FIRST – Project Number: 734599

H2020-MSC-RISE-2016 Ref. 6742023

D1.1 Overview of Manufacturing Assets and Services Classification and Ontology

Lead Editor: Lai Xu and Paul de Vrieze

With contributions from:

Reviewer: []

Deliverable nature	Report
Dissemination level	Public
Contractual delivery date	
Actual delivery date	
Version	
Keywords	
Revision History	5/10/2017 BU Structure of D1.1 17/10/2017 RuG Section 3 of D1.1 25/10/2017 UniMore Section 4 of D1.1 27/20/2017 RuG the updated version of Section 3 of D1.1

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 734599
H2020-MSC-RISE-2016 Ref. 6742023

Table of Contents

Executive Summary 3
1 Introduction 4
  1.1 Purpose and Scope 4
  1.2 Deliverable Structure 4
  1.3 Methodology 4
  1.4 Terminology 4
  1.5 The Relationship between the Digital and Smart Factory Concepts 6
  1.6 Relationship between the Digital and Virtual Factory concepts 8
2 Relevant Technologies, Standards and Frameworks 11
  2.1 STEP (Standard for the Exchange of Product Model Data) ISO 10303 11
  2.2 Open Services for Lifecycle Collaboration (OSLC) 11
  2.3 Reference Architecture Model for Industry (RAMI) 4.0 11
  2.4 Semantics for Product Life-cycle Management (PLM) Repositories 12
  2.5 Ontology Mediation for Collaboration of PLM with Product Service Systems (PSS) 12
  2.6 Interoperability of Product Lifecycle Management 13
3 Manufacturing Asset Description Languages 15
  3.1 Electronic Device Description Language (EDDI) 15
    3.1.1 History of EDDL 15
    3.1.2 EDDL Characteristics 16
    3.1.3 EDDL Distributions 16
    3.1.4 Electronic Device Description Interpreter 17
    3.1.5 Content and Structure EDD document 17
  3.2 Field Device Tool/Device Type Manager (FDT/DTM) 18
  3.3 Field Device Integration (FDI) 18
    3.3.1 FDI Technology 19
    3.3.2 FDI Architecture 20
    3.3.3 FDI and Existing Solutions 22
    3.3.4 Increasing Interoperability 24
    3.3.5 The Benefits 25
    3.3.6 FDI and Industry 4.0 25
4 Manufacturing Assets/Services Classification 28
5 Manufacturing Assets/Services Discovery Methods 31
  5.1 General purpose service discovery approaches 32
  5.2 Semantics for service discovery 35
6 Conclusions 37
7 References 38

Executive Summary

The FIRST (vF Interoperation suppoRting buSiness innovation) project aims to provide new technology and methodologies to describe manufacturing assets; to compose and integrate existing services into collaborative virtual manufacturing processes; and to deal with evolution of changes. This deliverable describes the state of the art as well as the progress that can be envisaged beyond the state of the art, thus creating a baseline for research, in relation to two major working packages WP2 and WP3.

In Section 1, this deliverable includes clarification of some basic concepts and relationships among basic concepts. In Section 2, the deliverable surveys a number of technologies, standards, and frameworks that are relevant from a theoretical or technical perspective for activities.

In Section 3, deliverable reports on manufacturing asset description languages. In relation to WP2, the deliverable focused on the envisaged progress in Electronic Device Description Language (EDDL), Field Device Tool/Device Type Manager (FDT/DTM), and Field Device Integration (FDI).

Manufacturing assets/services classifications are reviewed in Section 4, which is related to WP2. The deliverable reviewed different manufacturing asset and service descriptions, representations and classifications from Quantum Lifecycle Management (QLM), eCl@ss, UNSPSC (United Nations Standard Products and Services Code), and MSDL (Manufacturing Service Description Language) as well as applied industrial sectors of each description and classification are introduced.

Manufacturing assets and services discovery methods are discussed in Section 5. Main elements of a service discovery framework are reviewed; general purpose service discovery approaches are presented with mentioned some commercial service discovery platforms. Ontology-based service discovery are considered.

The overview of manufacturing assets and service classification and ontology provides blueprint for our further research on WP2 and WP3. The initial research in this deliverable is going to be refined in a later release of this deliverable, following the results of WP2 and WP3.

1 Introduction

1.1 Purpose and Scope

The main purpose of this deliverable is to set the common grounds for the research to be conducted in FIRST to achieve interoperability of virtual factories. It describes several related technologies such as semantic integration among product lifecycle management and product service systems, storage and analyse capabilities of product lifecycle data, collaborative mechanism, visualizing and edit the aggregated information and data in product life cycle, etc.

1.2 Deliverable Structure

In order to set the baseline for the research carried out in FIRST, manufacturing concepts, terminologies and relations among different concepts are reviewed in Section 1 as well as a set of technologies and existing approaches have to be taken into account in Section 2. Manufacturing asset description languages are reviewed in Section 3. In this deliverable, we describe the fundamental areas and technologies relevant to manufacturing assets/services classification, the area of ontology of manufacturing product design lifecycles and product service systems in Section 4. Manufacturing assets and services classification and discovery methods are specified in Section 5.

1.3 Methodology

The information in this deliverable has been provided by the FIRST workpackage leaders. These inputs are based on literature studies, their knowledge and expertise, as well as that of other WP contributors on the past and on-going projects in the area. When needed and within reason, however, these inputs have been edited by the deliverable editor for the sake of improved readability of the overall deliverable.

1.4 Terminology

The *Factory of the Future* initiative prompts the use of new technologies such as Internet of Things (IOT) and Virtual Reality (VR), to integrate and connect the manufacturer's processors in a way which was not previously possible. Using these technologies allows a manufacturing organisation to improve its productivity, cost per item and efficiency adhering to customer order trends.

One of the core initiatives for a factory of the future was introduced by Germany in their Industry 4.0 programme. A report from Ron Davies, put forward to the EU parliament on Industry 4.0 (Davies, 2015), provides more insight into the German initiative. This provides awareness into how a factory of the future will change businesses and lists the challenges organisations will face, especially SME's.

Smart Factories introduce several new concepts, including the Internet of Things technology and Cyber Physical Systems. The Smart factory level focuses on the hardware layer of a factory floor, through this it enables the sending out of its collected data which is then utilised by the Digital factory level. While it makes sense to do so, a Smart Factory does not imply deeper horizontal integration of the smart manufacturing process.

A *Digital factory* utilises the capability to quantify large amounts of data, this data is typically received from the smart factory level hardware (Bracht & Masurat, 2005). A digital factory this data is then used for a variety of purposes: based on exploiting data from the floor for various purposes: decision-making processes; on creating simulations of designed prototypes for speeding the process of going to market; on connecting people and hardware from the smart level manufacturing and outside partners within the virtual level (EFFRA, 2013).

From the perspective of the European Factories of the Future Research Association (EFFRA, 2013), digital factories address the front-end stages of manufacturing, in particular, early concept modelling, simulation, and evaluation (EFFRA, 2013). As well as the front-end stages, they also transform the knowledge time curve to ensure there is a greater acquisition of knowledge earlier, to allow better-informed manufacturing decisions to be made. The report (EFFRA, 2013) also states that in terms of ICT, digital factories focus on providing a better understanding and design of production and manufacturing systems, thus improving the product lifecycle management through the use of simulation, modelling, and knowledge management from the product conception level down to the manufacturing, maintenance, and disassembly/recycling.

The Industry 4.0 Working Group (Kagermann, Wahlster, & Helbig, 2013) states that a digital factory platform must connect people, objects, and systems together and must possess the following features:

- Flexibility provided by rapid and simple orchestration of services and applications, including Cyber Physical System-based software
- Simple allocation and deployment of business processes along the lines of the App Stores model
- Comprehensive, secure, and reliable backup of the entire business process
- Safety, security, and reliability for everything from sensors to user interfaces
- Support for mobile end devices
- Support for collaborative manufacturing, service, analysis, and forecasting processes in business networks.

Kuhn (2006) argues that the digital factory concept also requires the integration of tools for design, engineering, planning, simulation, communication, and control on all planning and manufacturing levels. He suggests that openness and interoperability are the key factors in implementing a digital factory concept. (Zäh & Reinhart , 2004) supports this view by explaining that a digital factory is an approach towards using common data for all applications. This suggests that the core of a digital factory is ubiquitous information and data that can be accessed by all entities within the digital factory network. Zäh & Reinhart further suggests that this common data will enable collaboration with virtual models for different purposes at different levels of detail.

Gregor and Stefan (2010) state that a very important property of a Digital Factory is the ability to integrate process planning and product development using common data. They further suggest that it is important to gain all of the required data once and manage it with uniform data control to allow all software systems to utilise it effectively. Overall they imply that the digital factory is a link between the 'what', product development (CAD), and the 'when and who', process planning (ERP), with the use of common data to provide the 'how' within the product lifecycle.

Zadeh, Lindberg, El-Khoury, and Sivard (2017) propose that NoSQL databases are best suited for the collection of the real-time data that is produced from the interconnected sensors and machines within the factory. They state that this is due to their scalability and their capability of doing millions of data transactions a second. They further suggest that the main difference between the design of an IT system that supports a digital factory and other software development projects is that a software development project will generally start from a clean slate, whereas a digital factory framework involves selection, configuration, and adaptation of a reasonably high-level commercial application. From this, it can be said that a digital factory framework is a means to bridge the gap between existing software systems to create a homogenous, more intelligent factory setting.

The report on the project Future Internet Technologies for MANufacturing (FITMAN, 2013) sums up the concept of a digital factory to be an efficient and comprehensive environment, able to support and optimise the design, modelling, simulation, and evaluation of products, processes, and systems before a new factory is built or any modifications are made to existing layouts, thus improving the quality and time to market of products. Additionally, the digital factory framework covers the life-cycle management of products from the design phase all the way through to the production, maintenance and disassembly and recycling.

The concept of *virtual factories* is a major expansion upon virtual enterprises in the context of manufacturing. The virtual organization approach only integrates collaborative business processes from different enterprises to simulate, model and test different design options to evaluate performance, thus to save time-to-production (Debevec, Simic, & Herakovic, 2014). In contrast, creating virtual factories requires the integration of product design processes, manufacturing processes, and general collaborative business processes across factories and enterprises. An important aspect of this integration is to ensure straightforward compatibility between the machines, products, processes, related products and services, as well as any descriptions of those. It also requires that the nature of the manufacturing processes is sufficiently well understood (and modelled) to support the efficient integration (aspects such as reconfiguration, maintenance, or warm-up time can be significant factors).

A Virtual Factory consist of a multi-layered integration of the information related to various activities along the factory and product lifecycle manufacturing related resources (Chungoora, et al., 2013) as well as real and virtual worlds. With support of cyber-physical systems, smart electronics, sensors, robots, and embedded systems (Da Xu, He, & Li, 2014; Monostori, 2014) data is constantly gathered to enable context-aware enterprise management to extensively support and speed-up the decision process (Davis, Edgar, Porter, Bernaden, & Sarli, 2012) not only in the design stage but also for management decisions (Bi, Da Xu, & Wang, 2014).

From a different perspective a virtual factory not only provides an integrated platform for manufacturing systems design and analysis (Tolio, Sacco, Terkaj, & Urgo, 2013), but it can also be an integrated into simulation model of the major subsystems in a factory. The model considers the factory as a whole and provides an advanced decision support capability (Sanjay, Ngai Fong, Khin Maung, & Ming, 2001). To avoid terminological confusion we will call this a *virtualised factory*.

Extending this virtualised factory concept, a virtual factory can be seen as utilising a new piece of technology in the form of Virtual Reality. Using this technology, organisations can create 3D interface models with their designed product, to have a better view of the intended product without the need for a prototype in its design process (Hao & Helo, 2017). We will call this a 3D or visually virtualised factory.

According to (EU, 2013), the *virtual factory* focuses on what the IT platforms, which can be considered as the hardware, system architectures and software necessary to undertake a range of related tasks. Typically these platform projects are focused on specific interests such as: manufacturing information and knowledge management; supply chain configuration; creation of virtual factories assembled from multiple independent factories; and collaborative engineering.

1.5 The Relationship between the Digital and Smart Factory Concepts

Hermann, Pentek, and Otto (2016) define a smart factory as a factory that is context-aware and assists people and machines in the execution of their tasks. They are able to do this by gathering and using information from the physical and virtual world. Information such as the position of a tool to electronic drawings of the product it is producing or of the tool itself. The smart factory is context-aware in the sense that it uses CPS to communicate over the IoT. This higher level of smartness is however still localised. A smart factory is integrated with a digital factory by enhancing the virtual models created within the digital factory with real time data to improve decision making processes as shown in Figure 1.

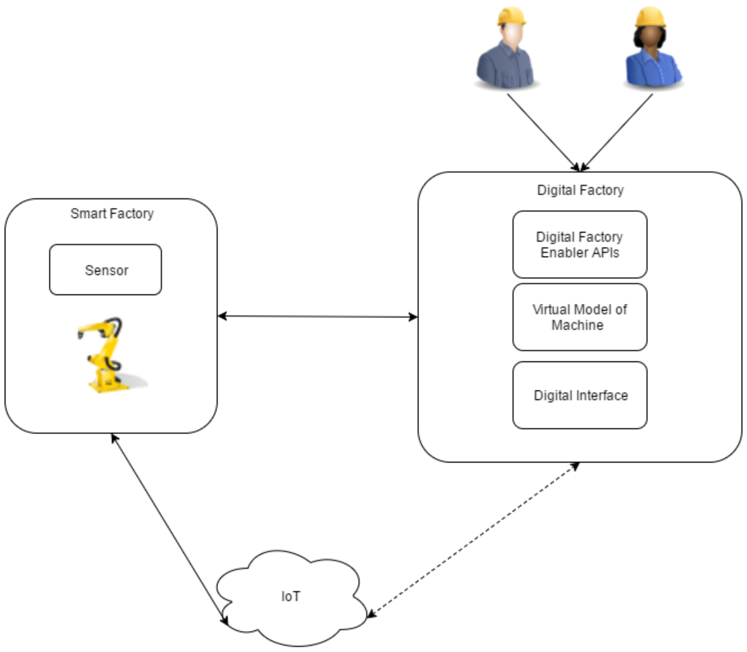


Figure 1: Smart and Digital Factory Integration

There is no clear line as to where the transition from a smart factory into a digital factory occurs. This separation can be very situational with respect to the design of the framework. IoT devices are an integral part of a smart factory's architecture and enable the connection between a digital and smart factory to exist. Yet the management of these devices can occur in either frameworks. Additionally, there is ambiguity over where the management of the data created by the IoT devices can be situated.

A smart factory can provide both the device management and data management within its domain. The FIWARE for Industrial IoT Reference Architecture, shown in Figure 2, demonstrates this by placing the device management at a lower level of the smart factory architecture and the data management at a high level. This can be beneficial for a factory that wants to focus on the fundamental behaviour of their shop floor. The devices and data they produce are able to be monitored and acted upon without the need to look at the product lifecycle as a whole.

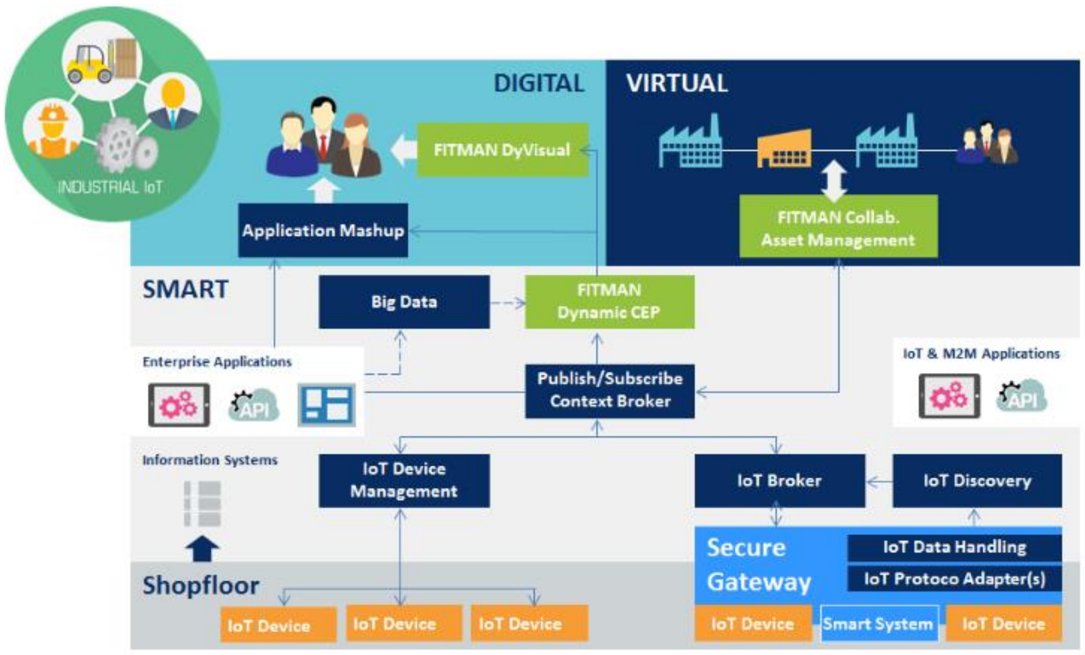


Figure 2: Industrial IoT Reference Architecture (FIWARE, 2015)

Additionally, a Digital factory can provide device and data management within its domain. The BeInCPPS reference architecture (BeinCPPS, 2016), as shown in Figure 3, demonstrates this by placing the field device management within the factory block, which can be translated to the digital factory domain. The architecture also shows that data management happens within the digital factory domain, through big data and event processing. The device management within the digital factory will appear at a lower level of the architecture whereas the data management will appear at a much higher level. Placing these within the digital factory domain allows an enterprise to define better knowledge-oriented decision making processes that impact the whole product lifecycle. Virtual models of the factory can be enhanced with the data collected by IoT sensors to assist in the decision making process of upgrading the shop floor to improve a current product or make room for a new product.

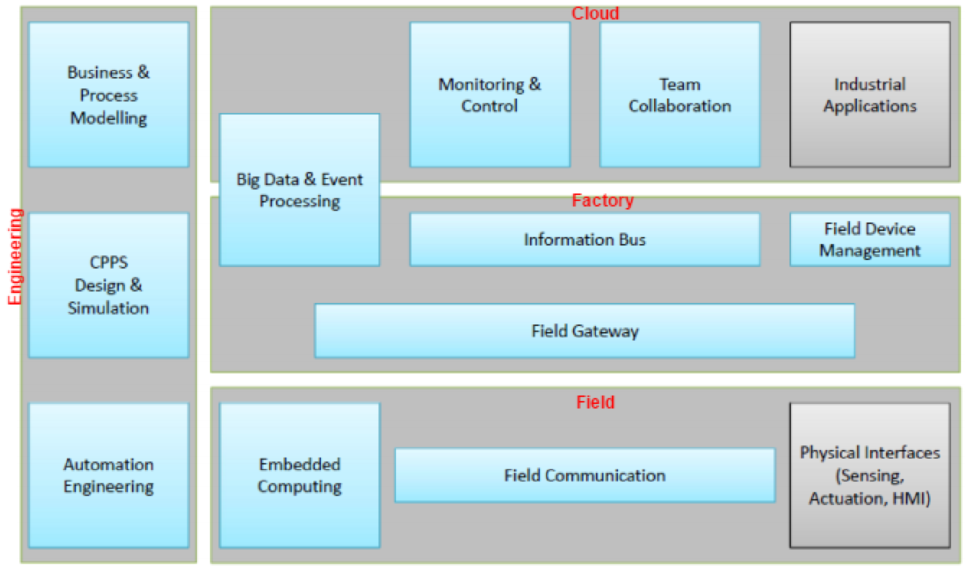


Figure 3: BeInCPPS Block Reference Architecture (BeinCPPS, 2016)

In conclusion, there are two unique areas where there is overlap between the smart and digital factory domains. The first area identified is the management of IoT devices; it was observed that it can function within the low level of a smart factory or the low level of a digital factory. The second area identified is the management of data created by the IoT devices within the smart factory. It was observed that it can function within the high level of a smart factory or a high level of a digital factory. The observations will be important requirements to include in the creation of the artefact.

1.6 Relationship between the Digital and Virtual Factory concepts

The FITMAN project has done extensive work in the area of digital and virtual factories. Within this project, the explored solutions (FITMAN, VOLKSWAGEN Trial – PLM Ramp up reducing Time to Market, 2015) are integrated from product inception to high-volume manufacturing across the factory and also the company boundaries. Within FITMAN, the aim of a virtual factory is to support the integration and exchange of data and physical assets through global networked operations to gain clear and exact useful knowledge while enabling and supporting the decision making process. Thus, from this perspective, the virtual factory deals with the collaboration of design, production process, and the extended supply chain.

Like for smart versus digital factories, there is a similar ambiguity with respect to where a digital factory stops and where a virtual factory begins. This can lead to scenarios where both accomplish the same tasks depending on whether virtual or digital factory reference architectures are used. Traditionally, a factory would have a supplier management module located within their supply chain management system (SCM). The SCM is located within the enterprise information system of the supply chain owner. Suppliers linked to the supply chain would have access to the SCM system.

However, within the realm of Industry 4.0 and increased competition for the prices of supplied materials, the SCM can be implemented within the virtual factory architecture through the means of a virtual supplier community. The virtual supplier community allows for more efficient operations and a reduction in administrative cost.

A key aspect of a virtual factory, in this definition, is that of being able to collaborate with other businesses through the use of software solutions expanding outside the company boundaries. The virtual factory offers the opportunity for the business and its suppliers to collaborate on business processes that affect the supply chain.

A lower level view of Figure 3, produced by Industrial IoT Reference Architecture (FIWARE, 2015) shown in Figure 4, allows us to see how the collaboration between the factory and its community of suppliers can occur. The (FITMAN, 2015) generic and specific enablers, for example the FITMAN SCApp which stands for Supply Chain and Business Ecosystem Apps which facilitates the collaboration among supply chain and business ecosystem partners. It allows the factory to easily engage with 3rd party suppliers or partners over a network depending on the business opportunity.

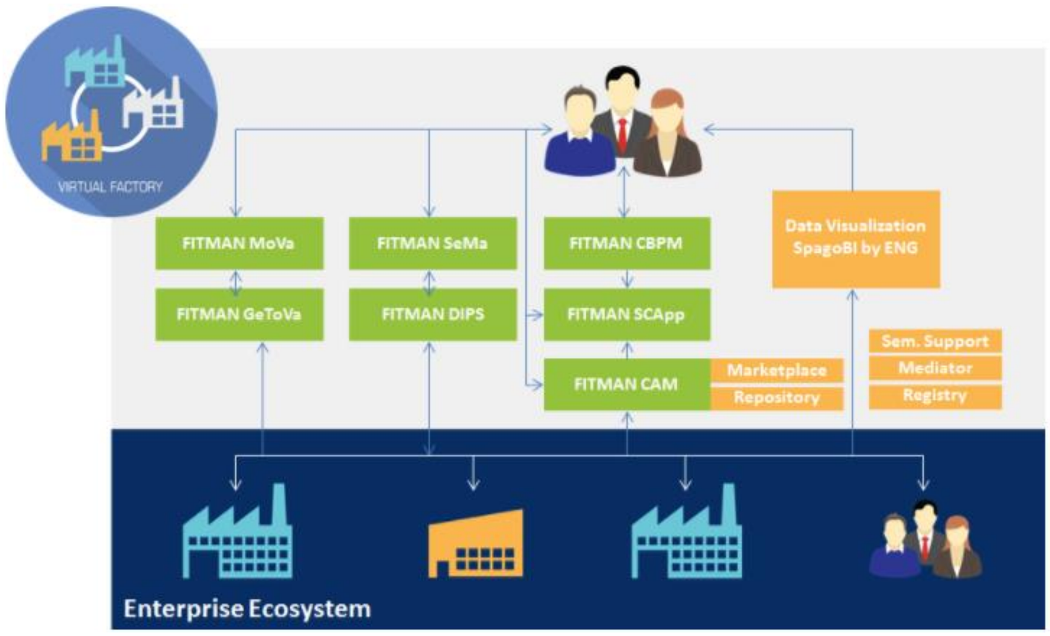


Figure 4: Industrial IoT Virtual Factory Reference Architecture (FIWARE, 2015)

The virtual factory model is suited towards a business that requires many independent suppliers with frequent design changes or product changes, for example, a mobile phone company. They are able to use the community to quickly and easily collaborate with different suppliers for new phones that require new or upgraded hardware.

One notices in Figure 4 that the design and simulation engineering module is located in line with the factory block, which is associated with the digital factory. This means that the design of new products can be kept within the company boundaries. However, it can also be seen that the team collaboration is located within the cloud block, this implies that the design assets can be the object of collaboration by teams within the company and the suppliers. It is important to note that no business processes will be changed by the collaboration, but the product designs are likely to change.

The digital factory is suited for businesses that need long term relationship with their suppliers, typically on a fixed term. A natural example of where this architecture best fits is an automobile manufacturer. Although they may release new products frequently, the lifecycle of the product is much longer compared to the production, and parts will need to be manufactured for many years.

The parts that are required from the supplier are unlikely to change and therefore a network connection between the two businesses is not essential.

Overall, we can see that there is a strong situational overlap between the digital and virtual factory. A virtual factory can provide collaboration between design, production and the supply chain of a product with a networked community of suppliers and 3rd party applications. In contrast, a digital factory provides collaboration between design, production and the supply chain of a product within the company boundaries. It can thus be said that a virtual factory provides collaborative business processes with different partners whereas a digital factory can provide collaborative asset management with different partners. These will be important requirements to consider in the creation of the artefact.

2 Relevant Technologies, Standards and Frameworks

Product lifecycle management is the process of dealing with the creation, modification, and exchange of product information through engineering design and manufacture, to service and disposal of manufactured products. In this section, we review the economic and technical aspects of an interoperation framework for product lifecycle management, related standards, technologies, and projects.

2.1 STEP (Standard for the Exchange of Product Model Data) ISO 10303

ISO 10303, also known as STEP (Standard for the Exchange of Product Model Data), is an international standard for industrial automation systems and integration of product data representation and exchange. It is made up of various parts that offer standards for specific topics. Part 242:2014 refers to the application protocol for managing model-based 3D engineering (ISO 2014). The standard will be essential to implementing a digital factory based model.

2.2 Open Services for Lifecycle Collaboration (OSLC)

Open Services for Lifecycle Collaboration (OSLC) is an open community that creates specifications for the integration of tools, such as lifecycle management tools, to ensure their data and workflows are supported in the end-to-end processes. OSLC is based on the W3C linked data (W3C 2015).

2.3 Reference Architecture Model for Industry (RAMI) 4.0

Reference Architecture Model for Industry 4.0 (RAMI 4.0) defines three dimensions of enterprise system design and introduces the concept of Industry 4.0 components (VDI/VDE GMA, 2015). The RAMI4.0 is essentially focused on the manufacturing process and production facilities; it tries to focus all essential aspects of Industry 4.0. The participants (a field device, a machine, a system, or a whole factory) can be logically classified in the model and relevant Industry 4.0 concepts described and implemented.

The RAMI4.0 3D model includes hierarchy levels, cycle and value stream, and layers. The layers represent the various perspectives from the assets up to the business process, which is most relevant with our existing manufacturing asset/service classification.

Currently RAMI4.0 does not provide detailed, strict indication for standards related to communication or information models. The devices/assets are provided using *Electronic Device Description* (EDD) (also see section 3.1) (Naumann & Riedl, 2011) (formalised using the IEC 61804-3 *Electronic Device Description Language*), which includes the device characteristics specification, the business logic and information defining the user interface elements (UID – User Interface Description).

The optional User Interface Plugin (UIP) that defines programmable components based on the Windows Presentation Foundation specifications, to be used for developing UI able to effectively interact with the device.

The *Functional and Information Layer the Field Device Integration (FDI) (also see section 3.3)* (FDI Cooperation, 2012 ; Fieldcomm Group) specification as integration technology. The FDI is a new specification that aims at overcoming incompatibilities among some manufacturing devices specifications. Essentially the FDI specification defines the format and content of the so-called *FDI package* as a collection of files providing: the device Electronic Device Description (EDD), the optional User Interface Plugin (UIP), and possible optional elements (called attachments) useful to configure, deploy and use the device (e.g. manual, protocol specific files, etc.).

An *FDI package* is therefore an effective mean through which a device manufacturer defines which data, functions and user interface elements are available in/for the device.

2.4 Semantics for Product Life-cycle Management (PLM) Repositories

OWL-DL is one of the sublanguages of OWL[1]. OWL-DL is the part of OWL Full that fits in the Description Logic framework and is known to have decidable reasoning. In building product lifecycle management repositories, OWL-DL is used to extract knowledge from PLM-CAD (i.e CATIA) into the background ontology automatically, other non-standard parts (i.e. not from CATIA V5 catalogue) manually into the background ontology. OntoDMU is used to import standard parts into concepts of the ontology.

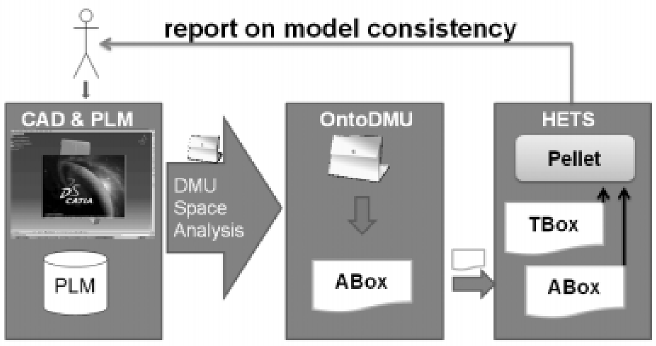


Figure 5. Ontology based on PLM Repositories (Franke M. , Klein, Schröder, & Thoben, 2011)

An ontological knowledge base consists of two parts offering different perspectives on the domain. In Figure 5, the structural information of a domain is characterized through its TBox (the terminology). The TBox consists of a set of inclusions between concepts. The ABox (the assertions) contains knowledge about individuals, e.g. a particular car of a given occurrence of a standard part in a CAD model. It can state either that a given named individual (i.e. 'myCar') belongs to a given concept (e.g., that myCar is, in fact, a car) or that two individuals are related by a given property (e.g. that myCar is owned by me).

2.5 Ontology Mediation for Collaboration of PLM with Product Service Systems (PSS)

The PSYMBIOSYS[2] EU Project addresses collisions of design and manufacturing, product and service, knowledge and sentiments, service-oriented and event-driven architectures, as well as business and innovations. Each lifecycle phase covers specific tasks and generates/requires specific information. Ontology mediation is proposed is proposed as a variant of ontology matching since the level of matching can be rather complex.

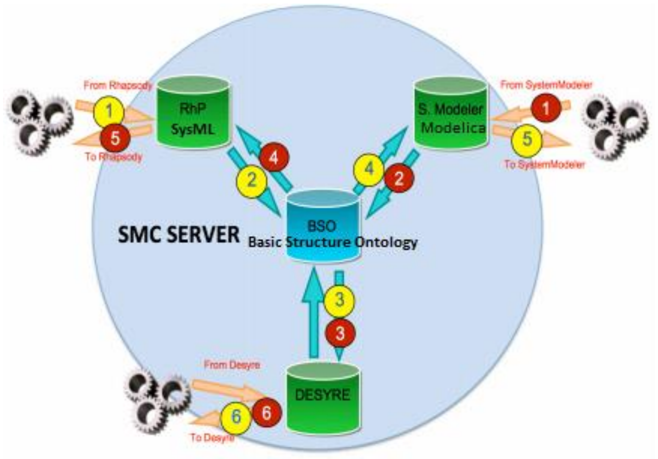


Figure 6. Ontology Mediation

[1] https://www.w3.org/TR/owl-guide/
[2] http://www.psymbiosys.eu/

When matching two different modelling languages, such as Modelica and SysML in Figure 6, the issue of completeness makes the mapping task impossible. The two languages are significant differences and overlaps. Figure above presents a ontology mediation approach, which Basic Structure Ontology (BSO) is at the centre, and the mediation among three different tools was working through three matching sets that connected the common structure ontology which each of the tools: Medelica tool, SysML tool and a 3rd party proprietary tool (Shani, Franke, Hribernik, & K. D. Thoben, 2017).

2.6 Interoperability of Product Lifecycle Management

Integrating among heterogeneous software applications distributed over stakeholders in closed-loop PLM. The capabilities of the Internet of Things are being extended to Cyber-Physical-Systems (CPS), which divide systems into modular and autonomous entities. The systems are able to communicate, to recognize the environment and to make decisions. Different companies with different IT-infrastructures adopt different roles in the product lifecycle.

In order to manage the interoperability of heterogeneous systems throughout the product lifecycle, different approaches could be used (Franke M. , et al., 2014)

- Tightly coupled approaches implement federated schema over the systems to be integrated. A single schema is used to define a combined (federated) data model for all involved data sources (Franke M. , et al., 2014). Any change of the individual system's data models need to be reflected by a corresponding modification of the entire federated schema.
- Object-oriented interoperability approaches are closely related to tightly couple ones. Different types of these approaches are described in (Pitoura, Bukhres, & Elmagarmid, 1995). Object-oriented interoperability approaches use common data models which are a similar problem of dealing with modification of the individual system.
- Loosely coupled interoperability approaches are more suitable to achieving scalable architecture, modular complexity, robust design, supporting outsourcing activities, and integrating third party components. Using Web services for a communication method among different devises, objectives, or databases is one of such loosely coupled interoperability approaches. The semantic meaning of a Web service can be described using OWL (Web Ontology Language). Web services described over third party ontologies (Martin, et al., 2007) are called Semantic Web Services.
- Service Oriented Architecture (SOA) has emerged as the main approach for dealing with the challenge of interoperability of systems in heterogeneous environment (Srinivasan V. , 2011), (Wang & Xu, 2013). SOA offers mechanisms of flexibility and interoperability that allow different technologies to be dynamically integrated, independently of the system's PLM platform in use (Jardim-Goncalves, Grilo, & Steiger-Garcao, 2006). Some of standards for PLM using SOA are: *OMG PLM Services* (PLM Services 2.1 , 2011) and *OASIS PLCS PLM Web Services* (Product Life Cycle support (PLCS) Web services V2).

OMG PLM Services. The current version, PLM Services 2.0 (PLM Services 2.1 , 2011), covers a superset of the STEP PDM Schema entities and exposes them as web services. This specification resulted from a project undertaken by an industrial consortium under the umbrella of the ProSTEP iViP Association. Its information model is derived from the latest ISO 10303-214 STEP model (which now includes engineering change management process) by an EXPRESS-X mapping specification and an EXPRESS-to-XMI mapping process. The functional model is derived from the OMG PDM Enablers V1.3. The specification defines a Platform Specific Model (PSM) applicable to the web services implementation defined by a WSDL specification, with a SOAP binding, and an XML Schema specification. More details on architecting and implementing product information sharing service using the OMG PLM Services can be found in (Srinivasan, Lämmer, & Vettermann, 2008).

OASIS PLCS PLM Web Services. Product Life Cycle Support (PLCS) is the phrase used for the STEP standard ISO 10303-239 (Product Life Cycle support (PLCS) Web services V2) (ISO, 2005)[33]. After the initial STEP standard was issued by ISO, a technical committee was formed in the OASIS organization to develop this further. A set of PLCS web services has been developed by a private company (Eurostep) as part of the European Union funded VIVACE project [34]. Eurostep has put this forward on behalf of VIVACE to the OASIS PLCS committee for consideration as the basis for an OASIS PLCS PLM web services standard.

ISA-95/OAGIS SOA in Manufacturing. ISA-95[3] and OAGi are jointly working on standards for manufacturing systems integration. They are actively looking into the suitability of SOA for such integration in manufacturing.

[3] https://isa-95.com/

3 Manufacturing Asset Description Languages

Nowadays, in automated production plants, field devices come from many diverse manufacturers. Even a small plant may employ several thousand devices of several hundred types from more than 10 vendors for its process automation facilities (Yamamoto & Sakamoto, 2008). In this situation, industrial control software faces enormous challenges in terms of association and interoperability. Ideally all the devices are integrated and provide a unique infrastructure for the production process, while the variety of devices increases the difficulty of device management, interconnection, and maintenance. Open and standardized device integration languages and technologies are there to mitigate the problem. Device integration makes data and functionality of devices available throughout the entire automation system in ways that support association, integration, data exchange, and possibly semantic descriptions. Currently, the most widespread and relevant technologies include Electronic Device Description Language (EDDL), Field Device Tool (FDT)/Device Type Manager (DTM) and Field Device Integration (FDI).

3.1 Electronic Device Description Language (EDDI)

EDDL is a device integration technology that uses an electronic file written in a common language to describe an intelligent device in a machine-readable format. It has been acknowledged as a generic language by IEC as an International standard for Interoperability (IEC 61804-3).

The electronic device descriptions are available for many millions of digital devices that are currently installed in the process industry. It supports parameter handling, operation, and monitoring of automation system components such as remote I/Os, controllers, sensors, and programmable controllers. Four major foundations have endorsed EDDL such as Fieldbus foundation, HART Communication Foundation (now is FieldComm Group), Profibus Nutzerorganisation (PNO), and the OPC foundation (Blevins, 2007).

The use of EDDL will increase as the event of Industrial Internet of Things (IIoT) and Industry4.0 brings more digitally networked devices into plants. Currently about 1800 devices from more than 100 manufacturers are described with EDDL and more than 16 million devices are in use in process industry (Naumann & Riedl, 2011).

3.1.1 History of EDDL

Device Description (DD) technology has been in use since 1992. The first generation of the technology did not include any visual elements or capability for long term storage of data. When the technology was standardized as IEC 61804-2 in March 2004 all three bus organizations adopted the common acronym EDDL, Electronic Device Description Language.

The Device Description standard was unilaterally initiated by the HART Communication Foundation (HCF), Fieldbus Foundation, PROFIBUS Nutzerorganisation e.V. (PNO), and OPC Foundation between 1992 and 2000. In april 2004, The EDDL Cooperation Team (ECT) was formed by those organizations to deliver an integrated standard IEC 61804-2 and later revised as 61804-3 in 2005.

The capabilities of EDDL then were extended to provide an industry standard solution for advanced visualization of intelligent device information closing the gap of early DD for visualization. The visual elements were added without deviating from the original concept. Graphs (stored waveform), charts (continuous trend), grids (tables), and images were added along with better menus and methods. Capability for persistent long term storage of data and retrieval for comparison was also added. EDDL was approved as the international standard IEC 61804-3 in 2006. EDDL is included in the OPC-UA specification.

3.1.2 EDDL Characteristics

As defined in (Electronic Device Description Language, 2017), EDDL has some notable characteristics as follows:

- **Text Based**: EDDL is not software. Instead, it is a text file that is independent of operating system. This brings benefits such as easy to manage and maintain. This makes EDDL secure, robust, and applicable to portable tools which are not PC-based, such as handheld communicators and calibrators. Moreover, it enables EDDL to avoid device version conflicts, makes integration and removal easier, and protects the investment.
- **Independent of communications protocol**: EDDL is applicable to different communication protocols and is independent of the underlying communication hierarchy. This makes it possible to integrate data from HART, WirelessHART, Foundation fieldbus, and PROFIBUS devices in the same tool. EDDL is not software and thus malicious code cannot get embedded. Complex procedures are scripted as text which is interpreted rather than executed.
- **International standard**: The International Electrotechnical Commissions (IEC) has released IEC 61804-3 standard to ensure compatibility among manufacturers.
- **Externally accessible information**: EDDL is not a display program and therefore the device information and the meta-information about the device information is also made available to other applications, for example through OPC-DA or OPC-UA.
- **Full support of device functionality**: EDDL describes procedures such as calibration and complex setup and presents waveform graphics, continuous trend charts, table grids, and images. Persistent storage is provided for test results. All functionality in even the most sophisticated device can thus be made available to the technician.
- **Different Tasks**: EDDL describes function blocks, device parameters, calibration procedures, menu structures, and presentation of diagnostics results and thus applies to control strategy building, commissioning, operation, and maintenance tasks. EDDL handles all life cycle aspects.
- **Fits Best in All Environments**: No limitation in EDDL implementation. EDDL is used from handheld device to Manufacturing Execution System (MES) as well as from simple devices to very complex devices, proving that EDDL is scalable.

3.1.3 EDDL Distributions

The Electronic Device Description file is distributed in either plain text or compressed text format depending on what the chosen software requires as illustrated in Figure 7. In the plain text format, EDDs are interpreted by EDDI (Electronic Device Description Interpreter) in the software whenever the data is used, to render the display or when it is printed, just like a web browser. If it is presented in compressed text, the source EDDL "tokenized" to a compressed format to prevent tampering and subsequent problems. The compilation process also includes checking of EDDL syntax etc. The tokenized files are relatively small and therefore files for many types and versions of devices can be stored also in the limited flash memory of a handheld communicator or in the device itself from where it can be uploaded by the software.

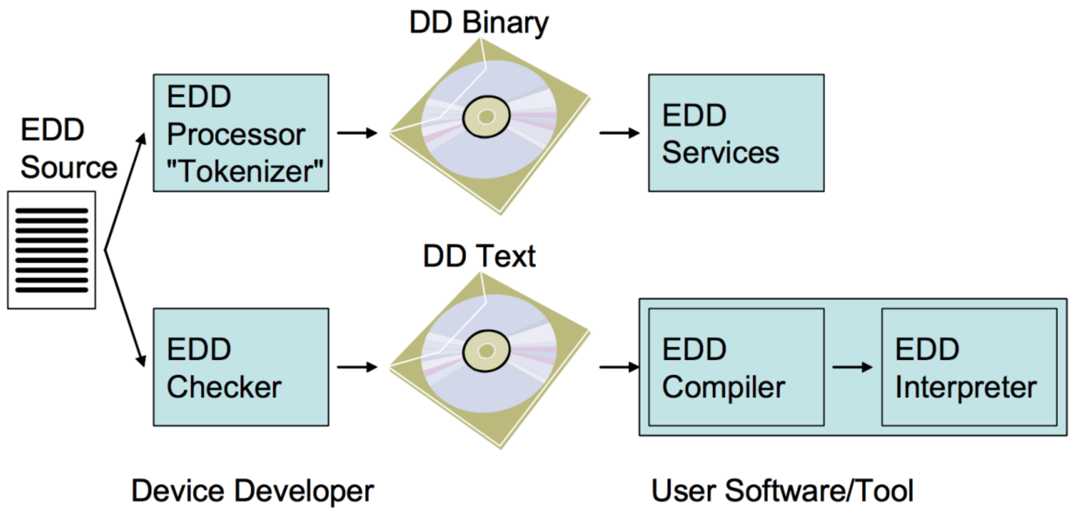


Figure 7: Illustration of EDDL Distributions, adopted from (Naumann & Riedl, 2011)

3.1.4 Electronic Device Description Interpreter

EDDL is based on text files which are interpreted, not executed on the system. Electronic Device Description Interpreter (EDDI) is used to render the display or when it is printed, just like a web browser.

3.1.5 Content and Structure EDD document

In describing device properties, EDDL incorporates a set of language elements. MANUFACTURER and DEVICE_TYPE identify vendor and type of device. VARIABLE is used to describe the parameters of device. COMMAND describes communication mapping and relates the VARIABLE to factual command used to read the variable. MENU is used to organize VARIABLE and METHOD and describe their display structure. The METHOD is used to describe the configuration and diagnosis functions. Such an element can be invoked to diagnose and maintain device. Other elements such as COLLECTION and ARRAY are also available to organize various variables and methods. The sample of data description is shown in Listing 1.

```
#define LINEAR 0
{
  VARIABLE trans1_temperature_unit
  {
    LABEL [digital_units];
    HELP  [temperature_unit_help];
    CLASS CONTAINED;
    HANDLING READ & WRITE;
    TYPE ENUMERATED(2)
  }
  {
    DEFAULT VALUE32;
    {32, [degC], [degC_help]},
    {33, [degF], [degF_help]},
    {34, [degR], [degR_help]},
    {35, [Kelvin], [Kelvin_help]
  }

  IF (trans1_sensor_type = LINEAR)
  {
    {36, [mV], [mV_help]},
    {37, [Ohm], [Ohm]},
    {39, [mA], [mA_help]},
  }
}
```

Listing 1. Sample of Data Description, adopted from (Blevins, 2007)

Listing 1 shows description of a variable `trans1_temperature_unit` of a temperature device. Parameter can be described with label, help text, data type, min and max vales, read/write handling. Data definitions can be used in structures like BLOCK, RECORD, COLLECTION, ARRAY, LIST, FILE, etc. The conditional expression is also allowed to define value ranges, read/write handling dependent of any other parameters.

3.2 Field Device Tool/Device Type Manager (FDT/DTM)

Field Device Tool/Device Type Manager is an alternative technology for device integration. It integrates instrumentation devices with the Distributed Control System (DCS) and asset management system. FDT/DTM comes as a Windows component object model (COM)-based technology (Spiegel; R. Spiegel, 2009). The FDT technology is supported by FDT Group[4] which also handle the certification of DTM vendors and users. As the technology is based on a software running in Windows operating system, it depends on windows upgrades and version control. DCS vendors need to execute FDT software in separate machine than the basic control and database server. In such a machine, several supporting parts are needed, namely FDT Frame Application, CommDTM, Device DTM (EDDL or FDT/DTM: Characteristics of EDDL and FDT/DTM, 2006).

Both EDDL and FDT are targeting communication with field devices. Though there are fundamental differences as shown in Table 1, these are complementary solutions that perform best in their respective areas of application. That is, EDDL is best suited for device data access while FDT/DTM is the recommended platform for advanced asset management applications and efficient Human Machine Interface (WIB Test Confirms Value Of FDT/DTM Technology For Asset Management, 2008). The future of device integration may be a matter of learning how to optimize the technologies side-by-side. FDI (Field Device Integration) might be an answer to integrate these technologies.

3.3 Field Device Integration (FDI)

The Field Device Integration (FDI) (FDI Cooperation, 2012) (Fieldcomm Group) is an integration technology. It is a new specification aiming at overcoming incompatibilities among different manufacturing devices. Essentially, the FDI specification defines the format and content of the so-called FDI package as a collection of files providing: the device Electronic Device Description (EDD), the optional User Interface Plugin (UIP), and possible optional elements useful to configure, deploy and use the device (FDI Cooperation, 2012). An FDI package is therefore an effective mean through which a device manufacturer defines which data, functions and user interface elements are available in/for the device (Neumann, Simon, & Diedrich, 2001; Simon, Diedrich, Riedl, & Thron, 2001).

Device integration enables functions and information from devices to be accessible at a higher level (Neumann, Simon, & Diedrich, 2001). The viable integration of devices requires multiprotocol, standards. The device information should be available across different manufacturers. The current solutions, which are EDDL (Electronic Device Description Language) in various formats and FDT (Field Device Technology), have their strengths and weaknesses

[4] https://fdtgroup.org/

(Simon, Diedrich, Riedl, & Thron, 2001). But they overlap to a large extent. This causes extra expense for users, manufacturers and in standardization.

With FDI, a technology has been developed that combines the advantages of FDT with those of EDDL in a single, scalable solution. FDI takes account of the various tasks over the entire lifecycle for both simple and the most complex devices, including configuration, commissioning, diagnosis and calibration.

Globally leading control system and device manufacturers, such as ABB, Emerson, Endress+Hauser, Honeywell, Invensys, Siemens and Yokogawa, along with the major associations FDT Group, Fieldbus Foundation, HART Communication Foundation, OPC Foundation, PROFIBUS & PROFINET International, are supporting and driving forward the development of the FDI Technology together (FDI Cooperation, 2012).

3.3.1 FDI Technology

The scalable FDI Package is the core of FDI Technology. This package is a collection of files in Figure 8: the Electronic Device Description (EDD), the device definition (Def), business logic (BL) and user interface description (UID). It is based on Electronic Device Description Language (EDDL, IEC 61804-3). The optional user interface plugin (UIP) offers the flexible user interfaces, like FDT, based on Windows Presentation Foundation (WPF). The device manufacturers define via the FDI Device Package which data, functions and user interfaces are stored on the FDI Server.

The FDI Package adds product documentation, protocol-specific files, such as GSD or CFF, etc. as attachments. FDI defines a single protocol independent encoded file format for the EDD part of the FDI Package.

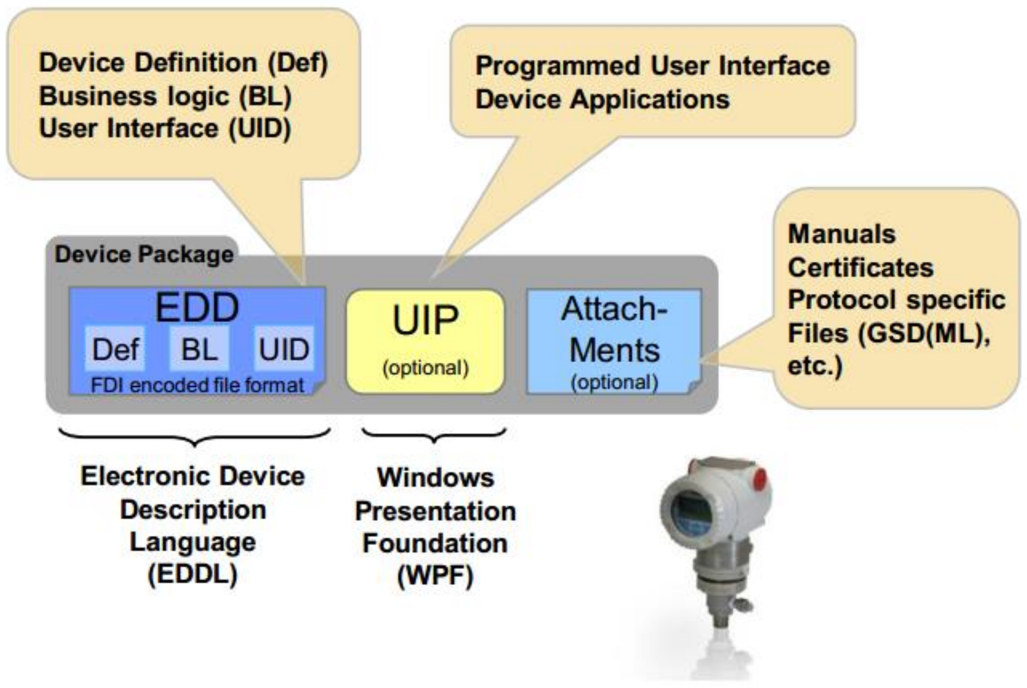


Figure 8: FDI Device Package (FDI Cooperation, 2012)

In the international standard IEC 61804-3, the Electronic Device Description Language (EDDL) is specified as a basis for FDI. Moreover, the standard defines in profiles which constructs from the overall language scope and which library functions are permitted to be used for the HART, Foundation Fieldbus and PROFIBUS protocols. In the FDI technology, EDDL is largely harmonized and standardized across the protocols (FDI Cooperation, 2012). A uniform EDDL is the foundation for uniform multiprotocol FDI Package development tools (FDI IDE), and uniform host components, such as EDD Engine and the client-side components UID Renderer and UIP

Hosting. The result is sustainable strengthening of the key factors interoperability and quality. At the same time, cost savings can be achieved for device and system manufacturers, fieldbus organizations and, last but not least, end users (FDI Cooperation, 2012).

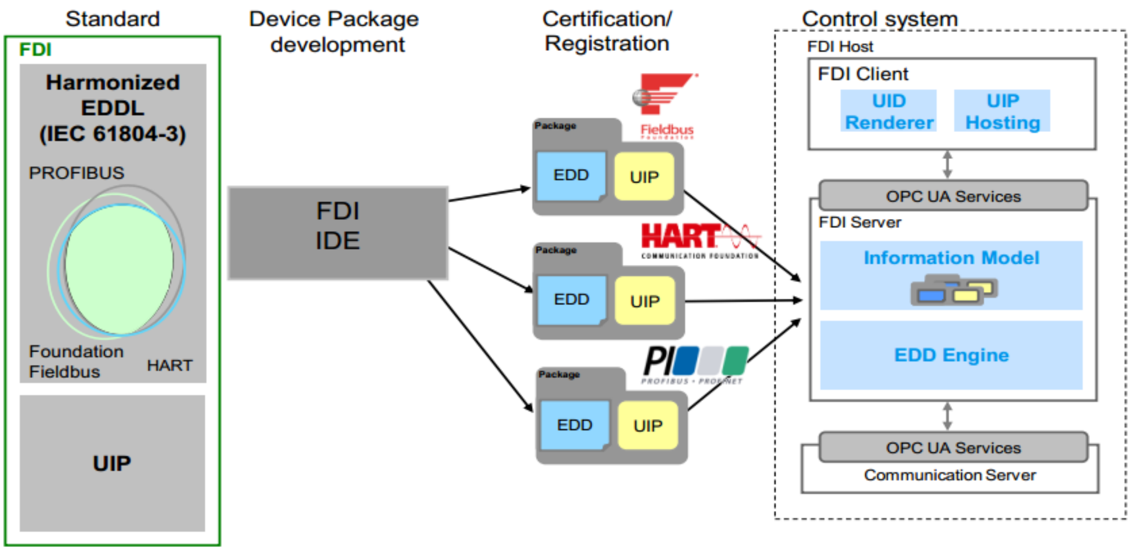


Figure 9 FDI workflow (FDI Cooperation, 2012)

3.3.2 FDI Architecture

The FDI architecture includes different kinds of host implementations. Device management software, a device configuration tool on a laptop or a hand held field communicator could act as a FDI host. A FDI host always supports all features of a FDI Device Package in Figure 10.

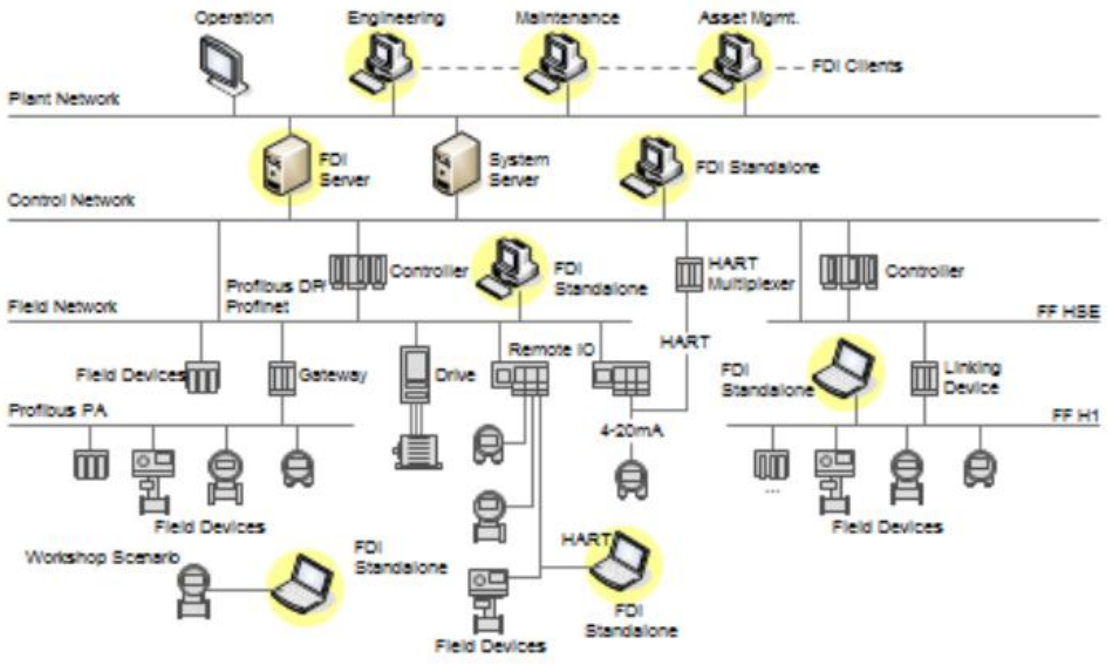


Figure 10 FDI host systems in various applications (FDI Cooperation, 2012)

In a FDI host client-server architecture, a server provides services which are accessed by various distributed or local clients. The FDI architecture is based on the information model used in the OPC Unified Architecture (OPC UA) providing advantages, like platform independence (Grossmann, Bender, & Danzer, 2008). The FDI Server imports FDI Device Packages into its internal device catalogue. Therefore, the version management of FDI Packages is handled centrally within the FDI

Server. Moreover, FDI Packages do not require registration in the sense of a software installation and there are no unpleasant side effects. The representation of device instances in the FDI Server takes place in the information model (Mahnke, Gössling, Graube, & Urbas, 2011). If an FDI Client wishes to work with a device, it accesses the information model and loads the user interface of the device in order to display it on the client side. By interpreting the EDD in its EDD Engine the FDI Server always ensures that the device data remain consistent (Li & Liu, 2011). If OPC UA communication is used between client and server, OPC UA authentication and encryption mechanisms take effect and prevent unauthorized access (Grossmann, Bender, & Danzer, 2008). The FDI hosts do not have to implement the client server architecture. The FDI architecture allows also for the implementation of standalone tools.

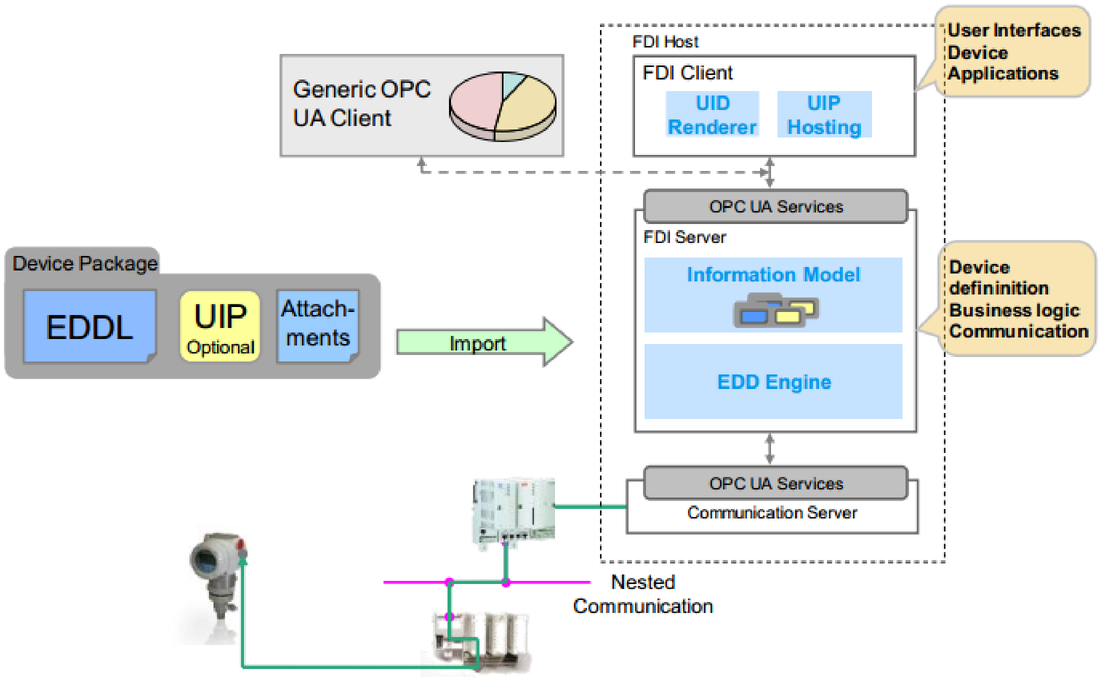


Figure 11: FDI host – client server architecture (FDI Cooperation, 2012)

FDI Packages can run in two system architectures – one purely FDI host and one FDT-based FDI Figure 12 host. Because FDI reused FDT 2.0 interfaces for the FDI UIP. For many FDT Frame manufacturers , the FDT-based architecture opens up an economically attractive migration route to FDI (Gunzert, Lindner, Wesner, & Kato, 2013). The FDT 2.0 Frame is retained without changes and simply supplemented with an FDI DTM. Therefore, the FDT Frame manufacturer will not have to develop the component themselves. The interoperability with FDT allows all system and tool manufacturers to support FDI. Over the longer term, this will lead to reductions in the number of device drivers per device type, and thus to significant savings in product development and maintenance. In the end, the end users benefit from the improved interoperability and smaller range of versions.

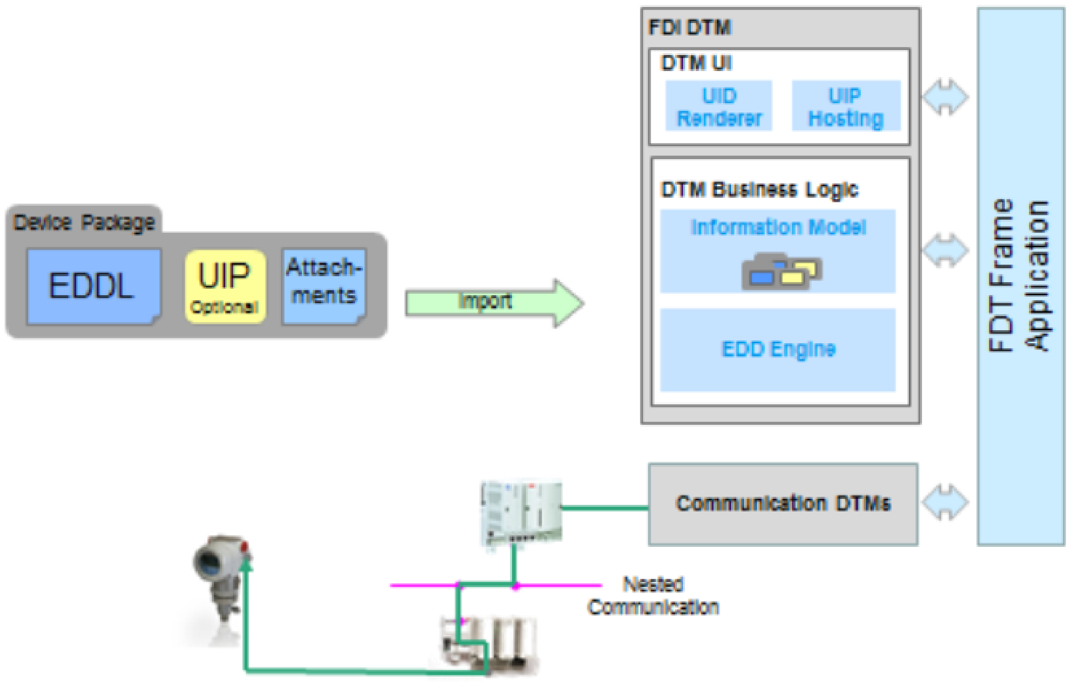
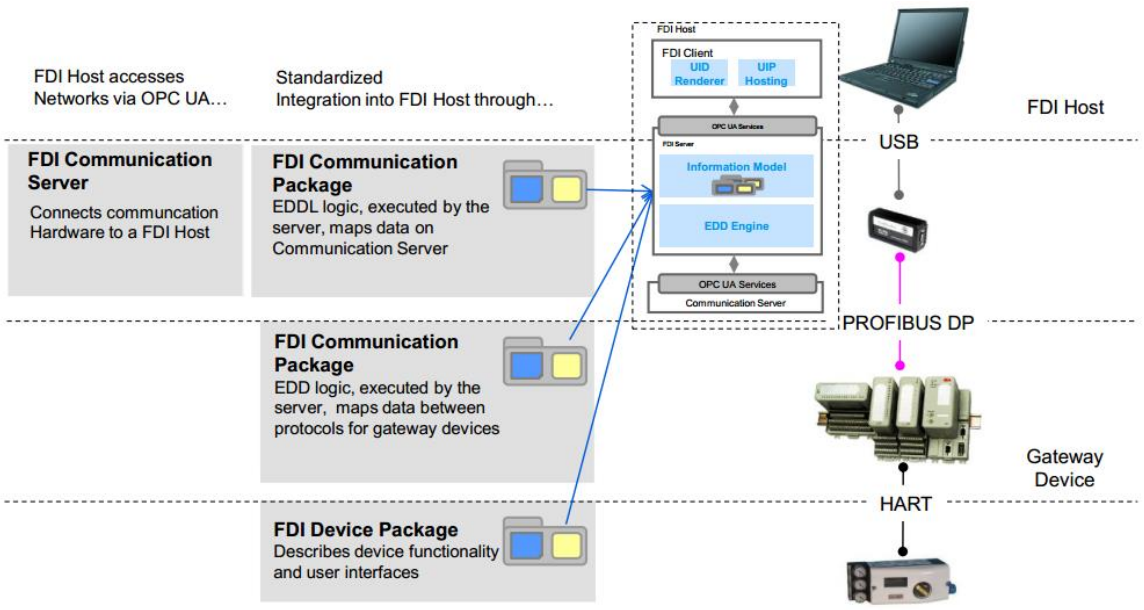


Figure 12: FDT based FDI host (FDI Cooperation, 2012)

The nested communication is another FDT concept used in FDI (Gunzert, Lindner, Wesner, & Kato, 2013) in Figure 13 . It is the open integration of gateways, and the integration of communication drivers via communication servers. In FDI, all operations and services that are necessary for communication are described and provided in a standardized format as part of an FDI Communication Package via EDDL code. The FDI Server takes care of the execution and management of all tasks. The FDI communications concept enables communication with devices in heterogeneous hierarchical networks and the use of any communication hardware.



FDI Host accesses Networks via OPC UA…
Standardized Integration into FDI Host through…
FDI Host
FDI Client
UID Renderer
UIP Hosting
OPC UA Services
FDI Server
Information Model
EDD Engine
OPC UA Services
Communication Server
FDI Host
USB
FDI Communication Server
Connects communcation Hardware to a FDI Host
FDI Communication Package
EDDL logic, executed by the server, maps data on Communication Server
PROFIBUS DP
FDI Communication Package
EDD logic, executed by the server, maps data between protocols for gateway devices
Gateway Device
HART
FDI Device Package
Describes device functionality and user interfaces

Figure 13: Hierarchical networks – nested communication (FDI Cooperation, 2012)

3.3.3 FDI and Existing Solutions

FDI intended in the long term to replace the existing EDDL and FDT technologies. Tools and system manufacturers are used to handle life cycle topics and provide concepts for maintaining the system hardware and software. The EDD, DTM and the new FDI Package exist only on the computer, not in the device. Therefore, it is possible to migrate from DTM or EDD to FDI without changing the devices in case of system software upgrade (Yamamoto & Sakamoto, 2008; Mahnke, Gössling, Graube, & Urbas, 2011). Moreover, it should be possible for device manufacturers to create FDI Packages efficiently and economically for both new and existing device (Li & Liu, 2011). Specifically, existing EDD sources or DTMs should be possible to be reused. The FDI Technology supports all these methods.

The installed base of EDD (Electronic Device Descriptions) is supported by two means:

1. The FDI Package development tool (IDE, Integrated Development Environment) allows existing EDD sources to be converted into the harmonized EDD and used in a device package in combination with a UIP.
2. The multiprotocol EDD Engine (EDD interpreter) is backward compatible with existing EDD formats. Existing EDDs can therefore be processed directly in an FDI host.

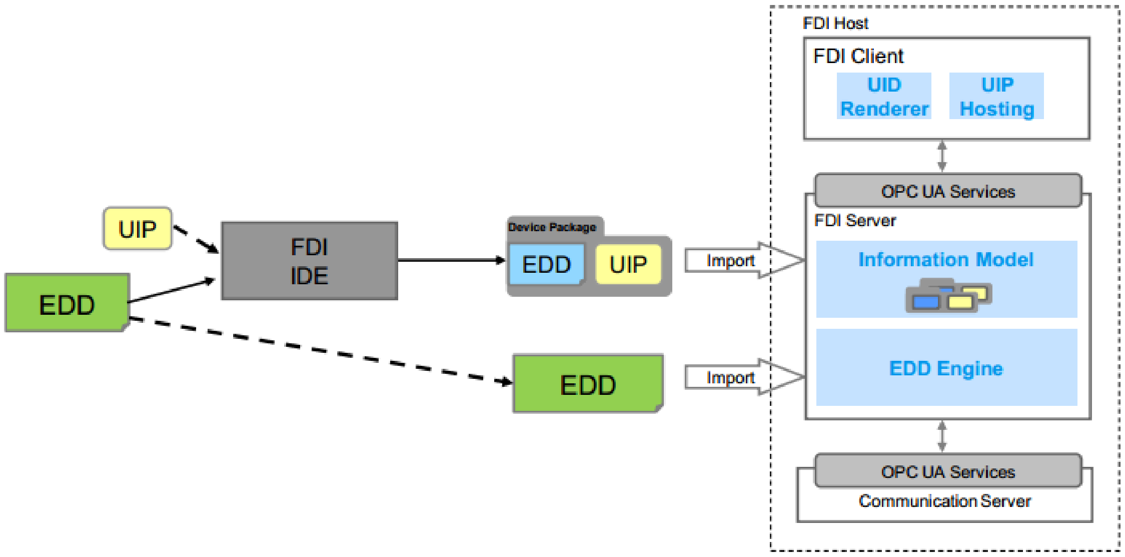


Figure 14: EDD Migration (FDI Cooperation, 2012)

Similar to the migration concept for EDDs, the migration of DTMs will be made possible:

1. The FDI Package development tool (IDE) allows existing EDD sources to be converted into the harmonized EDD and used in a device package in combination with a UIP.
2. Existing DTM software can be reused for creating UIPs and then used as a component of the FDI Package.
3. FDT frame applications can process device packages with the aid of an FDI DTM. As the multiprotocol EDD Engine (EDD interpreter) is also backward compatible with existing EDD formats, existing EDDs can also be processed using the FDI DTM.
4. Existing DTMs can still be processed in the FDT frame application.

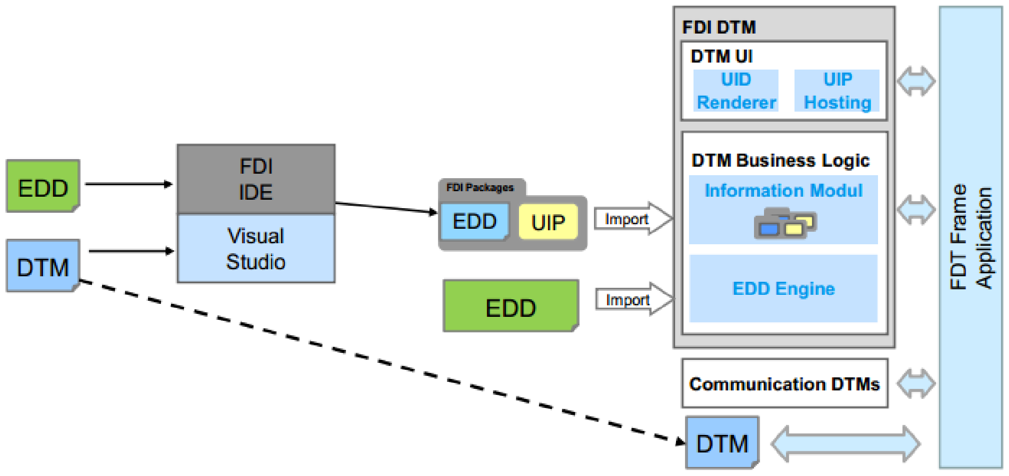


Figure 15: DTM Migration (FDI Cooperation, 2012)

3.3.4 Increasing Interoperability

To support the devices and system development, as well as increasing the interoperability of FDI products, multiprotocol software tools and standard host components are provided by the fieldbus organizations.

The Integrated Development Environment (IDE) helps device manufacturers create device packages for FF, HART, PROFIBUS and PROFINET devices. Essentially, the tool has four components (FDI Cooperation, 2012):

- EDDs with “tokenizing” (the binary coding of an EDD).
- The encoded EDD, the UIP.
- A runtime environment (reference host).
- The test engine.

The FDI Packages that a device manufacturer creates in this manner are certified and registered by Fieldbus Foundation, HART Communication Foundation and PROFIBUS & PROFINET International, together with the respective device hardware.

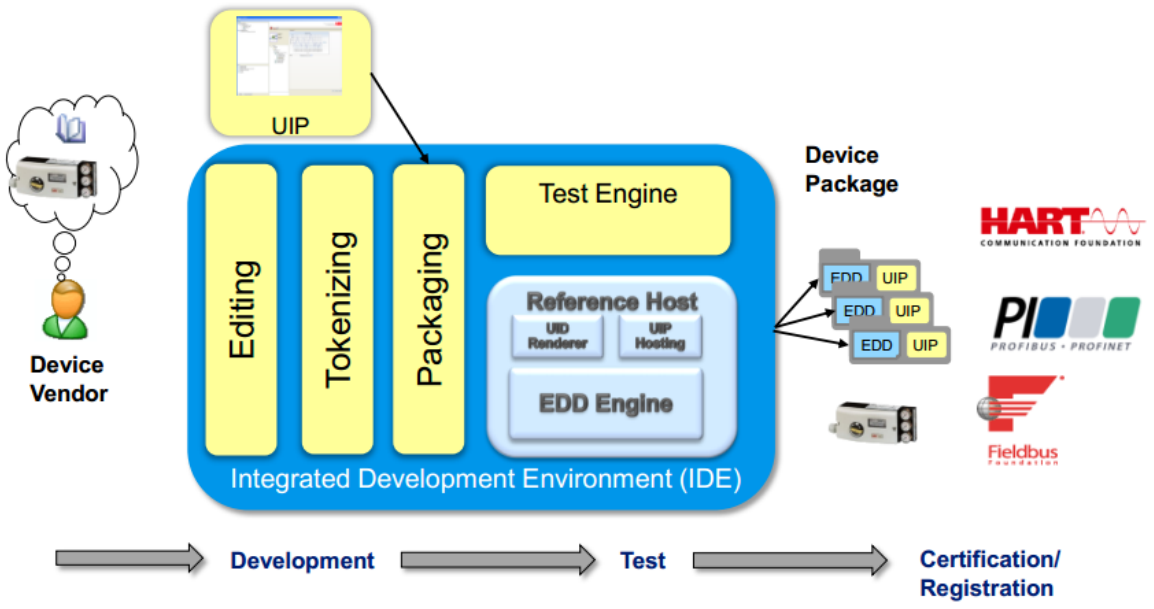


Figure 16 FDI Integrated Development Environment (FDI Cooperation, 2012)

The way out of the interpreter components available today is to develop uniform, multiprotocol standard FDI host components. EDD Engine (interpreter), UID Renderer and UIP Hosting components ensure that an FDI Device Package behaves in the same way in various systems. The EDD Engine supports the entire language scope of EDD in a multiprotocol manner, in accordance with IEC 61804-3, and is backward compatible with existing EDD formats. Therefore, in future, system manufacturers no longer need to integrate three interpreter components, but only one. It saves time and effort and contributes to improving the quality and interoperability.

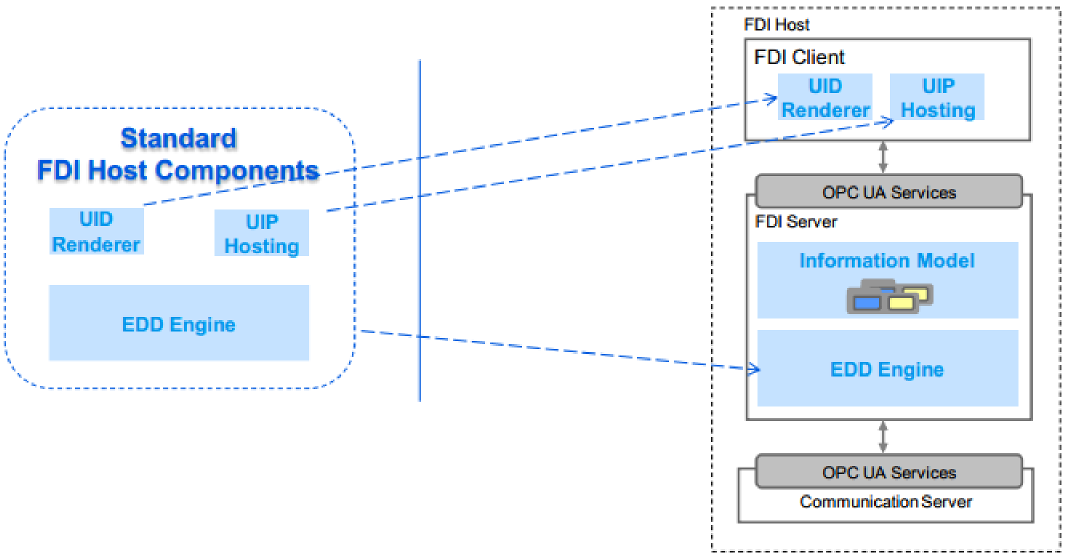


Figure 17: FDI standard host components (FDI Cooperation, 2012)

3.3.5 The Benefits

FDI provides benefits for control system manufacturers, device manufacturers and users. For control system manufacturers, the client-server architecture simplifies the use of device data and functions in powerful, distributed control systems. In addition, transparent access to device data and functions facilitates the integration of other applications. There are other benefits: the central management of data prevents inconsistencies; client-side installation is no longer required. FDI reduces effort and saves costs because for device manufacturers. Another advantage is the scalability of the FDI Device Package. For the customer, the main benefit of FDI is the standardized integration of field devices through a futureproof standard. It ensures unrestricted interoperability of device packages from a wide variety of device manufacturers with FDI systems (FDI hosts) from a wide variety of control system manufacturers.

3.3.6 FDI and Industry 4.0

The Industry 4.0, the fourth industrial revolution, is set on merging automation and information domains into the industrial Internet of Things (IoT), services, and people. Self-configuring and maintaining systems proclaim the dissolution of the automation pyramid. In this situation, FDI can be the bridge between the protection of past investment and the future of automation, moving from asset to service-oriented automation, and still keeping plant owners in control of their processes. The FDI can contribute to the key requirements summarized as (Schulz, 2015)

1. Confidentiality to protect trade secrets and brands
2. Functional integrity to protect human lives, production, and production assets
3. Barrier-free data access for increased efficiency and Internet of Services (IoS) and People (IoP).

FDI closes gaps in its predecessor technologies FDT and EDDL in a uniform way. The existing base of installed devices can be migrated to the FDI standard without any hardware modification. Thus the existing investment can be protected (Schulz, 2015). The essential point is that with FDI, the interface for any such data exchange can be easily provisioned. Therefore there is no necessity to develop new technologies and protocols for Industry 4.0 (Schulz, 2015).

In short, electronic device description language, field device tool/device type, and field device integration are reviewed in this section. A plant may employ more than a thousand devices. Moreover, the typical lifespan of such technologies is measured in a decade or more. While the challenges of Industry 4.0 require a fully integrated infrastructure, integration technologies, such as EDDL, FDT/DTM and FDI, are expected to play an increasingly important role in process and factory automation. In Table 1, we compare these technologies on the basis of the most relevant features. FDI combines both EDDL and FDT and it provides benefits for control system manufacturers, device manufacturers, and users. Firstly, FDI enables EDDL to be largely harmonized and standardized across the protocols. Moreover, FDI ensures interoperability with FDT. Therefore, it allows all system and tool manufacturers to support FDI. This will primarily benefit device manufacturers that currently have to provide both a DTM and an EDD for one device. In case of system software upgrade, it is always possible to migrate from DTM or EDD to FDI without changing the devices. Overall, FDI seamlessly integrates EDDL and FDT and takes all benefits from them. Table 1 provides a comparison among FDT/DTM, EDD, and FDI.

Table 1: The comparison of FDT/DTM, EDD, and FDI, adopted from (EDDL or FDT/DTM: Characteristics of EDDL and FDT/DTM, 2006)

Item	**FDT/DTM**	**Electronic Device Description**	**FDI**
Structure/type	Program	Text, data	Package - a collection of files
Functionality of field device determined by	Field device and component manufacturers	Host system manufacturers	FDI host
Flexibility for adding new functionality	High for device manufacturers, non for host system manufacturers	High for host system manufactures, low for device manufacturers	Low for all manufacturers
Presentation of device functionality	Is determined by DTM. Therefore full functionality for all device types	Dependent on host system. Must be supported by DCS vendor.	Handled by the information model in FDI host
Installation procedures	Software installation	File copy	Software installation
Dependency on operating system	FDT frame and DTM must be verified against operating system	No, but host application (EDDL interpreter) may be dependent on host operating system	No
User interface	DTM style guide	Proprietary, determined by host system	Windows Presentation Foundation (WPF)
International Standard	IEC 62453	IEC 61804-3	n/a

In our view, FDI is the most promising of these technologies because it creates a uniform standard for device integration which brings EDDL and FDT/DTM together. We also remark that current Industry 4.0 scenarios are mostly at a high level of abstraction. I.e., plug and produce, selforganizing system, horizontal integration, all require data exchange between individual devices and machines without detailed specification. With FDI, the interface for any such data exchange can be easily provisioned. This means that it is not necessary to re-invent new technologies and protocols for designing the details of Industry 4.0.

4 Manufacturing Assets/Services Classification

Digital Manufacturing Platforms will be fundamental for the development of Industry 4.0 and Connected Smart Factories. They are enabling the provision of services that support manufacturing in a broad sense by aiming at optimising manufacturing from different angles: production efficiency and uptime, quality, speed, flexibility, resource-efficiency, etc. For instance, services can aim at (EFFRA, 2016):

- Engineering of manufacturing
- Monitoring of manufacturing processes
- Data analytics through advanced automatic and human data science technics/technologies
- Manufacturing control involving an interaction among different agents, including machine-to-machine communication and the introduction of self-learning capabilities
- Simulation of manufacturing processes
- Assistance to factory workers and engineers, including augmented reality
- Planning of manufacturing, predictive and automated maintenance, etc.

All these services collect, store, process and deliver data that either describe the manufactured products or are related to the manufacturing processes and assets that make manufacturing happen.

As pointed out in (EFFRA, 2016), pre-requisites for digital platforms to thrive in a manufacturing environment include the need for agreements on industrial communication interfaces and protocols, common data models and the semantic interoperability of data, and thus on a larger scale, platform inter-communication and inter-operability. The achievement of these objective will allow a boundaryless information flow among the single product lifecycle phases (QLM, 2012) thus enabling an effective, whole-of-life product lifecycle management (PLM). Indeed, the most significant obstacle is that valuable information is not readily shared with other interested parties across the Beginning-of-Life (BoL), Middle-of-Life (MoL), and End-of-Life (EoL) lifecycle phases but it is all too often locked into vertical applications, sometimes called *silos*. Moreover, these objectives are strictly related to the need of achieving the full potential of the Internet of Things in the manufacturing industry. Indeed, without a trusted and secure, open, and unified infrastructure for true interoperability, the parallel development of disparate solutions, technologies, and standards will lead the Internet of Things to become an ever-increasing web of organization and domain-specific intranets.

The EU PROMISE project[5] developed the foundation of the Quantum Lifecycle Management (QML) Technical Architecture to support and encourage the flow of lifecycle data between multiple enterprises throughout the life of an entity and its components. QML was further developed by the Quantum Lifecycle Management (QLM)[6], a Work Group of The Open Group whose members work to establish open, vendor-neutral IT standards and certifications in a variety of subject areas critical to the enterprise.

The three main components of QML are the Messaging Interface (MI), the Data Model (DM), and the Data Format (DF) (Parrotta, et al., 2013). The Message Interface provides a flexible interface for making and responding to requests for instance-specific information. A defining characteristic of MI is that nodes do not have predefined roles, as it follows a “peer-to-peer” communications model. This means that products can communicate directly with each other or with back-end servers, but the MI can also be used for server-to-server information exchange of sensor data, events, and other information. The transmitted information is in XML format and mainly

[5] The PROMISE Project (2004-2008): A European Union research project funded under the 6th Framework Program (FP6) which focused on information systems for whole-of-life product lifecycle management.
[6] http://www.opengroup.org/subjectareas/qlm-work-group

intended for automated processing by information systems. The MI allows one-off or standing information request subscriptions to be made. Subscriptions can be made for receiving updates at regular intervals or on an event basis – when the value or status changes for the information subscribed to. The MI also supports read and write operations of the value of information items.

The Data Model, instead, enables detailed information about each instance of a product to be enriched with “field data”; i.e., detailed information about the usage and changes to each instance during its life. It also allowed the aggregation of instance-specific data from many different software systems; e.g., CAD, CRM, and/or SCM and other legacy systems as part of a company’s IT infrastructure in order to allow specific decision support information to be generated and made available through the PDKM system. DM is represented by different classes of information to individuate activities, processes, resources, documents, field data and other aspects through the whole product life. Each class contains dedicated attributes to explain information suggested collecting different information about the product.

Finally, the Data Format represents, through an XML schema, the structure of the message exchanged between many products and/or systems. The structure of the message is similar to the Data Model schema so that it could be easily recognize by a system QLM DM compatible, thereby automating the data collection.

Various works adopt QLM for manufacturing assets representation and classification. For instance the paper (Kubler, Främling, & Derigent, 2015) proposes data synchronization models based upon QLM standards to enable the synchronization of product-related information among various systems, networks, and organizations involved throughout the product lifecycle. These models are implemented and assessed based on two distinct platforms defined in the healthcare and home automation sectors. Främling, Kubler, & Buda (2014) describe two implemented applications using QLM messaging, respectively, defined in BoL and between MoL-BoL.

The former is a real case study from the LinkedDesign EU FP7 project, in which different actors work on a production line of car chassis. This process segment involved two robots to transfer the chassis part from machine to machine. The actors involved in the manufacturing plan expressed, on the one hand, the need to check each chassis part throughout the hot stamping process and, on the other hand, the need to define communication strategies adapted to their own needs. Accordingly, scanners are added between each operation for the verification procedure, and QLM messaging is adopted to provide the types of interfaces required by each actor.

The latter, instead, involves actors from two distinct PLC phases: 1) In MoL: A user bought a smart fridge and a TV supporting QLM messaging; 2) In BoL: The fridge designer agreed with the user to collect specific fridge information over a certain period of the year (June, July, August) using QLM messaging. Also in this case, the appropriate QLM interfaces regarding each actor have been set up in such a way that the involved actors can get the required information about the smart objects.

In most applications scenarios, taxonomies are usually adopted as common ground for semantic interoperability. Classifying products and services with a common coding scheme facilitates commerce between buyers and sellers and is becoming mandatory in the new era of electronic commerce. Large companies are beginning to code purchases in order to analyse their spending.

Nonetheless, most company coding systems today have been very expensive to develop. The effort to implement and maintain these systems usually requires extensive utilization of resources, over an extended period of time. Additionally, maintenance is an on-going, and expensive, process. Another problem is that company’s suppliers usually don’t adhere to the coding schemes of their customers, if any.

Samples of taxonomy including the description and classification of manufacturing assets and services are: eCl@ss, UNSPSC, and MSDL. eCl@ss[7] is an international product classification and

[7] http://www.eclasscontent.com/index.php?language=en&version=7.1

description standard for information exchange between customers and their suppliers. It provides classes and properties that can be exploited to standardise procurement, storage, production, and distribution activities, both intra-companies and inter-companies. It is not bound to a specific application field and can be used in different languages. It is compliant to ISO/IEC. It adopts an open architecture that allows the classification system to be adapted to an enterprise's own internal classification scheme, so granting flexibility and standardization at the same time. Thanks to its nature, it can be exploited in the Internet of Things field in order to enable interoperability among devices of different vendors. As of October 2017, there are about 41,000 product classes and 17,000 uniquely described properties which are categorized with only four levels of classification; this enables every product and service to be described with an eight-digit code. One of the aims of eCl@ass is to decrease inefficiencies, so that packaging and distribution take place automatically, relying on the classes and identifier available by the standard. The nature of eCl@ss enables the definition of several aspects in virtual factories.

The United Nations Standard Products and Services Code (UNSPSC)[8] provides an open, global multi-sector standard for efficient, accurate classification of products and services. The UNSPSC was jointly developed by the United Nations Development Programme (UNDP) and Dun & Bradstreet Corporation (D & B) in 1998. It has been managed by GS1 US since 2003. UNSPSC is an efficient, accurate and flexible classification system for achieving company-wide visibility of spend analysis, as well as, enabling procurement to deliver on cost-effectiveness demands and allowing full exploitation of electronic commerce capabilities. Encompassing a five-level hierarchical classification codeset, UNSPSC enables expenditure analysis at grouping levels relevant to the company needs. The codeset can be drilled down or up to see more or less detail as is necessary for business analysis. The UNSPCS classification can be exploited to perform analysis about company spending aspects, to optimize cost-effective procurement, and to exploit electronic commerce capabilities.

The Manufacturing Service Description Language (MSDL) (Ameri & Dutta, 2006) is a formal ontology for describing manufacturing capabilities at various levels of abstraction including the supplier-level, process-level, and machine-level. It covers different concepts like actors, materials, like ceramic and metal, physical resources, tools, and services. Description Logic is used as the knowledge representation formalism of MSDL in order to make it amenable to automatic reasoning. MSDL can be considered an "upper" ontology, in the sense that it provides the basic building blocks required for modeling domain objects and allows ontology users to customize ontology concepts based on their specific needs; this grants flexibility and standardization at the same time. MSDL is composed of two main parts: 1) MSDL core and 2) MSDL extension. MSDL core is the static and universal part of MSDL that is composed of basic classes for manufacturing service description; MSDL extension is dynamic in nature and includes a collection of taxonomies, sub-classes and instances built by users from different communities based on their specific needs; MSDL extensions drive evolution of MSDL over time.

The 2016 EFFRA document (EFFRA, 2016) highlights the need for activities that aim at validating the deployment of digital platforms for manufacturing with a focus on:

- The possibility to connect to additional services according to the 'plug-and-play' philosophy and considering the multi-sided ecosystem of service providers, platform providers and manufacturing companies;
- Integrating legacy system (hardware and software);
- Overcoming semantic barriers;
- Considering requirements of specific manufacturing sectors (process industry, consumer goods, capital equipment, ...);
- Generating accessible technical and non-technical software documentation.

[8] http://www.unspsc.org/

5 Manufacturing Assets/Services Discovery Methods

With the increasing number of assets/services, service discovery becomes an integral part of digital/virtual factories. Service discovery provides a mechanism which allows automatic detection of services offered by any component/agent/element in the system/network. In other words, service discovery is the action of finding a service provider for a requested service. When the location of the demanded service is retrieved, the requestor may further access and use it. The objective of a service discovery mechanism is to develop a highly dynamic infrastructure where requestors would be able to seek particular services of interest, and service providers offering those services would be able to announce and advertise their capabilities. Furthermore, service discovery should minimize manual intervention and allows the system/network to be self-healing by automatic detection of services which have become unavailable. Once services have been discovered, devices in the system/network could remotely control each other by adhering to some standard of communication.

The main elements of a service discovery framework are (Talal & Rachid, 2013):

- Service Description - In order to facilitate the service discovery process, each protocol has a description language to define the vocabulary and syntax used to describe the service and its properties. The available methods for this task vary according to the degree of expressiveness: key/value, template-based and semantic description. In the key/value approach, services are characterized using a set of attribute-value pairs. The template-based approach: uses the same technique as in the first approach, in addition it offers predefined set of common attributes which are frequently used. The semantic description relies on the use of ontology. It has richer expressive power than the first two approaches.
- Service Discovery Architecture - Architecture used by service discovery protocols can be classified as directory and non-directory based models, according to how the service descriptions are stored.
- The directory based model has a dedicated directory which maintains the whole service descriptions. In this case, the directory takes care of registering service descriptions and processing user requests. The directory can be logically centralized but physically distributed over the system/network. Therefore, service descriptions are stored at different locations (directories).
- The non-directory based model: has no dedicated directory, every service provider maintains its service descriptions. When a query arrives, every service provider processes it and replies if it matches the query.
- Service Announcement and Query - Service announcement and query are the two basic mechanisms for directories, service providers, and directories to exchange information about available services.
- Service Announcement: allows service providers to indicate to all potential users that a set of new services is active and ready for use. This will be accomplished by registering the appropriate service descriptions with the directory if it exists, or multicast service advertisements.
- Query approach: allows requestors to discover services that satisfy their requirement. To do this, users initiates (a) unicast query to the directory, or (b) multicast query. The query is expressed using the description language, and specifies the details about service it is looking for. The directory or service provider that holds the matching service description replies to the query.
- When a directory exists, service providers and users will first discover the directory location before services can be registered and queried. In this case, the directory can be seen as any service in the system/network and makes advertisement to advertise its existence.

- Service Usage (Service invocation) - After retrieving the desired services information, the next step is to access. However, apart from performing service discovery, most protocols offer methods for using the services. An example is Simple Object Access Protocol (SOAP) used in Universal Plug and Play (UPnP). We will not address further the service usage in this section.
- Configuration Update (management dynamicity) - Service discovery protocol must preserve a consistent view of the system/network and deliver valid information about available services while system/network is dynamic. Therefore, the management of such dynamicity is required. Configuration update allows requestors to monitor the services, their availability and changes in their attributes. There are two sub functions in Configuration Update:
    - Configuration Purge. Allows detection of disconnected entities through (a) leasing and (b) advertisement time-to-live (TTL). In leasing, the service provider requests and maintains a lease with the directory, and refreshes it periodically. The directory assumes that the service provider who fails to refresh its lease has left the system, and purges its information. With TTL, the user monitors the TTL on the advertisement of discovered services and assumes that the service has left the system if the service provider fails to re-advertise before its TTL expires.
    - Consistency Maintenance. Allows requestors to be aware when services change their characteristics. Updates can be propagated using (a) push-based update notification, where requestors and directories receive notifications from the service provider, or (b) pull-based polling for updates by the user to the directory or service provider for a fresher service description.

It is important to note that the features and techniques mentioned before representing the pillars around which an autonomic service discovery protocol is based. But, depending on characteristics of each protocol other functions have been already proposed in diverse approaches (e.g. service selection, security, scalability).

5.1 General purpose service discovery approaches

Over the past years, many organizations and major software vendors have designed and developed a large number of service discovery protocols. They are general-purpose, i.e., to specifically tailored for the domain of virtual/digital factories.

- SLP - Service Location Protocol (SLP) (Guttman, Perkins, Veizades, & Day, 1999) is an open, simple, extensible, and scalable standard for service discovery developed by the IETF (Internet Engineering Task Force). It was intended to function within IP network. SLP addresses only service discovery and leaves service invocation unspecified. The SLP architecture consists of three main components:
    - User Agent (UA): software entity that sends service discovery request on a requestor application’s behalf.
    - Service Agent (SA): advertises the location and characteristics of services on behalf of services.
    - Directory Agent (DA): a central directory collects service descriptions received from SAs in its database and process discovery queries from UAs.

  When a new service connects to the network, the SA contacts the DA to advertise its existence (service registration). Registration message contains: service lifetime, URL for the service, and set of descriptive attributes for the service. Both URL schemas and attributes are defined in the standard. Registration should be refreshed periodically by the SA to indicate its continuous existence. The same when the requestor needs a certain service, the UA sends request message to the DA which in turn responds with message containing URLs for all services matched against the UA needs. The requestor can access one of the services pointed to by the returned URL. The protocol used between the client and the service is outside the scope of the SLP specification. To perform their respective roles UA and SA have first to discover DA location. SLP provides three methods for DA discovery: static, active, and passive. In the static approach, SLP agents obtain the address of the DA using DHCP; with the active approach, SLP agent (UA/SA) sends service request to the SLP multicast group address, a DA listening on this address will respond via unicast to the requesting agent; in the passive approach, DA multicasts advertisements periodically, UAs and SAs learn the DA address from the received advertisements. It is important to note that the DA is not mandatory; it is used especially in large networks to enhance scalability. In smaller network (e.g. home network, office network) there may be no real need for DA, SLP is deployed without DA. In this case, UAs send their service requests to the SLP multicast address. The SAs announcing the service will send a unicast response to the UA. SLP provides a powerful filter that allows UAs to select the most appropriate service from among services on the network. The UA can formulate expressive queries using operators such as AND, OR, comparators (<, =,>, <=,>=) and substring. SLP is an open source; it does not depend on any programming language and scales well in large networks. The scalability is supported by various features such as scope concept, and multiple DAs.

- Jini (Arnold, O'Sullivan, Scheifler, Waldo, & Wollrath, 1999)is a distributed service discovery system developed by Sun-Microsystems in Java. The goal of the system is the federation of groups of clients/services within a dynamic computing system. A Jini federation is a collection of autonomous devices which can become aware of one another and cooperate if need be. To achieve this goal, Jini uses a set of lookup services to maintain dynamic information about available services and specifies how service discovery and service invocation is to be performed among Java-enabled devices. The Jini discovery architecture is similar to that of SLP:
    - Client: requests Lookup Service for available service.
    - Service provider: registers its services and their descriptions with Lookup Service.
    - Lookup Service (LS): directory which collects service descriptions and process match queries in manner analogous to DA in SLP. Unlike SLP, where DA is optional, Jini operates only as a directory based service discovery and requires the presence of one or more Lookup Services in the network.

  The heart of Jini is a trio protocols called: discovery, join, and lookup. Discovery occurs when a service provider or client is looking for Lookup Service. Join occurs when a service provider has located a LS and wishes to join it. Lookup occurs when the client needs to locate and invoke a service. Jini uses Java’s remote method invocation (RMI) facility for all interactions between either a client or a service and the lookup server (after the initial discovery of the lookup server). It allows data as well as objects to be passed through the network. In Jini, evaluation of requests is based on equality and exact correspondence between request parameters and attributes of services. Jini does not allow the evaluation of complex queries with Boolean operators or comparators such as SLP.
- UPnP is a Microsoft-developed service discovery technology aimed at enabling the advertisement, discovery, and control of networked devices and services. It is built upon IP that is used for communication between devices, and uses standard protocols like HTTP, XML, and SOAP for discovery, description, and control of devices. The architecture of UPnP network is as follow:
    - Device: can be any entity on the network that contains services or any embedded devices. A service is the smallest unit of control in UPnP and it consists of:
        - State table: models the state of the services at run time through state variable.
        - Control server: receives requests, executes them; updates the state table and returns responses.
        - Event server: publishes events to interested clients when service state changes.

   - Control point: any entity in the network that is able to discover, retrieve service descriptions, and control the features offered by a device.

  UPnP uses a non-directory based approach for service discovery where each device hosts a device description document. This document is expressed in XML and includes device information (e.g., manufacturer, model, serial number, etc.), list of any embedded devices or services, as well as URLs for the service description, control, and eventing. For each service, the description contains the service type, service ID, state table, and list of the actions that a service can perform. The UPnP discovery process is based on the Simple Service Discovery Protocol (SSDP), which allows UPnP devices to announce their presence to others and discover other devices and services. When a device comes on-line, it sends advertisement (ssdp: alive) via multicast to announce its presence. The advertisement message is associated with a lifetime and contains typically the type of the advertised service, and URL to the description. An UPnP device may send out many presence announcements. When the device wish to disconnect from the network, it should send an advertisement (ssdp:bye-bye) to notify control points that its services are no longer available. Any control point that comes on-line after the UPnP device has announced its presence sends out discovery request (ssdp: discover) via multicast. Devices listening for this multicast respond via unicast if they match the service. Control points can search only for: all services, specific service type, or specific device type since SSDP does not support attribute-based querying for services.

- UDDI - Universal Description, Discovery, and Integration (UDDI) is an XML-based registry for business internet services. Publishing a Web service involves creating a software artifact and making it accessible to potential consumers. Web service providers augment a Web service endpoint with an interface description using the Web Services Description Language (WSDL) so that a consumer can use the service. Optionally, a provider can explicitly register a service with a Web Services Registry such as Universal Description Discovery and Integration (UDDI) or publish additional documents intended to facilitate discovery such as Web Services Inspection Language (WSIL) documents. The service users or consumers can search Web Services manually or automatically. The implementation of UDDI servers and WSIL engines should provide simple search APIs or web-based GUI to help find Web services. Web services may also be discovered using multicast mechanisms like WS-Discovery, thus reducing the need for centralized registries in smaller networks.
- The current UDDI search mechanism can only focus on a single search criterion, such as business name, business location, business category, service type by name, business identifier, or discovery URL. In fact, in a business solution, it is very normal to search multiple UDDI registries or WSIL documents and then aggregate the returned result by using filtering and ranking techniques. As an example, IBM modularized this federated Web Services Discovery engine in 2001, releasing its Business Explorer for Web Services (BE4WS).
- Historically, UDDI was an open industry initiative, sponsored by the Organization for the Advancement of Structured Information Standards (OASIS), for enabling businesses to publish service listings and discover each other, and to define how the services or software applications interact over the Internet. It was originally proposed as a core Web service standard (August 2000), designed to be interrogated by SOAP messages and to provide access to WSDL documents describing the protocol bindings and message formats required to interact with the web services listed in its directory. UDDI was included in the Web Services Interoperability (WS-I) standard as a central pillar of web services infrastructure, and the UDDI specifications supported a publicly accessible Universal Business Registry in which a naming system was built around the UDDI-driven service broker. Unfortunately, UDDI has not been as widely adopted as its designers had hoped. IBM, Microsoft, and SAP announced they were closing their public UDDI nodes in January 2006; the group defining UDDI, the OASIS Universal Description, Discovery, and Integration (UDDI) Specification Technical Committee voted to complete its work in late 2007 and has been closed; in September 2010,

Microsoft announced they were removing UDDI services from future versions of the Windows Server operating system. Instead, this capability would be moved to BizTalk Server; in 2013, Microsoft further announced the deprecation of UDDI Services in BizTalk Server. UDDI systems are most commonly found inside companies, where they are used to dynamically bind client systems to implementations; however, much of the search metadata permitted in UDDI is not used for this relatively simple role.

A UDDI business registration consists of three components:

- White Pages — address, contact, and known identifiers. White pages give information about the business supplying the service. This includes the name of the business and a description of the business - potentially in multiple languages. Using this information, it is possible to find a service about which some information is already known (for example, locating a service based on the provider's name. Contact information for the business is also provided - for example the businesses address and phone number; and other information such as the Dun & Bradstreet.
- Yellow Pages — industrial categorizations based on standard taxonomies. Yellow pages provide a classification of the service or business, based on standard taxonomies. These include the Standard Industrial Classification (SIC), the North American Industry Classification System (NAICS), or the United Nations Standard Products and Services Code (UNSPSC) and geographic taxonomies. Because a single business may provide a number of services, there may be several Yellow Pages (each describing a service) associated with one White Page (giving general information about the business).
- Green Pages — technical information about services exposed by the business. Green pages are used to describe how to access a Web Service, with information on the service bindings. Some of the information is related to the Web Service - such as the address of the service and the parameters, and references to specifications of interfaces. Other information is not related directly to the Web Service - this includes e-mail, FTP, and telephone details for the service. Because a Web Service may have multiple bindings (as defined in its WSDL description), a service may have multiple Green Pages, as each binding will need to be accessed differently.

To the best of our knowledge, all service discovery frameworks/approaches proposed for digital/virtual factories are based on the above technologies, and not specific new frameworks have been developed so far. Depending on the specific virtual/digital factory technology and approach, service discovery is developed adopting some of the previous concepts.

5.2 Semantics for service discovery

As previously discussed, the core of the expressive power of a service discovery approach lies in the service descriptions. A service description should define the functionality and intention of a service in unambiguous way. This can potentially be accomplished if a suitable ontology for service descriptions has been adopted, so that semantic matching is possible and keyword similarity can be taken into account when searching for services.

By adopting such rich service descriptions, also context awareness can be considered, by taking into account different information in the discovery stage (e.g., requestor preferences, device capabilities, QoS, etc.), which again should be modelled in the ontology.

When service descriptions are built using ontologies, it is possible to pursue the Ontology-based Data Integration (OBDI) approach (Lenzerini, 2002), which is based on the idea of posing the semantics of the application domain at the centre of the scene. In the last years, the OBDI approach has been successfully used in several projects at European level, in particular the European projects on Artefact-Centric Service Interoperation (ACSI, FP7-ICT-2009-5). Ontology-Based Data Access (Kontchakov, Lutz, Toman, Wolter, & Zakharyaschev, 2011)has been thoroughly investigated in recent years from the theoretical point of view, to a large extent within previous European projects (Haarslev & Möller, 2008; Calvanese, De Giacomo, Lembo, Lenzerini, & Rosati, 2007; Lenzerini, 2002). Also, prototypical implementations exist (Acciarri, et al., 2005) which have been applied to minor industrial case studies (see, e.g., (Amoroso, Esposito, Lembo, Urbano, & Vertucci, 2008)). The OPTIQUE project (FP7 ICT-2011.4.4) aimed at building a system for scalable end-user access to big data exploiting semantic technologies. While OPTIQUE was mainly focused on providing a semantic end-to-end connection between users and ontologies, by means of techniques for transforming user queries into complete, correct and highly optimized queries over the data sources, it is feasible to investigate how to enhance ontologies with representation of the dynamics of the processes and services, in order to effectively build a cognitive base supporting the service discovery.

6 Conclusions

In this deliverable we summarized existing work on the manufacturing assets/services description, representation and classification, as well as manufacturing assets/services discovery approaches done so far. This D1.1 deliverable is the first version of ‘‘Overview of manufacturing assets and services classification and ontology’’ and provides the basis for WP2, WP3 and WP4. We have identifie the relations between digital factory and smart factory, as well as the relations between digital factory and virtual factory. In a state of the art overview we presented and discussed some main concepts: manufacturing semantics related standards, architecture, and interoperability of product lifecycle management. In addition to these, state of the art technologies and tools are presented.

The main contribution of D1.1 is the review manufacturing asset/services description languages, manufacturing asset/services classification, and asset/services discovery approaches, which are foundation for WP2 and WP3. Considering the review, it is clear that most current work is still stays in digital factory level interoperability of manufacturing assets and services descriptions, representations, and classifications for interoperability within product lifecycle management and related systems, such as produce service systems. It provides the requirements to the FIRST project, i.e. how to extend or adopt current research to virtual factory level.

It is clear that the content of this deliverable can only be the starting point for extensive discussion on how manufacturing assets/service des are described, classified and discovered in the context of virtual factory to support on the fly business process verification, etc. This deliverable D1.1 provides the blueprint for D2.1 and D3.1. The initial version of D1.1 is going to be refined in a later release of this deliverable.

7 References

Acciarri, A., Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Palmieri, M., & Rosati, R. (2005). QUONTO: querying ontologies. *20th National Conference on Artificial Intelligence (AAAI 2005)*, (pp. 1670-1671).

Ameri, F., & Dutta, D. (2006). An upper ontology for manufacturing service description. *In ASME Conference*, (pp. 651-661).

Amoroso, A., Esposito, G., Lembo, D., Urbano, P., & Vertucci, R. (2008). Ontology-based Data Integration with MASTRO-I for Configuration and Data Management at SELEX Sistemi Integrati. *16th Italian Conference on Database Systems (SEBD 2008)*, (pp. 81-92).

Arnold, K., O'Sullivan, B., Scheifler, R., Waldo, J., & Wollrath, A. (1999). *The Jini Specification.* Addison-. Wesley.

BeinCPPS. (2016). *BeinCPPS Architecture – Layers*. Retrieved from Milan: Politecnico Di Milano: https://beincpps.ems-innovalia.org/nfs/programme_5/call_2/call_preparation/BEinCPPS%20architecture.pdf

Bi, Z., Da Xu, L., & Wang, C. (2014). Internet of things for enterprise systems of modern manufacturing. *IEEE Transactions on industrial informatics, 10(2)*, 1537-1546.

Blevins, T. (2007, April 16). *EDDL Overview*. Retrieved from http://www.eddl.org/SiteCollectionDocuments/EDDL_SP104Presentation.pdf

Bracht, U., & Masurat, T. (2005). The Digtial Factory between vision and reality. *Computers in industry 56, no. 4* , 325-333.

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., & Rosati, R. (2007). Tractable reasoning and efficient query answering in description logics: The DL-Lite family. *Journal of Automated reasoning, 39(3).*, 385-429.

Chungoora, N., Young, R. I., Gunendran, G., Palmer, C., Usman, Z., Anjum, N. A., & Case, K. (2013). A model-driven ontology approach for manufacturing system interoperability and knowledge sharing. *Computers in Industry 64, no. 4*, 392-401.

Da Xu, L., He, W., & Li, S. (2014). Internet of things in industries: A survey. *IEEE Transactions on industrial informatics, 10(4)*, 2233-2243.

Davies, R. (2015). *Industry 4.0 Digitalisation for Productivity and Growth.* http://www.europarl.europa.eu/RegData/etudes/BRIE/2015/568337/EPRS_BRI(2015)568337_EN.pdf.

Davis, J., Edgar, T., Porter, J., Bernaden, J., & Sarli, M. (2012). Smart manufacturing, manufacturing intelligence and demand-dynamic performance. *Computers & Chemical Engineering, 47*, 145-156.

Debevec, M., Simic, M., & Herakovic, N. (2014). Virtual factory as an advanced approach for production process optimization . *International journal of simulation modelling 13 (1)* , 66-78.

EDDL or FDT/DTM: Characteristics of EDDL and FDT/DTM. (2006). *WIB Workshop EDDL or FDT/DTM.* Utrecht. Retrieved from “EDDL or FDT/DTM: Characteristics of EDDL and FDT/DTM,” WIB Workshop EDDL or FDT/DTM, Utrecht, October 4, 2006. .

EFFRA. (2013). *FACTORIES OF THE FUTURE MULTI-ANNUAL ROADMAP FOR THE CONTRACTUAL PPP UNDER HORIZON 2020.* EFFRA - European Factories of the Future Research Association.

EFFRA, H. 2. (2016, underAvailable at:). *Factories 4.0 and Beyond Recommendations for the work programme 18-19-20 of the FoF PPP*. Retrieved from Horizon 2020 EFFRA: http://effra.eu/sites/default/files/factories40_beyond_v31_public.pdf

Electronic Device Description Language. (2017, 10 16). Retrieved from http://www.eddl.org/

EU. (2013). *Digital Single Market.* https://ec.europa.eu/digital-single-market/en/virtual-factory.

European Union. (2017). *New European Interoperability Framework (EIF).* https://ec.europa.eu/isa2/sites/isa/files/eif_brochure_final.pdf.

FDI Cooperation. (2012). *Field Device Integration Technology.* http://www.fdi-cooperation.com/tl_files/images/content/Publications/FDI-White_Paper.pdf.

Fieldcomm Group. (n.d.). *Field Device Integration* . https://fieldcommgroup.org/technologies/field-device-integration .

FITMAN. (2013). *D1.5 FITMAN Reference Architecture.*

FITMAN. (2015). *Fitman Catalogue Specific Enablers*. Retrieved from Milan: Fitman: http://catalogue.fitman.atosresearch.eu/enablers

FITMAN. (2015). VOLKSWAGEN Trial – PLM Ramp up reducing Time to Market. Retrieved from FITMAN FI, 2015a. VOLKSWAGEN Trial – PLM Ramp up reducing Time to Market.

FIWARE. (2015). *Industrial IoT Reference Architecture* . Retrieved from Milan: Fiware for Industry: http://www.fiware4industry.com/?page_id=1043

Främling, K., Kubler, S., & Buda, A. (2014). Universal messaging standards for the IoT from a lifecycle management perspective. *IEEE Internet of things journal, 1(4),* , 319-327.

Franke, M., Klein, K., Hribernik, K., Lappe, D., Veigt, M., & Thoben, K. D. (2014). Semantic web service wrappers as a foundation for interoperability in closed-loop product lifecycle management. . *Procedia CIRP, 22*, 225-230.

Franke, M., Klein, P., Schröder, L., & Thoben, K. (2011). Ontological semantics of standards and plm repositories in the product development phase. *Proceedings of the 20th CIRP Design Conference* (pp. 473-482). Ecole Centrale de Nantes, Nantes, France: Springer, Berlin, Heidelberg,.

Gregor, M., & Stefan, M. (2010). Digital Factory–Theory and Practice. *In Engineering the Future. InTech.*

Grossmann, D., Bender, K., & Danzer, B. (2008). OPC UA based field device integration. *In 2008 SICE Annual Conference* (pp. 933-938). IEEE.

Gunzert, M., Lindner, K. P., Wesner, S., & Kato, M. (2013). Bridging FDT and FDI. *In SICE Annual Conference (SICE), 2013 Proceedings of* (pp. 332-337). IEEE.

Guttman, E., Perkins, C., Veizades, J., & Day, M. (1999). *Service Location Protocol, Version 2.* Internet Engeineering Task Force (IETF), RFC 2608.

Haarslev, V., & Möller, R. (2008). On the scalability of description logic instance retrieval. *Journal of Automated Reasoning, 41(2)*, 99-142.

Hao , Y., & Helo, P. (2017). The role of wearable devices in meeting the needs of cloud manufacturing: A case study. *Robotics and Computer-Integrated Manufacturing, Volume 45,*, Pages 168-179, ISSN 0736-5845, https://doi.org/10.1016/j.rcim.2015.10.001.

Hermann, M., Pentek, T., & Otto, B. (2016). Design principles for industrie 4.0 scenarios. *System Sciences (HICSS), 2016 49th Hawaii International Conference* (pp. 3928-3937). IEEE.

ISO. (2005). ISO 10303-239 2005. *Industrial automation systems and integration–Product data representation and xchange–Part 239: Application protocol: Product life cycle support.* International Organization for Standardization, Geneva (Switzerland).

Jardim-Goncalves, R., Grilo, A., & Steiger-Garcao, A. (2006). Challenging the interoperability between computers in industry with MDA and SOA. *Computers in Industry, 57(8)*, 679-689.

Kagermann, H., Wahlster, W., & Helbig, J. (2013). *Recommendations for Implementing the Strategic Initiative INDUSTRIE 4.0 -- Securing the Future of German Manufacturing Industry.* München: http://www.acatech.de/fileadmin/user_upload/Baumstruktur_nach_Website/Acatech/root/de/Material_fuer_Sonderseiten/Industrie_4.0/Final_report__Industrie_4.0_accessible.pdf.

Kontchakov, R., Lutz, C., Toman, D., Wolter, F., & Zakharyaschev, M. (2011). The combined approach to ontology-based data access. *IJCAI, Vol. 11*, 2656-2661.

Kubler, S., Främling, K., & Derigent, W. (2015). P2P Data synchronization for product lifecycle management. *Computers in Industry, 66*, 82-98.

Kuhn, W. (2006). Digital factory-simulation enhancing the product and production engineering process. *Simulation Conference, 2006. WSC 06. Proceedings of the Winter* (pp. 1899-1906). IEEE.

Lenzerini, M. (2002). Data integration: A theoretical perspective. *the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems* (pp. 233-246). ACM.

Li, Q., & Liu, F. (2011). The future of the device integration: Field device integration. *In Software Engineering and Service Science (ICSESS), 2011 IEEE 2nd International Conference on* (pp. 687-690). IEEE.

Mahnke, W., Gössling, A., Graube, M., & Urbas, L. (2011). Information modeling for middleware in automation. *In Emerging Technologies & Factory Automation (ETFA), 2011 IEEE 16th Conference on* (pp. 1-7). IEEE.

Martin, D., Domingue, J., Sheth, A., Battle, S., Sycara, K., & Fensel, D. (2007). Semantic web services, part 2. *IEEE Intelligent Systems, 22(6)*, 8-15.

Monostori, L. (2014). Cyber-physical production systems: Roots, expectations and R&D challenges. *Procedia CIRP, 17*, 9-13.

Naumann, F., & Riedl, M. (2011). *EDDL - Electronic Device Description Language.* Oldenbourg Industrieverl.

Neumann, P., Simon, R., & Diedrich, C. a. (2001). Field device integration. *In Emerging Technologies and Factory Automation, 2001.* (pp. Vol. 2, pp. 63-68). IEEE.

Parrotta, S., Cassina, J., Terzi, S., Taisch, M., Potter, D., & Främling, K. (2013). Proposal of an interoperability standard supporting PLM and knowledge sharing. *In IFIP International Conference on Advances in Production Management Systems* (pp. 286-293). Springer.

Pitoura, E., Bukhres, O., & Elmagarmid, A. (1995). Object orientation in multidatabase systems. *ACM Computing Surveys (CSUR), 27(2)*, 141-195.

PLM Services 2.1 . (2011, May). *Object Management Group*.

Product Life Cycle support (PLCS) Web services V2. (n.d.). *OASIS* . http://www.plcs-resources.org/plcs_ws/v2/ .

QLM. (2012). *An introduction to Quantum Lifecycle Management (QLM) by The Open Group QLM work Group*. Retrieved from http://docs.media.bitpipe.com/io_10x/io_102267/item_632585/Quantum%20Lifecycle%20Management.pdf

R. Spiegel. (2009, 4 1). *What is FDT*. Retrieved from https://www.automationworld.com/article/automation-strategies/control/what-fdt

Sanjay, J., Ngai Fong, C., Khin Maung, A., & Ming, L. (2001). Virtual factory: an integrated approach to manufacturing systems modeling", . *International Journal of Operations & Production Management, Vol. 21 Issue: 5/6, ,* pp.594-608, https://doi.org/10.11.

Schulz, D. (2015). FDI and the Industrial Internet of Things. . *In Emerging Technologies & Factory Automation (ETFA), 2015 IEEE 20th Conference on* (pp. 1-8). IEEE.

Shani, U., Franke, M., Hribernik, K. A., & K. D. Thoben. (2017). Ontology mediation to rule them all: Managing the plurality in product service systems. *2017 Annual IEEE International Systems Conference (SysCon),* (pp. 1-7). Montreal, QC: IEEE.

Simon, R., Diedrich, C., Riedl, M., & Thron, M. (2001). Field device integration. *In Industrial Electronics, Proceedings. ISIE 2001. IEEE International Symposium on.* (pp. (Vol. 1, pp. 150-155)). IEEE.

Spiegel, R. (n.d.).

Srinivasan, V. (2011). An integration framework for product lifecycle management. *Computer-aided design, 43(5)*, pp.464-478.

Srinivasan, V., Lämmer, L., & Vettermann, S. (2008). On architecting and implementing a product information sharing service. *Journal of Computing and Information Science in Engineering*, 1-11.

Talal, B. K., & Rachid, M. (2013). *Service Discovery--A Survey and Comparison.* Retrieved from Talal, B. K., & Rachid, M. (2013). Service Discovery--A Survey and Comparison. arXiv preprint arXiv:1308.2912.

Thoben, K. D., Wiesner, S., & Wuest, T. (2017). “Industrie 4.0” and Smart Manufacturing–A Review of Research Issues and Application Examples. *International Journal of Automation Technology Vol, 11(1).*

Tolio, T., Sacco, M., Terkaj, W., & Urgo, M. (2013). Virtual Factory: An Integrated Framework for Manufacturing Systems Design and Analysis. *Procedia CIRP, Volume 7,* 25-30, https://doi.org/10.1016/j.procir.2013.05.005.

VDI/VDE GMA. (2015). *Reference Architecture Model Industrie 4.0 (RAMI4.0).* https://www.zvei.org/fileadmin/user_upload/Presse_und_Medien/Publikationen/2016/januar/GMA_Status_Report__Reference_Archtitecture_Model_Industrie_4.0__RAMI_4.0_/GMA-Status-Report-RAMI-40-July-2015.pdf .

Wang, X., & Xu, X. (2013). An interoperable solution for Cloud manufacturing. *Robotics and Computer-Integrated Manufacturing, 29(4)*, pp.232-247.

WIB Test Confirms Value Of FDT/DTM Technology For Asset Management. (2008, 2 4). Retrieved from https://fdtgroup.org/wib-test-confirms-value-fdtdtm-technology-asset-management/

Yamamoto, M., & Sakamoto, H. (2008). FDT/DTM framework for field device integration. *In SICE Annual Conference 2008* (pp. 925-928). IEEE.

Zadeh, N. S., Lindberg, L., El-Khoury, J., & Sivard, G. (2017). "Service Oriented Integration of Distributed Heterogeneous IT Systems in Production Engineering Using Information Standards and Linked Data. *Modelling and Simulation in Engineering* .

Zäh, M., & Reinhart , G. (2004). *Virtuelle Produktionssystem-Planung Virtuelle Inbetriebnahme und Digitale Fabrik.* München: Utz Verlag.