Exploring the requirements process for a complex, adaptive system in a high risk software development environment

Sheridan Jeary[1], Katarzyna Musial[2], Keith Phalp[1]
[1] Software Systems Research Centre, [2]SMART Technology Research Centre,
Bournemouth University, Fern Barrow, Poole, Dorset BH12 5BB.
sjeary, kmusial, kphalp @ bournemouth.ac.uk

Abstract

This work ties together research from a number of different areas to show how the development of a complex adaptive system for an industrial company has a number of difficulties given the current state of the art. The INFER system which is a Complex Adaptive System (CAS) has a number of attributes which mean that current requirements and indeed development processes are not able to cope with them adequately. A CAS can be recognised by the fact that it consists of a number of agents acting together dynamically resulting in emergent behaviour. This emergent behaviour cannot be predicted and thus, along with other phenomena such as reaction to and with the environment and deciding the different responsibilities of the components means that the requirements process for such a system is a current research area. A retrospective case study is underway to capture the rich data available from the experiences of building such a system.

Keywords

requirements process, development process, complex adaptive systems, adaptive software systems, risk

Introduction

Traditionally, a successful software system is one that meets or exceeds the needs or requirements of the client, users or other stakeholders. There has been considerable discussion over the last few years about the difficulty of obtaining requirements from the user when they have the IWKIWISI (I will know it when I see it) issue [1] and how the capturing of their requirements is a problem for requirements engineers.

Alongside requirements risk, the technical complexity of a project and the distributed nature of the personnel are also likely to be high risk elements of a project. The INFER project brings together all these elements and, therefore, the aim is to explore the requirements process in a technically complex project that is distributed between across three locations. Thus a retrospective, qualitative and exploratory case study is being run to examine both the successes and difficulties of creating a specification. The creation of the specification is for an exploratory complex, adaptive predictive system and has required academics to articulate their ideas to allow them to identify the requirements and to produce a specification prior to building the system. The personnel were distributed among three countries within Europe.

This work explores the background to the requirements process, complex systems adaptive systems, the INFER system, risk and the requirements process for adaptive systems and explains the importance of future work. Section 2 looks at the traditional requirements process. In Section 3 complex, adaptive systems are explained along with the phenomena they demonstrate. Section 4 explores the INFER project and Section 5 discusses how the INFER system is an adaptive system. Section 6 describes the attributes of the INFER project that are considered a risk factor in the project management community before Section7 explores the requirements process for complex adaptive systems and a system like INFER. The work then concludes and discusses current work in carrying out a retrospective case study before highlighting future work in Section 8.

The traditional requirements and development process

For the purpose of this study the measure of success of a software system is how that system broadly matches the purpose for which it is intended [2]. In addition, the way that the requirements process is considered is in the spirit of the ideas put forward by Jackson [3] who considers that the requirements engineer, the work that they do and the artefacts that they produce are all clearly situated in the problem domain. The further elements of the software engineering process are considered to be a part of the solution domain. The specification is at the interface of the two domains.

Requirements traditionally have been about completeness, consistency, traceability and testability, with an emphasis on early and accurate determination, often motivated by the high cost of fixing requirements errors later in the life cycle [4].

Hence, plan-based approaches, such as defined by Boehm and Turner [5] have proved to be both a popular and useful process particularly when it was possible to specify requirements in advance, and have also been extended to include processes for distributed (or even global) development [6]. Systems, such as embedded systems or safety critical systems for example, work well on this premise. The systems need to be specified sometimes formally and often with an accuracy that ensures that the resulting inspections and tests are passed with very few errors. The Capability Maturity Model was later developed with the idea of improvement of the software process making it both repeatable and measurable [7].

However, there are many systems that do not fit this ideal. Examples include development of web applications where the business is reacting to both business and technology pressures, where there is a lack of time to create full specifications and thus an iterative, incremental approach has been recommended to requirements with a prototyping approach [8]. In the post methodological era many developers are turning to a more informal way of development and using ‘more flexible off-the-cuff approaches’ [9]. There is some evidence that the backlash against the traditional processes and methods has focussed developers minds about they work and increased the search for helpful methods and techniques [9]. It is in this development environment that agile methods have become increasingly important and popular. This change in the development process has affected the gathering of the requirements by the creation of techniques such as user stories [10] and the production of a specification is no longer necessarily part of the development process. The agile group of methods encourage the stakeholder to become part of the development team and, as such, can act as a bridge between the problem and solution domains.

A further broad group of systems that are problematic when it comes to use of the waterfall process are complex predictive, adaptive systems that are described in Section 3.

Complex Adaptive Systems

The area of Complex Adaptive Systems (CAS) is a wide research field, which brings together specialists with backgrounds in economics, physics, social sciences, biology, computer science, and others. CAS's are systems that consist of populations of interacting agents that are able to adapt. The interactions of those agents lead to complex non-linear dynamics, the results of which are emergent system phenomena [11, 12]. The complexity means that the behaviour of the whole system cannot be described by a simple sum of behaviours of the entities that create this system [13, 14]. Examples of CASs include social insect and ant colonies, ecosystems, the brain and the immune system, cell biology and genomics, manufacturing business and various human social group-based endeavours in socio-cultural systems such as health and related educational support communities. However, not only natural systems can be complex and adaptive; software is an artificial system that needs to adapt.

The increasing complexity of software systems means that new approaches for developing and maintaining them are required. They need to adapt to the changing external environment and to new users' requirements; in addition to integrating large amounts of information and using the different technologies that they need to employ. Thus complexity and adaptivity of software systems has become an extensively researched area [15-17]. Self-adaptive software systems are the next generation of systems that are able to evaluate their own behaviour. If as a result of evaluation the system does not perform in the way it should, then the system adapts its behaviour [18]. Whilst completing its' analysis, a self-adaptive system will also need to take into account the external environment because the behaviour of the system is modified in response to changes in the environment in which it operates [19] and also to the changes internal to the system itself [15]. Modelling dimensions for self-adaptive systems can be grouped in four main groups: (i) self-adaptation of the system goals, (ii) causes of self-adaptation, (iii) mechanisms to achieve self-adaptation, and (iv) effects of self-adaptation on the system [15], [16]. Their aim is to support creation of models for representing important aspects of the self-adaptation and these factors should be considered when building this kind of system.

In the next Section, an example of a self-adaptive system that is being built within the Computational Intelligence Platform for Evolving and Robust Predictive Systems (INFER) project is presented.

The INFER Project

INFER is a major EU-funded project involving 25 academics from organisations in three different countries. This includes Evonik Industries from Germany, Research and Engineering Centre (REC) from Poland and Bournemouth University. INFER is a project funded by the European Commission within the Marie Curie Industry and Academia Partnerships & Pathways (IAPP) programme.

The project focuses on pervasively adaptive software systems for the development of a modular platform applicable in various commercial settings and industries. The main innovation of the project is a novel type of environment in which the "fittest" predictive model for whatever purpose will emerge – either autonomously or by user high-level goal-related assistance and feedback. Such a system is beneficial for businesses relying on accurate predictions of any type (e.g., customer behaviour, market conditions) and, at the same time, requiring an automated ability to react to changes in market or operational conditions.

System Goals

The INFER system aims to address current challenges in developing adaptive predictive models such as (i) how to build many models that can also be a combination of other models, (ii) what and how the adaptation mechanisms can be introduced, and (iii) which parts of the process and in what way they can be automated. The high level aim of the INFER project is to build a modular software platform for predictive modelling applicable in different industries. In order to achieve this, the predictive system that consists of a population of predictive models will be developed. The 'fittest' predictive model for whatever purpose will be chosen and used. Once applied, the predictive system will exploit any available feedback for its performance monitoring and adaptation. Based on the above description it can be summed up that the main goal of the software system is to deliver accurate predictions all the time.

High Level System Architecture

Figure 1: The main INFER system components.

The main components of the system are presented in Figure 1. The data input/output infrastructure enables the integration of the external data sources, i.e. import to the system of data in different formats and export of output data in the format that any defined external system can work with. The GUI interface is the second main component. Three GUI interfaces will be developed and the layout and functions available for users will differ depending on the access rights that a given user has. In general, these interfaces will provide an Administration and Maintenance interface that enables the user to change each part of the system; an Expert interface that has access to an expert knowledge module within the Platform core; and an Operator GUI from which a user can monitor the work of the system. This component will evolve to provide the user access to the newly developed functionalities of the system.

The Platform core will enable the processing of input data, build the predictive models that can compete or collaborate with each others, self-assess built models (also based on the meta-knowledge about previous predictive models), adapt the models, and in a result of all above deliver accurate predictions.

The INFER system as an Adaptive System

In this section the INFER system will be presented in the context of CASs. It has been done using four groups of modelling dimensions for self-adaptive systems [15], and a full description according to these dimensions is made available at [16]. The following now considers the main issues for requrememnts determination with INFER with respect to System Goals, Changes within INFER and the impact of adaption.

INFER system Goals

The INFER system has one main goal and three sub-goals as described in section 4.1. They are static and will not change during the system lifetime. However, the system is innovative and there are no solutions for some of the issues in modelling self-adaptive predictive systems. This means that one cannot say that the list of system goals is final and there will not be any additional goals. The continuous evolution of the requirements specification means that it is possible that new detailed aims may emerge during system analysis and design.

Changes in the INFER system

The next phase is to identify what kind of changes can occur in the system and in the environment in which it operates. The environment external to the system will change when the INFER system is applied in different application areas and this will mean that different input data will be sent to the system. Even within the same industry the input data can vary and change continuously. As a consequence, models the system builds need to be adapted or redeveloped. There can be also changes in the internal system environment when the system develops a better predictive model for a given problem. In such a case the system will reconfigure its behaviour to create and start using the more accurate model. Whilst these changes are all functional there will also be changes that are non-functional because for different industries different quality requirements can be important. In addition, the changes can occur very often in some application areas and extra adaptation will be required to cope with it e.g., if the system operates in a very dynamic environment. The types of dynamic changes in the system can be predicted in terms of their types but not when they will occur. In general, the more dynamic environment, the more unpredictable changes. In contrast, if there is a static external environment the system will continuously try to find a better solution for a given problem and data.

Mechanisms to Cope with Changes in INFER system

The mechanisms describing the reaction of the system to the changes can vary. thus for all dimensions presented in [15] there is no one concrete value in the case of INFER system. The adaptation can be related to both the parameters and the whole components of the system. In general it should be an autonomous process but it can also be assisted by the end user or be a mixture of both approaches. In system design the adaptation will be centralised however in the process of system development it may turn out that some decentralised mechanisms will also be needed. Also the scope of adaptation can vary from a local component to several components or even the whole system. Depending on the application area and the complexity of the models the time of adaptation can be short, medium or long. Regarding the timeliness, in the case when the predictive model does not deliver accurate enough predictions then the time for adaptation must be guaranteed. Finally, the adaptation is event-triggered. It means that it is done when the predictions are not accurate enough or there are free resources to improve working models. Thus the range of adaptation is very wide and different adaptation mechanisms can have different scope, duration and can be organised and administered in a variety of ways.

The effects of adaptation on the INFER system

The effects of adaptation will vary depending on the application area and also the type of adaptation that has been done. Because there are a lot of possible mechanisms that can be applied and various industries in which the system can work, one cannot predict all effects of the various adaptations.

The important thing to emphasize is that the overhead of adaptation must be insignificant and it means that the system must be able to deliver predictions all the time. There cannot be situation in which the system does not produce the output required or demanded by the user.

The elements of risk in a project

Risk as a factor in a software project is considered important particularly with respect to the management of the project and to allow for mitigation against project failure. Barki [20] defines software development risk as the project uncertainty multiplied by the magnitude of the potential loss due to project failure. Wallace and others identify 27 software project risks clustered into 6 dimensions which are risk factors and will affect project performance. One of the dimensions is requirements and they identify continually changing requirements, unidentified, unclear and incorrect requirements as risk [21]. The use of new technology is also identified as a risk along with a high level of technical complexity, immature technology and technology that has not been used in other projects. Han and Huang [22] use the same risk factors and show that the requirements phase in the 115 projects they examined is the most important factor across all projects and they show that of the ten most important software risk factors, four of the top five relate to requirements and number eight relates to the use of new technology.

Zowghi [6] argues that the requirements process when carried out as part of a global software development will need to be different in order to handle the issues of a distributed team. These include such things as cultural differences, communications, coordination and control and time difference. Requirements elicitation is a highly interactive process and can prove more difficult when the ‘problem owing and problem solving communities’ have distance between them [6, 23]. This is highlighted in Bhat, Gupta and Murthy’s case study [24] involving real projects when conflicting requirements approaches by teams at a distance caused problems. Issues that could be resolved by face-to face meetings were not resolved at a distance. Hanisch and Corbett [25] agree and highlight that the use of structured requirements processes by distributed teams may affect the social aspects of requirements elicitation and this may cause misinterpretation and miscommunication of the requirements. There are also cultural issues that affect virtual teams because a shared vision of the requirements and the project objectives are required. These issues are based on a variety of factors such as hidden meanings and interpretation of requirements caused by different cultural backgrounds of team members [25].

The requirements process for a complex, predictive and adaptive system

There is general agreement that there needs to be a new way of handling the requirements process for complex, predictive and adaptive systems [15, 23, 26]. Shaw argues that the system ‘health’ is difficult to predict when the designer has to deal with requirements from the user that are incomplete and the dynamics of the system with emergent factors are such, that future behaviour is not known [26]. The CAS may be termed a ‘large-scale’ system in terms of its complexity or the number of components and requirements for it will come from varying sources and be at a variety of different abstractions [23]. CAS’s that are particularly difficult are those systems such as the INFER system that are reliant on the physical environment. There are issues involving scope and boundaries, assigning responsibilities to peer software systems and components, hardware interfaces and human operators [23]. The integration of all these elements is the weakest area of current RE knowledge and practice [23].

The INFER project is thus about the design and building of a CAS by a distributed team and has a number of high risk elements to it. There is no clear requirements process for this type of system certainly none known to industry at this time, the technology is new and complex, it has requirements attributes that are unknown and behaviour which cannot be predicted. The development process the team could follow is also unknown for this type of system and finally, the development team is distributed.

Conclusions and Further work

Complex Adaptive Systems such as INFER are a new type of system that consists of a number of interacting agents whose emergent behaviour cannot be described as the sum of the entities that make up the system. Behaviour of the individual and the system as a whole will change as the system performs. Some of the attributes of a CAS are emergent behaviour, self organising, co-evolution, continual evolution, sub-optimal, diversity, connectivity and systems of systems. This means that a CAS is constantly changing, is often complex and large scale and reacts to both its environment and human users often in ways that cannot be predicted.

The traditional requirements and development processes are not able to cope with systems such as CAS's and we have shown that there is a very high element of risk attached to the building of a system like INFER. This has most of the previously discussed attributes and there will be rich data available from the requirements process which has been completed and a retrospective case study is underway.

There is considerable further work to be done in this area. The requirements process needs considerable exploration, in addition the way that development is done and the process it follows is also interesting. Finally the elements of risk that are attached to building this type of system will need further work as this type of system becomes more common place.

Acknowledgements. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 251617.

Literature

1. Boehm, B.: Requirements that Handle IKIWISI, COTS and Rapid Change. IEEE Computer 33 (2000)
2. Nuseibeh, B., Easterbrook, S.: Requirements Engineering: A roadmap. International Conference on Software Engineering, Limerick, Ireland (2000)
3. Jackson, M.: Problem Frames: Analyzing and structuring software development problems. Addison Wesley/ ACM Press, Harlow (2001)
4. Fagan, M.E.: Design and code inspections to reduce errors in program development. IBM Systems Journal 15 (1976)
5. Boehm, B., Turner, R.: Balancing Agility and Discipline. Addison-Wesley, Boston (2004)
6. Zowghi, D.: Does Global Software Development Need a Different Requirements Engineering Process? : International Workshop on Global Software Development - ICSE 2002, Orlando, Florida (2002)
7. Humphrey, W.: Characterizing the software process: a maturity framework. IEEE Software 5 (1988) 73-79
8. Boehm, B.: A Spiral Model of Software Development and Enhancement. Computer 21 (1988) 61-72
9. Avison, D.E., Fitzgerald, G.: Where Now for Development Methodologies? Communications of the ACM 46 (2003) 79-82
10. Cohn, M.: User Stories Applied: For Agile Software Development. Pearson Education, Boston (2004)
11. Brownlee, J.: Complex Adaptive Systems. Complex Intelligent Systems Laboratory, Swinburne University of Technology, , Melbourne, Australia. (2007)
12. Holland, J.H.: Hidden Order: How Adaptation Builds Complexity. Basic Books, New York (1996)
13. Levin, S.A.: Ecosystems and the Biosphere as Complex Adaptive Systems. Ecosystems, Biomedical and Life Sciences and Earth and Environmental Science 1 (1998) 431-436
14. Arthur, W.B.: Introduction: Process and Emergence in the Economy in the Economy. In: Arthur, W.B., Durlauf, S., Lane, D.A. (eds.): The Economy as an Evolving Complex System II. Addison-Wesley Pub. Co, Reading, Mass, USA (1997) 1-4
15. Cheng, B.H.C., Lemos, R., de, Holger, G., Inverardi, P., Magee, J.: Software Engineering for Self-Adaptive Systems: A Research Roadmap. In: Cheng, B.H.C., Lemos, R., de, Holger, G., Inverardi, P., Magee, J. (eds.): Software Engineering for Self-Adaptive Systems LNCS 5525 Springer-Verlag, Heidelberg (2009) pp. 1-26
16. Dimensions for INFER; Available at: http://sjeary.co.uk/eurospi.html
17. Salehie, M., Tahvildari, L.: Self-adaptation Software: Landscape and Research Challenges. ACM Transactions on Autonomous and Adaptive Systems 4 (2009.)
18. Laddaga, R., Robertson, P., Shrobe, H.: Introduction to Self-adaptive Software: Applications. In: Laddaga, R., Robertson, P., Shrobe, H. (eds.): Self-adaptive Software: Applications. Springer-Verlag, Heidelberg (2003)
19. Oreizy, P., Gorlick, M.M., Taylor, R.N., Heimbigner, D., Johnson, G., Medvidovic, N., Quilici, A., Rosenblum, D.S., Wolf, A.L.: An architecture-based approach to self-adaptive software. IEEE Intelligent Systems 14 (1999) 54-62
20. Barki, H., Rivard, S., Talbot, J.: Toward an Assessment of Software Development Risk. Journal of Management Information Systems 10 (1993) 203-225
21. Wallace, L., Keil, M., Rai, A.: How Software Project Risk Affects Project Performance: An Investigation of the Dimensions of Risk and an Exploratory Model. Decision Sciences 35 (2004) 289-321
22. Han, W.-M., Huang, S.-J.: An empirical analysis of risk components and performance on software projects. The Journal of Systems and Software 80 (2007) 42-50
23. Cheng, B.H.C., Atlee, J.M.: Research Directions in Requirements Engineering. Future of Software Engineering (FOSE 07). IEEE, Minneapolis, MN, USA. (2007)
24. Bhat, J.M., Gupta, M., Murthy, S.N.: Overcoming Requirements Engineering Challenges: Lessons from Offshore Outsourcing. IEEE Software 23 (2006) 38-44
25. Hanisch, J., Corbitt, B.J.: Requirements Engineering during Global Software Development: Some Impediments to the Requirements Engineering Process: A Case Study. 12th European Conf. Information Systems Turku, Finland (2004)
26. Shaw, M.: "Self-Healing": Softening Precision to Avoid Brittleness. First Workshop on Self-Healing Systems (WOSS02), Charleston, South Carolina, USA (2002)

Author CVs

Sheridan Jeary

Sheridan Jeary is a Senior Lecturer in Software Entrepreneurship at Bournemouth University and has research interests that span Software Development Methods and processes, Web Systems and business processes. Her PhD is in the area of requirements and web development methods.

She had several years of management and systems experience across a variety of domains before moving into academia. She was the BU Project Manager for the EU Commission funded VIDE project on Model driven development and has just completed a 3 month secondment with REC in Wroclaw, Poland as part of the INFER project.

Katarzyna Musial

Katarzyna Musial was born in 1982 in Poland. She received her M.Sc. degree in Computer Science from the Wroclaw University of Technology, Poland in 2006. In the same year she received her second MSc degree in Software Engineering from the Blekinge Institute of Technology, Sweden. She obtained Ph.D. in 2009 from the Institute of Informatics, Wroclaw University of Technology, Poland. She is a Lecturer in Informatics at the Bournemouth University, UK.

Katarzyna Musial focused her Ph.D. thesis on the calculation of individual's social position in the virtual social network. She is interested especially in complex social networks and dynamics and evolution of complex networked systems. Katarzyna Musial is a BU Transfer of Knowledge Coordinator in INFER project She is one of the founder members of the 'Social Network Group' at WUT (SNG@WUT) established in October 2006.

Keith Phalp

Keith Phalp originally read for a first degree in Mathematics, which he then taught for a few years, before completing a Masters in Software Engineering, followed by Ph.D. in Software Process Modelling. He then spent three years as a post-doctoral research fellow at the University of Southampton, again in the area of process modelling. In 1997, Dr. Phalp took up a lectureship at Bournemouth University, and has been there ever since in a variety of roles. He is currently an Associate Dean within the School of Design, Engineering & Computing and Head of Computing and Informatics.