rolland, C. and Pernici, B. (editors), *Information System Development Process*, Elsevier Science, pp117-134.

Berrisford, G., 1998. Improving object-oriented analysis methods. *Journal of Object-Oriented Programming*, March / April, pp6-7.

Boehm, B., 1981. *Software Engineering Economics*, Hemel Hempstead, Prentice Hall.

Booch, G., 1994. Scenarios. *Report on Object Analysis and Design*, 1 (3), pp3-6.

Booch, G., 2000. Software Development Best Practices. In: Krutchen, P., *The Rational Unified Process: An Introduction 2nd Edition*, Harlow, Addison-Wesley, pp3-16.

Booch, G., Rumbaugh, J. and Jacobson, I., 1999. *The Unified Modeling Language User Guide*, Harlow, Addison-Wesley.

Bødker, S., 2000. Scenarios in user centred design – setting the stage for reflection and action. *Interacting with Computers*, 13, pp61-75.

Bransford, J. and Franks, J., 1971. The abstraction of linguistic ideas. *Cognitive Psychology*, 2, pp331-350.

Bransford, J. and Johnson, M., 1973. Considerations of some problems of comprehension. In: Chase, W. (editor), *Visual Information Processing*, New York, Academic Press, pp383-438.

Bransford, J., Barclay, J. and Franks, J., 1972. Sentence memory: a constructive versus interpretative approach. *Cognitive Psychology*, 3, pp193-209.

Bray, I., 2002. *An Introduction to Requirements Engineering*, Harlow, Addison-Welsey.

Britton, C. and Doake, J., 2000. *Object-Oriented Software Development: A Gentle Introduction*, Maidenhead, McGraw-Hill.

Budgen, D., 1994. *Software Design*, Harlow, Addison-Wesley.

Bustard, D., He, Z. and Wilkie, F., 2000. Linking soft systems and use-case modelling through scenarios. *Interacting with Computers*, 13, pp97-110.

Campbell, R., 1992a. Will the real scenario please stand up? *SIGCHI Bulletin*, 24 (2), pp6-8.

Campbell, R., 1992b. Categorizing scenarios: a quixotic quest? *SIGCHI Bulletin*, 24 (4), pp16-17.

Carroll, J. and Rosson, M., 1992. Getting around the task-artifact cycle: how to make claims and design by scenario. *ACM Transactions on Information Systems*, 10 (2), pp181-212.

Carroll, J. (editor), 1995a. *Scenario-Based Design: Envisioning Work and Technology in System Development*, Chichester, Wiley.

Carroll, J., 1995b. Introduction: the scenario perspective on system development. In: Carroll, J. (editor), *Scenario-Based Design: Envisioning Work and Technology in System Development*, Chichester, Wiley, pp1-17.

Carroll, J., Rosson, M., Chin Jnr, G. and Koenemann, G., 1998. Requirements development in scenario-based design. *IEEE Transactions on Software Engineering*, 24 (12), pp1156-1170.

Carroll, J., 2000a. Editorial: introduction to the special issue on 'Scenario-Based System Development'. *Interacting with Computers*, 13, pp41-42.

Carroll, J., 2000b. Five reasons for scenario-based design. *Interacting with Computers*, 13, pp43-60.

Carey, T. and Rusli, M., 1995. Usage representations for reuse of design insights: a case study of access to on-line books. In: Carroll, J. (editor), *Scenario-Based Design:*

Envisioning Work and Technology in System Development, Chichester, Wiley, pp165-182.

Castell, N. and Hernandez, A., 2000. Knowledge management in the Sarel system. Second International Workshop on Natural Language Information Systems, London, 6-8 September 2000. In: Tjoa, A., Wagner, R. and Al-Zobaidie, A. (editors), *Proceedings of DEXA'2000, 11th International Workshop on Database and Expert Systems Applications*, Los Alamitos, CA, IEEE Computer Society Press, pp91-95.

Chomsky, N., 1975. *Syntactic Structures*, New York, Peter Lang Publishing.

Christerson, M. and Jacobson, I., 1995. A growing consensus on use cases. *Journal of Object-Oriented Programming*, March / April, pp15-19.

Clark, H., 1997. Dogmas of understanding. *Discourse Processes*, 23 (3), pp567-598.

Coad, P. and Yourdon, E., 1991. *Object-Oriented Analysis*, Englewood Cliffs, NJ, Prentice-Hall.

Coad, P., North, D. and Mayfield, M., 1995. *Object Models: Strategies, Patterns and Applications*, Upper Saddle River, NJ, Prentice Hall.

Cockburn, A., 1997. Using Goal-Based Use Cases. *Journal of Object-Oriented Programming*, 10 (5), pp56-62.

Cockburn, A., 2001. *Writing Effective Use Cases*, Harlow, Addison-Wesley.

Constantine, L. and Lockwood, S., 1999. *Software for Use: A Practical Guide to the Models and Methods of Usage Centred Design*, Harlow, Addison-Wesley.

Cox, K., 2000a. Fitting scenarios to the requirements process. 2nd International Workshop on the Requirements Engineering Process: Innovative Techniques, Models, Tools to support the RE Process, London, 6-8 September 2000. In: Tjoa, A., Wagner, R. and Al-Zobaidie, A. (editors), *Proceedings of DEXA'2000, 11th International Workshop on*

Database and Expert Systems Applications, Los Alamitos, CA, IEEE Computer Society Press, pp995-999.

Cox, K., 2000b. Cognitive dimensions of use cases: feedback from a student questionnaire. In: Blackwell, A. and Bilotta, E. (editors), *12th Annual Meeting of the Psychology for Programming Interest Group*, Corigliano Calabro, Italy, 10-13 April 2000, Cosenza, Memoria, pp99-121.

Cox, K., 2000c. Taking to scenarios to improve the requirements process: an experience report. Presented as: Scenarios in a financial system, *IEE Seminar on Scenarios through the System Lifecycle* (Ref 00/138), London, 7 December 2000, IEE, pp8/1-8/10.

Cox, K. and Phalp, K., 1999. Semantic and structural difficulties with the Unified Modelling Language use case notation version 1.3. *Workshop on Rigorous Modelling and Analysis of the UML: Challenges and Limitations, OOPSLA'99*, Denver, 2 November 1999.

Cox, K. and Phalp, K., 2000a. A case study implementing the Unified Modeling Language use case notation version 1.3. In: Opdahl, A., Pohl, K. and Rossi, M. (editors), *6th International Workshop on Requirements Engineering: Foundation for Software Quality*, Stockholm, 5-6 June 2000, Essen, Essener Informatik Beitrage, pp69-78.

Cox, K. and Phalp, K., 2000b. Use case authoring: replicating the CREWS guidelines experiment. *4th International Conference on Empirical Assessment in Software Engineering*, Keele, Keele University, 17-19 April 2000.

Cox, K. and Phalp, K., 2000c. Replicating the CREWS use case authoring experiment. *Empirical Software Engineering Journal*, 5 (3), pp245-268.

Cox, K., Phalp, K. and Shepperd, M., 2001. Comparing use case writing guidelines. In: Achour-Salinesi, C., Opdahl, A., Pohl, K. and Rossi, M. (editors), *7th International Workshop on Requirements Engineering: Foundation for Software Quality*, Interlaken, Switzerland, 4-5 June 2001, Essen, Essener Informatik Beitrage, pp101-112.

Davis, A., 1991. *Software Requirements Analysis and Specification*, Hemel Hempstead, Prentice Hall.

Davis, A. and Hickey, A., 2002. Requirements researchers: do we practice what we preach? *Requirements Engineering Journal*, 7, pp107-111.

Dennis, A. and Wixom, B., 2000. *Systems Analysis and Design – An Applied Approach*, Chichester, Wiley.

Denscombe, M., 1998. *The Good Research Guide*, Buckingham, Open Univeristy Press.

Dunsmore, A., Roper, M. and Wood, M., 1999. The role of comprehension in software inspection. In: *Papers from the 3rd International Conference on Empirical Assessment in Software Engineering*, Keele, Keele University, 12-14 April 1999.

Dutoit, A. and Paech, B., 2000. Supporting evolution: using Rationale in use case driven software development. In: Opdahl, A., Pohl, K. and Rossi, M. (editors), *6th International Workshop on Requirements Engineering: Foundation for Software Quality*, Stockholm, 5-6 June 2000, Essen, Essener Informatik Beitrage, pp99-112.

Dutoit, A. and Paech, B., 2001. Developing guidance and support for Rationale-based use case specification. In: Achour-Salinesi, C., Opdahl, A., Pohl, K. and Rossi, M. (editors), *7th International Workshop on Requirements Engineering: Foundation for Software Quality*, Interlaken, Switzerland, 4-5 June 2001, Essen, Essener Informatik Beitrage, pp85-100.

Dutoit, A. and Paech, B., 2002. Rationale-based use case specification. *Requirements Engineering Journal*, 7 (1), pp3-19.

Dzida, W. and Freitag, R., 1998. Making use of scenarios for validating analysis and design. *IEEE Transactions on Software Engineering*, 24 (12), pp1182-1196.

Erickson, T., 1995. Notes on design practice: stories and prototypes as catalysts for communication. In: Carroll, J. (editor), *Scenario-Based Design: Envisioning Work and Technology in System Development*, Chichester, Wiley, pp37-58.

Eriksson, H. and Penker, M., 1998. *UML Toolkit*, Chichester, Wiley.

Feblowitz, M. and Greenspan, S., 1998. Scenario-based analysis of COTS acquisition impacts. *Requirements Engineering Journal*, 3 (3/4), pp182-201.

Fenton, N., 2001. Viewpoint article: conducting and presenting empirical software engineering, *Empirical Software Engineering Journal*, 6 (3), pp195-200.

Fenton, N. and Pfleeger, S., 1996. *Software Metrics: A Rigorous and Practical Approach, 2nd Edition*, Cambridge, International Thomson Computer Press.

Fernandes, J. and Machado, R., 2001. From use cases to objects: an industrial information systems case study analysis. In: Wang, Y., Patel, S. and Johnston, R. (editors), *7th International Conference on Object-Oriented Information Systems*, Calgary, 27-29 August 2001, London, Springer-Verlag, pp319-328.

Filippidou, D., 1998. Designing with scenarios: a critical review of current research and practice. *Requirements Engineering Journal*, 3 (1), pp1-22.

Fillmore, C., 1968. The case for case. In: Holt, Rinehart and Winston (editors), *Universals in Linguistic Theory*, Chicago, Bach and Harms, pp1-88.

Firesmith, D., 1995. Use cases: the pros and cons. *Report on Object Analysis and Design*, 2 (2), pp2-6.

Fletcher, C., van den Broek, P. and Arthur, E., 1996. A model of narrative comprehension and recall. In: Britton, B. and Graesser, A. (editors), *Models of Understanding Text*, Mahwah, NJ, Lawrence Erlbaum Associates, pp141-163.

Fliedl, G., Kop, C., Mayerthaler, W., Mayr, H. and Winkler, C., 2000. Linguistic aspects of dynamics in requirements specifications. Second International Workshop on Natural Language Information Systems, London, 6-8 September 2000. In: Tjoa, A., Wagner, R. and Al-Zobaidie, A. (editors), *Proceedings of DEXA'2000, 11th International Workshop on Database and Expert Systems Applications*, Los Alamitos, CA, IEEE Computer Society Press, pp83-90.

Fowler, M., 1997. *Analysis Patterns*, Harlow, Addsion-Wesley.

Fowler, M. with Scott, K., 2000. *UML Distilled 2nd Edition*, Harlow, Addison-Wesley.

Galliers, R. and Land, F., 1987. Choosing appropriate information systems research methodologies. *Communications of the ACM*, 30 (11), pp900-902.

Garnham, A. and Oakhill, J., 1996. The mental models theory language of comprehension. In: Britton, B. and Graesser, A. (editors), *Models of Understanding Text*, Mahwah, NJ, Lawrence Erlbaum Associates, pp313-339.

Gause, D. and Weinberg, G., 1989. *Exploring Requirements: Quality Before Design*, New York, Dorset House Publishing.

Gelbukh, A., Sidorov, G. and Bolshakov, I., 2000. Coherence maintenance in human-machine dialogue: indirect antecedents and ellipses. Second International Workshop on Natural Language Information Systems, London, 6-8 September 2000. In: Tjoa, A., Wagner, R. and Al-Zobaidie, A. (editors), *Proceedings of DEXA'2000, 11th International Workshop on Database and Expert Systems Applications*, Los Alamitos, CA, IEEE Computer Society Press, pp96-100.

Gernsbacher, M., 1996. The structure-building framework: what it is, what it might also be and why. In: Britton, B. and Graesser, A. (editors), *Models of Understanding Text*, Mahwah, NJ, Lawrence Erlbaum Associates, pp289-311.

Gernsbacher, M., 1997. Two decades of structure building. *Discourse Processes*, 23 (3), pp265-304.

Glass, R., 1995. Pilot studies: what, why and how. *The Software Practitioner*, January, pp4-11.

Glass, R., 1998. *Software Runaways*, Harlow, Prentice Hall.

Glass, R., 2000. From *Keynote to 4th International Conference on Empirical Assessment in Software Engineering*, as quoted by Brereton, P., 2000. EASE 2000 editorial. *Empirical Software Engineering Journal*, 5 (3), pp171-174.

Goldin, L. and Berry, D., 1994. AbstFinder, a prototype abstraction finder for natural language text for use in requirements elicitation: design, methodology, and evaluation. *1st International Conference on Requirements Engineering*, Colorado Springs, 18-22 April 1994, Los Alamitos, CA, IEEE Computer Society Press, pp84-93.

Goldman, S., Varma, S. and Cote, N., 1996. Extending capacity-constrained construction integration: toward 'smarter' and flexible models of text comprehension. In: Britton, B. and Graesser, A. (editors), *Models of Understanding Text*, Mahwah, NJ, Lawrence Erlbaum Associates, pp73-113.

Goldman, S., Graesser, A. and van den Broek, P., 1999. Essays in honor of Tom Trabasso. In: Goldman, S., Graesser, A. and van den Broek, P. (editors), *Narrative Comprehension, Causality and Coherence: Essays in Honor of Tom Trabasso*, Mahwah, NJ, Lawrence Erlbaum Associates, pp1-10.

Graddol, D., Cheshire, J. and Swann, J., 1994. *Describing Language – 2nd Edition*, Buckingham, Open University Press.

Graesser, A. and Britton, B., 1996. Five metaphors for text understanding. In: Britton, B. and Graesser, A. (editors), *Models of Understanding Text*, Mahwah, NJ, Lawrence Erlbaum Associates, pp341-351.

Graesser, A., Singer, M. and Trabasso, T., 1994. Constructing inferences during narrative comprehension. *Psychological Review*, 101, pp371-395.

Graesser, A., Swamer, S., Baggett, W. and Sell, M., 1996. New models of deep comprehension. In: Britton, B. and Graesser, A. (editors), *Models of Understanding Text*, Mahwah, NJ, Lawrence Erlbaum Associates, pp1-32.

Graham, I., 1996. Tasks scripts, use cases and scenarios in object-oriented analysis. *Object-Oriented Systems*, 3 (3), pp123-142.

Graham, I., 1998. *Requirements Engineering and Rapid Development*, Harlow, Addison-Wesley.

Graham, I., Henderson-Sellers, B. and Younessi, H., 1997. *The OPEN Process Specification*, Harlow, Addison-Wesley.

Greene, J. and D'Oliveira, M., 1982. *Learning to use statistical tests in psychology: a student's guide*, Milton Keynes, Open University Press.

Grice, H., 1975. Logic and conversation. In: Cole, P. and Morgan, E. (editors), *Syntax and Semantics 3: Speech Acts*, New York, Academic Press.

Guindon, R., Krasner, H. and Curtis, B., 1987. Breakdowns and processes during the early activities of software design by professionals. In: Olsen, G. (editor), *Empirical Studies of Programmers: Second Workshop*, Norwood, NJ, Ablex, pp65-82.

Haberlandt, K., 1984. Components of sentence and word reading times. In: Kieras, D. and Just, M. (editors), *New Models in Reading Comprehension Research*, Mahwah, NJ, Lawrence Erlbaum Associates, pp219-252.

Hall, T., Beecham, S. and Rainer, A., 2002. Requirements problems in twelve software companies: an empirical analysis. *Proceedings of the 6th International Conference on Empirical Assessment in Software Engineering*, Keele, Keele University, 8-10 April 2002.

Halliday, M. and Hasan, R., 1976. *Cohesion in English*, Harlow, Longman Group.

Ham, G., 1998. Four roads to use case discovery – there is a use (and a case) for each one. *CrossTalk*, December. Version taken from: www.stsc.hill.af.mil/CrossTalk/1998/dec.ham.as.

Harwood, R., 1997. Use case formats: requirements, analysis and design. *Journal of Object-Oriented Programming*, January, pp54-66.

Haumer, P., Heymans, P., Jarke, M. and Pohl, K., 1998. Bridging the gap between past and future in requirements engineering: a scenario-based approach. Version taken from: http://sunsite.informatik.rwth-aachen.de/CREWS/reports.htm, Report Series 98-38.

Henderson-Sellers, B., 1997. *A Book of Object-Oriented Knowledge: An Introduction to Object-Oriented Software Engineering, 2nd Edition*, Upper Saddle River, NJ, Prentice-Hall.

Heymans, P. and Dubois, E., 1998. Scenario-based techniques for supporting the elaboration and the validation of formal requirements. *Requirements Engineering Journal*, 3 (3/4), pp202-218.

Hofmann, H. and Lehner, F., 2001. Requirements engineering as a success factor in software projects. *IEEE Software*, July/August, pp58-66.

Holbrook, H., 1990. A scenario-based methodology for conducting requirements elicitation. *ACM SIGSOFT Software Engineering Notes*, 15 (1), pp95-105.

Holt, A., 1999. Formal verification with natural language specifications: guidelines, experiments and lessons so far. *South African Computer Journal*, 24, November, pp253-257.

Homrighausen, A., Six, H. and Winter, M., 2002. Roundtrip prototyping based on integrated functional and user interface requirements specifications. *Requirements Engineering Journal*, 7 (1), pp34-45.

Hooks, I., 2000. Requirements engineering: is it 'mission impossible'? *Requirements Engineering Journal*, 5, pp194-197.

Host, M., Regnell, B. and Wohlin, C., 2000. Using students as subjects – a comparative study of students and professionals in lead-time impact assessment. *Empirical Software Engineering Journal*, 5 (3), pp201-214.

Hughes, J., 2000. Informing requirements: ethnography and social issues. In: *IEE Seminar on Scenarios in the System Lifecycle* (Ref 00/138), London, 7 December 2000, IEE, pp6/1-6/4.

Hughes, J., O'Brien, J., Rodden, T., Rouncefield, M. and Sommerville, I., 1995. Presenting ethnography in the requirements process. *2nd International Symposium on Requirements Engineering*, York, 27-29 March 1995, Los Alamitos, CA, IEEE Computer Society Press, pp27-34.

Hurlbut, R., 1997. A survey of approaches for describing and formalizing use cases. Technical Report XPT-TR-97-03, Expertech Ltd. Version taken from: http://www.iit.edu/~rhurlbut/xpt-tr-97-03.html

Hsia, P. and Yaung, A., 1988. Screen-based scenario generator: a tool for scenario-based prototyping. *Proceedings of the 21st IEEE Conference on System Science*, Hawaii, 4-7 January 1998, IEEE Computer Society Press, pp455-461.

Hsia, P., Samuel, J., Gao, J., Kung, D., Toyoshima, Y. and Chen, C., 1994. Formal approach to scenario analysis. *IEEE Software*, March, pp33-41.

Insfran, E., Pastor, O. and Wieringa, R., 2002. Requirements engineering-based conceptual modelling. *Requirements Engineering Journal*, 7 (2), pp61-72.

Isoda, S., 2001. Object-oriented real-world modelling revisited. *Journal of Systems and Software*, 59, pp153-162.

Jackendoff, R., 2002. *Foundations of Language*, New York, Oxford University Press.

Jackson, M., 1995. *Software Requirements and Specifications: A Lexicon on Principles, Prejudice and Practice*, Wokingham, Addison-Wesley.

Jackson, M., 1998. A discipline of description. *Requirements Engineering Journal*, 3, pp73-78.

Jackson, M., 2001. *Problem Frames*, Harlow, Addison-Wesley.

Jacobson, I., Christerson, M., Jonsson, P. and Overgaard, G., 1992. *Object-Oriented Software Engineering: A Use Case Driven Approach*, Wokingham, Addison-Wesley.

Jacobson, I., 1994a. Basic use case modelling, *Report on Object Analysis and Design*, 1 (2), pp15-19.

Jacobson, I., 1994b. Basic use case modelling (continued), *Report on Object Analysis and Design*, 1 (3), pp7-9.

Jacobson, I., 1995. The use-case construct in object-oriented software engineering. In: Carroll, J. (editor), *Scenario-Based Design: Envisioning Work and Technology in System Development*, Chichester, Wiley, pp309-336.

Jacobson, I., Ericcson, M. and Jacobson, A., 1995. *The Object Advantage: Business Process Reengineering with Object Technology*, New York, Addison-Wesley.

Jacobson, I., Booch, G. and Rumbaugh, J., 1999. *The Unified Software Development Process*, Reading, MA, Addison-Wesley.

Jacobson, I., 2001. Introduction. In: Armour, F. and Miller, G., 2001, *Advanced Use Case Modelling*, Harlow, Addison-Wesley, ppxiii-xiv.

Jarke, M., 1998. Guest editorial: interdisciplinary uses of scenarios. *Requirements Engineering Journal*, 3 (3/4), pp153-154.

Jarke, M. and Kurki-Suonio, R., 1998. Guest editorial – special issue on scenario management. *IEEE Transactions on Software Engineering*, 24 (12), pp1033-1035.

Jarke, M., Pohl, K., Haumer, P., Weidenhaupt, K., Dubois, E., Heymans, P., Rolland, C., Ben Achour, C., Cauvet, C., Ralyté, J., Sutcliffe, A., Maiden, N. and Minocha, S., 1997. Scenario use in european software organisations – results from site visits and questionnaires. Taken from: http://sunsite.informatik.rwth-aachen.de/CREWS/reports.htm, Report Series 97-10.

Jarke, M., Tung Bui, X. and Carroll, J., 1998. Scenario management: an interdisciplinary approach. *Requirements Engineering Journal*, 3 (3/4), pp155-173.

Jarke, M., 1999. CREWS: towards systematic usage of scenarios, use cases and scenes. In: *Wirtschaftinformatik 99*, Saarbrucken, 3-5 March 1999, Springer Aktuell. Version taken from: http://sunsite.informatik.rwth-aachen.de/CREWS/reports.htm, Report Series 99-02.

Jarvenpaa, S., 1988. The importance of laboratory experimentation in IS research. *Communications of the ACM*, 31 (12), pp1502-1504.

Johnson, P., Johnson, H. and Wilson, S., 1995. Rapid prototyping of user interfaces driven by task models. In: Carroll, J. (editor), *Scenario-Based Design: Envisioning Work and Technology in System Development*, Chichester, Wiley, pp209-246.

Kaakinen, J., Hyona, J. and Keenan, J., 2002. Perspective effects on online text processing. *Discourse Processes*, 33 (2), pp159-173.

Kaindl, H., 1998. Combining goals and functional requirements in a scenario-based design process. In: Johnson, H., Nigay, L. and Roast, C. (editors), *People and Computers XIII: Proceedings of HCI'98*, Sheffield, September 1998, London, Springer-Verlag, pp101-121.

Kamsties, E., Hormann, K. and Schlich, M., 1998. Requirements engineering in small and medium enterprises. *Requirements Engineering Journal*, 3, pp84-90.

Karat, J., 1995. Scenario use in the design of a speech recognition system. In: Carroll, J. (editor), *Scenario-Based Design: Envisioning Work and Technology in System Development*, Chichester, Wiley, pp109-133.

Kavakli, E., 2002. Goal-oriented requirements engineering: a unifying framework. *Requirements Engineering Journal*, 6, pp237-251.

Kitchenham, B., Pfleeger, S., Pickard, L., Jones, P., Hoaglin, D., El-Emam, K. and Rosenberg, J., 2000. Preliminary guidelines for empirical research in software engineering. *IEEE Transactions on Software Engineering*, accepted for publication.

Korn, J., Scenarios through linguistic modelling. In: *IEE Seminar on Scenarios in the System Lifecycle* (Ref 00/138), London, 7 December 2000, IEE, pp3/1-3/7.

Kösters, G., Six, H. and Winter, M., 2000. Validation and verification of use cases and class models. In: Opdahl, A., Pohl, K. and Rossi, M. (editors), *6th International Workshop on Requirements Engineering: Foundation for Software Quality*, Stockholm, 5-6 June 2000, Essen, Essener Informatik Beitrage, pp59-68.

Kösters, G., Six, H. and Winter, M., 2001. Coupling use cases and class models as a means for validation and verification of requirements specifications. *Requirements Engineering Journal*, 6 (1), pp3-17.

Kovitz, B., 1999. *Practical Software Requirements: A Manual of Content and Style*, Greenwich, CT, Manning Publications.

Krutchen, P., 2000. *The Rational Unified Process: An Introduction, 2nd Edition*, Harlow, Addison-Wesley.

Kulak, D. and Guiney, E., 2000. *Use Cases – Requirements in Context*, Harlow, Addison-Wesley.

Kuuti, K., 1995. Work processes: scenarios as a preliminary vocabulary. In: Carroll, J. (editor), *Scenario-Based Design: Envisioning Work and Technology in System Development*, Chichester, Wiley, pp19-36.

Kyng, M., 1992. Scenario? Guilty! *SIGCHI Bulletin*, 24 (4), pp.8-9.

Kyng, M., 1995. Creating contexts for design. In: Carroll, J. (editor), *Scenario-Based Design: Envisioning Work and Technology in System Development*, Chichester, Wiley, pp85-107.

Lauesen, S., 2002. *Software Requirements: Styles and Techniques*, Harlow, Addison-Wesley.

Lawrence, B., Wiegers, K. and Ebert, C., 2001. The top risks of requirements engineering. *IEEE Software*, November / December, pp62-63.

Leech, G., 1991. *An A-Z of English Grammar and Usage*, Walton-on-Thames, Nelson.

Leibundgut, R., 2002. Use cases in the project lifecycle. Presented at: Requirements Engineering Specialist Group of the British Computer Society Workshop: Scenarios Work! Improving Requirements Engineering with Use Cases and Scenarios, July 10 2002, U.C. London, http://mcs.open.ac.uk/computing/resg2/documents/Leibundgut.pdf

Leite, J., Hadad, G., Doorn, J. and Kaplan, G., 2000. A scenario construction process. *Requirements Engineering Journal*, 5 (1), pp38-61.

Liang, J. and Palmer, J., 1994. A pattern matching and clustering based approach for supporting requirements transformation. *1st International Conference on Requirements Engineering*, Colorado Springs, 18-22 April 1994, Los Alamitos, CA, IEEE Computer Society Press, pp180-183.

Long, D., Seely, M., Oppy, B. and Golding, J., 1996. The role of processing in reading ability. In: Britton, B. and Graesser, A. (editors), *Models of Understanding Text*, Mahwah, NJ, Lawrence Erlbaum Associates, pp189-214.

Lubars, M., Potts, C. and Richter, C., 1993. Developing initial OOA models. In: *15th International Conference on Software Engineering*, Baltimore, May 17-21 1993, IEEE Computer Society Press, pp255-264.

MacLean, A. and McKerlie, D., 1995. Design space analysis and use representations. In: Carroll, J. (editor), *Scenario-Based Design: Envisioning Work and Technology in System Development*, Chichester, Wiley, pp183-207.

Magliano, J., 1999. Revealing inference processes during text comprehension. In: Goldman, S., Graesser, A. and van den Broek, P. (editors), *Narrative Comprehension, Causality and Coherence: Essays in Honor of Tom Trabasso*, Mahwah, NJ, Lawrence Erlbaum Associates, pp55-75.

Magliano, J. and Graesser, A., 1991. A three-pronged method for studying inference generation in library text. *Poetics*, 20, pp193-232.

Maiden, N., 2002. Tales from the trenches: using scenarios to specify an air traffic control system. Presented at: Requirements Engineering Specialist Group of the British Computer Society Workshop: Scenarios Work! Improving Requirements Engineering with Use Cases and Scenarios, July 10 2002, U.C. London, http://mcs.open.ac.uk/computing/resg2/documents/Maiden.pdf

Maiden, N. and Corrall, D., 2000. Scenario-driven systems engineering. *IEE Seminar on Scenarios through the System Lifecycle* (Ref 00/138), London, 7 December 2000, IEE, pp2/1-2/3.

Mannes, S. and St George, M., 1996. Effects of prior knowledge on text comprehension: a simple modelling approach. In: Britton, B. and Graesser, A. .(editors), *Models of Understanding Text*, Mahwah, NJ, Lawrence Erlbaum Associates, pp115-139.

Marr, D., 1982. *Vision*, San Francisco. Freeman.

Mattingly, L. and Rao, H., 1998. Writing effective use cases and introducing collaboration cases. *Journal of Object-Oriented Programming*, October, pp.77-87.

McGraw, K. and Harbison, K., 1997. *User-Centred Requirements: The Scenario-Based Engineering Process*, New Jersey, Lawrence Erlbaum Associates.

McKoon, G. and Ratcliff. R., 1992. Inference during reading. *Psychological Review*, 99, pp440-466.

McNamara, D. and Kintsch, W., 1996. Learning from texts: effects of prior knowledge and text coherence. *Discourse Processes*, 22, pp247-288.

Melchisedech, R., 1998. Investigation of requirements documents written in natural language. *Requirements Engineering Journal*, 3, pp91-97.

Miles, M. and Huberman, A., 1994. *Qualitative Data Analysis, 2nd Edition*, Thousand Oaks, CA, Sage Publications.

Miller, J., Daly, J., Wood, M., Roper, M. and Brooks, A., 1997. Statistical power and its subcomponents – missing and misunderstood concepts in empirical software engineering. *Information and Software Technology*, 39, pp285-295.

Miller, J., Wood, M. and Roper, M., 1998. Further experiences with scenarios and checklists. *Empirical Software Engineering Journal*, 3, pp37-64.

Muller, M., Tudor, L., Wildman, D., White, E., Root, R., Dayton, T., Carr, R., Diekmann, B. and Dykstra-Erickson, E., 1995. Bifocal tools for scenarios and representations in participatory activities with users. In: Carroll, J. (editor), *Scenario-Based Design: Envisioning Work and Technology in System Development*, Chichester, Wiley, pp135-163.

Murphy, R., 1985. *English Grammar in Use, 2nd Edition*, Cambridge, Cambridge University Press.

Nardi, B., 1992. The use of scenarios in design. *SIGCHI Bulletin*, 24 (4), pp13-14.

Nielson, J., 1995. Scenarios in discount usability engineering. In: Carroll, J. (editor), *Scenario-Based Design: Envisioning Work and Technology in System Development*, Chichester, Wiley, pp59-83.

O'Brien, E. and Myers, J., 1999. Text comprehension: a view from the bottom up. In: Goldman, S., Graesser, A. and van den Broek, P. (editors), *Narrative Comprehension, Causality and Coherence: Essays in Honor of Tom Trabasso*, Mahwah, NJ, Lawrence Erlbaum Associates, pp35-53.

Object Management Group (OMG), 1999. *Unified Modeling Language Specification.* Proposed Final Version UML 1.3, June 1999. Version taken from: http://www.omg.org/cgi-bin/doc?ad/99-06-09

Object Management Group (OMG), 2001. *Unified Modeling Language v1.4 – Semantics*, Document 01-09-73. Version taken from: http://www.omg.org/pub/docs/formal/01-09-73.pdf

Ould, M., 1995. *Business Processes: Modelling and Analysis for Reengineering and Improvement*, Chichester, Wiley.

Overmyer, S., Lavoie, B. and Rambow, O., 2001. Conceptual modeling through liguistic analysis using LIDA. In: *23rd International Conference on Software Engineering*, Toronto, 12-19 May 2001, Toronto, IEEE Computer Society Press, pp401-410.

Ozyurek, A. and Trabasso, T., 1997. Evaluation during understanding of narratives. *Discourse Processes*, 23, pp305-335.

Pais, A., Oliveira, P. and Leite, P., 2001. Robustness diagram: a bridge between business modelling and system design. In: Wang, Y., Patel, S. and Johnston, R. (editors), *7th International Conference on Object-Oriented Information Systems*, Calgary, 27-29 August 2001, London, Springer-Verlag, pp530-539.

Perfetti, C., 1997. Sentences, individual differences and multiple texts: three issues in text comprehension. *Discourse Processes*, 23, pp337-355.

Pfleeger, S., 1994. Design and analysis in software engineering part 1: the language of case studies and formal experiments. *ACM SIGSOFT Software Engineering Notes*, 19 (4), pp16-20.

Phalp, K., 1995. *An Empirical Investigation of Software Process Modelling*, PhD Thesis, Bournemouth University, UK.

Phalp, K. and Cox, K., 2000a. Picking the right problem frame: an empirical study. *4th International Conference on Empirical Assessment in Software Engineering*, Keele, Keele University, 17-19 April 2000.

Phalp, K. and Cox, K., 2000b. Picking the right problem frame - an empirical study. *Empirical Software Engineering Journal*, 5 (3), pp215-228.

Phalp, K. and Cox, K., 2001. Guiding use case driven requirements elicitation and analysis. In: Wang, Y., Patel, S. and Johnston, R. (editors), *7th International Conference on Object-Oriented Information Systems*, Calgary, 27-29 August 2001, London, Springer-Verlag, pp329-332.

Phalp, K. and Cox, K., 2002. Supporting communicability with use case guidelines: an empirical study. *6th International Conference on Empirical Assessment in Software Engineering*, Keele, Keele University, 8-10 April 2002.

Phalp, K. and Shepperd, M., 1999. Analysing process models quantitatively. *3rd International Conference on Empirical Assessment in Software Engineering*, Keele, Keele University, 12-14 April 1999.

Pickard, L., 1992. *DESMET Case Study Procedures*, NCC Consultancy, 28 July 1992.

Pohl, K., Brandenburg, M. and Gulich, A., 2001. Integrating requirements and architecture information: a scenario and meta-model based approach. In: Achour-Salinesi, C., Opdahl, A., Pohl, K. and Rossi, M. (editors), *7th International Workshop on Requirements Engineering: Foundation for Software Quality*, Interlaken, Switzerland, 4-5 June 2001, Essen, Essener Informatik Beitrage, pp68-84.

Pooley, R. and Stevens, P., 1999. *Using UML – Software Engineering with Objects and Components*, Harlow, Addison-Wesley.

Potts, C., 1993. Software engineering research revisited. *IEEE Software*, September, pp19-28.

Potts, C., Takahashi, K. and Anton, A., 1994. Inquiry-based requirements analysis. *IEEE Software*, March, pp21-32.

Pressman, R., 1997. *Software Engineering: A Practitioner's Approach*, London, McGraw Hill.

Rational Software Corporation, 1997. *UML Notation Guide Version 1.1*, http://www.rational.com/uml/

Ratjens, M., 1997. 7 tips for writing better use cases. *Class Technology*. Version taken from: http://www.class.com.au/newsletr/97sep/usecases.htm

Raven, J., 2000. Ethical dilemmas. *The Psychologist*, 13 (8), pp404-406.

Reenskaug, T. with Wold, P. and Lehne, O., 1996. *Working With Objects: The OOram Software Engineering Method*, Greenwich, CT, Manning Publications.

Regnell, B. and Davidson, A., 1997. From requirements to design with use cases – experiences from industrial pilot projects. In: Dubois, E., Opdahl, A. and Pohl, K. (editors), *3rd International Workshop on Requirements Engineering: Foundation for Software Quality*, Barcelona, 16-17 June 1997, Namur University Press.

Regnell, B., Kimber, K. and Wesslen, A., 1995. Improving the use case driven approach to requirements engineering. In: *2nd International Symposium on Requirements Engineering*, York, March 1995, Los Alamitos, CA, IEEE Computer Society Press, pp40-47.

Reisner, D., 1992. Further uses of 'scenario'. *SIGCHI Bulletin*, 24 (4), p15.

Robertson, S., 1995. Generating object-oriented design representations via scenario queries. In: Carroll, J. (editor), *Scenario-Based Design: Envisioning Work and Technology in System Development*, Chichester, Wiley, pp279-308.

Robertson, S., 2001. Are we afraid of the dark? *IEEE Software*, July / August, pp12-15.

Robson, C., 1993. *Real World Research*, Oxford, Blackwell.

Rolland, C. and Ben Achour, C., 1998. Guiding the construction of textual use case specifications. *Data and Knowledge Engineering Journal*, 25 (1-2), pp125-160.

Rolland, C., and Proix, C., 1992. Natural language approach to conceptual modelling. In: Loucopoulos, P. and Zacari, R. (editors), *Conceptual Modelling, Databases and CASE – an integrated view of information systems development*, Chichester, Wiley, pp447-463.

Rolland, C., Ben Achour, C., Cauvet, C., Ralyté, J., Sutcliffe, A., Maiden, N., Jarke, M., Haumer, P., Pohl, K., Dubois, E. and Heymans, P., 1998a. A proposal for a scenario classification framework. *Requirements Engineering Journal*, 3 (1), pp23-47.

Rolland, C., Souveyet, C. and Ben Achour, C., 1998b. Guiding goal modelling using scenarios. *IEEE Transactions on Software Engineering*, 24 (12), pp1055-1071.

Rolland, C., 2002. L'E-Lyee: coupling L'Ecritoire and LyeeALL. *Information and Software Technology*, 44, pp185-194.

Rosenberg, D. with Scott, K., 1999. *Use Case Driven Object Modelling with UML: A Practical Approach*, Harlow, Addison-Wesley.

Rosson, M. and Carroll, J., 1995. Narrowing the specification implementation gap in scenario-based design. In: Carroll, J. (editor), *Scenario-Based Design: Envisioning Work and Technology in System Development*, Chichester, Wiley, pp247-278.

Rubin, K. and Goldberg, A., 1992. Object Behaviour Analysis. *Communications of the ACM*, 35 (9), pp48-62.

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F. and Lorensen, W., 1991. *Object-Oriented Modelling and Design*, Upper Saddle River, NJ, Prentice Hall.

Rumbaugh, J., 1994. Getting started: using use cases to capture requirements. *Journal of Object-Oriented Programming*, September, pp8-23.

Saeki, M., Horai, H. and Enomoto, H., 1989. Software development process from natural language specification. *11th International Conference on Software Engineering*, Pittsburgh, 15-18 May 1989, Los Alamitos, CA, IEEE Computer Society Press.

Salkind, N., 2000. *Statistics for people who (think they) hate statistics*, London, Sage Publications.

Satzinger, J. and Orvik, T., 2001. *The Object-Oriented Approach: Concepts, System Development and Modelling with the UML, 2nd Edition*, Cambridge, Thomson Learning.

Schneider, G. and Winters, J., 1998. *Applying Use Cases: A Practical Guide*, Harlow, Addison-Wesley.

Sieber, J., 2001. Protecting research subjects, employees and researchers: implications for software engineering. *Empirical Software Engineering Journal*, 6 (4), pp329-341.

Sim, S., Singer, J. and Storey, M., 2001. Beg, borrow or steal: using multidisciplinary research approaches in empirical software engineering research. *Empirical Software Engineering Journal*, 6 (1), pp85-93.

Sindre, G. and Opdahl, A., 2001. Templates for misuse case description. In: Achour-Salinesi, C., Opdahl, A., Pohl, K. and Rossi, M. (editors), *7th International Workshop on Requirements Engineering: Foundation for Software Quality*, Interlaken, Switzerland, 4-5 June 2001, Essen, Essener Informatik Beitrage, pp125-136.

Sommerville, I. and Sawyer, P., 1997. *Requirements Engineering: A Good Practice Guide*, Chichester, Wiley.

Standish Group, 1995. The Standish Group Report: Chaos. http://www.scs.carleton.ca/~beau/PM/Standish-Report.html

Stiemerling, O. and Cremers, A., 1998. The use of cooperative scenarios in the design and evaluation of a CSCW system. *IEEE Transactions on Software Engineering*, 24 (12), pp1171-1181.

Storey, M., Phillips, B. and Maczewski, M., 2001. Is it ethical to evaluate web-based learning tools using students? *Empirical Software Engineering Journal*, 6 (4), pp343-348.

Sutcliffe, A., 1996. A conceptual framework for requirements engineering. *Requirements Engineering Journal*, 1, pp170-189.

Sutcliffe, A., 1997. Task-related information analysis. *Human Computer Studies*, 47, pp223-257.

Sutcliffe, A., 1998. Scenario-based requirements analysis. *Requirements Engineering Journal*, 3, pp48-65.

Sutcliffe, A., Maiden, N., Minocha, S. and Manuel, D., 1998. Supporting scenario-based requirements engineering. *IEEE Transactions on Software Engineering*, 24 (12), pp1072-1088.

Swan, M., 1980. *Practical English Usage*, Oxford, Oxford University Press.

Tichy, W., 1998. Should computer scientists experiment more? *IEEE Computer*, May, pp32-40.

Trabasso, T., van den Broek, P. and Suh, S., 1989. Logical necessity and transitivity in causal relations in stories. *Discourse Processes*, 12, pp1-25.

Traxler, M. and Gernsbacher, M., 1995. Improving coherence in written communication. In: Gernsbacher, M. and Givon, T. (editors), *Coherence in Spontaneous Text*, Philadelphia, John Benjamins, pp215-237.