FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

A Graphical Toolkit for ISA-95

Empowering End Users to Develop Bridges between ERP and MES

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Business Informatics

by

Laurens Lang, B.Sc.
Registration Number 11719751

to the Faculty of Informatics

at the TU Wien

Advisor: a.o. Univ.-Prof. Mag. Dr. Christian Huemer
Assistance: Bernhard Wally, MSc

Vienna, 1st April, 2020

Laurens Lang | Christian Huemer

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Danksagung

An dieser Stelle möchte ich allen Personen danken, die mit ihrer Unterstützung zum Gelingen dieser Arbeit beigetragen haben. Ich möchte meinem Betreuer Bernhard Wally danken. Er hat mir nicht nur mit großer Hilfsbereitschaft und kritischen Hinweisen bei der Erstellung dieser Arbeit geholfen, sondern auch viele neue Einblicke in das wissenschaftliche Arbeiten gegeben. Die Exkurse nach Prag, Linz und Hamburg haben mir sehr gut gefallen und ich möchte mich herzlich bedanken, dass mir so viel Raum für das Ausleben meiner Ideen gegeben wurde. Außerdem konnte ich extrem viel von ihm lernen.

Außerdem möchte ich Herrn Prof. Dr. Christian Huemer danken, der es mir ermöglicht hat, diese Arbeit in der Business Informatics Group (BIG) zu schreiben.

Desweiteren möchte ich Prof. Dr. Manuel Wimmer und dem Christian Doppler Forschungslabor für Intelligente Modellintegrierte Produktion, die es ermöglicht haben, dass ich die Inhalte der Arbeit in zwei Papers publizieren durfte.

Außerdem danke ich dem Czech Institute of Informatics, Robotics and Cybernetics für die Einblicke in ihre Arbeit und den zwei Einladungen nach Prag. Die Case-Study hat der Arbeit die nötige Erdung gegeben.

Ich danke darüber hinaus allen Probanden, die sich Zeit genommen haben und an meiner Usability Studie teilgenommen haben.

Weiterhin danke ich meiner Familie, die mir durch ihre uneingeschränkte Unterstützung das Studium und dadurch auch diese Arbeit ermöglicht hat.

Kurzfassung

Als die vierte industrielle Revolution ausgerufen wurde, begeisterte Industrie 4.0 Unternehmen und Forscher, indem sie ihnen die Möglichkeit aufzeigte, den Weg für die Vision eines vernetzten Smart-Factory-Ökosystems zu ebnen. Um diese tiefgreifende Digitalisierung der industriellen Produktion zu erreichen, ist eines der Forschungsschwerpunkte die horizontale und vertikale Integration, die sich durch die Integration von Industriestandards realisieren lässt. Um die Vernetzung der Vielzahl von Industriestandards zu bewältigen, ist ein modellgetriebener Ansatz zur Transformation dieser Standards sinnvoll. Unter der Fülle von Industriestandards, die im Kontext der intelligenten Fertigung verfügbar sind, wird eine Normenreihe für das Management von Fertigungsabläufen konsequent erwähnt: IEC 62264. Der zweite Teil enthält ein konzeptionelles Modell für die Beschreibung von Produktionssystemen und deren Fähigkeiten, einschließlich Laufzeitinformationen wie konkrete Wartungspläne oder erreichte Produktionsziele. In dieser Arbeit präsentieren wir eine konkrete grafische Syntax und ein Toolkit für die Erstellung und Darstellung von IEC 62264-2 konformen Modellen unter Verwendung von Techniken aus dem modellgetriebenen (Software-) Entwicklung. Der ISA-95 Designer wurde evaluiert, indem wir eine Nutzerstudie durchgeführt haben, um die Usability und Funktionsfähigkeit zu evaluieren. Infolgedessen ermöglicht das ausgereifte grafische Toolkit Endbenutzern ohne Programmierkenntnisse die Modellierung automatisierter Produktionssysteme.

Abstract

The dawn of the fourth industrial revolution, Industry 4.0 has created great enthusiasm among companies and researchers by giving them an opportunity to pave the path towards the vision of a connected smart factory ecosystem. In order to achieve this extensive digitization of industrial production, among the main research issues are the horizontal and vertical integration, which can be realized with the help of industrial standards. In order to handle the interconnected nature of the huge variety of industry standards, a model-driven approach to transform these standards is meaningful. Among the plethora of industrial standards available in the context of smart manufacturing, one series of standards is consistently being mentioned for dealing with manufacturing operations management: IEC 62264. Its second part provides a conceptual model for the description of production systems and their capabilities, including runtime information such as tangible maintenance schedules or achieved production goals. In this work, we present a concrete graphical syntax and toolkit for the creation and presentation of IEC 62264-2 compliant models, using techniques from model-driven (software) engineering. We have evaluated our tool by conducting a user study for assessing its usability and effectiveness. As a consequence, the engineered graphical toolkit empowers end users with no coding skills to model automated production systems.

Contents

CHAPTER 1

Introduction

1.1 A Brief History of Manufacturing Technology

In order to realize the high significance of the fourth industrial revolution, it is important to study the previous industrial revolutions [DK15]. In recent history of humankind, there have been three major stages of manufacturing development, also called *industrial revolutions*, which had an enormous and lasting impact on economic and social conditions [DK15].

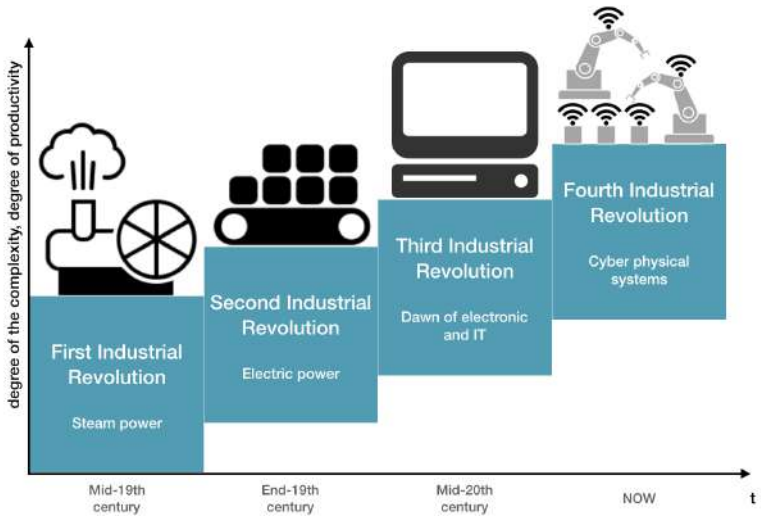


Figure 1.1: Time-dependent increasing complexity and productivity of manufacturing technology based on [DK15] [DH14].

Figure 1.1 shows evolution stages from a production engineering point of view over

time. The diagram shows that there has been a continuous increase in complexity and productivity at shorter intervals and that it will continue with a fourth revolution.

The **first industrial revolution** was initiated with the spread of mass production through the use of machinery, such as the mechanical loom. Inventions, such as the steam engine, led to fundamental innovations in many industries, such as heavy industry, transportation, and coal mining. The novel use of hydropower served as the basic energy for increasing production at this stage of development [DH14].

At the end of the 19th century, the availability of national electric power grids, the unstifled growth of pre-existing industrial endeavors in petrochemical, electronic and steel production gave birth to what is now documented as the **second industrial revolution**. By using engines, it was now possible to automate work in the factories more efficiently. In addition, the increase in production through assembly line work, new telecommunication techniques, and inventions in the aviation and shipping industries were the first steps towards the formation of a global trade market [DK15].

In the 1970s, the development of calculating machines and digitally programmable control units replaced the prevailing analogue logics and formed the basis for the continuing trend towards automation. This **third industrial revolution** peaked in highly automated industrial plants and modern process control systems for more efficient, nearly fully automated production systems [DK15].

1.1.1 Computer Integrated Manufacturing

In the third industrial revolution, **Computer Integrated Manufacturing** (CIM) systems were used. They integrated hard- and software, which were needed for computer graphics, computer-aided modeling, computer-aided design, manufacturing activities, from its initial product concept through its production and distribution in the marketplace and made it possible to (i) respond to rapid changes in product design modifications and therefore for changing market demands, (ii) better the use of materials, machinery and human resources, (iii) reduce inventory, (iv) improve the overall management of the manufacturing production.

The following innovations are components of *CIM* [KSS16] [Sch13]: (i) *Computer numerical control (CNC)*: These are machine tools that use modern control technology to automatically produce workpieces with high precision even for complex shapes with the help of coded instructions in the form of numerical data. (ii) *Adaptive control* Adaptive control is called in cybernetics control a control that can adapt its parameters to the process. Adaptive control is used because a normal control can only compensate relatively small changes in the process dynamics. Therefore, there is the adaptive control for larger, slow changes where a static controller can not work satisfactorily. Examples include supersonic aircraft, whose behavior changes greatly with each Mach, as well as unknown or changing delays or disturbances. (iii) *Industrial robots* Industrial robots can replace humans, especially in highly repetetive, dangerous and boring operations, such as assembling industry. As a result, variability and errors in product quality are decreased

and productivity is improved. (iv) *Automated material handling* such as moving a part from one machine to another; transport them to inventory, inspection and to shipment. (v) *Automated assembly systems* Assembly costs are high. Products are designed to faster by automated machinery, thus reducing the total manufacturing cost. (vi) *Computer-aided process planning* are systems, which are capable of improving productivity, product quality, and consistency. (vi) *Group technology* Parts can be grouped and produced by classifying them into families, so part designs and process plans are standardized. (vii) *Just-in-time production* Materials are delivered just in time to be used in order to lower inventory costs. (viii) *Cellular manufacturing* This system utilizes workstations that consist of a number of manufacturing cells, each containing various production machines controlled by a central robot, with each machine performing a different operation on the part, including inspection. (ix) *Flexible manufacturing systems* Consisting basically of complex computer programs, these systems have the capability of performing various tasks and solving difficult real-life problems, much as human experts would, including expediting the traditional iterative process in design optimization. (x) *Expert systems* An expert system contains the functionality to create and improve the knowledge base, to process (inference engine) and to make the user understandable (explanatory component). (xi) *Artificial intelligence* Systems integrate the automation of intelligent behavior and machine learning. (xii) *Artificial neural networks* CIMs use information processing vaguely inspired by the biological neural networks that constitute animal brains, which include concepts like Neurons, connections, weights, and propagation functions.

CIM systems comprise the following subsystems, which are integrated: (i) *Business planning and support* integrate tasks like forecasting, scheduling, material-requirements planning, invoicing, and accounting (ii) Product design (iii) Manufacturing process planning (iv) Process automation and control (v) Production-monitoring systems.

CIM systems have on the one hand the following benefits: (i) Better process control increases product quality and uniformity (ii) Lower Product costs and higher productivity through efficient use of resources (iii) Better control of the whole manufacturing operation (iv) Enables shorter product life cycles in the context of global competition and changing market demands.

On the other hand, these CIM systems are often monolithic, too complex and non-reusable [CAGT15]. This resulted in the so-called *CIM crisis* [Zü10].

1.2 Industrie 4.0

The **fourth industrial revolution** was called into being by the German government in terms of their high tech strategy. So, for the first time, an industrial revolution is predicted apriori, not observed ex-post [DH14]. The *Platform Industry 4.0*[1] defines it as digitization and networking of entire value chains and follows mechanization, electrification, and automation at all stages of the value chain. Industry 4.0 involves both, upstream and

[1] Industrie 4.0 Österreich – die Plattform für intelligente Produktion – https://plattformindustrie40.at/

downstream, actors such as suppliers or logistics companies as well as internal processes such as procurement, production, sales or maintenance. Industry 4.0 leads to higher productivity and flexibility, more innovation as well as lower resource consumption.

Acatech[2] defines the concept in its report and includes all areas of entrepreneurial creativity: Industry 4.0 basically means the technical integration of cyber-physical systems (CPS) into production and logistics, as well as the application of the Internet of Things and services in industrial processes - including the consequent consequences for value creation, the business models as well as the downstream services and work organization [aca13].

All in all, this catch-all phrase summarizes the following three interconnected factors [ZMVS16]: (i) Digitization and integration of any simple technical-economical relation to technical-economical complex networks (ii) Digitization of products and services (iii) New business models.

Besides, Industry 4.0 is based on six design principles which are (i) interoperability (ii) virtualization (iii) decentralization (iv) real-time capability (v) service orientation and (vi) modularity [HPO15].

Across the big pond, the transatlantic cousin of Industrie 4.0, the *Industrial Internet Consortium* (IIC)[3], was founded by GE, IBM, CISCO, Intel, and AT&T. In contrast to European research, it focuses not only on manufacturing systems but also on energy, healthcare, and infrastructure [LMF16]. In this context, it is also named *Industrial Internet of Things*.

The Acatech report also mentions important action fields, on which to make industrial and industrial policy decisions should be made in order to support the economy in implementing the digital transformation [aca13] [Hei15]: (i) Standardization and a reference architecture for cross-company networking (cf. Subsection 1.2.1) (ii) Mastery of complex systems through adapted explanatory models (iii) Nationwide broadband infrastructure (iv) Operational and security of intelligent production systems (v) Change in work organization, as well as continuous education and training of employees (vi) Legal framework for data protection and innovative corporate structures (vii) Identification of methods for increasing resource efficiency

1.2.1 RAMI 4.0

In order to approach the manifold challenges and technologies in a structured manner, *The Reference Architectural Model for Industrie 4.0* (RAMI 4.0) was developed by the industrial associations BITCOM [4], VDA [5] and ZVEI [6], see Figure 1.2 [HR15] [Deu16].

[2] Deutsche Akademie der Technikwissenschaften: https://www.acatech.de/
[3] https://www.iiconsortium.org
[4] Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V.
[5] Verband Deutscher Maschinen- und Anlagenbau e.V.
[6] ZVEI – Zentralverband Elektrotechnik- und Elektronikindustrie e.V.

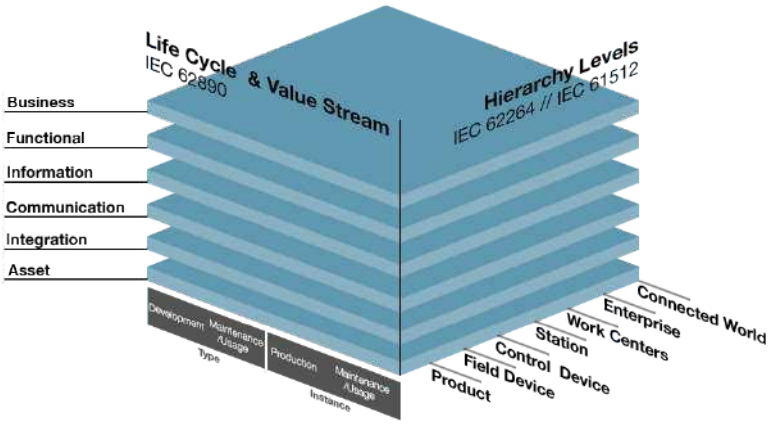


Figure 1.2: Reference Architecture Model Industrie 4.0 (RAMI4.0) [HR15] [Deu16].

RAMI 4.0 helps to better identify possible gaps and overlaps between applications. The three dimensions of the model are represented by the aspects Life Cycle and Value Stream (IEC 62890), Hierarchy Levels (according to IEC 62264 and IEC 61512, expanded by Product and Connected World) and interoperability in corresponding layers [ZMVS16]. RAMI 4.0 is a reference three-dimensional model, which combines the crucial elements of Industrie 4.0. Based on this framework, Industrie 4.0 technologies can be classified and further developed [Deu16]. The three dimensions are:

Life Cycle and Value Stream are represented on the left horizontal axis. It integrates IEC 62890. It is differentiated between product types and instances, which are physically manufactured.

Layers are represented on the vertical axis, which describes the decomposition of a machine into its properties structured layer by layer, i.e. the virtual mapping of a machine. Such representations originate from information and communication technology, where properties of complex systems are commonly broken down into layers. Started from the bottom, the layers are: asset, integration, communication, information, functional and business.

Hierarchy Levels are represented on the right horizontal axis, derived from IEC 62264, also called ISA-95. In order to represent the Industrie 4.0 environment, the hierarchy levels *Product*, and the connection to the Internet of Things and Services, named *Connected World*, were added.

The following subsection 1.3 will give a systemic overview of the standards in the context of the *Automation Hierarchy*, which are defined in these standards.

1.3 Industry 4.0 Standardization Landscape

Nowadays, markets change rapidly and consumers demand constant increasing product quality. The ability of different systems to exchange, understand, and utilize product, production, and business data relies heavily on information standards [LMF16]. The usage of standards makes it possible for business owners to adopt technologies and innovations. Taking a closer look at the *Standard Landscape of Industry 4.0*, the following separation is within the framework of the Automation Hierarchy of IEC 62264, which is shown in Figure 1.3 [Int13a]. Lu et. al. gives an overview of the *Current standards landscape for smart manufacturing* [LMF16]:

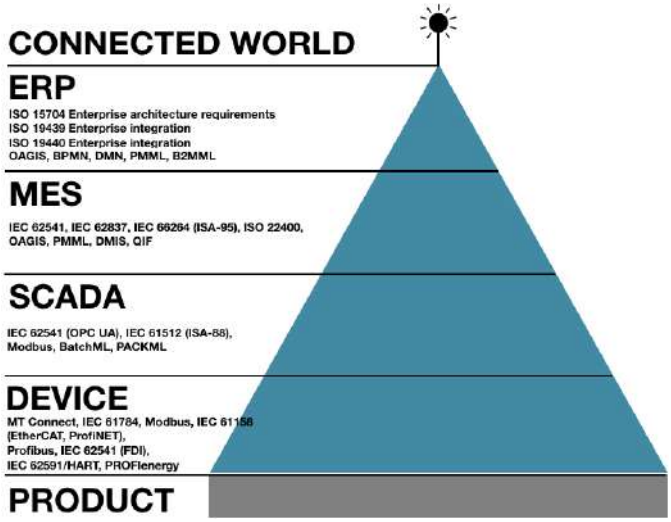


Figure 1.3: Automation hierarchy derived from IEC 62264 [Int13a].

1.3.1 Enterprise Level: Enterprise Resource Planning

ISO Standard 15704 defines requirements for enterprise reference architectures and methodologies [ISO05]. *ISO 19439* and *ISO 19440* define the enterprise integration: a framework and specific characteristics of the main components needed for enterprise activities [ISO06b] [ISO07]. With the help of the *Predictive Model Markup Language (PMML)* by the Data Mining Group (DMG) statistical and data mining models can be developed, which can also be used on the MES level [Pec09]. *Decision Model and Notation (DMN)* by OMG is defined to close the gap between business decision design and decision implementation [ASE16]. *ISO 20140* defines general principles of a method for assessing the environmental influences of manufacturing production systems [ISO13]. OMG's famous *Business Process Model Notation (BPMN)* is a notation used for designing manufacturing business processes [OMG11], which is also manifested in ISO/IEC 15944-4:2015 [Sta15]. Besides, OAGIS was developed to establish a common content model for enterprise application integration [KD10]. Lastly, REA is an onthology to model economic exchanges such as accounting transactions (cf. Section 6.2.1 and 5.2.1) [McC82].

1.3.2 Manufacturing Execution System/Manufacturing Operations Management Level

OPC UA defines a communication standard. *IEC 62264* also called ISA-95 defines enterprise control system integration: activity models, function models, and object models in the MES domain. *Business to Manufacturing Markup Language (B2MML)* is an implementation of *IEC 62264* to link ERP and MES systems (cf. Section 5.2.2). *ISO 22400* defines key performance indicators (KPIs) in manufacturing operations management [ISO14]. *QIF* defines the flow of information of computer-aided quality measurement systems. *PMML* from DMG can also be used at this level.

1.3.3 SCADA and device level standards

As communication systems, EtherNet/IP, DeviceNet, ControlNet, PROFINET, and EtherCAT are used [Bro01]. Fieldbuses like PROFIBUS, CAN bus, HART, and Modbus connect the PLCs to the field components. As communication protocol, *Modbus* is often used. In addition, *OPC* and *OPC UA (Unified Architecture)*, *MTConnect*, *PackML* and *BatchML* are used to link SCADA to the MES level. *MTConnect* defines the access of real-time data from shop floor manufacturing equipment. *ISA 88* defines the physical model, procedures, and recipes, which is also called *IEC 61512*. *PackML* defines the packing used in the batch processing industry according to ISA 88 standards. *BatchML* is the implementation of ISA 88 for linking batch control systems to MES [LMF16].

1.4 Model-driven Smart Factory

Approaches like Industry 4.0 aim to enable flexible production networks that require horizontal integration across companies. Thus, there is also production-related information exchanged in the network, which must be vertically forwarded to the corresponding service endpoints of the local production system [WHM17b] [MWH+19]. Manufacturing companies run different software in order to control the broad range of operations from strategic management to the actual production process. The consequence of the heterogenity of industries, vendors, brands, and technologies is a great variety of solutions.

Cadavid et al. [CAGT15] point out the following requirements for a *Smart Factory* in order to develop, apply and distribute innovative technologies for production and industrial plants: (i) Such systems have to be highly adaptable and reconfigurable. (ii) They have to be modular in order to stay flexible. CPS are complex systems, which consist of a large number of components (sensors, actuators, tools, software), which have to work together seamlessly as one entity and interact with other complex systems as well. Nevertheless, modularity has to be considered in order to prevent a second *CIM crisis*. The components have to be reusable. (iii) The factory has to be connected to the upper management levels of the enterprise. (iv) Governmental regulators require a system that is precisely traceable throughout and beyond the production chain. (v) The concerns

of mechanical, electrical, and control designs have to be separated and semantically decoupled.

The following artifacts are derived to realize the vision of a model-driven smart factory [CAGT15]: (i) **Model library of manufacturing processes** offers a collection of manufacturing actions, which can be chained to form complex processes. (ii) **Architecture Viewpoint Description** is composed of an architecture, views, viewpoints, and stakeholders. (iii) **Automated and Context-Aware Generation of User Interfaces** The advances in recent times make possible the capitalization of models also to conceive the automatic generation of context-aware UIs. (iv) **Traceability Management across Models** can connect the technical system model, a production process model, and a MES functional model. (v) **Tailoring of Manufacturing Processes** in order to realize flexible production systems (vi) **Model-Based Simulation of Manufacturing Processes** to accurate and comprehensible simulation.

Mazak et. al. proposes a framework for model-based horizontal and vertical integration of standards for Industry 4.0, in which concrete industry standards can be aligned [MWH+19]: REA Ontologie, ISA-95, AutomationML, OPC UA (c.f. Section 5.2). The international standard IEC 62264 has been specified for data exchange and information alignment between enterprise resource planning (ERP) systems and MES and allows the modeling of MESs. This standard defines data models that relate the systems ERP and MES [WSH+17] [Int13b]. Considering its capabilities, IEC 62264 represents a promising future MES standard [LMF16].

1.5 Problem Statement

When companies with different systems choose to work together, interoperability needs to be addressed to enable smart manufacturing. This can be achieved through standards and interfaces (cf. Section 1.3). Taking the dynamic and complex nature of manufacturing into account, there is a need for various kinds of standards [TWW17]. Considering its capabilities, IEC 62264 represents a promising future for the MES domain [LMF16] and among the plethora of industrial standards available, this series of standards is consistently being mentioned for dealing with manufacturing operations management. It consists of five parts. Each part deals with another aspect of enterprise control system integration [Int13b]: IEC 62264-1:2013 describes models and terminology, IEC 62264-2:2013 describes objects and attributes for enterprise-control system integration, IEC 62264-3:2007 describes activity models of manufacturing operations management, IEC 62264-4:2015 describes object model attributes for manufacturing operations management integration, IEC 62264-5:2011 describes business to manufacturing transactions [Wal18].

However, models and requirements for ISA-95 are only abstractly sketched. Important implementation details and definitions are missing. Therefore, contemporary domain experts do not have the appropriate tool support to model ISA-95 compliant models. Because of the multidisciplinary nature of automated production systems, it is necessary to support collaborative development by a variety of stakeholders with diverse back-

grounds [FHK+15]. Therefore an appropriate toolchain could play a decisive role in the adoption of this standard. This toolchain is not available yet.

Note: Since ISA-95 and IEC 62264 are essentially the same, this thesis will from now on refer to both standards by using the term ISA-95. In ISA-95, basic ressources like equipments, physical assets, materials and personnel can be defined. These can be synthesized to production segments and operation definitions.

CHAPTER 2

Research Method

In order to set the purpose for this thesis, the following paragraph sets forth research questions (c.f. Section 2.1) and research contributions (c.f. Section 2.2) and research method (c.f. Section 2.3).

2.1 Research Questions

The research questions are based on the initial problem statement, that contemporary domain experts do not have appropriate tool support to model ISA-95 compliant models.

ISA-95 is an international standard that defines structures that can be used to link the enterprise resource planning layer to the manufacturing execution system. Therefore, in this thesis, a graphical domain-specific language will be implemented.

The answering of Research Question 1 gives some indication of the expressiveness of ISA-95 and therefore of the effectiveness of such a tool. The effectiveness of a tool is the breadth of ideas that can be represented and communicated in that language. The more expressive a language, the greater the variety and quantity of ideas it can be used to represent. However, not only the effectiveness but also the efficiency of the tool is relevant. Effectiveness means that all information is presented clearly and quickly in a cost-effective manner. order to empower domain experts to implement manufacturing execution systems.

Research Question 1

How helpful is ISA-95 Designer in contrast to a generic editors based on a metamodel?

Design science research propagates an iterative approach. Therefore in a second iteration, the usability could be improved through advanced visualizations, simplifications, abstractions, shortcuts and interaction concepts, which will be addressed by Research Question 2 [HMPR04].

Research Question 2

Which simplifications and shortcuts of ISA-95 models are required by domain experts to structure and accelerate the development process?

The expressiveness of the graphical domain-specific language (DSL) and ISA-95, in general will be tested by modeling parts of the *Industry 4.0 Testbed* of the Czech Institute of Informatics, Robotics, and Cybernetics [1] (CIIRC) in Prague.

2.2 Research Contributions

This thesis aims to contribute two main artifacts along with corresponding evaluation: a graphical DSL and a sample model created with it.

2.2.1 Graphical DSL for Manufacturing Environments

Based on a metamodel, a graphical domain-specific language is to be developed in order to empower domain experts to model production systems and processes. This will be realized with the help of Eclipse sirius [2].

2.2.2 Modeling Of Sample Model

The graphical DSL will be tested by modeling Industry 4.0 Testbed of the Czech Institute of Informatics, Robotics, and Cybernetics (CIIRC) in Prague.

2.3 Research Method

Information System (IS) research is based on two paradigms: behavioral science and design science. Design Science is aimed at solving problems, derived from the engineering domain - behavioral science at explaining or predicting human phenomena [HMPR04]. Design science research targets building and evaluating innovative IT artifacts and is, therefore, a good fit for the development and evaluation of a graphical domain-specific language for ISA-95.

[1] https://www.ciirc.cvut.cz/
[2] http://www.eclipse.org/sirius/overview.htmlx

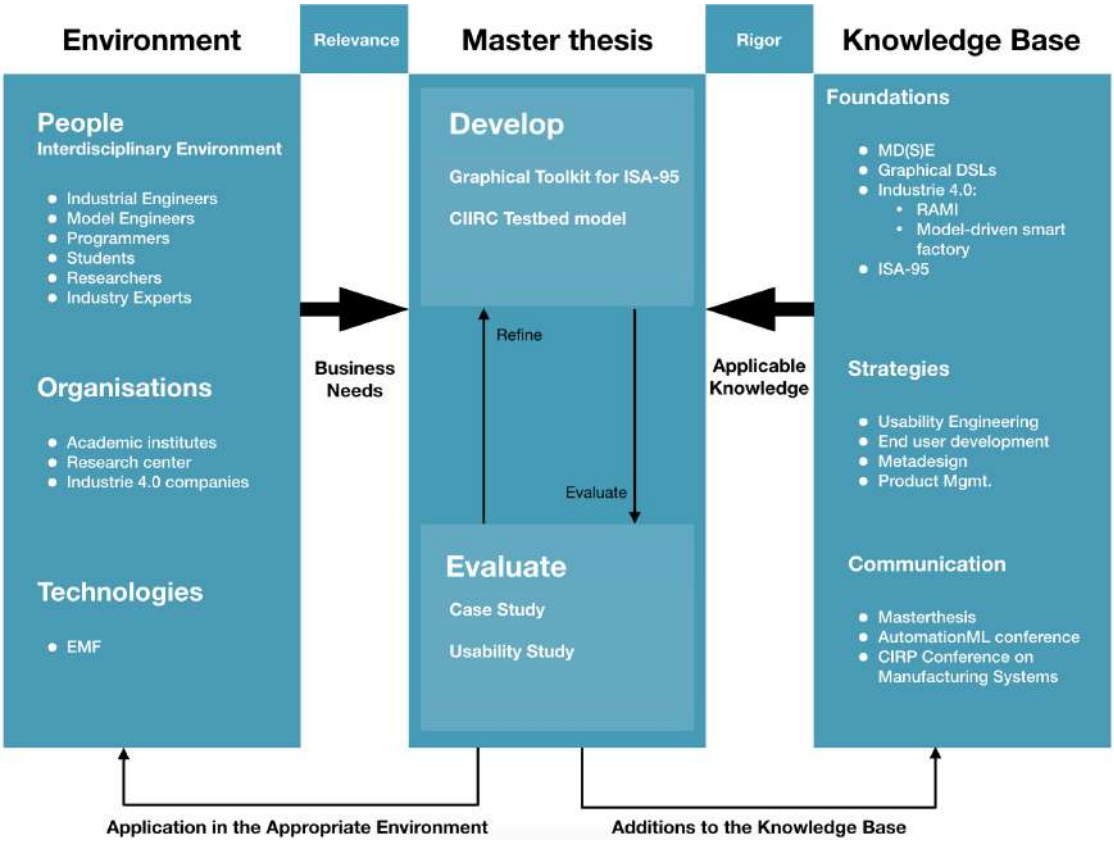


Figure 2.1: Research Framework adapted from [HMPR04].

In the following, the research framework and contributions are discussed using the seven design science research guidelines in order to prove that the framework meets reliable research standards [HMPR04] .

Guideline 1 (Design as an Artifact). This thesis aims to produce a viable instantiation artifact. In particular, a graphical domain-specific language is created in order to empower domain experts to develop and maintain an ISA-95 model. The proof-of-concept prototype demonstrates the feasibility in terms of the defined research questions. The evaluation of the developed constructs and methods is accomplished through a case study and a usability study.

Guideline 2 (Problem Relevance). The relevance of conducted design science research shall be assessed towards the community. ISA-95 is an international standard describing structures that can be used to link the enterprise resource planning (ERP) layer to the manufacturing execution system (MES) layer and to model ISA-95 Models [WHMW18a]. In contrast to other standards, there is no appropriate tool for *ISA-95* available.

Guideline 3 (Design Evaluation). The artifacts must be evaluated based on empirical evidence. Relevant metrics are: 1. time 2. effectiveness (accuracy and completeness) 3. satisfaction questionnaire Therefore we conduct a user study and a case study.

Guideline 4 (Research Contribution). This thesis aims to contribute a graphical modeling language for production systems and processes, a reference model of the CIIRC testbed in Prague as well as the evaluation of both artifacts.

Guideline 5 (Research Rigor). The point of origin of this thesis is existing literature and artifacts. Starting from this, the artifacts and methods are created. In particular, we make use of existing knowledge about modeling, visualization, abstraction, evolution of production standards. Figure 2.2 shows, which research fields are addressed. While there is research in the combination of *Model-driven Software Engineering* and *ISA-95* and the combination of *MDSE* and *Usability*, there is no research in the combination of the three fields of *ISA-95* and *Usability* as well as *MDSE*, *ISA-95* and *Usability*.

Guideline 6 (Design as a Search Process). Design science suggests an iterative process to create a satisfying and effective solution for realistic IS problems.
Models and artifacts are created, discarded, optimized und refined in each step. In the first step, the ISA-95 and the existing Ecore Models are reviewed. In the second step, a graphical domain-specific language is created and checked if it meets the requirements. In the third step, usability improvements are proposed, discussed, implemented and evaluated. From this evaluation and further literature reviews, possible future improvements and changes can be derived.

Guideline 7 (Communication of Research). The developed results are communicated to researchers as well as to practitioners. They should benefit from these research results. This will be realized through this thesis, its defense, several seminars on the one side. In addition, it was presented at the *AutomationML Plugfest 2019* in Hamburg [WLW+19]. Furthermore, it was accepted for publication to the *53rd CIRP Conference on Manufacturing Systems 2020*, Chicago [LWH+]. In order to promote the tool and its research, a website was set up and it will be filled up with textual and video documentation on `http://isa95.info/`. Subsequently, to rank higher in search engine results, backlinks are collected. Suitable collaborations can be MDE, Sirius and automation niche blogs.

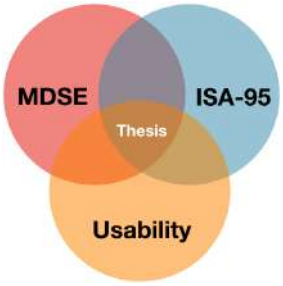


Figure 2.2: The combination of the different research fields implies the research rigor.

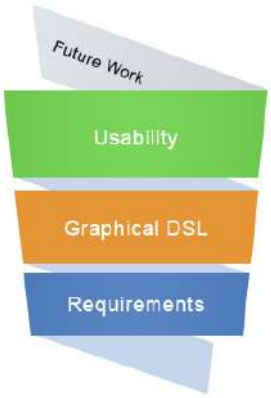


Figure 2.3: Iterative Process

CHAPTER 3

Model-Driven Engineering

"You never change things by fighting the existing reality. To change something, build a new model that makes the existing model obsolete."

— Richard Buckminster Fuller

Why modeling is at the core of many disciplines, it is important in engineering because it facilitates communication and helps to handle complexity by dividing into smaller cognitively managable parts [Sel03] [Tho04b]. Model engineering treats software development as a set of transformations between successive models from requirements to analysis, to design, to implementation and to deployment [Tho04a].

This chapter is structured as followed: Section 3.1 defines the term *model*. In Section 3.2, *Model-driven Engineering* is explained, Section 3.3 presents Model-driven Architecture, a more standardized process. Section 3.4 expands on the requirements of Modeling Languages. Finally, in Section 3.5 the *Technologies* explained the technologies used for the artifacts.

3.1 What is a Model?

A model is a simplified image of reality [Sel03]. According to Stachowiak, three fundamental model properties exist [Sta73]:

Mapping A model is always a model of a representation or representation of an original. These originals can be physical, ideas, symbols or terms.

Reduction A model reflects only a relevant selection of the original's attributes.

Pragmatism A model is not only a model of something, but also for someone or for some purpose.

The most important of these characteristics is the *abstraction* (cf. reduction) in order to reduce complexity. It allows modelers and viewers to focus on relevant aspects only.

3.2 Model-Driven Engineering

In the 1980s, computer-aided software engineering (CASE) arose, which focused on developing software methods and tools with the help of general-purpose graphical programming representations, e.g. state machines, structure diagrams, and dataflow diagrams, which promised to separate concerns [Cas85]. But in practice, it was not widely adopted, because of its poorly mapping onto the underlying platforms and inability to scale to handle complex, production-scale systems in a lot of application domains.

Model-driven Engineering (MDE) is a promising approach to address platform complexity - and the inability of third-generation languages to alleviate this complexity and express domain concepts effectively - is to develop Model-Driven Engineering that makes use of the following concepts [Sch06] [Tho04b]:

Domain-specific modeling languages are used to formalize the application structure, behavior, and requirements within particular domains in a model.

Transformation engines and generators analyze certain aspects of models and then generate various types of artifacts, such as source code, simulation inputs, XML deployment descriptions.

First, a meta-model is developed. Second constraints define restrictions to the instances of the meta-model, which minimize error-prone behavior. Therefore, it is often much easier to develop, debug, and develop MDE tools and applications created with these tools [Sch06].

So, *Model-Driven Engineering* (MDE) focuses more on the core domain and its concepts and logics rather than on algorithmic and implementation solutions. MDE follows the *Don't repeat yourself*-Principle (DRY-Principle), which implies to avoid or reduce redundancy. MDE also called model-driven development (MDD), models are used to achieve a higher level of abstraction in the engineering process.

Subsection 3.2.1 gives an overview of the overall vision of *Model-driven (Software) Engineering* and the main interconnected components of MDE.

3.2.1 MD(S)E Vision

The core concepts of MD(S)E are models and manipulating operations on these models, called model transformations. The famous formula *Algorithms + Data Structures = Programs* from Niklaus Wirth can be recontextualized within *Model-driven Software Engineering* [Wir78] [BCW12]:

Models + Transformations = Software

Figure 3.1 shows this refined vision for system development of *Model-driven Software Engineering* (MDSE) [BCW12].

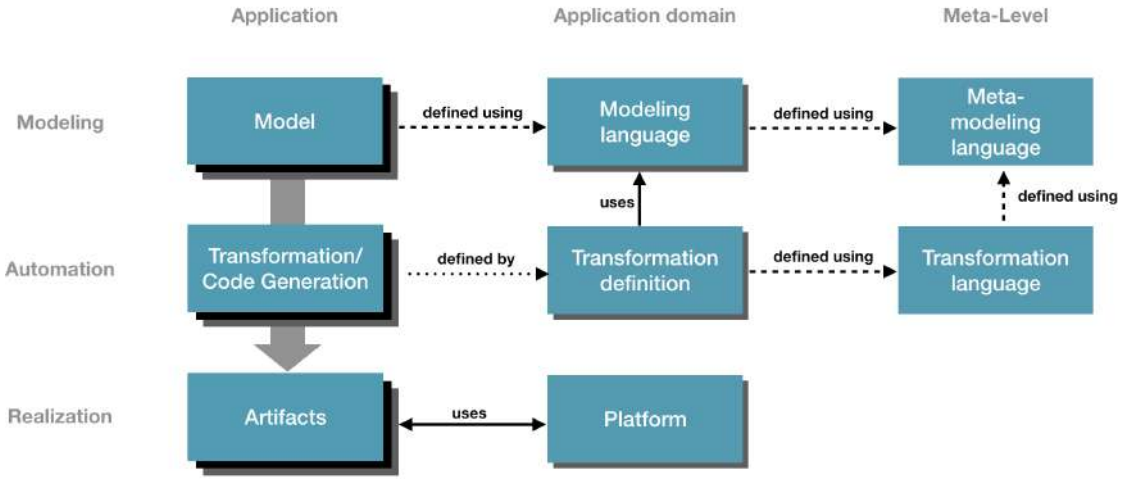


Figure 3.1: The overall MDSE Vision based on [BCW12].

The three vertical dimensions deal with **implementation**. The implementation is realized with the mapping of models to existing or future artifacts:

Modeling Level On the modeling level, modeling language, and meta-modeling language defined. The model is defined by the modeling language, which is defined by the meta-modeling language.

Automation level On the automation level, the mapping from the modeling level to the realization level is here located.

Realization Level On the realization level the solutions are implemented through artifacts. The artifacts are generated through the transformations and use a platform.

The horizontal dimension deals with the **conceptualization**, which has also three levels:

Meta-Level On the meta-level the conceptualization of models and language transformations are here defined.

Application domain level On the application domain level, the modeling language, transformations and implementations platforms for a specific domain defined.

Application level Here are models of the application defined. The transformation rules are performed and executable components are generated.

The software development process runs from the application model down to the running artifacts through transformations. In order to run the software on a specific platform, the models are specified in the paradigm of a modeling language, which relies again on a meta-modeling language.

3.2.2 Why MDE?

The model-driven approach offers the following advantages in contrast to common software development:

Abstraction MDE allows developing on a higher abstraction level. This reduces complexity because one layer contains the mapping of the model to the programming language and the other contains the actual complexity of the model. The modeler does not need specific implementation details of the used programming language [SVEH07].

Productivity arises as a result of a well-defined architecture and automated code generation, which directly affects the speed of development. As a result, this productivity boost manifests from a business perspective in shorter time to market, faster customer feedback loops, lower development costs, allows later changes, allows multiple customer segments through variants and a reduced need for outsourcing [PK02].

Consistent Architecture Using automated transformations in order to generate code from a formal model, MDE is a consistent architecture [SVEH07].

Reusability Using models promotes the usage of existing assets in some form within the software product development process. These assets can be products or by-products of a software development lifecycle. In MDE this can be: Code, models, metamodels, transformations, domain-specific languages notations and documentation [Sch06].

Interoperability and platform independence The approach increases the interoperability among target platforms and facilitates the platform independence since most of the system parts are defined in a technology-independent model. An impressive example of this strategy is OPC UA, which is further described in Subsection 5.2.4.

Software quality Although software quality is multifactorial, MDE promises to enable software quality through automated transformations and formally-defined modeling languages and it provides a process and tools to handle complexity [SVEH07]. MDE offers a productive environment in technology, engineering, and management fields through its use of process building blocks and best practices. It thus contributes to meeting the goals described here.

3.2.3 Metamodeling

A model represents an image of an original. In the context of model-driven software development, the model reflects one aspect of a software system. Models exist in

different representations e.g. in a graphical form of an UML class diagram or a textual representation in an XML file. It can be human and/or machine-readable. A metamodel represents an abstraction of a model. The metamodel, therefore, stands on a higher abstraction level than the model. An example is the *Meta Object Facility* (MOF). Also, the meta-model represents the formal description of the domain. This can be formulated in UML. The elements of a model represent the instances of the model elements of the associated meta-model. The relationships between meta-model, model and domain are shown in Figure 3.2. M3 and M2 are therefore language engineering and M1 and M0 domain engineering.

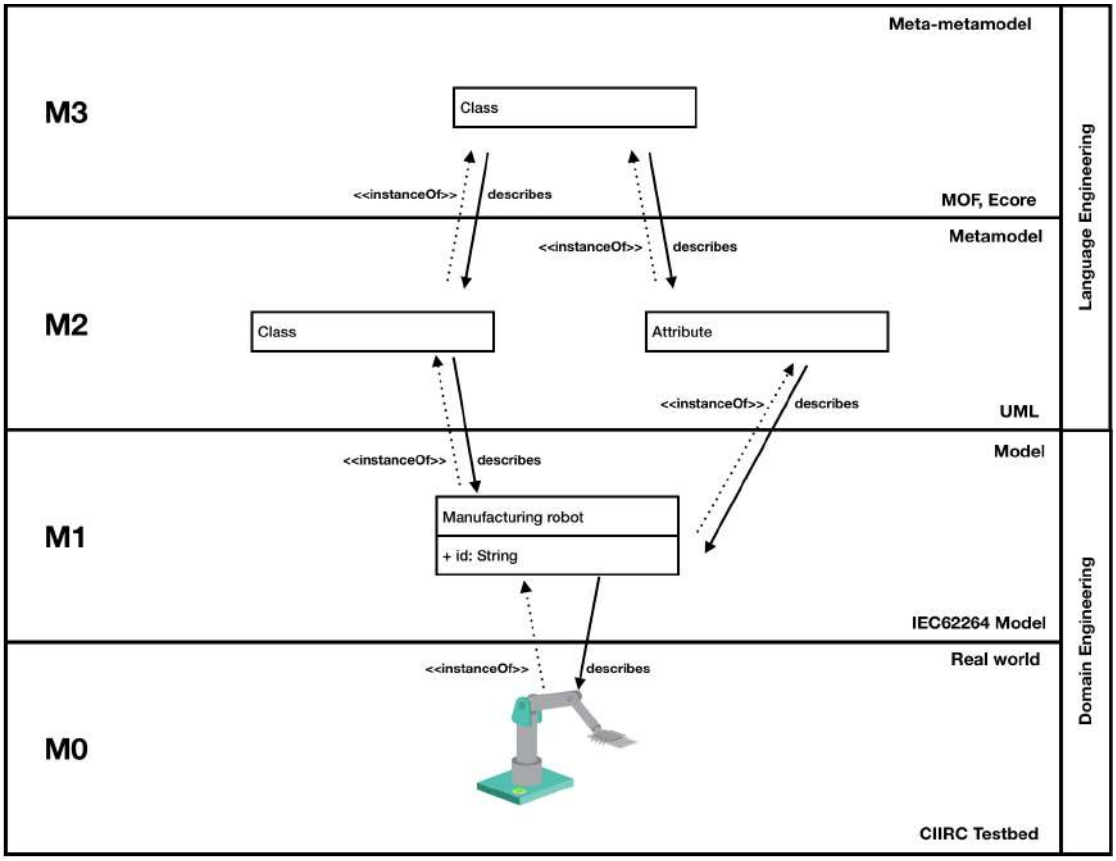


Figure 3.2: Four layer model hierarchies [BCW12].

In addition to the levels, there is an M3 model, which is called meta-metamodel.

Meta-metamodel The meta-metamodel represents the next level of abstraction of the model. The meta-metamodel is the metamodel of the metamodel. In theory, there can be an infinite number of meta-levels for a model. These include the meta-model level, the meta-model level and the model level itself. The meta-meta model is usually described by itself. The Meta Object Facility (MOF) is a popular meta-metamodel of the OMG.

For example, UML is based on this model. Another example of a meta-metamodel is Ecore, which is the abstraction of the Eclipse Modeling Framework (EMF). Ecore is also used for the approach described in this thesis.

3.2.4 Transformations

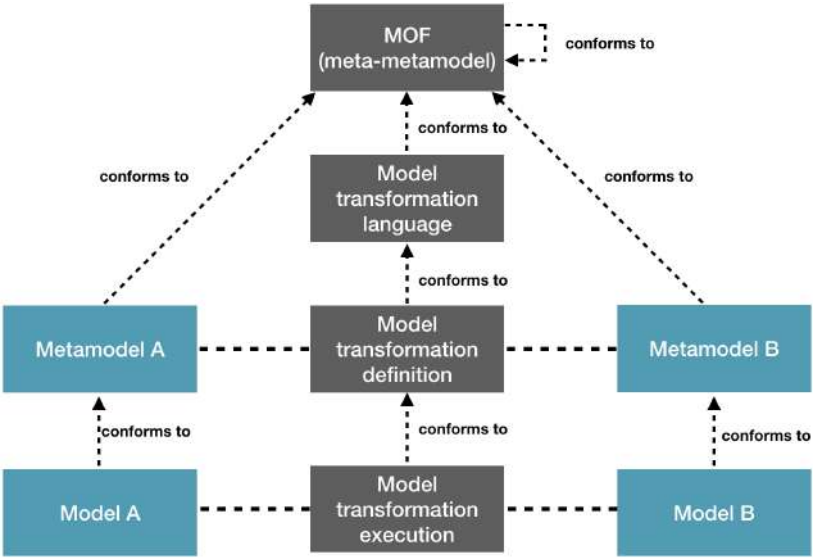


Figure 3.3: Role and definition of transformations between models [BCW12].

Consistency is a key challenge when working with multiple, interconnected models. The alignment, as well as other activities like view generation, reverse engineering, refactoring, can be optimized with the help of automation. These automated processes, which take one or more models as input and produce one or more target models as output, are called *model transformations* [SK03].

Model transformation can be formulated with a DSL, in which rules can be defined or produced automatically by higher-level mapping rules. Transformations are defined at the metamodel level, and then applied at the model level, upon models. The figure shows model A, which conforms to meta-model A and model B, which conforms to metamodel B, and a model transformation, that transforms model A to model B. Metamodel A and metamodel B as well as model transformation conform to the same meta-metamodel.

3.3 Model-Driven Architecture

Model-Driven Architecture (MDA) is an international standard of *Object Management Group* (OMG). The chief object of MDA is the standardization of model-driven techniques for software systems. *OMG* defines four principles of *MDA*: (i) Models have to be expressed in a **well-defined notation**. (ii) An **organized design**, based on a multi-layered and multi-perspective architectural framework is achieved through a system specification

with models, transformations, mappings, and relationships of these models. (iii) Models must be defined with **conformance to the meta-model**. (iv) The modeling framework should increase **acceptance and adoption**.

3.3.1 Modeling Levels

Figure 3.4 shows the three different level of abstraction of models [GDD06]:

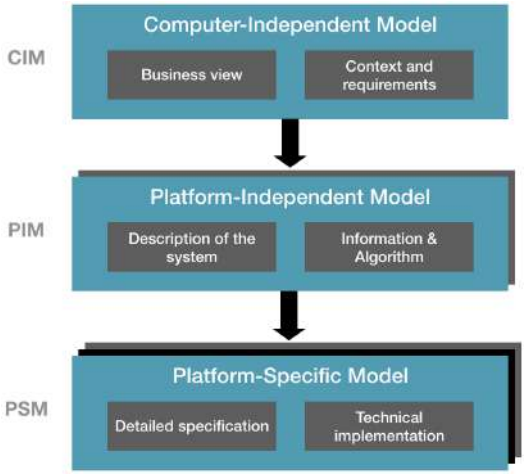


Figure 3.4: Modeling Levels of MDA [BCW12] [GDD06].

Computation-Independent Model (CIM) This level describes the context, requirements and purpose of the solution without any specification of computation implications. It is the most abstract level. This represents only the business or domain view. The CIM can be realized with several *Platform-Independent Models*.

Platform-Independent Model (PIM) This level describes the behavior and structure of the application without technology-related specifications. So PIM can be mapped to multiple *Platform-Specific Models*.

Platform-Specific Model (PSM) This level contains all the required information on the behavior and structure of an application for a specific platform, which can be used to implement executable code.

3.4 Modeling Languages

As Figure 3.5 shows, three parts are needed to develop a metamodel-centric modeling language [SVEH07] [BCW12]:

(i) **Semantics** are the meaning of elements of the language and their combination. Semantics define the meaning of the abstract syntax. (ii) **Abstract syntax** describes

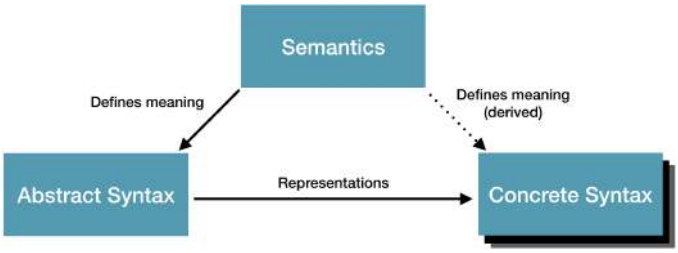


Figure 3.5: Parts of Languages [BCW12].

the primitives and their combination without a particular representation. (iii) **Concrete syntax** describes specific representations of the abstract syntax. This can be textual or graphical. The semantic therefore defines the meaning of concrete syntaxes transitively.

The following section describes the technological stack of the aforementioned concepts. The abstract syntax is realized in *Eclipse Modeling Framework* (cf. 3.5.1) and the graphical concrete syntax in *Eclipse sirius* (cf. 3.5.2).

3.4.1 Modeling Constraints

Modeling and metamodeling is a constructive approach: A limited set of modeling constructs are given through the meta-metamodel in order to define a new modeling language. It is just possible to restrict the possible links between different elements through cardinality. *Object Constraint Language (OCL)* is used to complement metamodels and add restrictions to the metamodels. Textual rules can be defined to which each model conforming to the metamodel yields. OCL has the following features: (i) **Declarative** A contract for each operation must be provided. The contract consists of a set of pre and postconditions [Cab07]. (ii) **Typed** Each expression has a type. Well-formed expressions must obey the type of conformance rules of OCL. Each classifier defined in the metamodel represents a distinct OCL type and OCL includes a set of supplementary predefined types (iii) **Side-effect free** OCL constraints can query or constraint, but not modify the state of the system.

3.5 Model-driven Engineering Technologies

Various tools support model-driven approaches. The Eclipse Modeling Framework is an open-source modeling framework and code generation facility for building tools and other applications based on a structured data model. Sirius is an open-source Eclipse project which allows creating own graphical modeling workbench by leveraging the Eclipse Modeling technologies, including EMF and Graphical Modeling Framework (GMF) [JMJ+16].

3.5.1 Eclipse Modeling Framework

The Eclipse Modeling Framework (EMF) is part of the Eclipse project. Eclipse is a project that comprises tools and projects for developing software, which its most prominent part is the Integrated Development Environment (IDE). The meta-model and extensions are based on Ecore, which is very similar to UML, and are implemented in EMF: You can define meta-classes, packages, different types of connections (composition, inheritance, and reference) can be modeled. A meta-class can contain different typed attributes or operations. It is possible to generate the following four plug-ins for each meta-model [Jun16].

Model Plug-in contains the interfaces and classes, which are designed in the meta-model.

Edit Plug-in contains the UI-independent code of the editor. ItemProvider classes for each meta-class and colored icons are generated.

Editor Plug-in contains the UI-dependent portion of the editor code.

Test Plug-in contains test code for each entity in the meta-model within the JUnit Framework.

Developing models with the generic tree-based editor can be hard for complex meta-models. Therefore, it is possible to define concrete syntaxes both, textual and graphical. Since the aim of the thesis is to develop a graphical concrete syntax, Section 3.5.2 describes the graphical domain-specific language development with the help of *Eclipse Sirius*.

3.5.2 Eclipse Sirius

Sirius is an Eclipse project which allows you to easily create a graphical modeling workbench by leveraging the Eclipse Modeling technologies, including EMF and GMF. Sirius has been created by the companies *Obeo* and *Thales* to provide a generic workbench for model-based architecture engineering that could easily be tailored to fit specific needs.

Based on a viewpoint approach, Sirius makes it possible to equip teams who have to deal with complex architectures on specific domains. The crux of the approach lies in declarative descriptions, which are dynamically interpreted using GMF runtime to materialize the workbench within the Eclipse IDE. So Sirius encapsulates GMF and users can customize the GMF level, too. The specifier of a graphical domain-specific language defines a model in a **.odesign* file, in which different viewpoints can be created.

Inside of this viewpoint, the following items can be created:

Representation Descriptions The following representations can be chosen: diagrams, sequence diagrams, tables and cross-tables, and trees.

Representation Extensions contain the data needed to display the diagrams.

Validation Rules can be defined, which can be applied to all representations defined inside viewpoint.

Java Extensions It is not necessary to write Java-Code, but it can be extended with *Java services*.

In diagrams nodes, edges and containers can be mapped to classes or their relationships. These elements can be visually designed and arranged on different layers [MP15].

To define a graphical concrete syntax, a domain class has to be set for the diagram which resembles the starting point. In order to define further model elements, the developer has to navigate through the meta-model starting from the chosen domain class with the help of several query notations, e.g. Acceleo Query Language [VMP14]. Besides, new tools can be added, e.g. node or edge creation.

To make the actual development clearer, Figure 3.6 shows the .odesign file in the specific editor. This file describes the modeling workbench that is created. It will be interpreted by the Sirius runtime.

In this file, the wizard has created a first viewpoint named Process Segments. The Model File Extension property defines the kind of models associated with the designer. A viewpoint provides a set of representations (diagrams, tables or trees) that the end user will be able to instantiate. These representations again can have different nodes and connections, which can be mapped on metamodel elements.

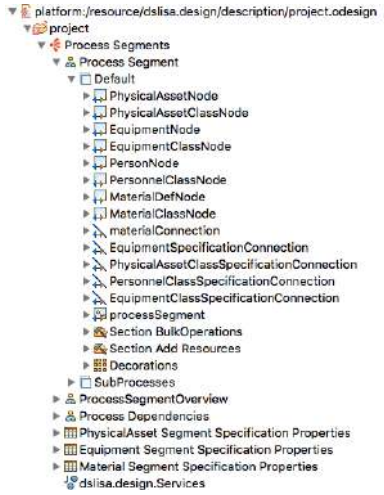


Figure 3.6: The development of a graphical DSL in Eclipse Sirius.

CHAPTER 4

Approach

In order to empower domain experts systemically, a systemic approach to tackle the manifold challenges (cf. 4.1. In order to create an intuitive user interface, Section 4.1.

Most of the viewpoints are diagrams. In order to model complex software abstractions, we make use of interdisciplinary research of cognitive psychology and requirements engineering of software artifacts, which is a suitable analogy for this issue. We therefore present the physics of a visual notation in Section 4.2. Section 4.3 deals with end user development in order to enable meta design. In order to not only reveal problems but increasing the possibility to initiate future development in the context of other projects (e.g. bachelor theses), Section 4.4 presents basic concepts of Product Management.

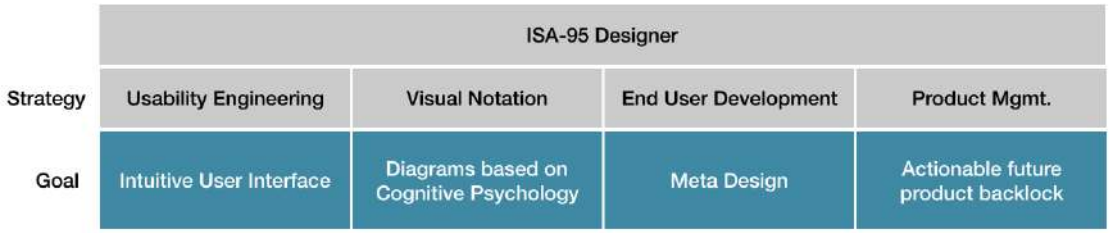


Figure 4.1: The goals with the approach of ISA-95 Designer.

4.1 Usability Engineering

The literature on Usability Engineering is quite confusing. Several related terms exist, which are used equivalently as well as differently, which are: *Usability*, *software ergonomics*, *User Experience*, *human-computer interaction*, *Usability engineering*, *Human factors*, *Interaction Design*, *User-centered design*, *Design thinking*. This thesis makes use of all

these disciplines. In order to make this process transparent, the development strategy is further explained in this chapter.

4.1.1 Usability

Usability is a quality attribute of a system. The *ISO* defines usability in *EN ISO 9241 Part 11: Usability: Definitions and concepts* as follows [Sta98]: The objective of designing and evaluating systems, products, and services for usability is to enable users to achieve goals effectively, efficiently and with satisfaction, taking account of the context of use. (i) **Effectiveness** describes the accuracy and completeness with which a user reaches a specific goal. (ii) **Efficiency** is achieved when the effort relative to the effectiveness is as small as possible to complete a task. (iii) **Satisfaction** is the subjective component. It refers to the freedom from interference by the system and the positive attitude of the user towards the use of the system. Satisfaction is a result of high effectiveness and high efficiency. Effectiveness and efficiency have a relationship of tension: The more precisely a goal is achieved, the more effort must be invested by the user [Lan13]. *Usability Engineering* is the art of finding the right ratio for a system for its users.

4.1.2 Usability principles

ISO 9241 is a multi-part standard of ISO that standardizes various aspects of human-computer interaction. Part 110 defines principles for the evaluation and design of user interfaces, the so-called seven quality criteria of software ergonomics [ISO06a]:

(i) **Suitability for the Task** A dialogue is task-suitable when the user can perform his tasks effectively and efficiently. The availability of the system is a prerequisite. moreover (ii) **Conformity with User Expectations & Suitability for Learning** The offered functions should be useful to complete the tasks. Comfort should be that all information and features should be offered in a way that does not cause handling problems and thus the user is not deterred from his work. Clarity should save short-term memory. (iii) **Self-Descriptiveness** The system is intuitively understandable or will be explained on request. (iv) **Controllability** The user must be able to control the dialog flow, ie Course, direction and speed until the destination is reached. (v) **Conformity with User Expectations** A dialogue is in line with expectation if it is aware of the knowledge of the user in his field of work, education or experience. The dialogue is also consistent and consistent with recognized Conventions. (vi) **Error Tolerance** The user can, despite recognizable error inputs, with no or minimal correction effort to achieve the intended work result.

4.1.3 Usability Engineering

Usability engineering aims to build user-oriented systems. In order to approach this in a structured manner, there is a need for a systematic, empirical and consistent user-oriented development process model.

Richter et.al. presents all relevant techniques, which influence the usability of a system [RF16]. Since using all proposed techniques and steps would result in impossibly high expenditure of time, the process model is tailored in order to have a fast and consistent approach.

Based on the famous *The Usability Engineering Lifecycle* some of the above-mentioned techniques are added, removed or updated [May99]:

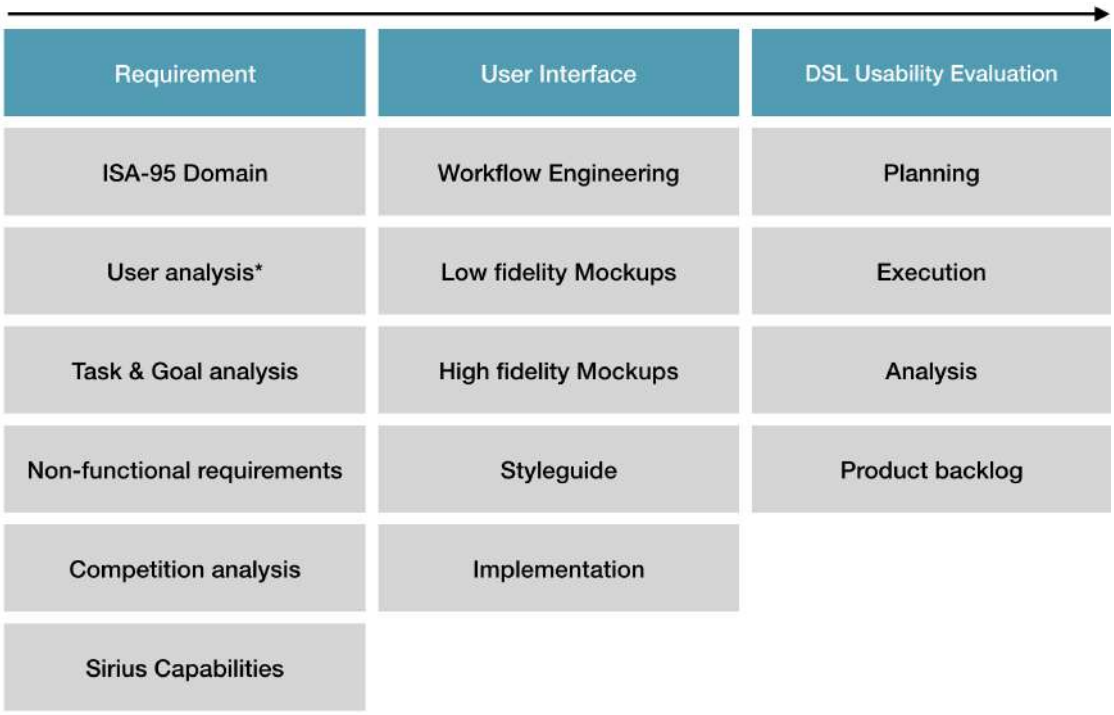


Figure 4.2: The custom-tailored Usability Engineering process model derived from [May99].

Requirements Analysis

With the requirements analysis usability-relevant information is sought:

ISA-95 Domain This addresses the detailed analysis of the ISA-95 and the implementation details within EMF.

User analysis Instead of time-costing user analysis, the thesis make use of *Personas*. Personas are a powerful technique to capture our knowledge about the users and customers of a product. Personas are fictional characters that usually consist of a name and a picture; relevant characteristics, behaviors, and attitudes, and a goal [Pic10].

Task and Goal Analysis detects the sign stimulus, the goal, the frequency and the possible user wishes of tasks.

Non-functional requirements are the definition of the order of non-functional requirements. What sounds at the first moment unnecessary, will turn out to be an effective tool to discuss the advantages and disadvantages of mockups - especially for A/B testing.

Competition analysis brings up important information, such as existing techniques, which can be used to optimize the product. This takes place in the chapter *Related Work* (cf. 6).

Sirius capabilities The approach of this thesis make use of existing technologies. This gives advantages in terms of speed of development. On the downside, this approach restricts the features of the tool.

User Interface Development

With the help of the usability-relevant information of the requirements analysis, user interfaces can be developed.

Workflow Engineering is done by sorting the result of the *task and goal analysis* and manifest it in a consistent flow/state diagram. The result is how the user can navigate through the different views.

Low fidelity Mockups are rough sketches of UIs, which allows focussing on essential aspects, instead of implementation details. We start with paper-based sketches and go on with non-functional digital prototypes. Low fidelity mockups are refined iteratively in a fluent passage.

High fidelity Mockups are detailed drafts, created with Photoshop. They look very similar to the resulting UIs.

Styleguide The same facts should be presented uniformly. A consistent design leads to predictability and gives users a sense of control. Therefore certain principles are manifested in a styleguide.

Implementation After the design phase, the software is, finally, implemented and the plans are executed.

It is important to understand the development of the *User Interface* not as a rigid scaffold, but a constant refinement with the help of fractal-like nano-iterations [SH10]. Interim results are analyzed and as a consequence discarded or refined.

4.2 Visual Notations in Software Engineering

Visual notations play an enormous role in software engineering. In order to design cognitively effective visual notations and use the whole spectrum of design scope, it is necessary to have a systemical overview of the physics of visual notations [Moo09].

The Principles for designing cognitively effective visual notations are [LMHM10]:

Semiotic Clarity There should be a 1:1 correspondence between graphical symbols and semantic constructs.

Perceptual Discriminability Symbols have to be clearly distinguishable from each other.

Semantic Transparency uses symbols whose appearance suggests their meaning.

Complexity Management Explicit mechanisms have to deal with complexity.

Cognitive Integration Explicit mechanisms have to support the integration of information from different diagrams.

Visual Expressiveness Make use of the full spectrum and capacities of visual variables.

Dual Coding Text can complement graphics.

Graphic Economy The number of different graphical symbols should be small and therefore cognitively manageable.

Cognitive Fit use different visual dialects for different tasks and/or audiences.

4.3 End-User Development

The omnipresence of computer systems in academic, professional and every day life leads to growing diversity in its requirements and is under continuous change. People with diverse knowledge, skills, cultural, social, cognitive and physiological backgrounds work in a huge variety of contexts, tasks, and fields of activity. Involving the end user by designing *software for change* leads to cost reduction. In the era of digitization, software applications will be developed, customized, and maintained by non-professional programmers. There are about 90 million estimated end users, who perform basic programming activities in American workplaces (165 million employed in total) [SSM05].

As a consequence, the goal of human-computer interaction shifts from *easy of use* to *easy to develop*. For this purpose, environments have to be created which allow end users, who are familiar with basic computer functionalities and interactions to develop, modify and change systems. End-user development summarizes methods, techniques, and tools, which help end users who are neither skilled nor interested to create, adapt or evolve software or software artifacts [BCFP19]. There exist three main types of end user development from a user's perspective and several subgroups [LPW06]:

Parameterization or Customisation Activities that allow users to decide among alternative behaviors, presentations or interaction mechanisms already available in the application.

Program Creation and Modification Activities that imply some modification, aiming at creating from scratch or modifying an existing software artifact. These can be

subdivided in the following points: (i) **Programming by Example** Users provide example interactions and the system con- cludes general rules. (ii) **Incremental Programming** enables to change small parts of software artifacts. (iii) **Extended Annotation Parameterisation**

Model-based Development The user provides a description of the activity to be supported and the system generates the corresponding interactive application.

Model-based development can be achieved with the help of domain-specific modeling, which is the creation of DSLs with the Model-driven Engineering approach (cf. 3.5.2).

Graphical diagrams are more effective than text in the goal-oriented requirements process [LMHM10]: In diagrammatic reasoning, which shows that the form of representations has an equal, if not greater, influence on human understanding and problem-solving performance as their content [LS87]. Empirical studies confirm that the visual appearance of RE notations significantly affects understanding, especially by novices [LMHM10].

Metadesign From a more socio-cultural perspective this represents an empowerment of modelers. With the help of our tool, it is possible to create a functional ISA-95 model without any further technical knowledge. Fischer et al. introduce *Metadesign*, which is an emerging conceptual framework aimed at defining and creating social and technical infrastructures, in which new forms of collaborative design can take place [FGY+04]. Fischer et al. determine, that these systems promote the transcendence of the individual mind (through collaboration), support sustained participation and enable the mutual adaptation and continuous evolution of users and systems [GF08].

4.4 Product Management

As a result of the evaluation, we want not only reveal problems from the end user perspective but develop a strategy on how to tackle the problems and derive actionable future requirements. We want to make use of existing strategies, frameworks, tools and methods from *Scrum Product Ownership*. As a result, we fill a so-called *Product Backlog* with requirements from the viewpoint of the user.

The *Product Backlog* contains the outstanding work necessary to develop a successful product. Its entries describe any future work to be done by the team to complete the product. These can be functional and non-functional requirements, but may also include examining customer needs or technology options, market introduction tasks, eliminating errors, or refactoring measures [Pic10]. This product backlog has the following features:

Prioritized All entries are prioritized. The highest prioritized entries are implemented first.

Adequately detailed Product backlog is different detailed. Higher prioritized entries are described in more detail than less prioritized ones.

Estimated Estimates are rough and often expressed as story points or ideal days. Detailed estimations are done in the sprint backlog.

Visible The Product Backlog must be visible and easily accessible to all project stakeholders, because it is the central internal communication.

Changeable The Product Backlog is constantly changing and growing. New requirements are created, modified, re-prioritized, refined or removed.

Since the thesis is just a snapshot of the iterative product evolution, not all axioms have to be fulfilled. The product backlog here can not be changed.

CHAPTER 5

ISA-95

First, an introduction into ISA-95 is given. In Section 5.1, part 2 of ISA-95 is presented in detail. After that, the synergy

The information flow of industrial systems grows in regards to its structural complexity and amount [HLML19]. In a fully integrated system, business and manufacturing functions are aligned in the contemplation of facilitating an efficient data flow [Wal18]. Using ISA-95 reduces cost, risk, and errors associated with implementing, the data exchange between *Enterprise Resource Planning* and the *Manufacturing Execution System* has been standardized in the ISA-95 series [Sch07]. This standard is also known as a standard of the American National Standards Institute (ANSI) and the International Society of Automation (ISA): ANSI/ISA-95.

An XML serialization of this standard has been published by Manufacturing Enterprise Solutions Association (MESA), which is called Business To Manufacturing Markup Language (B2MML) [Int].

The standard consists of the five following parts, which each deals with different aspects of enterprise control system:

Part 1 , published in 2013, gives an overview of the *basic models and terminology.*

Part 2 presents the *Objects and Attributes for Enterprise-Control System Integration.* As this is the main part of the tool, Section 5.1 provides a profound explanation.

Part 3 defines activity operations management that enables enterprise system to control system integration. It is focused on the activities between business planning and logistics functions, defined as the Level 4 functions and the process control functions, defined as the Level 2 functions of Part 1.

Part 4 focuses on the integration of manufacturing operations management information.

Part 5 desfines a transaction model for information exchange.

In this work, we concentrate mainly on *ISA-95-2:2013*. Therefore various facets are described in the following sections. First, the data structure of *Resources*, *Process Segments*, and *Operations Management Information* are presented.

Subsequently, the model transformations, which are available: *B2MML*, *UPC-UA*, *REA*, and *AutomationML* are presented.

5.1 ISA-95 Part 2

Figure 5.1 shows the scope of ISA-95 Part 2. There are four primary types of resources, that must be exchanged among MOM and ERP systems: (i) Information about equipment **Equipment** of an organization in terms of a role-based model. (ii) Information about the **Physical Assets** that bundle the equipment, (iii) Information about **Personnel** and its qualifications and roles, which are involved in the manufacturing processes. (iv) Another resource is **Material**, which expresses raw, intermediate and finished materials as well as consumables, which can be defined in unique material lots or sub lots. **Process Segments** are atomic elements of manufacturing activities. A hierarchical level can define different levels of abstraction. They represent an abstract production step, which connects different personnel, equipment, physical asset and material resources. **Operations Definition** defines a specific operation step, which can be in production, maintenance, quality management or inventory. Operation Definitions are therefore concrete steps of abstract *Process Segments*. **Operations Schedules** are requests for a specific operations to be performed within a defined date and time frame. **Operations Performances** represents specific operation, which has actually been performed. **Operations Capabilities** are collections of information about all resources for operations for a defined date and time frame (past and future), including terms, statuses, and quantities. Process Segment Capabilities are logical groupings of resources that are committed, available, or unavailable for a defined process segment for a specific time.

ISA-95 provides a standard manner to uniquely describe this information for exchange, including the interrelationships between the various types of information, which are described in detail in the following subsections.

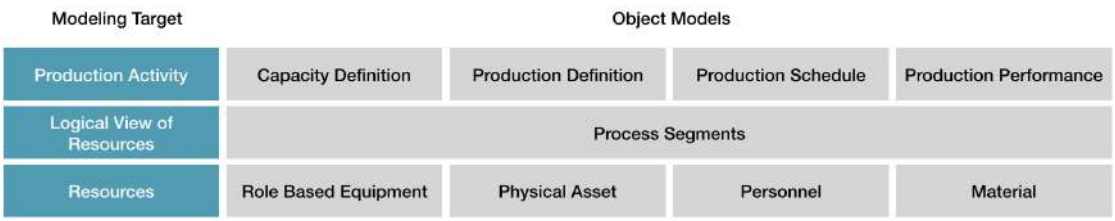


Figure 5.1: ISA-95-2: Overview [OAHB13]

There are different kinds of resources, that must be exchanged among MES and ERP systems: (i) Personnel (ii) Equipment (iii) Physical Asset (iv) Material. In the following subsections, the design of the metamodel is presented.

5.1.1 Personnel Model

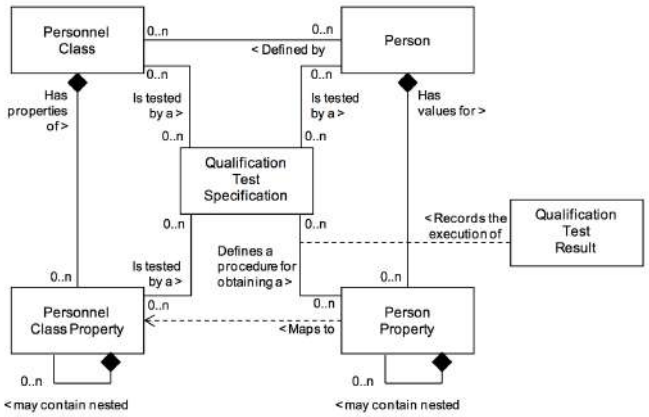


Figure 5.2: Personnel Model as defined in [Int13b]

Figure 5.2 shows the ISA-95 personnel model, which enables to specify the personnel classes, individual persons, their properties and qualifications, which are described in the following classes [Int13b].

Person

A *Person* is a representation of an individual person. Any person can be a member of zero or more *Personnel classes*. Each *Person* has an id, a description, and a name.

Person Property

Each *Person* can have zero or more recognized properties. Each property can have an *id*, a *description*, a *value*, and a *value unit of measure*. A property can contain other properties.

Personnel Class

The *Personnel Class* is a representation of clustering of persons with similar characteristics for a specific purpose. Each *Personal Class* has an id and a description.

Personnel Class Property

A *Person Property* can be defined in an analogous manner to the *personnel class property*. Each property has an *id*, a *description*, a *value*, and *value unit of measure*. A property can contain other properties.

Qualification Test Specification

Qualification Test Specification represents a qualification test. It can be associated with a person, a person property, a personnel class, a personnel class property. It is used to specify and test the required competencies for specific operations. Each *Qualification Test Specification* has an id, a description, and a version.

Qualification Test Result

The results of a test for a specific person can be defined in the *Qualification Test Result*. Each result requires a date, result, and an expiration, and can have an ID, a description and a result unit of measure.

5.1.2 Equipment Model

Figure 5.3 shows the ISA-95-2 equipment model, which enables to specify the equipment classes, equipment, their properties, and capabilities, which are described in the following classes [Int13b].

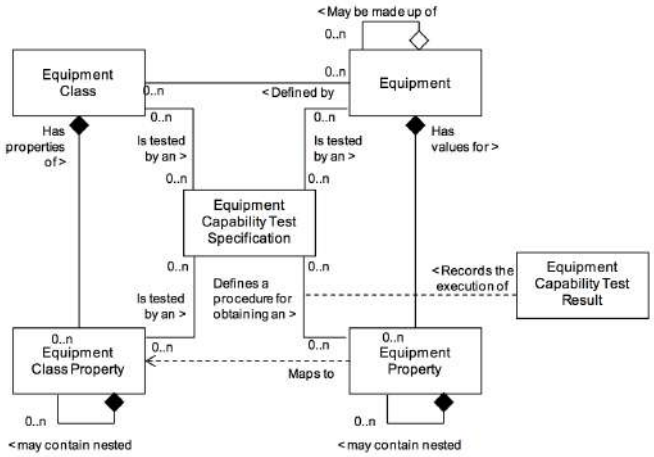


Figure 5.3: Equipment Model as defined in [Int13b]

Equipment

Equipment is a representation of a piece of equipment. Any equipment can be a member of zero or more *Equipment classes*. Each *Equipment* has an id and a description and equipment level, which defines the scale of a hierarchical composition. This enables, that equipment can be composed of other equipment.

Equipment Property

Each *Equipment* can have zero or more recognized properties. Each property can have an *id*, a *description*, a *value*, and *value unit of measure*. A property can contain other

properties.

Equipment class

The *Equipment class* is a representation of clustering of equipment with similar characteristics for a specific purpose. Each *Equipment Class* can have an id, a description, and an equipment level.

Equipment Class Property

An *Equipment Class Property* can be defined in an analogous manner to the *equipment property*. Each property can have an *id*, a *description*, a *value*, and *value unit of measure*, the type of the value. A property can contain other properties.

Equipment Capability Test Specification

Equipment Capability Test Specification represents a capability test. It can be associated with equipment class, equipment class property, equipment or equipment property. It is used, where a test is required to ensure that the equipment has the necessary capability and capacity. Each *Equipment Capability Test Specification* has an id, a description, and a version.

Equipment Capability Test Result

The results of an equipment capability test for a specific piece of equipment can be defined in the *Qualification Test Result*. Each result requires a date, result (passed-failed or quantitative result), and an expiration, and can have an ID, a description and a result unit of measure.

5.1.3 Physical Asset Model

Figure 5.4 shows the ISA-95-2 Physical Asset Model, which enables to specify the equipment classes, equipment, properties, and capabilities, which are described in the following classes [Int13b].

Physical Asset

Physical Asset is a representation of a physical piece of equipment. Any equipment can be a member of zero or more *Physical asset classes*. Each *Physical Asset* has an id and a description, physical location, a fixed asset id, and a vendor ID. A recursive composition relationship enables, that *Physical Asset* can be nested.

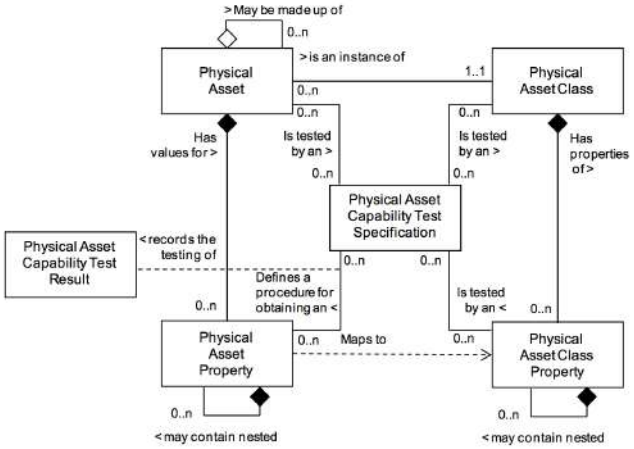


Figure 5.4: Physical Asset Model as described in [Int13b]

Physical Asset Property

Each *Physical Asset* can have zero or more recognized properties. Each property can have an *id*, a *description*, a *value* and *value unit of measure*. A property can contain other properties.

Physical Asset Class

The *Physical Asset class* is a representation of a clustering of equipment with similar characteristics for a specific purpose. Each *Physical Asset Class* can have an id, a description, and an equipment level.

Physical Asset Class Property

A *Physical Asset Class Property* can be defined in an analogous manner to the *Physical Asset Class Property*. Each property can have an *id*, a *description*, a *value* and *value unit of measure*, the type of the value. A property can contain other properties.

Physical Asset Capability Test Specification

Physical Asset Capability Test Specification represents a capability test. It can be associated with physical asset class, equipment class property, equipment or equipment property. It is used, where a test is required to ensure that the physical asset has the necessary capability and capacity. Each *Physical Asset Capability Test Specification* has an id, a description, and a version.

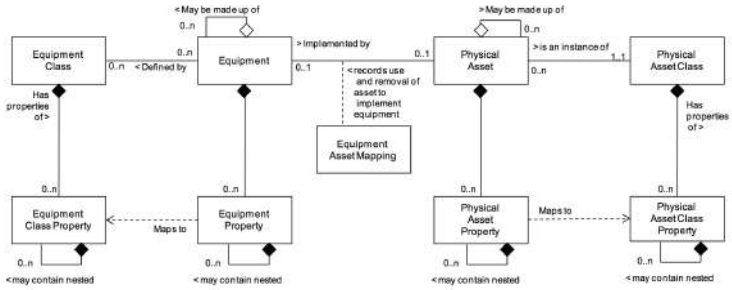


Figure 5.5: Physical Asset and Equipment Mapping as described in [Int13b]

Physical Asset Capability Test Result

The results of a physical asset capability test for a specific physical asset can be defined in the *Physical Asset Capability Test Result*. Each result requires a date, result (passed-failed or quantitative result) and an expiration, and can have an ID, a description and a result unit of measure.

5.1.4 Physical Asset and Equipment Mapping

Figure 5.5 shows the temporal relationship between physical assets and the equipment. The relationship is not rigid and changes over time, which is maintained in the mapping. Therefore, besides the *id*, and the *description*, the time period has to be saved with a *Start Time* and an *End Time*.

5.1.5 Material Model

Figure 5.6 shows the Material Model, which enables to specify the involved materials, material classes and other information.

Material definition

With the *Material definition*, you can specify materials. You can set the *ID*, *description*, *Assembly Type*, which can be physical or logical, and *Assembly Relationship*, which can be permanent or transient.

Material Class

Material Classes are the groupings of materials. The data model takes account of the distinction between raw, intermediate, finished materials and consumables. You can set the *ID*, *description*, *Assembly Type*, which can be physical or logical, and *Assembly Relationship*, which can be permanent or transient.

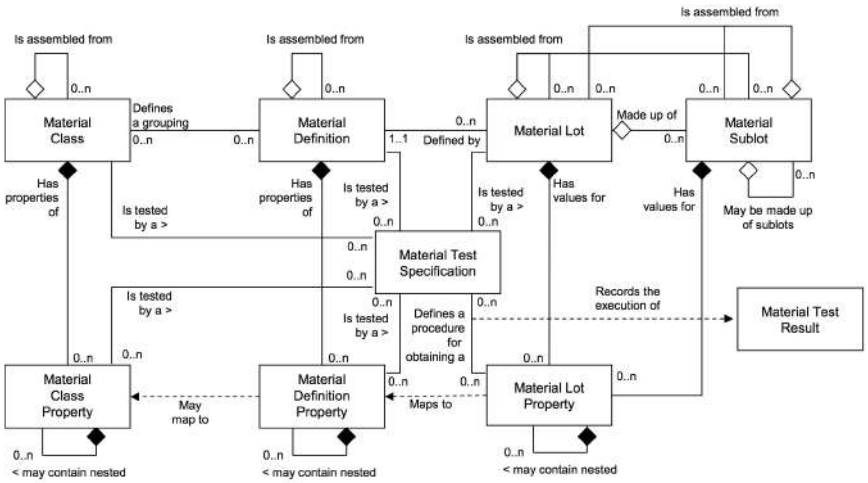


Figure 5.6: Material Model as described in [Int13b]

Material Definition

Material Definitions are representations with similar name characteristics for the purpose of manufacturing operations definition, scheduling, capability, and performance shall be presented as a material definition.

Material Lot

Material Lots are the countable or weighable amount of material are named *Material Lot*. A material lot has a unique identification, an amount of material (count, volume, weight), the unit of measure of the material, the location, a status of the lot. A material lot can be made up of material sub lots.

Material Sublot

MaterialSubLot are countable or weighable amount of material. A material Sublot has a unique identification, an amount of material (count, volume, weight), the unit of measure of the material, the location, a status of the lot. A material lot can be made up of material sublots.

Material Misc

The *Material Class Property*, *Material Definition Property*, *Material Lot Property*, *Material SubLot Property* have the same attributes as the properties of the other resources. The same applies to the *Material Test Specification* and *Material Test Result*.

5.1.6 Process Segment Model

The Process Segment Model is hierarchical and can be defined in different granularity. *Process segments* are the atomic elements of manufacturing that are visible to the business process. As figure 5.7 shows, they connect the personnel resources equipment resources, physical asset resources, and material resources, described in the chapter above, in order to perform a generic manufacturing operations step. The resources are connected through Resource Segment Specifications, in which further properties and quantities are defined. Logical and temporal dependencies can be formulated through the *Process Segment Dependency*. IEC defines them weakly by providing a few examples.

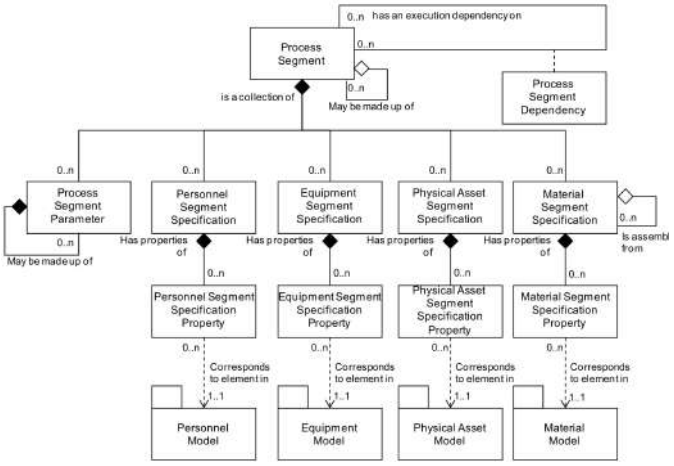


Figure 5.7: ISA-95-2: Process Segment Model [Int13b].

Process Segment have an *id*, description, Operation Types, durations, and value unit of measure. Parameters can be defined with reflexive *Process Segment Parameters*.

5.1.7 Operations Definition Information

OperationDefinitions are resources required to perform specified operation. In most cases, this is a product. It has an *ID*, a *Version*, a *Description*, an *Operations type*, and a *Hierarchy Scope*. For the external Documents there is a *Bill of Material ID*, *Work Definition ID* and *Bill of Resource ID* (c.f. Figure 5.8).

OperationDefinitions can have multiple *Operations Segments*, which have the same strucuture like Process Segment. Resources are connected through Specifications. Unlike Process Segments, they are actual production steps. In the same manner as Process dependencies, Dependencies can be defined.

OperationDefinitions can have multiple hierarchical *Operations Material Bills*. Each *Operations Material Bill* on a bill can have multiple *Operations Material Bill Items*, which can refer to a *Material*.

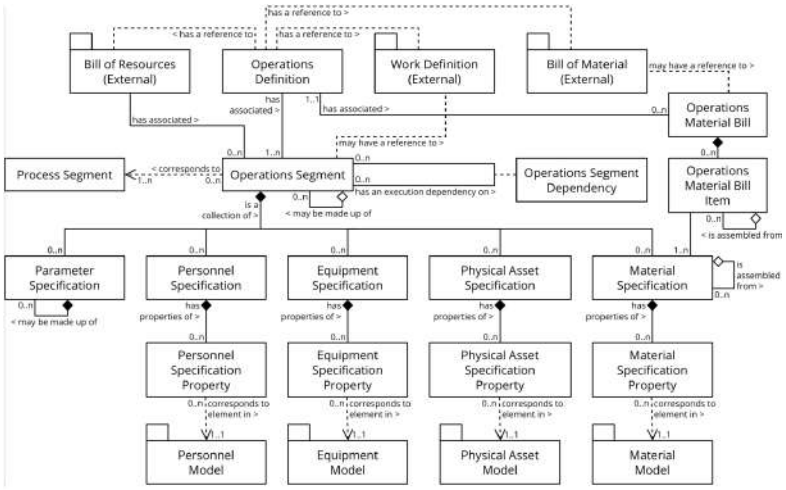


Figure 5.8: Operations Model as described in [Int13b]

Operations Segments have an *id*, description, Operation Types, Durations and Unit of Measure. Parameters can be defined with reflexive *Operations Segment Parameters*. Logical and temporal dependencies can be formulated through the *Operations Segment Dependency*.

5.2 Synergies with other Interconnected Standards

One of the main research issues for Industry 4.0, identified by *The German working committee*, is the horizontal integration through value networks and the vertical integration of network manufacturing systems [MH15] [KWH13]. In order to align and integrate related standards, model transformations (c.f. Model transformations Section 3.2.4) can be used in order to adjust overlapping information from these standards [WHM17b]. The use of model-based approaches is a crucial and competitive factor for successful engineering processes in the automated production systems domain, and hence an emerging practice in the industry. It offers the following possibility: Before the transformation can be executed, metamodels of the standards have to be available or developed. In the next step, the semantic mapping of the metamodels can be achieved.

Figure 5.9 shows the standards of interconnected standards of ISA-95. The x-axis functions as orientation and show the hierarchy levels of ISA-95 (c.f. RAMI 4.0 Section 6). ISA-95 can be used to connect the lower levels of the automation hierarchy with the business models. REA is a popular model for accounting information systems. The alignment will be presented in Subsection 5.2.1. There exists an end-user development tool to build REA models [MH12]. The second transformation presented in this thesis is the mapping from the model to the serialization. This is important in order to understand one of the core ideas of the model-based approach in general and will be presented in Subsection 5.2.2. The transformation of AutomationML to elements of ISA-95 further

fosters the integration of manufacturing layers, which will be presented in Subsection 5.2.3. AutomationML conforming models can be developed with the AutomationML editor. The connection of these standards is, therefore, an important step towards a fully integrated smart manufacturing ecosystem. The following subsection gives an insight into the interconnectedness of ISA-95. Therefore basic ideas of the transformation process are revealed.

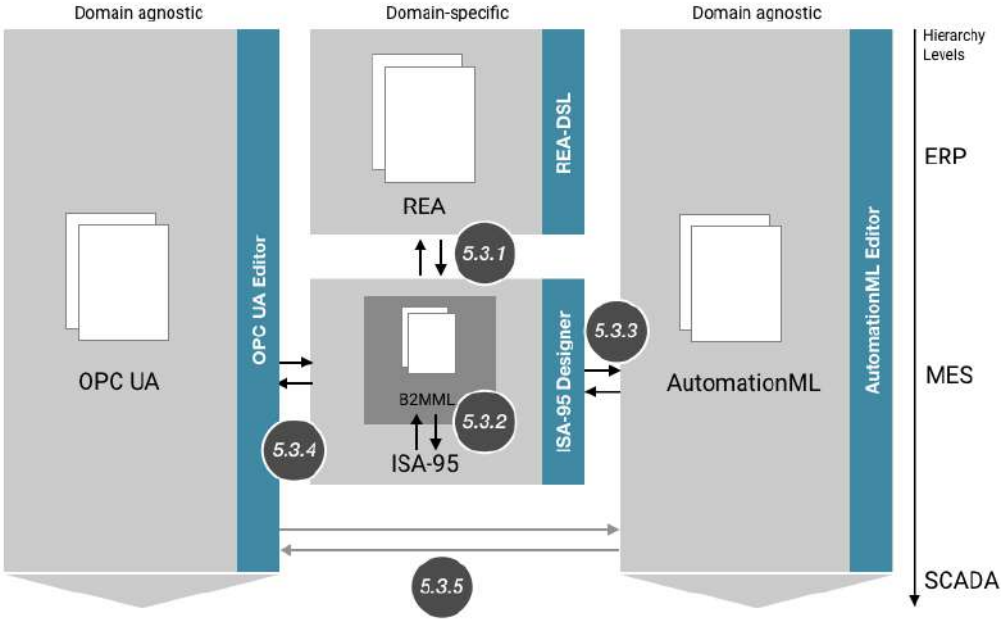


Figure 5.9: Landscape of selected standards in the proximity of ISA-95

End User Development Tools The purpose of this thesis is *Empowering End Users To Develop Automated Production Systems*. Figure 5.9 shows in the turquoise fields the tool support of the interconnected standards. The different tools can be viewed as different viewpoints of the same CPPS in an analogous manner to which the various diagrams are different viewpoints of ISA-95. One of the main challenges in the process of the requirements engineering to develop an automated production system is the communication problem, which results from the different backgrounds of stakeholders, as well as their individual and their different personality trait, roles and objective targets [Par98]. The following exemplary scenario arises: Production Managers can start to model with the help of the *ISA-95 Designer*. This model can be transformed to REA, which can be extended and refined with the REA-DSL by accounting managers, which extends the production with business models. The new insights may have an impact on the ISA-95 model as well, which can be again processed by the Production Manager. These models can be transformed into domain agnostic e.g. OPC UA models, which can be extended by engineers responsible for the communication and security of the APS.

In a more generic way, the models from one standard can be used as a skeleton of the definition of the next standard [WHM17a].

The tools as a whole can be viewed as a toolset, which enables to develop an Automated Production Systems with all its characteristics and facets with domain-specific respectively domain-agnostic, but tier- or implementation-specific languages, which the appropriate experts understand. This process enables continuous refinement and helps to overcome myopic views of single experts through a collaborative development process.

The granularity of concepts can give information about the development process. Since some concepts of ISA-95 are more granular than REA's concepts (cf. figure 5.13: 31 are mapped on 23 REA entities), it makes sense to start with the ISA-95 modeling. The result is an ISA-95 Model. This model can be converted to either *REA*, *AutomationML* or *OPC UA* and be edited and refined in the appropriate tool (REA DSL, AML Editor or OPC Modeler). After that, it can be retransformed into an ISA-95 model again as described in fig. 5.10.

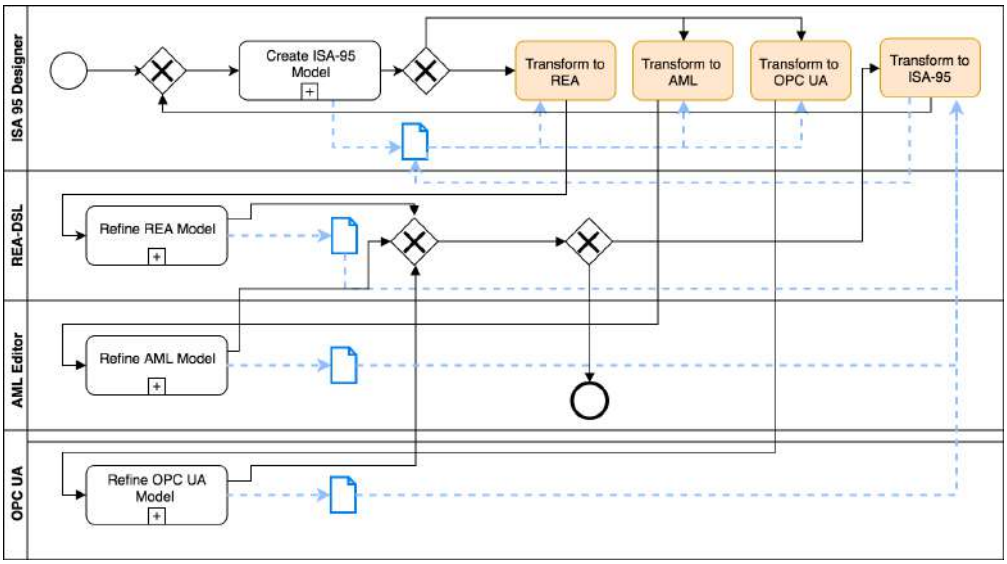


Figure 5.10: Development process model expressed in BPMN 2.0.

However, this idea goes along with a few challenges. Different standards have (i) different views on a specific entity or (ii) different granularity of a single entity [WHMW18a].

5.2.1 Resource-Event-Agent Business Ontology

The *Resource-Event-Agent business ontology* (REA) was developed in 1982 by William McCarthy. REA, which is used in *ISO 15944-4*, proposes concepts and patterns to derive semantic models of economic exchanges and transformations [McC82]. The underlying assumption is, that business enterprises acquire financial resources, engage in a chain

of economic exchanges with other parties. In each exchange, they give up an economic resource in return for another resource of greater value.

In order to model economic exchange, REA proposes three main objects, which are shown in Figure 5.11 on the left[MH12]: (i) **Resources** are exchanged between the agents, e.g. commodities and assets. (ii) **Event** triggers the executing economic activities. (iii) **Agent** gains access to a resource of another agent in exchange to giving up control of another resource.

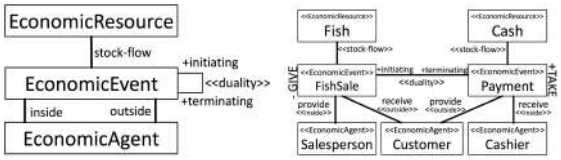


Figure 5.11: Simplified Metamodel of REA on the left and REA Object Constellation on the right [MH12].

The following axioms are premised [MH12]: (i) For each economic resource at least one take-event and one give-event exist, which guarantees an exchange. (ii) An event, which triggers a resource decrement must be eventually paired in duality relationships with events affecting an increment and vice versa. (iii) Each exchange needs an inside and outside agent, which ensures an exchange between two competing parties.

The right side of Figure 5.11 shows a very simple example of an exchange of a fish sale. A customer receives via FishSale a fish from a salesperson. In exchange for the fish, he provides cash to a cashier via payment. The fishSale is a decrement of fish, and the payment is an increment of cash [MH12].

Figure 5.12 shows the highly granular metamodel of the REA Duality, which was extended with several concepts: e.g. commitments, schedules, policies. Commitment, for example, enables the planning layer equivalent of events: It contains information about what has been agreed on in the future, which are fulfilled by events [WHM17a]. Besides, the type-object pattern, also known as powertypes, is used. This offers, that there is for each entity a type class and class available e.g. `Resource Type` Classes are as well as `Resource` Class. The type-object submodel is depicted with gray shading. The upper part contains the type layer (realized with powertype concept of UML) - the lower part of the object layer.

Mapping Now, we want to give a short version of the main idea of the mapping of the metamodels of `REA` and *ISA-95* [WHM17a]. The mapping of the entities includes the integration of their attributes. (i) `Resource Type` is mapped on *Physical Asset Class*, *Equipment Class*, *Material Class* and *Material Definition*. (ii) `Resource` is mapped on *Physical Asset*, *Equipment*, *Material Lot*, and *Material Sublot*. `Linkage` realizes the relationships of *Material Class*, *Material Definition*, *Material Lot* and *Material Sublot*. (iii) `Agent` and `Agent Type` can be mapped on *Personnel Class* and *Person*. (iv) The

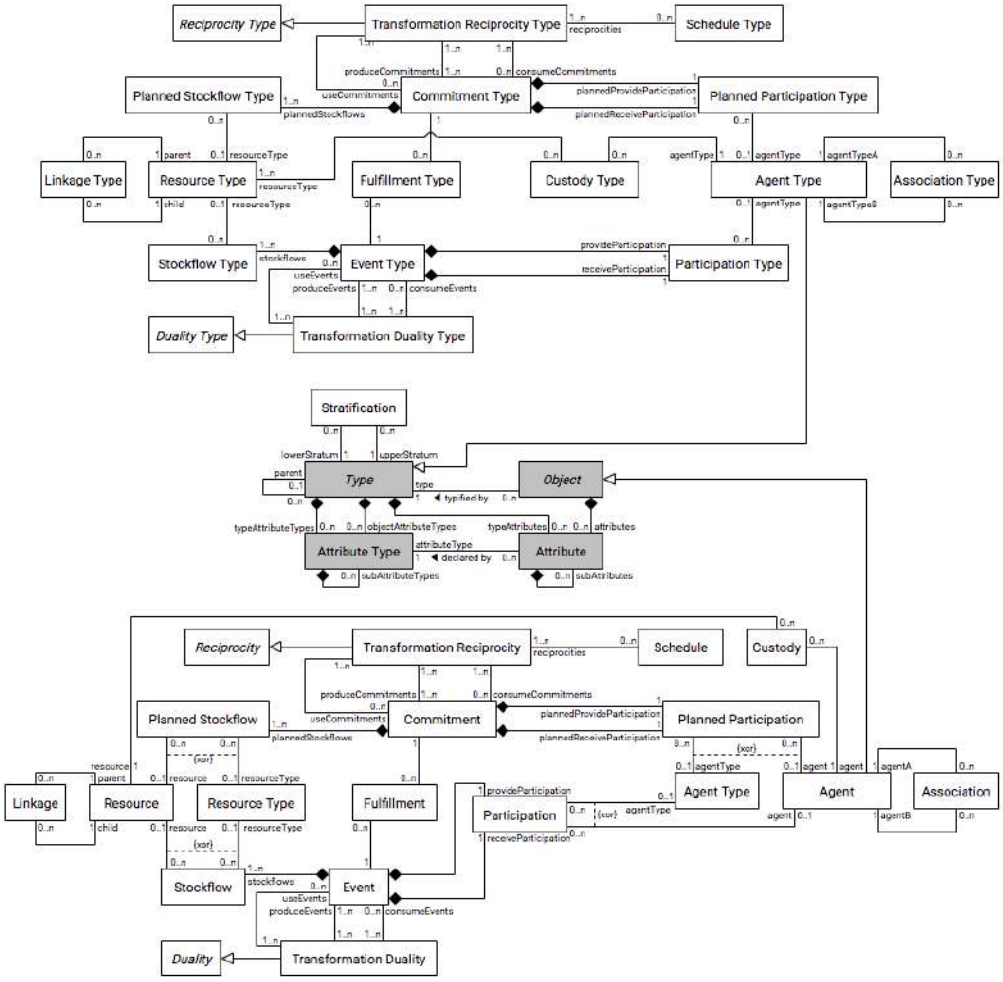


Figure 5.12: Detailed class diagram of REA. The upper part contains the type layer (realized with powertype concept of UML) - the lower part the object layer. The type-object submodel is depicted with gray shading [WHM17a].

Process Segment corresponds to the Type Layer, because it is more generic and does not address a specific production run. A *Process Segment* is therefore equivalent to `Transformation Reciprocity` and `Duality Type`. The attached `Commitment` or `Event Type` has the following `Participation Type`, which can be mapped on *Personnel Segment Specification*, and `Planned Stockflow Type`, which can be mapped either on *Equipment Segment Specification*, *Physical Asset Segment Specification* or *Material Segment Specification*. (v) *Operations Definition* is a `Schedule Type` - the *Schedule* an `Operations Request`. (vi) `Transformation Reciprocity` can be mapped on *Segment Requirement* - `Transformation Duality` on *Segment Response*. The attached entities can be transformed with the same concepts of (iv).

5.2.2 B2MML

B2MML, is an open-source XML implementation of ISA-95, published by the *MESA International*[1]. The current version is v0600[2] [Int]. B2MML is used as the de-facto standard interface to exchange the contents defined in ISA-95 [Tho04a]. B2MML consists of a set of XML schemas written in a XML Schema Definition and therefore defines the data models in the ISA-95 standard [Int] syntactically.

Every facet of ISA-95 is defined in separate files. Listing 5.1 shows an example of a specific material using B2MML. The material is specified as a "100 mm steel pipe" (ID: "FBFG3836744"), belonging to the class "raw material" with specified minimum and maximum levels of inventory (0kg and 1000kg, respectively) [KNH16].

Listing 5.1: B2MML example of material based on [KNH16]

```
<?xml version="1.0" encoding="utf-8"?>
<MaterialInformation
xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://www.wbf.org/xml/B2MML-V05">
<ID>ExampleMaterial</ID>
<Description>Small test example</Description>
<PublishedDate>2014-06-15T00:00:00</PublishedDate>
<MaterialDefinition>
<ID>FBFG3836744</ID>
<Description>100mm steel pipe</Description>
<MaterialDefinitionProperty>
<ID>Min</ID>
<Value>
<ValueString>0</ValueString>
<DataType />
<UnitOfMeasure>kg</UnitOfMeasure>
<Key />
</Value>
</MaterialDefinitionProperty>
<MaterialDefinitionProperty>
<ID>Max</ID>
<Value>
<ValueString>1000</ValueString>
<DataType />
<UnitOfMeasure>kg</UnitOfMeasure>
<Key />
</Value>
</MaterialDefinitionProperty>
<MaterialClassID>raw material</MaterialClassID>
</MaterialDefinition>
</MaterialInformation>
```

In contrast to the XML, the XMI generated by the model-driven approach has several advantages and disadvantages by design. XMI is a specific application of XML. Therefore, XML can be used for all XML applications, including XMI. XMI-based implementations are richer in terms of their semantic because it uses the developed information from the metamodel, defined with UML. The XML approach connections can only be realized through ids, which cannot be semantically validated.

B2MML files can be generated from the model with model-to-text transformations (c.f. 3.2.4).

[1] https://mesa.org/
[2] http://www.mesa.org/xml/B2MML-V0600

5.2.3 AutomationML

AutomationML is a neutral XML-based standard targeting the modeling of production environments. It was developed by an industrial-academic consortium including the companies Daimler, ABB, Siemens, Rockwell, Kuka, Zühlke, netAllied and the universities of Magdeburg and Karlsruhe and was published by the International Electronic Commission, released as *IEC 62714* [Int18]. AutomationML enables data exchange between manufacturing engineering tools and therefore supports the interoperability between them [DLPH08]. Various extensions were developed e.g. with the help of annotations of AML models geometry and kinematics of machines. In addition, their internal behavior can be enriched.

AutomationML is based on the *Computer-Aided Engineering Data-Exchange-Metamodel* (CAEX), an XML dialect standardized in *IEC 62424* [Int16]. CAEX is a neutral data format for model hierarchical object information. With CAEX can e.g. the hierarchical plant structure be mapped, which consists of coupled and parameterized modules and components at a certain level of abstraction. CAEX allows to save these modules and components using the Object Approach. It explicitly supports concepts such as encapsulation, classes, class libraries, instances, instance hierarchies, inheritance, relations, attributes, and interfaces. CAEX is based on XML and is defined as an XML schema. AutomationML adds a set of instance elements as well as rules and restrictions to CAEX, which results in computer-readable semantics and simplify the modeling of automated production systems [WHMW18b].

There are several entities, which are redundantly defined in both standards of *ISA-95-2* and *AutomationML*.

In order to transform actual models, the semantics of the appropriate meta models have to be aligned. First, the entities are semantically coupled to the CAEX-specific entities *Attributes*, *Role Class*, *System Unit*, *Role Class* and *Internal Elements*, which is visualized in Figure 5.14 on the left side: (i) All properties of all resources can be mapped to `Attribute`. Both metamodels support subattributes/subproperties. (ii) `System Units` are vendor-specific reusable entities, which can be represented through Physical Asset Classes. Material Definitions can also be seen as vendor-specific of raw material (e.g. coating powder). (iii) `Role Classes` are classifications of `Internal Elements` and `System Unit Classes`, which can be mapped to either Personnel Class, Equipment Class or Material Class. (iv) All other entities can be mapped to `Internal Elements`. (v) References between ISA-95-2 Elements, which can not be mapped to super or subclassifications can be realized with `External Interface and Internal Link`.

Second, Figure 5.14 shows on the right the mapping of ISA-95 entities categorized in the lower granular concepts of AutomationML Libraries: *Product*, *Process (Structure)* *Resources* (PPR) and AutomationMLBaseRoleClass: (i) Equipment, Equipment Class, Physical Asset and Physical Asset Class can be mapped to Resource. (ii) Material Definition, Material Class, Material Lot, Material Sublot are **Products**. (iii) Operations Segment, Process Segment are `Processes` - Operations Definition can be mapped to

`Process Structure`. (iv) All personnel-related and all specification entities can be mapped to `AutomationMLBaseRoleClass`.

5.2.4 OPC Unified Architecture

The *OPC Foundation*[3] has released a series of standards widely accepted in industry, which enables interoperability in industrial automation: *OPC HDA* allows accessing historical data, *OPC DA* allows accessing current data, and *OPC A&E* accessing alarms and events. The reasons for the development of *OPC UA* are manifold [LM06]: (i) **Unified data access:** *OPC UA* connects the aforementioned standards: Current data, historical data, and events are related to each other. OPC UA unifies all these data in its address space. (ii) **Additional requirements:** Multiple hierarchies, providing commands and accessing historical data are now possible. Whereas the old specification only provided a single hierarchy with items containing data, OPC UA provides an extensible meta model where those items are typed and multiple hierarchies can be developed. (iii) **Technology migration:** The old standards were based on Microsoft's COM/DCOM technology, which is renewed to the advantage of cross-platform capable Web Services and SOA. OPC UA specifies an abstract set of services and maps them to different technologies, which promotes platform independence. (iv) **Additional Areas of Applications:** With the possibility of mapping the service to different technologies, it is now possible to access devices in a performant way as well as providing web services for MES and ERP systems.

The *OPC Foundation* released an extension of the overall OPC Unified Architecture standards and defines an information model that conforms to ISA- 95 [OAHB13]. Figure 5.15 shows the mapping of the meta-models of OPC UA and ISA-95. Resource and Resource Classes of ISA-95 are mapped as OPC `Object` with an appropriate `ObjectType`, properties as `Variables` with an appropriate `VariableTypes`.

Process Segments, Operations Definitions as well as other production activities are not covered by the mapping.

[3] https://opcfoundation.org/

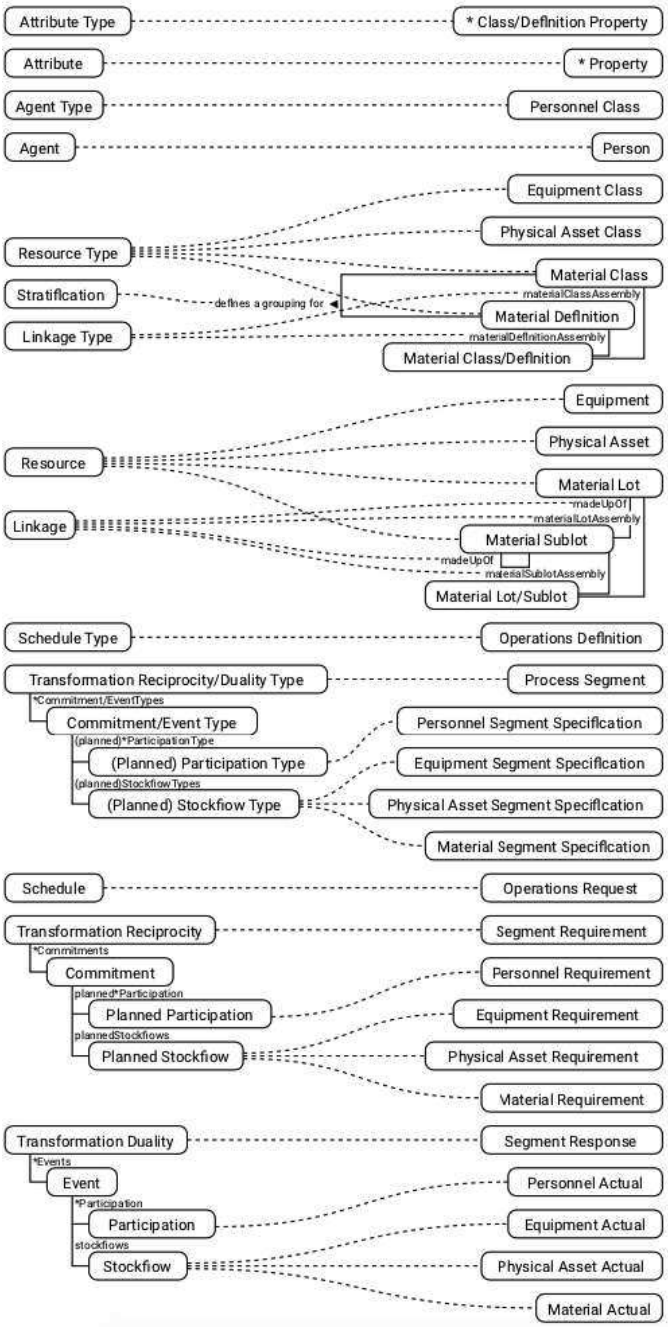


Figure 5.13: Visualisation of transformation of ISA-95 and REA, based on [WHM17a].

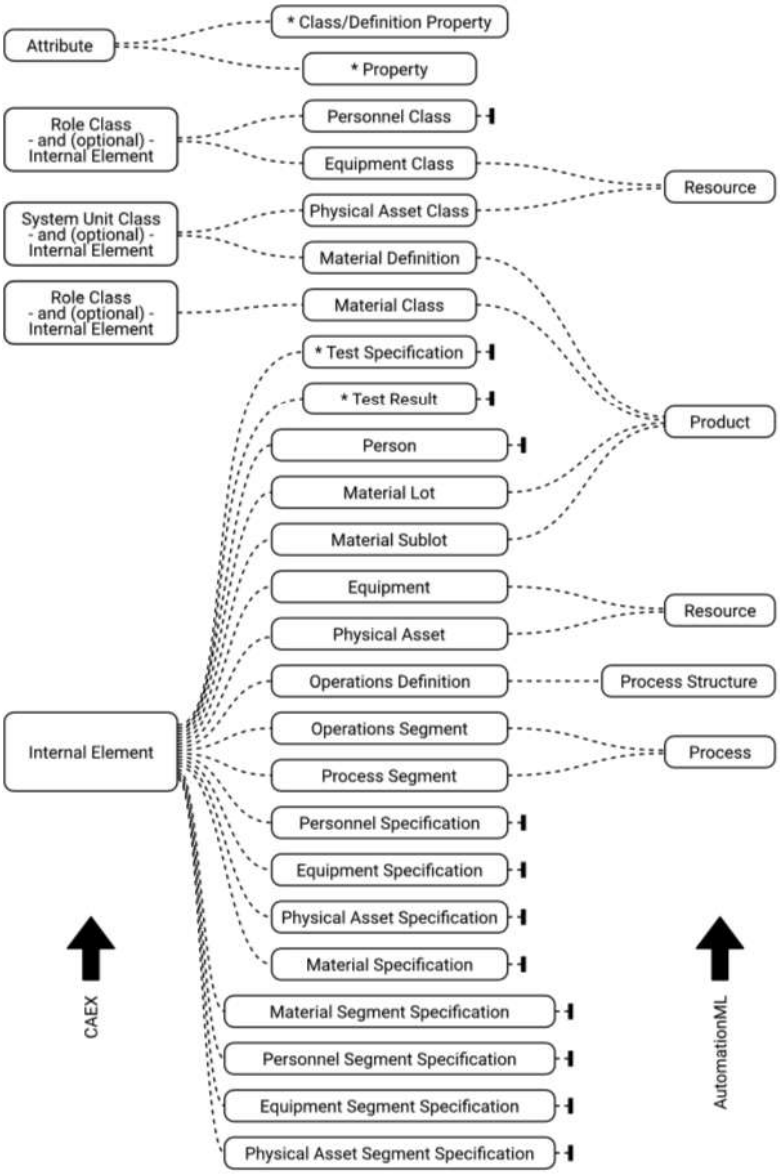


Figure 5.14: Visualization of the mapping from ISA-95-2 to the CAEX metamodel (left) and to existing AutomationML elements of the AutomationMLBaseRoleClassLib (right) [WHMW18b] [Wal18].

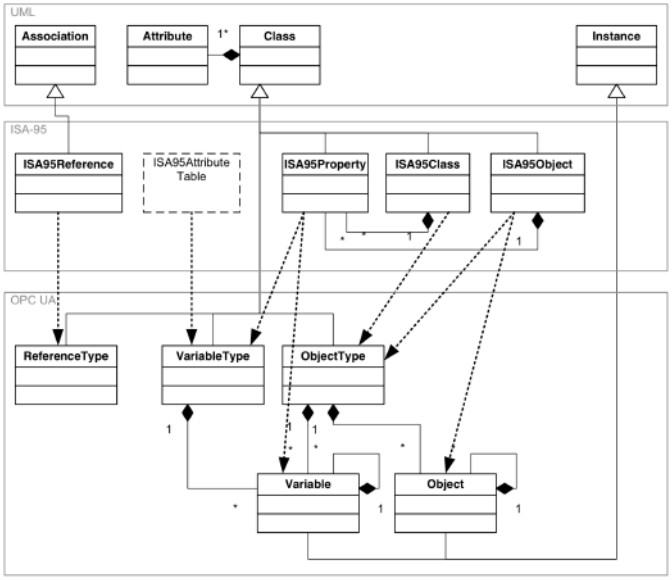


Figure 5.15: Mapping of the meta models of OPC UA and ISA-95 [OAHB13].

CHAPTER 6

Related Work

In order to approach this subject in a structured manner, we identify the following relevant sections of this literature and tool review. Firstly, we discuss previous studies for tools on ISA-95. Secondly, we present graphical domain-specific languages for other manufacturing standards. Thirdly, different concepts from different graphical notations are listed and it is analyzed, how it can be applied to the DSL.

6.1 ISA-95 Tool for Enterprise Modeling

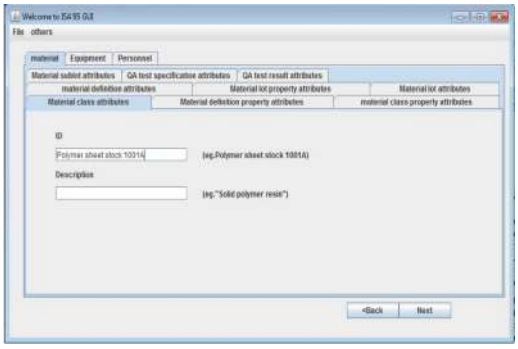


Figure 6.1: ISA-95 Tool

Dazhuang et al. present a tool that makes it easier to automatically integrate product information in a production system [HLML19]. The output is a .xml file following B2MML's .xsd templates. The tool seems to hardcode a lot of aspects. The hardcoded classes from the model inherit classes from ISA-95. This is a valid approach, but it does not fully exhaust all possibilities which the vision of a model-driven smart factory offers [CAGT15].

Besides, as Figure 6.1 shows, it only provides user interface components a generic editor would offer and lacks elaborated visualizations, abstractions and interaction techniques of a graphical DSL.

6.2 Similar Graphical DSLs for Smart Manufacturing

This section is divided into five parts: The first presents a graphical DSL for *REA*. The second gives an overview of variability, the third an overview on BPMN, the fourth an overview on DECLARE. In the last Section, the *CAEX Workbench* is presented.

6.2.1 REA-DSL

From the broader perspective of model-integrated smart production, the popular *Resources, events, agents (accounting model)* plays an enormous role in this work, because it is used to model *ERP* business models, which is one hierarchy level above *MES* in the *automation pyramid*. Viewing REA from a production environment, the bottom layer can be understood as the production process underlying REA transformation primitives. Consequently, there is redundant information in both systems [WHM17c]. It is proposed a domain-specific modeling language, the *REA-DSL*, to support the conceptual modeling of economic events based on the REA ontology [MH12].

6.2.2 Variability

The focus of *Feature Oriented Domain Analysis* is the identification of reusable, parameterized features of a software systems in a domain, which includes Context Analysis, Domain Modelling, Architecture Modelling, and its limitations. The cardinality-based feature modeling notation is presented which is an extension of several other extensions of the feature model. Thereby the lower and upper bound of the number of subfeatures can be set. This enables not just the variability, but also restriction with concrete constraints within this visual notation, which is a good basis to model a product line in this context [Cza05].

The famous swiss astrophysicist Fritz Zwicky invented the *Zwicky Boxes*, which is a creative heuristic method to tackle problems. The method helps to find solutions without prejudices and to structure and present ideas efficiently. Attributes of an entity, which are independent of each other, are defined and written one underneath the other. After that, the manifestations of the attribute are placed next to the attributes, by which a matrix is created, in which every combination of manifestations of all attributes is a possible solution. This notation can, therefore, be used to model instances of variability models [Rit14].

6.2.3 BPMN

Business Process Modeling Notation (BPMN) is defined as readily understandable by all businesses from the OMG. Compared to EPC, the constructs are well defined and specified [OMG11].

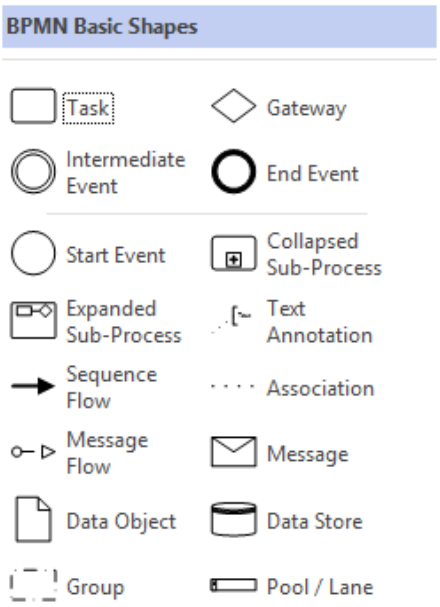


Figure 6.2: Business Process Model and Notation [OMG11]

Figure 6.2 presents an introduction to the main BPM notation elements. The processes are mainly based on activities. Subprocesses can be attached to these activities. Connectors are paths that represent the sequence flow of a process model. They connect individual elements, e.g. activities with each other. Gateways are used to split the sequence flows between the individual activities or to merge them again. These include e.g. data-based exclusive, event-based, parallel, inclusive and complex gateways. The beginning is visualized with a start event (e.g., message, time, escalation, condition error, compensation start event). Intermediate events are distinguished into incoming (e.g., message intermediate) and intervening intermediate events (e.g., intermediate time events). End events (e.g., message escalation, error, abort, and end-of-completion events) terminate a process. It is also possible to integrate notation elements that provide additional information. These include e.g. artifacts or data objects. These elements are attached to activities via associations. By using pools and lanes, the process can be further structured and subordinated by assigning roles to the activities.

6.2.4 DECLARE

Process Segment Dependencies describe temporal and logical dependencies and restrictions. Beside the imperative business processes description e.g. BPMN, another paradigm exists:

declarative process models. These offer more flexibility in execution. Various process modeling languages were developed. In ISA-95 these constraints are poorly described. The integration of formalized dependencies offers the integration of previous knowledge in these research fields on the one hand. On the other hand, it offers experienced modelers the possibility to make use of visual concepts based on cognitive psychology as well as already learned concepts. One constraint-based process modeling language for the development of declarative models describing loosely-structured processes is *DECLARE* [PSv07].

The tables below show basic notation elements of *DECLARE* to represent declarative process models. The explanations always refer to one process instance.

Atomic Constraints

Atomic constraints only affect individual segments (cf. table 6.1).

ID	Description
INIT	A must be executed first
LAST	A must be executed last
EXACTLY	A may be executed exactly n times
EXISTENCE	A must be executed at least n times
AT_MOST	A may be executed a maximum of n times

Table 6.1: Atomic constraints of DECLARE [PSv07].

Relationship Constraints

Relationship constraints affect two or more segments and describe how they are related (cf.6.2).

However, DECLARE lacks a detailed description of temporal dependencies. Especially for business processes, it is very important to depict time events such as deadlines or optimized response times. Therefore another dialect of *DECLARE* arose, which is called *timed DECLARE*. This guarantees the correct execution of a process in terms of latency and deadlines. Timed Declaring is an extension of the process modeling language DECLARE and allows monitoring of metric temporal constraints during runtime [Mag14].

ID	Description
RESPONSE	if A was executed, then B must also be executed afterward (but not necessarily directly after A)
PRECEDENCE	B may only be executed, if A was previously executed
RESPONSE	if A was executed, then B must also be executed afterward (but not necessarily directly after A)
SUCCESSION	if A was executed, then B must also be executed after that (but not necessarily directly after A) and B may only be executed if A was previously executed
CHAINED_RESPONSE	if A was executed, B must be executed immediately afterward
CHAINED_PRECEDENCE	b may only be executed if a has been executed immediately before
CHAINED_SUCCESION	if A was executed, B must be executed immediately afterward and B must be executed only if A was executed immediately before
RESPONDED_EXISTENCE	if A was executed, B must be executed either before or after
CO_EXISTENCE	if A was executed, B must be executed either before or after; if B was executed, then A must be performed either before or after
NOT_CO_EXISTENCE	A and B are not together in a process instance
NEGATIONRESPONSE	after A has been executed, B must not be executed afterward
RESPONDED_EXISTENCE	if A was executed, B must be executed either before or after

Table 6.2: Relationship constraints of DECLARE [PSv07].

6.2.5 CAEX Workbench

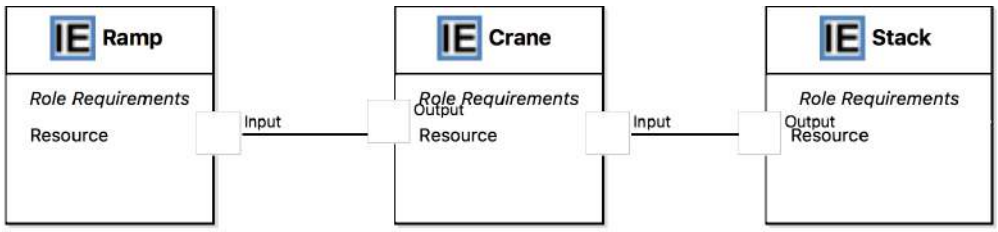


Figure 6.3: CAEX Workbench Graphical DSL [MWBD18]

Computer Aided Engineering Exchange (IEC 62424:2016), in short *CAEX*, is one of the most promising standards when it comes to data exchange between engineering tools in the production system automation domain and is used by AutomationML. The *CAEX Workbench* addresses the challenge of specification of the language including the formalization and validation of additional usage rule, as well as the evolution of the standard. It is of particular importance for this thesis because it uses the same approach and technologies: A Metamodel of CAEX was formalized using EMF's metamodeling

language Ecore based on the standardized CAEX XML schema [MWBD18]. It is argued, that a metamodel is more expressive than an XML schema when it comes to the definition of typed graphs. XML restricts a hierarchical data model, so it is not possible to define references across the hierarchy based on typing information. Besides, metamodels allow different serializations of models with XML being only one alternative. Furthermore, having a metamodel opens up the possibility for model visualization and editing frameworks, model validation frameworks, and model transformation framework. These possibilities are realized for the CAEX workbench through the following MDE tools of EMF: Besides the tree-based editing, textual editing, model querying, model validation Model comparison is enabled as well as model transformation (e.g. CAEX 2.15 models to CAEX 3.0, for which there is also a graphical DSL developed).

CHAPTER 7

Requirements Analysis

This chapter deals with the requirements of ISA-95 Designer. Section 7.1 gives an analysis of the future users. Section 7.2 presents the non-functional requirements. Section 7.3 shows the constraints of Eclipse Sirius. After that, the task and the appropriate goals are shown in Section 7.4.

7.1 User Analysis

In this chapter it is summarized, what it is known about the future users and what can be derived from these facts. While a detailed analysis of the users will be given in the next iteration (cf. Subsubsection 10.3.1), an educated guess of the future users are industrial engineers and IT business engineers. They are employees of companies that develop or maintain specific aspects of production systems in different lifecycle phases of the engineering domains or members of academic institutions or universities. It is assumed, that they have profound technical knowledge. In terms of graphical domain-specific languages, it is assumed that they know UML, AutomationML or SysML. For modeling processes, it is assumed that they have knowledge of BPMN. These assumptions were proven right (cf. Subsubsection 10.3.1).

7.2 Non-functional Requirements

In addition, the following design principles, defined in accordance with *ISO standard 9241-10:1996*, are used as the basis for the evaluation of Usability of the user interface used: (i) **Suitability for the task** A dialogue is task-appropriate if it helps the user to do his work effectively and efficiently. (ii) **Self-descriptiveness** A dialogue is self-describing if every single dialogue step through Feedback from the dialog system is immediately understandable or explained to the user on request. (iii) **Controllability** A dialogue is controllable when the user is able to start the dialogue flow as well as to

influence its direction and speed until the target is reached. (iv) **Conformity with user expectations** A dialogue is in line with expectation if it is consistent and the characteristics of the user, e.g. his knowledge of the field, his education, and experience, as well as the generally accepted conventions. (v) **Error tolerance** A dialogue is fault-tolerant if the intended result of work can be achieved despite the fact that the input was incorrect, either with no or with minimal correction by the user. (vi) **Suitability for individualization** A dialogue can be individualized if the dialogue system adapts to the requirements of the work task as well as the individual abilities and preferences of the user. (vii) **Suitability for learning** A dialogue is suitable for learning if it supports and guides the user in learning the dialogue system.

7.3 Sirius Capabilities

The graphical DSL syntax is developed with the help of *Eclipse Sirius*. Sirius provides some features, but this goes along also with the following restrictions (cf. table 7.1:

Issue	Description
Tables	Tables come up with a lot of restrictions and is therefore not that flexible
Focus	It is not possible to define the mouse/keyboard focus
Selected Tools	It is not possible to define tools for multiple selected nodes or containers.

Table 7.1: Constraints of Eclipse sirius.

7.4 Task and Goal Analysis

This table shows the Action and Goals - ordered mainly by the Model. It is defined, in which view they take place.

Model	Actions and Goals	View
Physical Assets	Overview of Physical Asset Classes and Physical Assets, CRUD of Physical Assets and Physical Asset Classes	PhysicalAssetTable
	Connections of Physical Assets, CRUD Physical Assets	PhysicalDetail
	Connections of Physical Assets, CRUD Physical Assets	PhysicalDetail
Equipments	Overview of Equipment Classes and Equipments, CRUD of Equipments and Equipment Classes	EquipmentTable
	Connections of Equipments, CRUD Equipments	EquipmentOverview
Personnel	Overview of Personnel Classes and Persons, CRUD of Personnel Classes and Persons	PersonnelTable
Material	Overview of Material Classes, MaterialDefinitions, MaterialLots	MaterialTable
	Variability of Materials, Connections of MaterialClasses, MaterialDefinitions, MaterialLots	MaterialDetail
Process Segments	All Process Segments, which don't have a parent.	ProcessSegmentOverview
	All Process Segments without a parent and their material flow	ProcessSegmentDetail
	Temporal and logic constraints and dependencies - atomic and relationship dependencies	ProcessSegmentDependencies
OperationSegments	All Operation Segments without a parent and their material flow	OperationsSegmentOverview
	Details of the resource specifications of the OperationSegments, children of the OperationSegments and their material flow	OperationsSegmentDetail
	Temporal and logic constraints and dependencies - atomic and relationship dependencies	OperationsSegmentDependencies
*properties	CRUD of all resources, resource specification and resource segment specification properties	*properties
*parameters	CRUD of process segment and operationsegment parameters	*parameters

Table 7.2: Action and Goals of the different views.

CHAPTER 8

User Interface Development

"I begin with an idea..
..and then it becomes something else."
— Pablo Picasso

One of the central tasks of a user-oriented approach is to develop a user interface concept suitable for the tasks of the user and the desired user experience. Therefore, it is key to define the principles of the user interface: How do users navigate through menus and dialogues? How is the information structured and presented? Does the system need to be optimized for special technologies or within a special framework? The starting point is the developed personas and scenarios and the agreed requirements. Simple prototypes help the user interface designer to play through the scenarios and work out a suitable concept.

8.1 Workflow Engineering

IEEE 1471 propagates notions of view and viewpoints; a viewpoint identifies the set of concerns and the representations/modeling techniques [Gro00]. Although no graphical domain-specific language for ISA-95 exists, there exists visualizations, interactions and evolution techniques for similar submodels. Since we do not want *Don't Reinvent The Wheel*, we first identify different submodels and later existing research or tools for similar submodels in order to recontextualize them with the purpose of simplifying the modeling process of ISA-95 compliant models.

The results of the *Task and Goal Analysis* are the definition of the views, respectively diagrams. In Figure 8.1 these views are connected to a business process: Through the navigation, there are three main categories: (i) **Resources** In order to manage the *Physical Asset Model* there exists the *PhysicalAssetTable* for CRUD Operations of

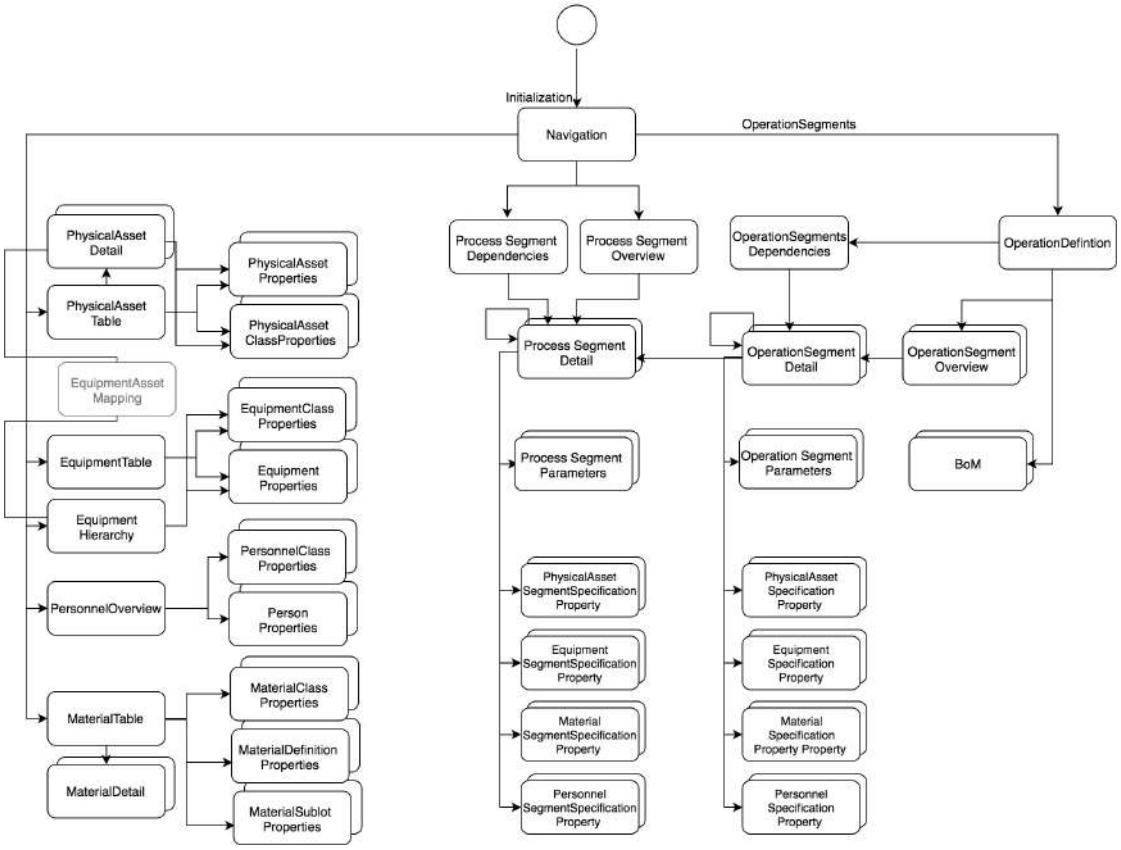


Figure 8.1: Workflow as statechart

PhysicalAssets and *PhysicalAsset Classes*. For handling the Equipment Model, there exists a tabularized and a hierarchical version, which is similar to an organigram. Besides, there is a tabular listing of Equipments and Classes. For the Personnel Model and the Material Model, there exists only a Personnel Table and a Material Model, respectively.

For all resource properties as well as for the Process Segment and Operations Segment exist a table view: (ii) **Process Segment Model** There is a global *Process Segment Overview* and *Process Segment Dependencies*, where you can see the dependencies and material flow between the segments without parent segments, where you can access the detail view. From there you can see and access also the children segments recursively. (iii) **Operation Segment Model** Here you can view the Products in the *OperationDefintion* view. From there you can access the OperationSegmentsDependencies, the BoM and the OperationSegments Overview. where you can see the dependencies and material flow between the segments without parent segements, where you can access the detail view. From there you can see and access also the children segments recursively.

For both, *Process Segment Detail* and *Operation Segment Detail* exists each a Parameter View and for all resource types property views.

8.2 Low Fidelity Mockups

Low-fidelity mockups are rough sketches of UIs. The quick and easy production of these mockups promotes the focus on the essential aspects, which are realized with pen and paper. Using *Pen and paper* is a good way to realize time-efficient prototypes.

8.2.1 Resources

The following chapter deals with selected low-fidelity mockups of the resources and thus gives an insight into the creation process.

Physical Asset

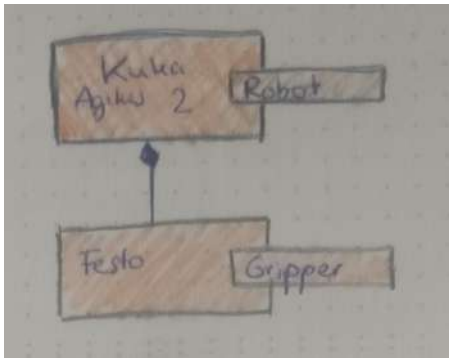


Figure 8.2: Low-fidelity mockup: Physical Asset

Figure 8.2 shows the low-fidelity mockup of the *Physical Asset Model*. Since a Physical Asset can have children of a Physical Asset, we make use of the UML notation for compositions. Physical Asset Classes of the Physical Asset are attached to the Physical Asset on the right side.

Equipment

Figure 8.3 shows the low-fidelity mockup of the *Equipment Model*. Equipments can be hierarchically connected. Equipment Classes of the Equipment are attached to the Equipment on the right side. The connections are derived from organigram like notations, which are used in the context of ISA 95. With this, a user can evolve a tree. Branches can be collapsed or expanded to focus on the creation process.

Personnel

For the Personnel Model, *Persons* are listed and arranged according to the *Personnel Class*. For both, Persons and Personnel Classes, the id and description are shown. The classes, which have a different visual appearance, can be expanded and collapsed.

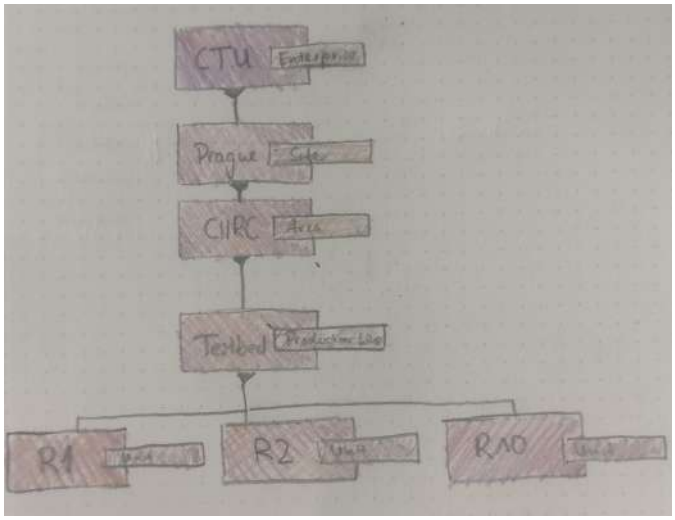


Figure 8.3: Low-fidelity mockup: Equipment Model

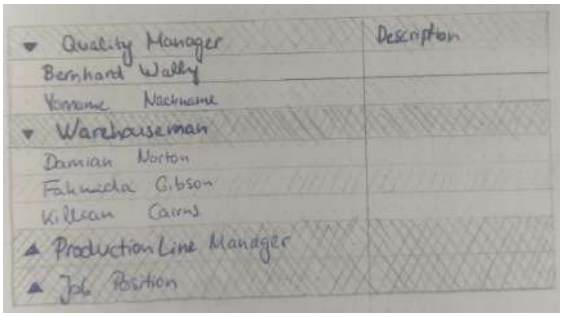


Figure 8.4: Low-fidelity mockup: Personnel Model

8.2.2 Process Segments

The following subsection gives an overview of the development process of process segments. After, that the subprocesses, then the process segment details are presented. Finally the Process Icons are descibed.

Process Segment Overview

The process segments define the logical connection between the ressources for each step on the one hand and on the other hand the connection of these segments in a logical and therefore a temporal way. Research from *business process* and *workflow* management can be used for the segment model. In order to show the overview of process segments, we make use of the Business Process Model and Notation (BPMN) 2.0, which is a graphical specification language in business informatics and process management. Process Segments are abstract operation steps. They function as scaffold of e.g. real production steps.

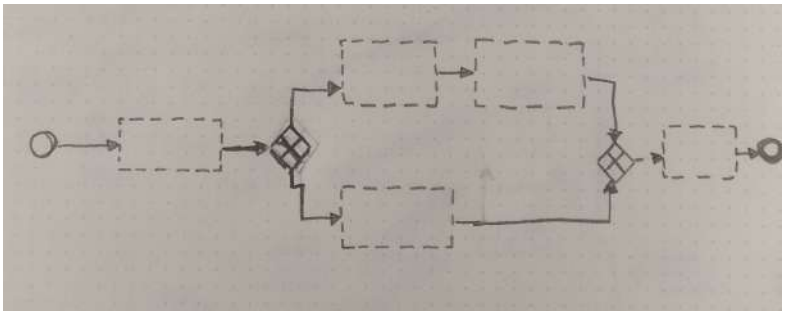


Figure 8.5: Low-fidelity mockup: Process Segment Overview

Therefore, the notation is that the border is dashed as you can see in figure 8.5. Process segments are logically and temporally connected through the Material.

Subprocesses (Hierarchical Operationsteps)

The number of modeling errors and the average or maximum number of elements in a business process model are strongly correlated. For models with more than 50 elements, the error probability tends to be higher than 50%. Mendling et al. propose to structure processes in subprocesses to reduce the complexity for users [MRvdA10]. This concept can be transferred to ISA-95, because the hierarchical composition is supported by the metamodel.

In Business Process Model and Notation these subprocesses are denoted with a plus sign [OMG11].

Process Segment Detail

Process segments have specifications to which the resources are linked. There is need for a cognitively manageable and fast way to get information. In Figure 8.6 a process segment is visualized as a dashed rectangle. Squares on the edge of the sides symbolize the resource specifications. (i) **North** Equipment and Physical Asset specifications are placed here. (ii) **South** Personnel specifications are placed here. (iii) **East** Material specifications, which are consumed, are placed here (iv) **West** Material specifications, which are produced, are placed here.

A click on the Specification Element opens the property view of the specification. Attached to this specification element with the appropriate resources, which are also shown. The connection between the material specification and the material implies the material use type: consumed, produced or consumable. This is realized as follows: (i) consumed materials have an arrow to the process segment (ii) produced materials have an arrow away from the process segment (iii) consumables have no directed arrow.

Figure 8.7 shows the advanced low-fidelity mockup of a *Process Segment Detail* view. The resources, as well as their specifications, are colored to be recognized faster. A plus

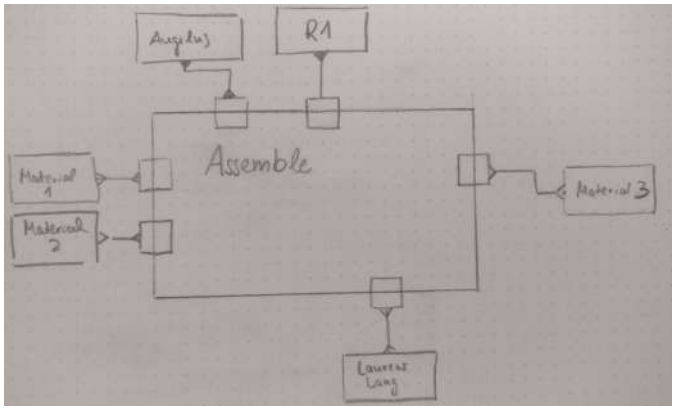


Figure 8.6: Low-fidelity mockup: Process Segment

icon shows if the process has subprocesses. These subprocesses also visualized like in the process overview. The generic visualization of just two layers of the hierarchy enables an unlimited number of reflexive instantiated subprocesses.

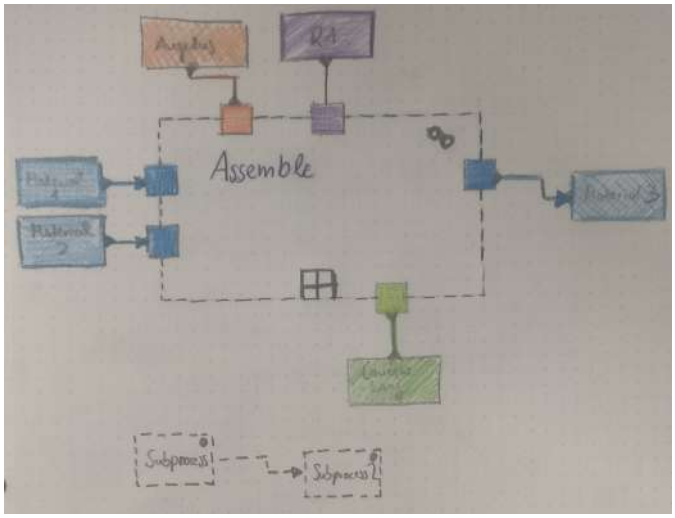


Figure 8.7: Low-fidelity mockup: Coloured Process Segment and Sub Process Segments.

Icons

Process Segments and Operations Segments must be categorized. Possible values are: (i) Production, (ii) Maintenance, (iii) Quality, (iv) Inventory (v) Mixed. Figure 8.8 shows the Icons maintenance, quality, inventory, and production. Mixed is an empty icon.

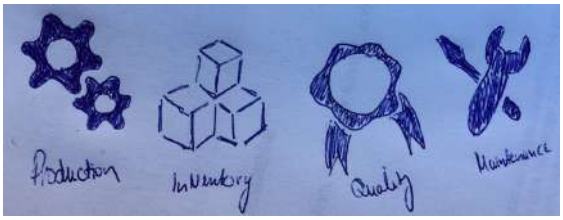


Figure 8.8: Low fidelity icons of production, inventory, quality and maintenance process segments.

8.3 High Fidelity Mockups

As seen above, low-fidelity mockups are rough sketches of UIs. If they reach a certain degree of maturity, electronic prototypes, so-called **high-fidelity mockups**, are created, which are generated with the help of Sketch and in such a way that they look very similar to the end system. Sketch is a vector graphics editor, developed by the Dutch company Bohemian Coding. Sketch focuses on these prototypes also undergo constant iterations, incorporating improvements [Mor15].

8.3.1 Resources

The following section deals with selected high-fidelity mockups of the resources and thus gives an insight into the creation process.

Equipment

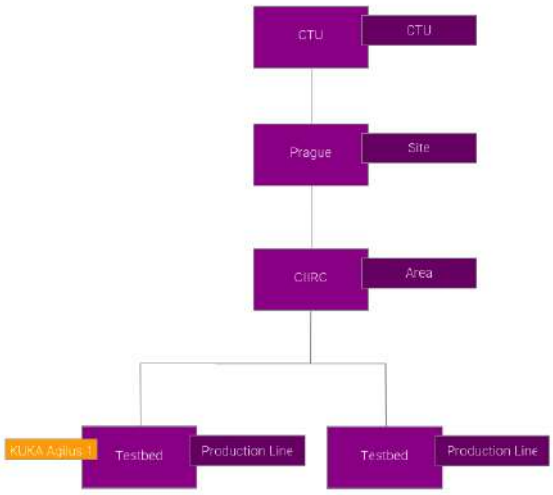


Figure 8.9: Equipment high-fidelity mockup.

In Figure 8.9 the high-fidelity mockup of the *Equipment Model* is presented. Equipments can be hierarchically connected. Equipment Classes of the Equipment are attached to the Equipment on the right side. The connection is derived from organigram like notations, which are used in the context of ISA 95. With this, the user can evolve a tree. Branches can be collapsed and expanded to focus on the creation process. Physical Assets can now be mapped to the appropriate equipment. In this case, the *KUKA Agilus* is mapped to *Testbed*.

8.3.2 Process Segment and Operations Segment

The Business Process Model and Notation (BPMN) 2.0 is a graphical specification language in business informatics and process management. It provides symbols that specialists, methodologists and information technology specialists can use to model and document business processes and workflows. BPMN 2.0 notation consists of activities, which are arranged with the help of *Flow Objects*. Separately artifacts like data objects are created as input or output.

8.3.3 Segment detail

Figure 8.11 shows the high-fidelity mockup of a *Process Segment Detail* view. The resources, as well as their specifications, are colored to be recognized faster. A plus icon shows if the process has subprocesses. The generic visualization of just two layers of the hierarchy enables an unlimited number of reflexive instantiated subprocesses.

Figure 8.10 shows a discarded high-fidelity mockup. A differentiation between the resource and the resource specification like in Figure 8.11 gives more interaction possibilities.

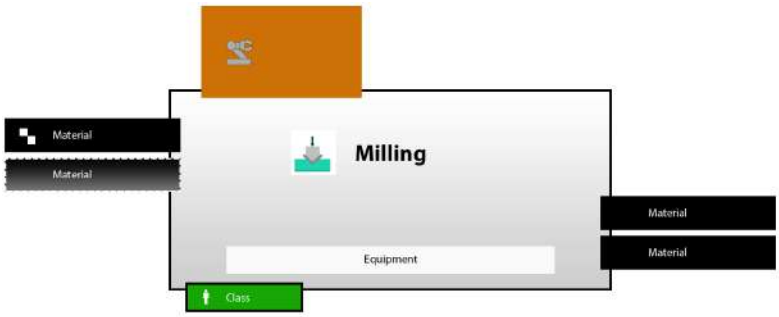


Figure 8.10: Discarded High-fidelity mockup of an Operation Segment

8.3.4 Process Evolution

Hierarchical Processes

There exist a strong positive correlation between the number of modeling errors and the average or maximum degree of elements in a business process model. For models with

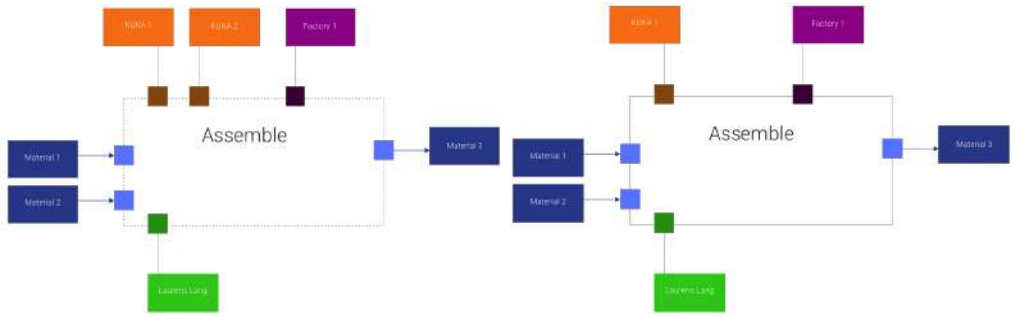


Figure 8.11: High-fidelity mockup: Process Segment on the left. Operation Segment on the right.

more than 50 elements, the error probability tends to be higher than 50%. Mendling proposes to structure process models based on sub-processes to reduce the complexity for users [MRvdA10]. This is supported through the metamodel of ISA-95 This can be transferred to the process segments and operation segments of ISA-95.

Control flow = Production flow?

It is important to analyze the mapping of the concepts of *Process Segment* and *Operations Segments* with BPMN. Process Segment and Operations Segments are activities. Sequence flows describe the order in which activities are executed. If you transfer the concepts there is this sequence flow. The counterpart of material would be data objects. But since modern production consists mostly of assembling, the control flow is in most cases identical to the material flow. Therefore it is not differentiated. However, it is important to detect exceptions: There are two process segment categories, which mostly have the same material as input and output: (i) **inventory** Mostly it is just about the transportation of materials. (ii) **Quality management** The input material is inspected. The output of this process segment is the same material. The upper side of Figure 8.12 shows the resulting connectivity problem. If B would have the same input as output, this results in 2 unnecessary connections, which are eliminated through conditional style if the process segment is defined as inventory or quality.

8.4 Styleguide

Based on the previous requirement analysis from Chapter 5 and the usability drafts from Chapter 6, a style guide was developed for ISA95 Designer. More important than an apposition of single visual ideas is a selection of ideas that are used in several places in the system. Consistency is one of the key principles of usability (c.f. Section 4.1.2), saving users from the requirements to learn patterns, thus fostering concentration on the content. In order to achieve consistency, it is necessary to design guidelines that apply to the entire system (c.f. Semiotic Clarity Section 4.2 and Consistency Section 4.1.2).

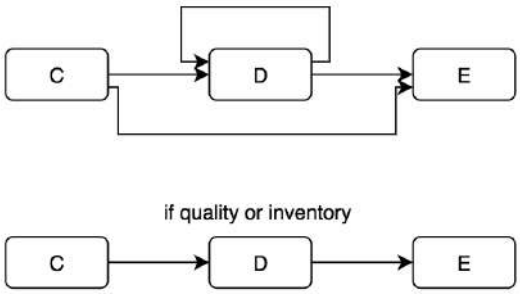


Figure 8.12: Hided connections between process segments.

These elements should be clearly distinguishable from each other (cf. Perceptual Discriminability and Semantic Transparency in Section 4.2).

8.4.1 Visual Design

In this subsection we cover all aspects of the visual design: Consistency throughout the whole system plays an enormous role in the usability 4.1. The universal color scheme, icons, and sizes are defined.

Icons

In this subsection, we present a selection of the created icons.

Resource Icons In figure 8.13 the resource icons are shown: Equipment, Equipment Class, Material, Material Class, Person, Personnel Class, Physical Asset and Physical Asset Class Icon.

Figure 8.13: Equipment, Equipment Class, Material, Material Class, Person, Personnel Class, Physical Asset and Physical Asset Class Icon

Segment Types Process Segments and Operations Segments must be categorized. Possible values are: (i) Production, (ii) Maintenance, (iii) Quality, (iv) Inventory (v) Mixed. Figure 8.14 shows the Icons of these. Mixed is an empty icon. In figure **??** the segment icons are shown: The sub process icon, quality icon, production icon, maintenance icon, inventory icon, and gateway icon.

Figure 8.14: Icons of the Process.

Color palette

Materials are colored dark blue, operation definition turquouise, physical assets orange, equipment purple. Classes are darkened and properties are brighter.

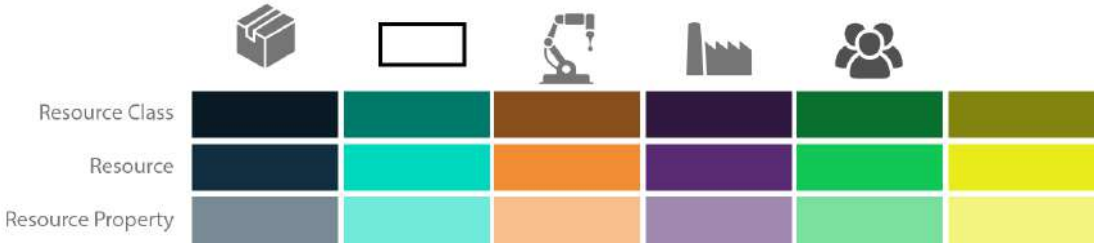


Figure 8.15: Color schema

Sizes and Borders

All visual representations of resources have the same size and no border. Process Segments and Operation Segments have two different sizes. If they are in an overview of multiple process tasks, they are small. If it is a detail view, the corresponding task is big.

8.4.2 Interactions

Not only the static view is relevant in order to create appropriate notations and usability components, but also the dynamic behavior and the evolution techniques play an enormous role. This can be realized with the tools on the right side of a view, which you can see on the right side of the workbench in Figure 8.16. Besides, these tools can be triggered through double-click or right-click. These tools can have an impact on visual nodes, but also on the model elements.

This chapter was a collection of design and implementation ideas. Instead of documenting all views again, for the concrete implementation the following chapter will introduce an example of the ISA-95 Designer.

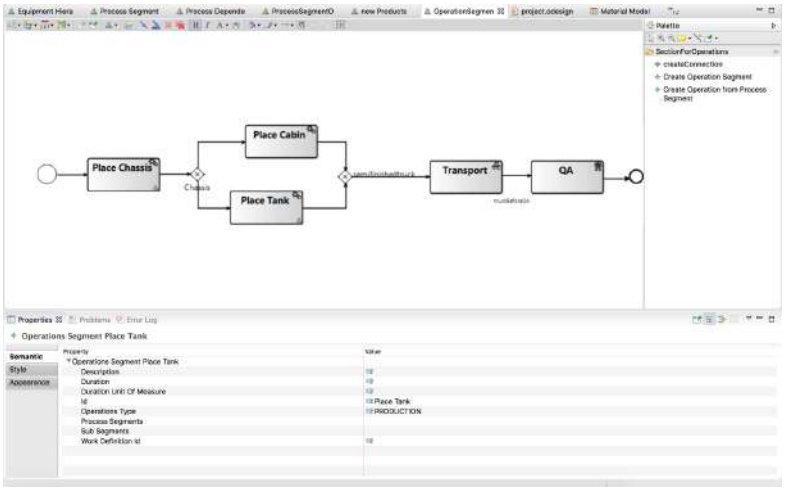


Figure 8.16: Workbench of the ISA-94 Designer.

CHAPTER 9

Case Study: CIIRC Testbed

The Czech Institute of Informatics, Robotics, and Cybernetics (CIIRC) is a department of *Czech Technical University in Prague* (CTU).

In the new building of the CTU CIIRC, a research and experimental environment was created under the name *Testbed for Industry 4.0* (Czech: Testbed pro Průmysl 4.0). Here, automated and digitized production according to Industry 4.0 principles can be tested for innovative solutions for smart factories, verify their compatibility, functionality, and efficiency, simulate, and optimize manufacturing and related in-house processes. Figure 9.1 shows the testbed in detail.

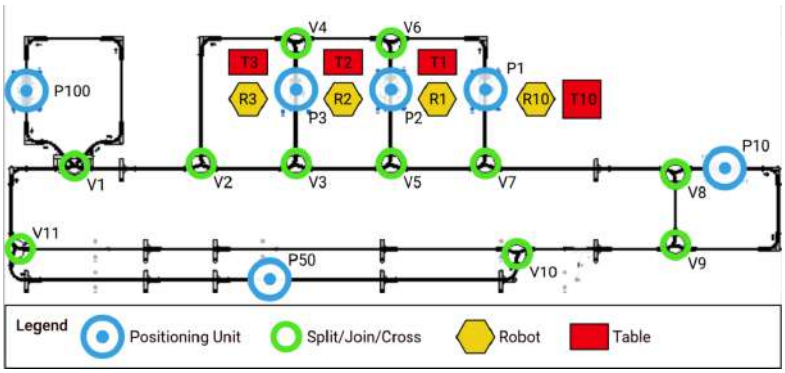


Figure 9.1: Conceptual strucuture of the CIIRC Testbed for Industry 4.0.

In this chapter, the testbed is modeled in the paradigm of ISA-95 in depth. With this, we want to prove, that ISA-95 can be used to describe a production line and contribute a realistic scenario to the knowledge base. It's a fully automated production line. The transportation system is a monorail transportation system by *montrac*. The rails are connected through ten controllable split/join/cross elements. On this rail system, several

shuttles transport small pallets with material, semi-products, and final goods. There exist six positioning units, where objects can be stored. Three of them function as a passive deposition to avoid crossing of positioning units. The other four tables are next to industrial robots. The system is equipped with three KUKA KR Agilus robots and one KUKA LBR iiwa robot[1], performing production operations according to a given production plan.

9.1 Resources

In this chapter we want to descibe the resources in detail with the help of the ISA-95 Designer.

9.1.1 Equipment

Organization

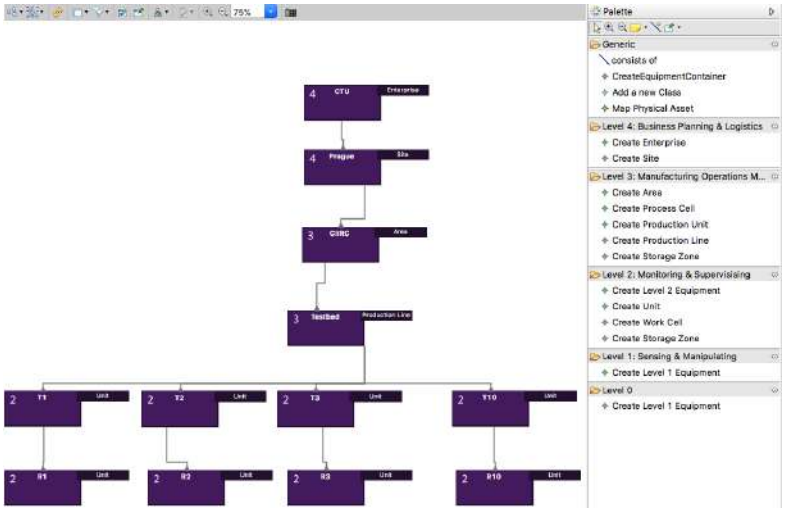


Figure 9.2: Organizational Equipment Model

Figure 9.2 shows the top-down organization structure. The *Czech Technical University*, CTU is an *Enterprise*. *Prague* is the *Site*. Czech Institute of Informatics, Robotics and Cybernetics, *CIIRC* is the textitArea. The *Testbed* is instantiated as *Production Line*, which has several Tables. All tables, Table 1, Table 2, Table 3 and Table 10 are *Units*.

Production Line

The monorail system by Montrac is a modular automation and transport system, which enables flexibility and scalability. The separated shuttles, which rides on the rails, provide vibration-free transport with safety sensor-controlled autostop function to avoid collision

[1] http://www.kuka.com

with possible obstacles. Each shuttle can be controlled independently. The products transported by one conveyor can be different and have different manufacturing processes. The CIIRC Testbeds conveyor system is equipped with three shuttles, that can park in three stations. It is controlled by Siemens S7-1200 PLC. Montratec offers a configurator, which generates a xml file. With the help of model-based techniques, this file can be transformed into an ISA-95 model and offers therefore high flexibility [NKW17]. The instances and classes are shown in figure 9.3.

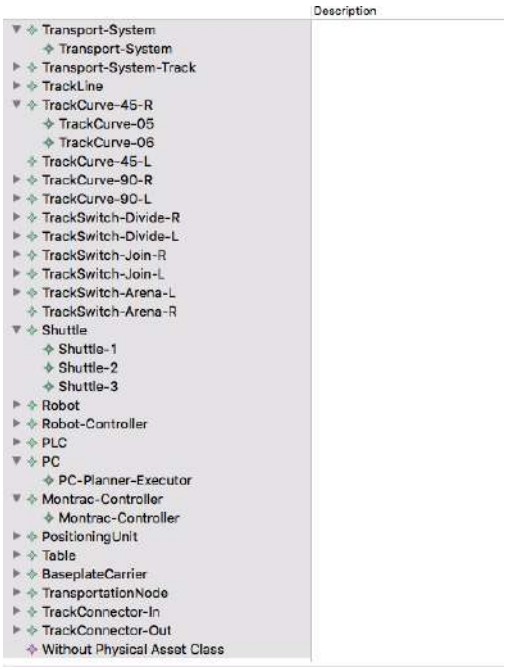


Figure 9.3: Table view of the production line

Physical Asset Mapping Mode

In a special tab in the property view, this mapping can be created and updated, but there is also a graphical notation. The mappings are on a different layer and which can activated and deactivated, which therefore functions as a *Physical Asset Mapping Mode*. If you activate the mapping mode, you can drop the physical asset on the appropriate equipment. In the appearing wizard the physical asset can be selected and the mapping can be refined. As shown in the Figure 9.4, the physical assets are then appended to the left of equipment.

9.1.2 Physical Asset Model

In this Subsection, the views of the Physical Asset Model of the CIIRC Industry 4.0 testbed are presented.

Figure 9.4: Physical Assets are mapped to Equipments.

KUKA Agilus

The Agilus robot is a six-axis robot that performs high-precision movements (repeatability of position up to 0.03 mm) regardless of the installation position. The Agilus robot is designed primarily for high working speeds and is therefore well suited for assembly operations. The robot has been designed to handle payloads of up to 6 kg even in confined spaces. Because of its lifetime lubrication, it requires no lubricant change in the gears and is therefore low maintenance and ensures a truly continuous run. Kuka offers four variants of Agilus: standard version, clean room version, waterproof variant and EX variant, the waterproof version with explosion protection. The robots available in the testbed are standard variants of the Agilus robot. They are controlled by the Siemens ET 200SP I/O system.

KUKA iiwa

The LBR iiwa is a lightweight robot, which meets all ISO criteria for human robots Collaboration. Therefore this robot can work collaborativly with humans. Any cooperation is possible due to the common torque Sensors in each of the robot's seven joints. This enable the reduction of the speed and force whenever the robot detects contact. The accuracy of the torque sensor is 2% of the axis-specific maximum. The torque and the repeatability of the position are 0.15 mm, so the robot is ideal for precise cooperative manipulation. KUKA iiwa Robot can also be programmed with java.

Grippers

All available grippers from Festo are documented in Table 9.1. Stroke implies the maximum stroke of an axis. The data is extracted from the official KUKA website:

The Physical Asssets can be viewed in a Table, which is similar to the MaterialTable in Figure 9.8. Figure 9.5 shows the DetailView of the Physical Asset *KR-Agilus-3*, which is an instance of *KR-Augilus*. The Gripper *DHPS-10-A* is attached to this physical asset.

9.1.3 Material

The goal is to produce a small truck out of a few components, as you can see in the photos.

Name	Stroke	Accuracy
DHPS-10-A	3 mm	0.02 mm
DHDS-16-A	2.5 mm	0.04 mm
DHWS-10-A	20°	0.04 mm
HGPL-14-40-A-B	40 mm	0.03 mm
Suckers ESG	8 mm	-

Table 9.1: The different grippers available at the testbed.

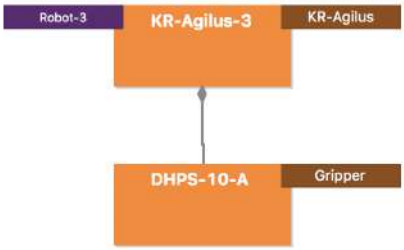


Figure 9.5: Physical Asset Detail View.

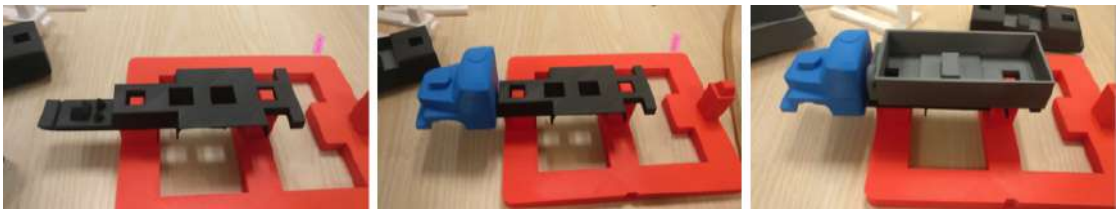


Figure 9.6: Photo of the material components.

Figure 9.7 shows the components in a more conceptual way: (i) **Chassis** has the function to carry the drive, the body and the payload and to stabilize against external forces. (ii) **Cabin** is the part of the structure of the truck that forms the space for drivers. (iii) **Naked Truck** is a composition of the two mentioned materials. (iv) **Tank** (v) **Finished Truck** Finally, this is the finsihed product.

Figure 9.8 shows the Material Table. The Material Class *Truck Component* has the following Instances: Chassis, Cabin, Naked Truck, Tank and Finished Truck. Each instance has a Material Lot. Through righ-click, the properties can be viewed and manipulated in a table.

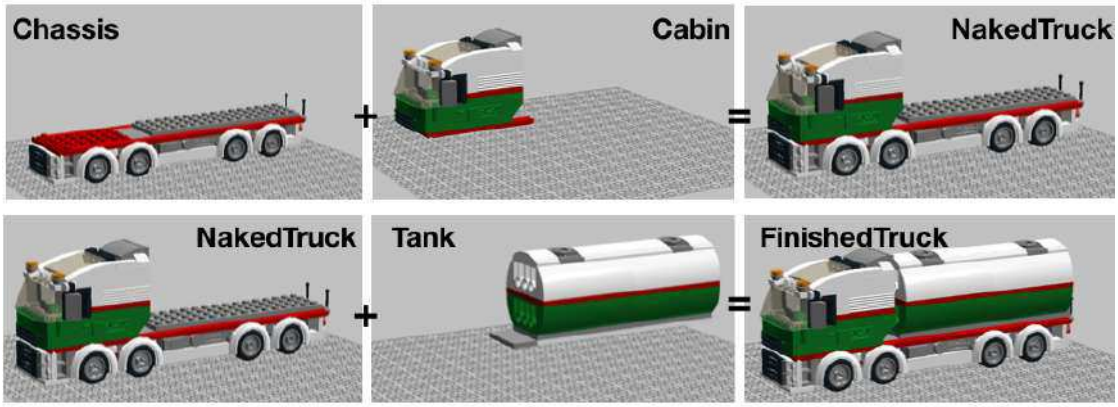


Figure 9.7: Truck Material in a conceptual way.

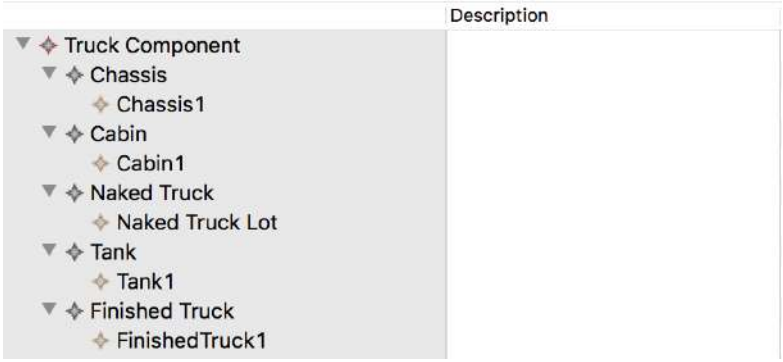


Figure 9.8: Truck Material in the MaterialTable

9.1.4 Personnel

The problem is that the smart factory at CIIRC is fully automated. We therefore integrate to the fully automated process a final operation step, in which a person takes the final product, checks it and deposit it.

9.2 Process Segment Model

This section deals with the process segment model of CIIRC. First, the process segments are listed. Second, one of the process segment is shown in detail.

9.2.1 Process Segment Overview

We identify 4 abstract operation steps, which are shown in Figure 9.9: (i) **Move:** means transportation. The transportation is done over the transportation system. (ii) **Assemble:** is the combination of the following both process segments. (iii) **Pick:** means that a robot picks a specific material. (iv) **Place:** means that a robot places a material on the

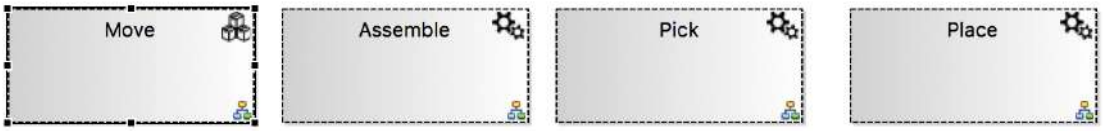


Figure 9.9: Process Segments: *Move*, *Assemble*, *Pick* and *Place*.

platform. As you can see, *Move* has a inventory icon, the other have a production icon. A double-click on the Process Segments opens the Process Segment Detail View.

9.2.2 Assemble in detail

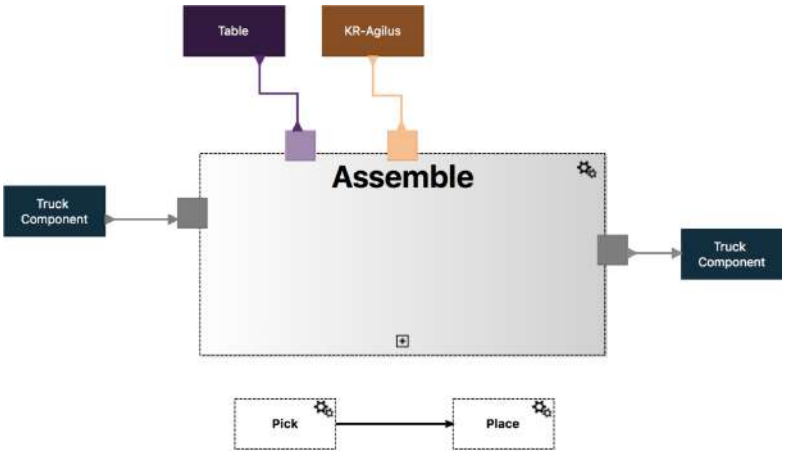


Figure 9.10: Process Segment *Assemble* in detail with the resources.

Figure 9.10 shows the Process Segment *Assemble*. As you can see at the icon, it is a production step. It has Truck Component as consumed Material as well as produced Material. As an equipment specifcation, the equipment class Table and, as a physical asset specification, a physical asset class KR-Augilus is defined. No personnel specification is defined.

As it is also symbolized with the subprocess-cross, it has subprocesses. The two subprocesses are visualized below: Pick and Place, which are connected through a Material Connection.

9.3 Operations Definition Model

This section deals with the operation defintion model of CIIRC. First, the operation defintion and its material bill are presented. Second, the operations segments overview are presented. Third, one of the operations segment is shown in detail.

9.3.1 Operations Definition and Material Bill

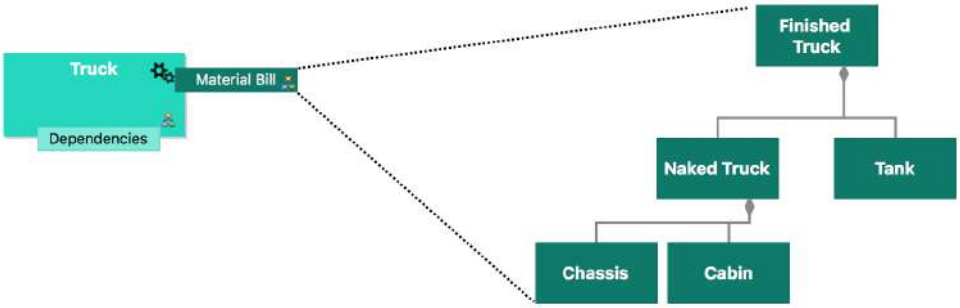


Figure 9.11: Operations Definition and Material of the Truck

Figure 9.11 shows the Operations Definition View on the left. A double-click on *Dependencies* opens the appropriate *Operations Segments Dependencies* View. A double-click on *MaterialBill* opens the MaterialBill View, which is shown on the right side of the Figure. MaterialBillItems can be imported from MaterialDefinitions or MaterialClasses. These can be hierarchically arranged.

9.3.2 OperationSegment Overview

In Figure 9.12 an overview of the Operations Segments is given. This view is for the Operations Definition. This view supports concepts of BPMN. The process model does not support fully the manufacturing process of the truck, but an excerpt. The startsymbol symbolizes, that Place Chassis is the first Process Segment. After that Place Cabin or Place Tank is executed. Afterwards, Transport and, finally, QA is executed. Operations Segments can be created. Material Connection can connect the Operations Segmentss.

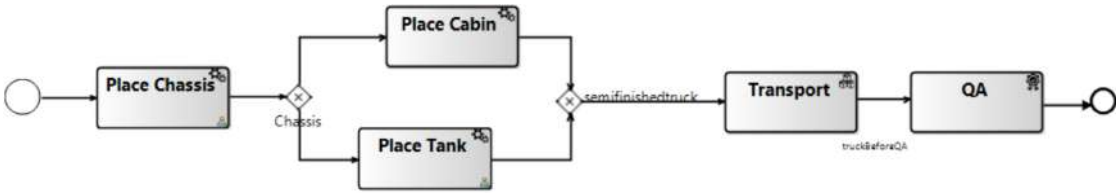
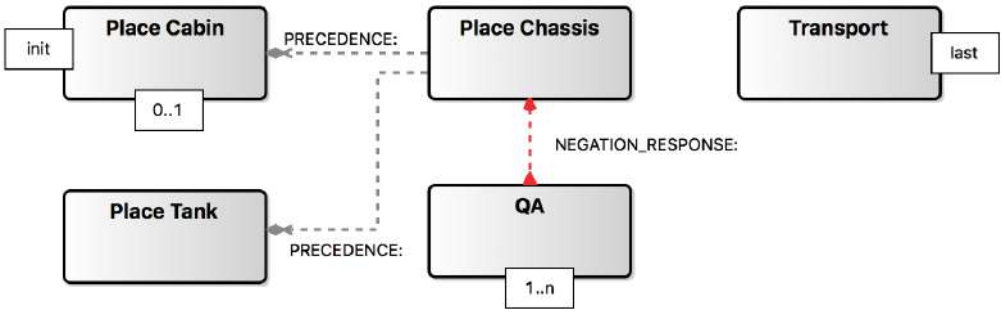


Figure 9.12: Operation Overview with BPMN concepts.

9.3.3 Operations Segments Dependencies

Figure 9.13 shows a selection of the Operations Segments Dependencies: (i) Place Cabin is the start segment. (ii) Place Cabin can be executed maximum once. (iii) QA is executed minimum once. (iv) Transport is the end segment. (v) If Place Cabin is executed, Place Chassis should be exectued beforße. (vi) If Place Tank is executed, Place Chassis should be exectued before. (vii) After QA cannot follow Place Chaissis.

Figure 9.13: Operation Dependencies.



CHAPTER 10

User Study

MDE has a flourishing research community. However, these powerful ideas and possibilities are not well established in the industry [BLR+17]. Usability is a key factor for the adoption of technology, which therefore could be one of the main challenges of MDE. The evaluation with potential users is crucial for the creation of usable products (c.f. 4.1.3) [RF16].

To evaluate the ISA-95 Designer, it is needed to get to know how effective and efficient participants work with it on the one hand. On the other, their mental effort for modeling tasks and the complexity perceived in this context were assessed. Finally, profound insights into the participants' process of modeling instruments are gained — derived from the quantitative process before.

10.1 Pilot Study

First, a *pilot study* was performed on a very small scale in order to refine the study design as well as transcend the own learning process. In order to avoid to distort the results with a learning effect, the participants did not take part in the *Usability Study*.

10.2 Usability Study

A usability study is conducted in order to gain insights about if and to which extent users perform with the Designer compared to the generic tree-based editor.

10.2.1 Study design

Figure 10.1 shows the study design: participants are introduced to the study. Before collecting demographic data, a tutorial about the main ideas of ISA-95 is presented adjusted to their prior knowledge. After that, they have to perform tasks, in which they

modeled different aspects of a production system - alternating with the help of ISA-95 Designer and the generic tool. After that, they to answer a short questionnaire collecting data of their perceived

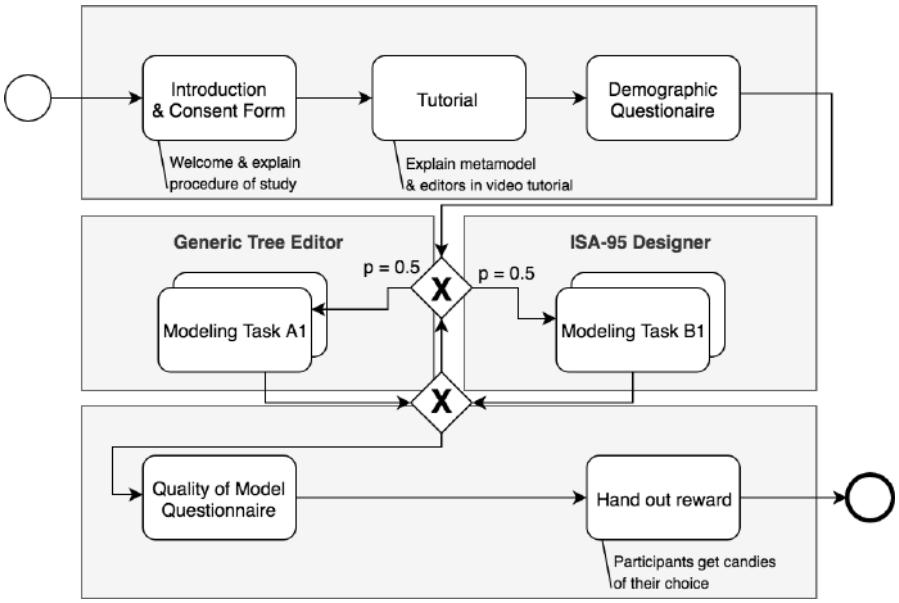


Figure 10.1: Study design of usability study

10.2.2 Participants

Participants are mainly students and research associates – recruited from different departments of Technical University of Vienna, Johannes Kepler University Linz, and the CIIRC department at Czech technical university in Prague. As far as it is possible, practitioners or researchers, who already worked with ISA-95, are involved.

Since this study has participants from various European countries, all materials (e.g., task descriptions, consent form, questionnaires) were provided in English. According to the study design, participants who answered the question with yes were classified as experts. On the other, participants who answered this question with no were classified as novices for the subsequent analysis.

10.2.3 Performance Measures

Hard metrics

Hard metrics use quantitative data points to evaluate how well the user performed assigned tasks with the tool. In this context the following four hard metrics were measured:

(i) **Speed of comprehension** indicates the efficiency of the visual notations of the tool.
(ii) **Speed of development** indicates the efficiency of the development techniques of

the tool. (iii) **Correctness of development** indicates the quality of the development techniques of the tool. (iv) **Correctness of comprehension** indicates the quality of the visual notations of the tool.

Soft Metrics

We make use of the HEART framework, developed by Google, for user-centered metrics, as well as a process for mapping product goals to metrics [RHF10].

(i) **Happiness** This metric relates to subjective aspects, like satisfaction, visual appeal, likelihood to recommend, and perceived ease of use. (ii) **Engagement** Engagement is the user's level of involvement with a product; in the metrics context, the term is normally used to refer to behavioral proxies such as the frequency, intensity, or depth of interaction over some time period. (iii) **Adoption** Adoption tracks how many users adopt the product (iv) **Retention** Track how many of the users from a given time period are still present in some later time period (for example, the percentage of seven-day active users in a given week who are still seven-day active three months later) (v) **Task Success** In order to evaluate the task success, the effectiveness (errors) and the efficiency (time) was measured.

10.2.4 Setting

In order to speed up the evaluation and prevent the noise of the setting, the participants use the same laptops with the same operating system with a computer mouse. They have a cheatsheet, a catalog of the metamodel — separated in chunks, they can easily process for single tasks.

10.2.5 Tasks

All 8 tasks that needed to be processed were comparable regarding their complexity. Note that a difference in the complexity of the models might limit the validity of the obtained results. A change of assessed performance measures, in turn, might be attributed to a diverse model complexity or a measurable learning effect. All tasks were designed in the same way. This includes the textual representation handed out to participants as well as the number of operations needed in the best case. The data collection instruments to be modeled were selected from different submodels of ISA-95. in order to evaluate the feasibility and practical applicability in these settings.

Questionnaires

Additional data is collected using questionnaires through Google Forms. They are logically connected through an ID. A demographic questionnaire collecting personal information was handed out to the participants. More specifically, information regarding prior knowledge on modeling, production, automatization and the used tools was assessed,

as this information was used to classify participants into novices and experts in the shape of a 5 point Likert-scale.

10.3 Results

This Subsection presents all the collected data.

10.3.1 Demographic Questions

Throughout the evaluation, additional data was collected using google forms. For example, a demographic questionnaire collecting personal information (e.g., gender, age or education) was handed out to the participants. Diagram 10.2 shows the demographic data.

1. As you can see, most of the participants (47.7%) are between 25-34. 33,3% are 35-50 years old. The last 20% are between 18 and 24. 2. Most of the participants have a master's degree, 26.7% a PhD, 20% a bachelor's degree and just. 3. Although it was tried to have a balance, 80% of the participants were male, 20% female. 4. The fourth question was about the field of expertise.

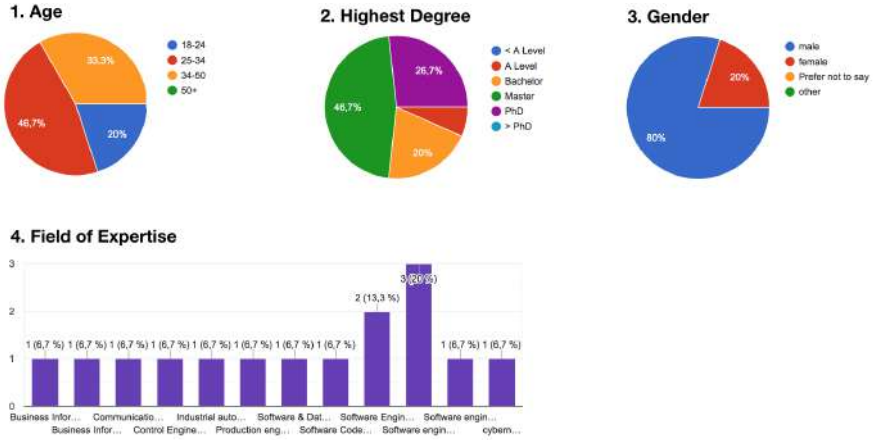


Figure 10.2: Demographic Data

Domain Knowledge Information regarding prior knowledge on the domain was assessed. Figure 10.3 shows, that 53.3% just know about the context of automation. Another small percentage

The next questions were about in which context the participants heard of or worked with ISA-95, (i) Research Project in the Area of Industry 4.0 (ii) Research in models related to I4.0 (RAMI 4.0, ISA, B2MML, ...) (iii) Project Research (RAMI 4.0) (iv) Basic modeling of production systems mainly in conjunction to AutomationML (v) I saw on a tutorial of ISA-95.

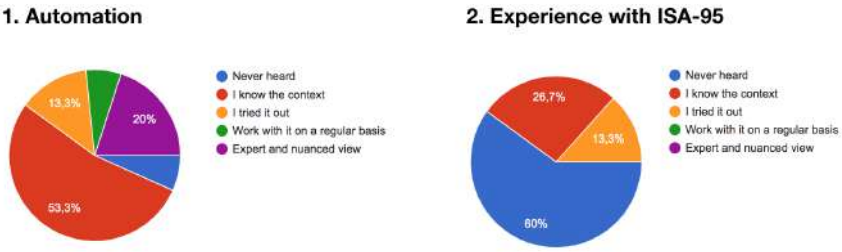


Figure 10.3: Domain knowledge

Modeling Skills Information regarding prior knowledge on the modeling was assessed. As you can see in figure 10.4.

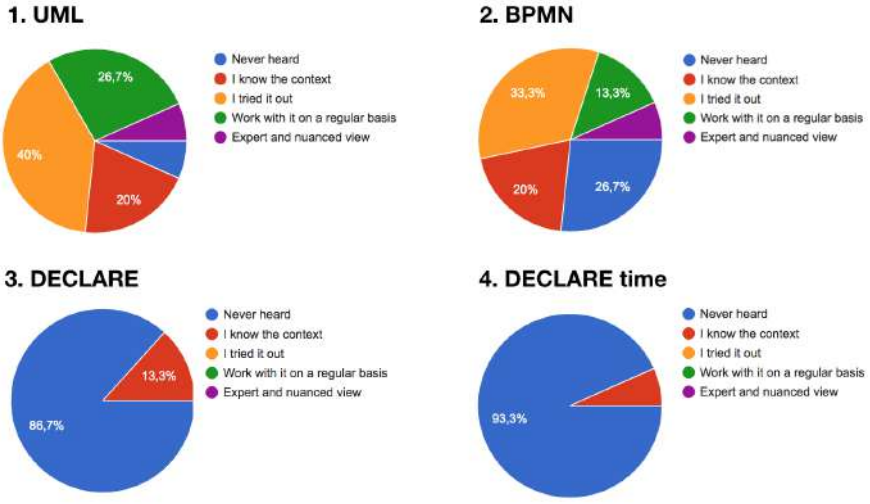


Figure 10.4: Modeling expertise of the participants.

Another question is, what other languages the users had already used. The answers were the following: (i) GLSP, S-BPM, IDEF, SysML, ArchiMate, GraphQL (ii) ARIS, ARCHIMATE (iii) LabView, UPPAAL, Scratch, PLC functional blocks, some databases have graphical interfaces (iv) Quartz Composer, PureData, Automator

Tooling Information regarding prior knowledge on the tooling was assessed and it is documented in figure 10.5

10.3.2 Performed Tasks

Now the evluated data are presented in the following subsections: Firstly, the times, then the number of errors are presented.

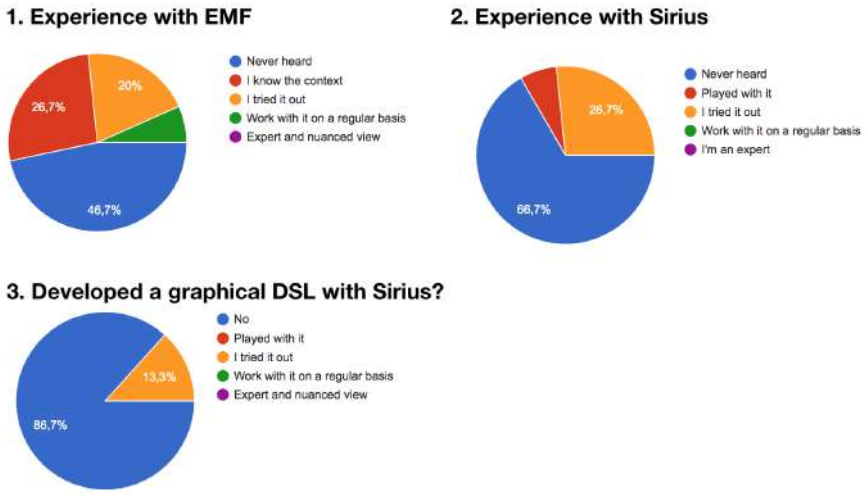


Figure 10.5: Expertise about similar tooling of the participants.

Time

The table shows the average time, which the users needed to perform the tasks. A more detailed list is shown in the appendix.

min	ISA-95 Designer	Generic Editor
Task 3	6 :31	7 :56
Task 4	16 :53	16 :40
Task 5	7 :29	7 :37
Task 6	10 :01	10 :45
Task 7	3 :59	11 :03

Table 10.1: Average time, which the users needed to perform the tasks.

Figure 10.6: Time of the tasks (red: generic editor; blue: ISA-95 Designer

Number of Errors

In order to create a meaningful interpretation it is important to design the data extraction process as transparent as possible. The number of errors was not automatically recognized, but manually, since the automatic approach would overvalue aftereffects, would correct models regard as errors (freedom of modeling), which result in false assumptions.

Nevertheless it is important to regulate this process. In the following, it is documented for each task, what is counted as an error:

	Counted Errors
Task 3	Errors of connections are only counted per hierarchy; other modeling errors like missing elements are individually counted
Task 4	Errors of connections are only counted per hierarchy; other modeling errors like missing elements are individually counted; missing mapping between equipment and physical assets
Task 5	Errors of connections are only counted per hierarchy; other modeling errors like missing elements are individually counted
Task 6	missing resource specifications; missing material connection between sub-processes; wrong resource specifications; missing hierarchical connection of processes; other modeling errors like missing elements are individually counted
Task 7	questions answered incorrectly

Table 10.2: rules, which define, what is counted as an error.

This systematic approach resulted in the following intersection of errors:

avg. errors	ISA-95 Designer	Generic Editor
Task 3	0,5	1,14
Task 4	0,67	1,33
Task 5	0,33	1
Task 6	0,14	1,33
Task 7	0	0,86

Table 10.3: Average errors of the performed tasks.

10.3.3 Process sketches

The ideas behind the sketch task were different. Despite this heterogenity, the sketches can be classified in the following categories:

(i) **BPMN** An excepert of the sketches, which rely mostly on BPMN can be seen in diagram 10.7.

(ii) **Control flow graph** An excepert of the sketches, which rely mostly on Control flow notation can be seen in diagram 10.8.

(iii)**misc** This category are sketches, which cannot really be categorized and are therefore summarized under *misc* and can be seen in diagram 10.9.

The distribution can be viewed in the following table:

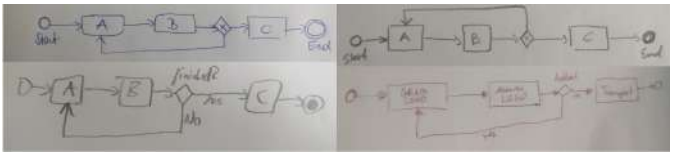


Figure 10.7: BPMN

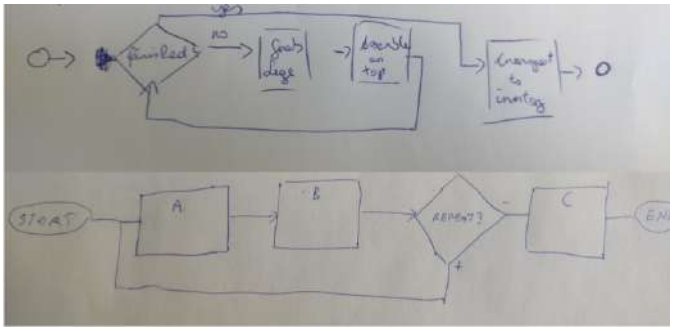


Figure 10.8: Control flow

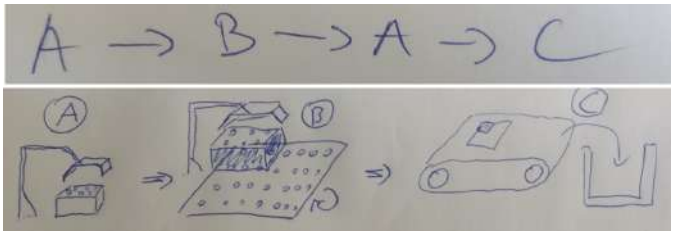


Figure 10.9: Misc

10.3.4 Soft metrics

The Figures of this subsection show the soft metrics. They include the distribution of the answers. On the left side is more likely ISA-95. On the right side is more likely Generic Editor.

Fun

Figure 10.10 shows the distribution of the answers of the question, which editor was more fun to use. 57,1 % answered, it was more fun to use ISA-95 Designer. 35,7 gave neutral answers. Only 7,1% stated, that it was more fun to use the generic editor.

Easier to Use

Figure 10.11 shows the distribution of the answers of the question, which editor was easier to use. 35,7% answered, it was more fun to use ISA-95 Designer. 35,7 gave neutral answers. 28,6% stated, that it was easier to use the generic editor.

Notation	BPMN	Control flow	Misc	Sum
Number	6	3	4	13
Percentage	46,15%	23,08%	30,77%	100%

Table 10.4: Distribution of the different topologies for sketching processes.

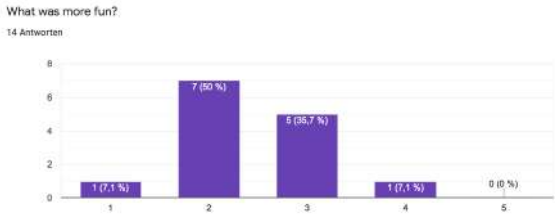


Figure 10.10: Distribution of which was more fun. Left is more ISA-95. Right is more Generic Editor.

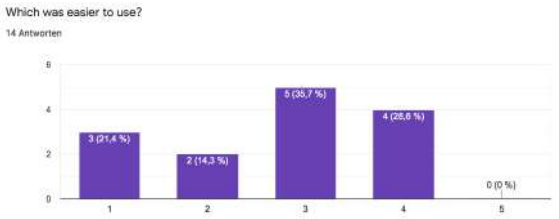


Figure 10.11: Distribution of *Easier to use*?

Engagement

Figure 10.12 shows the distribution of the answers of the question, if they would check the website for upcoming features. 57,1% answered, they would not do it. 42,9% answered, they would.

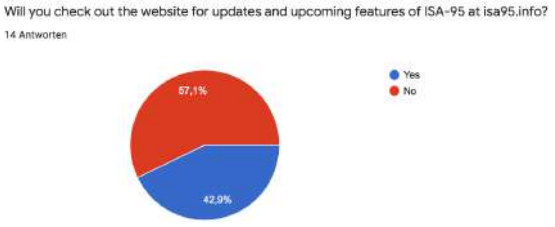


Figure 10.12: Distribution of Engagement.

Adoption

Figure 10.13 shows the distribution of the answers of the question, if they would use the ISA-95 Designer to model ISA-95 models. 92,9% answered they would. Only 7,1% would not.

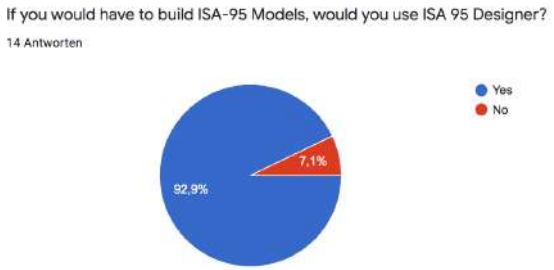


Figure 10.13: Distribution of Adoption.

10.4 Discussion

The overall goal of the study was to evaluate whether end-users are able to develop ISA-95 models when using the ISA-95 Designer. In order to measure their performance, various measures were assessed. The time needed to complete the tasks and the errors were tracked.

In order to create meaningfulness we try to discuss the data in order to arrive as high as possible in the DIKW Hierarchy [Row07].

In order to measure their performance, various measures were assessed. The time and operations needed to complete the tasks were automatically tracked by the configurator. Furthermore, all created models were manually assessed to evaluate potential errors. Finally, participants assessed their performance required to solve the given tasks by filling in self-reporting questionnaires.

10.4.1 Process sketches

The analysis of the topology of the sketches results in a clear statement (c.f. Section 10.3.3):

Process notation

BPMN is the notation, which user intuitively use, for modeling processes. Most of the participants use this notation intuitively.

10.4.2 Resources

Physical Asset

The participants, who used the ISA-95 designer, took in average about 90 seconds less time than in the generic editor. Besides, they made significantly fewer mistakes. Most mistakes done in the generic editor were missing hierarchical connections, which is directly noticeable in the Physical Asset Table.

Equipment

In average, no optimization of efficiency was found. But the evolution of an equipment model is less error-prone. Most mistakes done in the generic editor were missing hierarchical connections or missing connection between the classes and the instances, which is directly noticeable in the Equipment Hierarchy View. In terms of that, ISA-95 is clearly the better solution. What is noticeable is, that the times are so different in the task, that were performed with the ISA-95 Designer. It seems that some participants understood some usability concepts very early on and were therefore able to complete the task quickly, while others could not use them for themselves. In terms of that, it would be interesting to see how the learnability of the tool would improve the efficiency.

Material

The evolution of a material model is in average just minimal more efficient with the ISA-95 Designer. A significant optimization is not noticable. But in terms of quality, ISA-95 is clearly the better solution. The errorrate could be halved. Most mistakes done in the generic editor were missing hierarchical connections, which is directly noticeable in the Material Table.

10.4.3 Process Segments and Operation Segments

The speed of development was not improved. However, the advantages of ISA-95 Designer come to light especially in terms of reading Process segments and Operation Segments. The notation result in a significant better in speed of comprehension and quality. A possible explanation would be, that the process segments have a lot of relationships. Graphical notations provide powerful techniques to visualize these connection in contrast to tree-based editors. The notation supports the overview view, which is a natural way to understand processes. In the generic editor, this facet lies hidden in segment specification and has to be browsed cumbersomely by the end user. This is especially seen in Task 7. Users only needed one-third the time with the ISA-95 Designer than they needed than in the generic editor. Despite the efficiency, the tool also helped them get the task done right.

10.4.4 Process Segment and Operations Definition Dependencies

In this task no times were tracked. It shows that the dependencies were quite clear. No participant made mistakes at Subtask 1, 2, 3, 4, 6 and 7. Therefore the constraints *co-existence*, *starting*, *not co-existence*, *last*, *init* and *existence*. These dependencies were recognized correctly without any doubt.

One Participant, which is 7,69% of all participants, made a mistake in subtask 8, which corresponds to the *maximum* constraint. Due to the small percentage, no notation weakness can be implied, but probably a personal mistake.

But it looks quite different when it comes to subtask 5, which corresponds to *negation-response*. 69,23% made a mistake.The correct answer is $C \rightarrow D$. All the wrong answer marked $B \rightarrow C$ with a cross. A good explanation would have been that between B and C there is no connection. This reveals a deeper problem: What these participants did not understand is a very fundamental concept: Basically, declarative process description enable every combination, unless it violates the constraints.

Overall, it can be clearly stated, that the ISA-95 Designer supports the domain expert to increase the quality of the model. In every task the average error rate was lower than in the models of the generic one.

10.4.5 Soft metrics

The subjective evaluation of the tool revealed that the user happiness, as well as the engangement and adoption potential of the ISA-95 designer were significantly greater. Besides, it was subjectivly easier to use.

Figure 10.14 shows the collected answers. For the first two questions about the users' happiness, a five-point Likert scale with respective answers ranging from "EMF editor" to "Designer" is used. The value in the center denotes a neutral opinion. The polar questions below surveyed the users' engagement, adoption, and retention. Participants seemed to have liked the Designer. Users reported that it was more fun to use the designer than the generic tool and that they would adopt the tool, if they had to model IEC 62264 information.

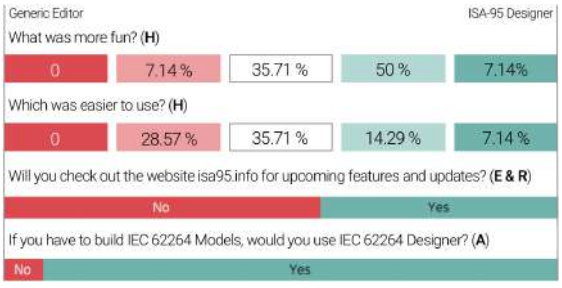


Figure 10.14: Results of the questionnaires.

10.4.6 Summary

Process notation

BPMN is the notation, which user intuitively use, for modeling processes. Most of the participants use this notation intuitively.

Model Quality

The users developed evidently higher quality models.

Speed of Reading Model

The speed of reading models is significantly reduced with the help of ISA-95 Designer.

HEART

The users want to use ISA-95 Designer.

Altogether, the usability study generated valuable findings. In particular, results confirm that end users were able to properly model automated production systems. More precisely, the users developed evidently higher quality results in every single task. For the development of process models, they use intuivly BPMN-like notations and model with the help of ISA-95 Designer significantly more efficient and less error-prone.

CHAPTER 11

Summary

In this chapter, we want to summarize the thesis. Therefore, we give a conclusion in Subsection 11.1 and present the future work in Subsection 11.2.

11.1 Conclusions

11.1.1 Summary

Among the plethora of industrial standards available in the context of smart manufacturing, one series of standards is consistently being mentioned for dealing with manufacturing operations management: ISA-95. Despite its popularity, the integration of this standard is cumbersome and yet not-accessible for non-technical people. ISA-95 Designer empowers no-coders to develop and maintain these systems:

ISA-95 Designer was developed for automated production systems. In an iterative search process mockups were sketched, evaluated, discarded, discussed and refined until they reached maturity. The developed concepts were implemented. After that, the artifacts were evaluated in three different locations (CIIRC in Prague, JKU Linz and TU Wien). The integration of usability engineering methods reulted in a significantly positive impact on the quality of the modeled systems and partially faster development process.

(i) No coding skills required. (ii) Improves the quality of the developed models. (iii) Speeds up the development time. (iv) ISA-95 and therefore ISA-95 Designer offers a sematic validation. (v) Access to other technologies. (vi) Can be used to teach ISA-95. (vii) Can be used to learn exploratively ISA-95.

11.1.2 Lessons learned

User Study effort

My personal time estimation of the user study were underpredicted. Although that has taken so much time, only parts of the tool have been tested in very artificial tests and it is far away from a holistic test.

Hofstadter's Law

It always takes longer than you expect, even when you take into account Hofstadter's Law [Hof79].

Usability Study Setup

The laptops used for the study were not optimal. Due to the low resolution many tools of the ISA-95 Designer were hidden and one did not have a large design space. In conclusion, it can be said that unfortunately they have badly influenced the study and therefore the evaluation of the tool.

Eclipse Sirius

Eclipse Sirius advertises the reduction of time to develop graphical toolkits. This statement is only correct to some extent. In the initial learning phase I made a lot of progress. For example, I had developed diagrams relatively quickly. However, with more complex problems and the development of tools was extremly cumbersome. Some ideas were not possible to implement without massive workarounds. Due to the lack of documentation of more complex examples and lack of possibilities to debug, it can be concluded that Sirius is more likely to create prototypes of design toolkits. The lack of documentation must not necessarily be a result from poor engineering but a business model. The company behind sirius provide consulting and development of visial modeling tools[1].

11.2 Future Work

This section describes the derived future requirements of the system. This includes Process Model Evolution techniques in Section 11.2.1, which are summarized, prioritized and estimated in a Production Backlog in Section 11.2.2. After that, the possibility to move it to the cloud is presented in Section 11.2.3.

[1] Obeo cf. http://obeo.fr

11.2.1 Process Evolution Techniques

The future development of process segments and operations defintion model can further optimized by using techniques from existing software tools[2] and existing research [Bob08].

Delete Between

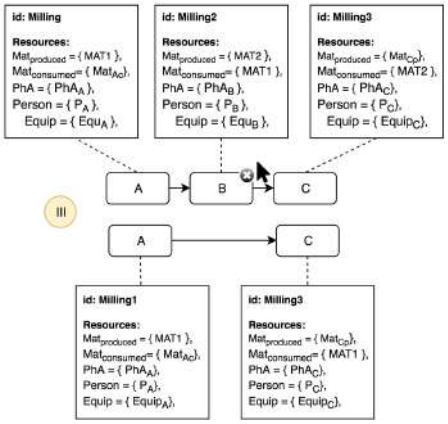


Figure 11.1: Storyboard of *Delete Interaction*.

Delete is used to remove segments from the process model. Application finds this tperation, for example, if unnecessary details are hidden in the process visualization. This operations is used if the user realizes a redundant segment. However, the user don't want to break the material flow. Figure 11.1 shows the constellation before and after the deletion of the segment. On a algortithmic level the segment is deleted. All the resources of the remaing segments stay unchanged except one: The consumed material of C is replaced with the produced material of A.

Create Between

Create between inserts a segment B between nodes A and C visually. From the model perspective, a proess segment is generated. For the physical asset, personnel and eqipment resource is created. The idea, which is presented in Fig. 11.2 behind that is, that it is much easier to reduce the resources than to create them. However, there is need of input, since the process model should have an on-going material flow: A dialog queries the name and the material, which is inserted as produced material of B. C changes the consumed material to the new material as well.

Create Similiar

The CIIRK model shows that there are a lot of consecutive process segment with similar resource specifications. Since the deletion or change of a resource specification is much

[2]Camunda BPMN tool: https://camunda.com/

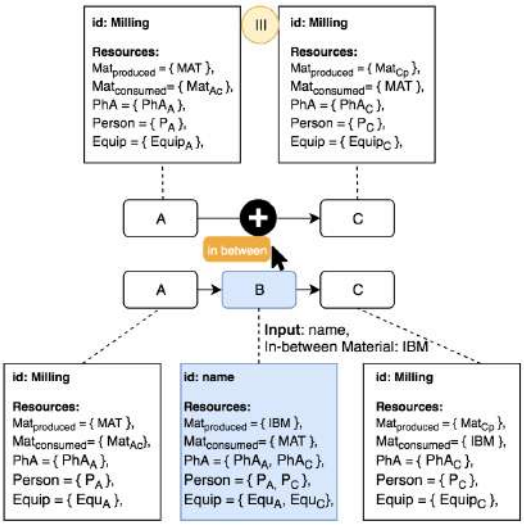


Figure 11.2: Storyboard of *Create between*.

easier than the creation, it makes sense to provide a tool *Create similar*, which is presented in Figure 11.3. *Create similar after* copies the process segment B. However, the materials are deleted from that, a new material is queried through a dialog. This material is defined as produced material of C. The consumed material is defined from the material produced of B. *Create similar before* copies the process segment B. However, the materials are deleted from that, a new material is queried through a dialog. This material is defined as consumed material of A. The produced material is defined from the consumed material of B.

11.2.2 Join

In various situations two process branches have to be joined. Especially from the point of view of the 7PMG guideline G3 ("use only one end event"): So there is a best practice, that all branches always should be joined to a single branch [MRvdA10]. Figure 11.4 shows the three steps of the interaction of two process branches. In this case a new segment E is created, which takes the produced material of both selected segments, B and C, as consumed material. As in the examples above, the name and the produced material is queried through a dialog.

Figure 11.5 shows the three steps of the interaction of three process branches. In this case a new segment G is created, which takes the produced material of three selected segments, B, D, and F, as consumed material. As in the examples above, the name and the produced material is queried through a dialog.

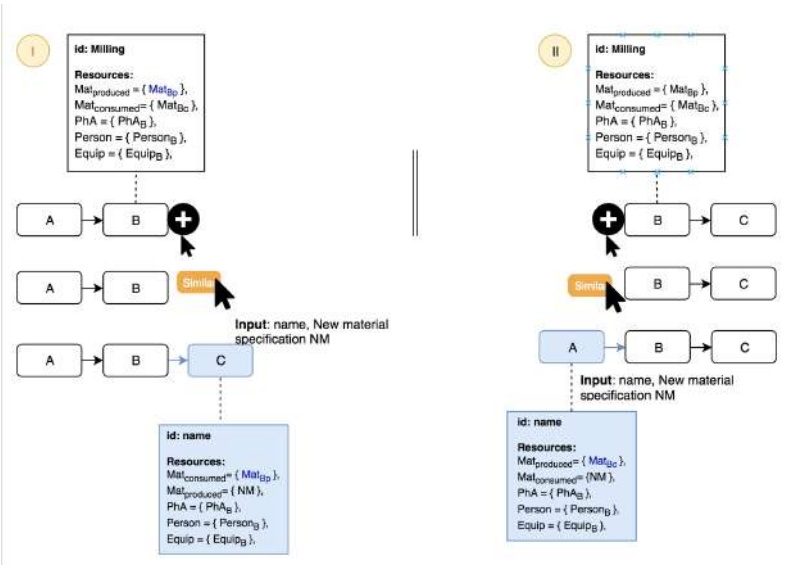


Figure 11.3: Storyboard of *Create similar*. On the left side is the creation of a segment ater, on the right side before a sequence of segments.

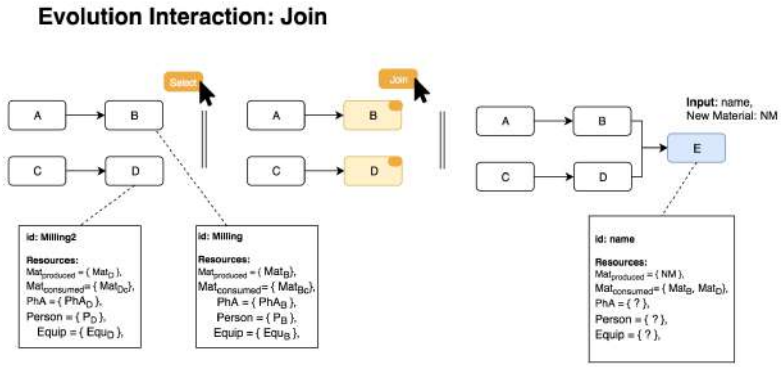


Figure 11.4: Storyboard of *Join* of two process tasks.

Backlog

Table 11.1 shows the current product backlog of the ISA-95 Designer. The backlog is prioritized and estimated.

11.2.3 To the web

Currently, model-driven engineering is usually done with the help of traditional Integrated Development Environments (IDEs) such as the Eclipse platform. The Eclipse Modeling Framework (EMF) provides support for modeling projects in Eclipse itself. If such a modeling environment for a domain-specific language needs to be made available for other

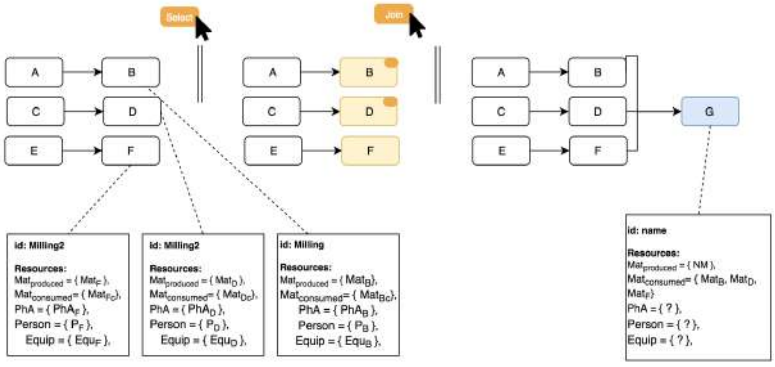


Figure 11.5: Storyboard of *Join* of three process tasks.

Feature	Description	Priority	Hours
Delete between	Errors of connections are only counted per hierarchy; other modeling errors like missing elements are individually counted	1	30
Create between	Errors of connections are only counted per hierarchy; other modeling errors like missing elements are individually counted; missing mapping between equipment and physical assets	2	30
Create similar	Errors of connections are only counted per hierarchy; other modeling errors like missing elements are individually counted	3	25
Join	missing resource specifications; missing material connection between subprocesses; wrong resource specifications; missing hierarchical connection of processes; other modeling errors like missing elements are individually counted	4	45

Table 11.1: Backlog of future requirements.

users to implement a domain specific task, the setup of such IDEs may be cumbersome and time-consuming. Throughout this paper we want to explore the possibility that web-based (or Electron-based) IDEs combined with a Language Server Protocol (LSP) offer for model engineering [REIWC18].

Web-based IDEs promise to enable access to projects from anywhere or any computer without the need to install a lot of software locally. Minimal configuration is needed and there is a centralized workspace, which enables development on the go. Additionally, this approach allows development from inexpensive machines, since stuff like validation,

compiling, etc. happens on a separate machine. To enable this approach, the Language Server Protocol can be used. It provides a common protocol to provide features like auto-completion, error highlighting or content assist to the client (e.g an IDE which may be an editor running in a browser). This features are provided by a language server which the client communicates with.

Bibliography

[aca13] acatech – Deutsche Akademie der Technikwissenschaften e.V. Umsetzungsempfehlungen für das Zukunftsprojekt Industrie 4.0. Abschlussbericht, 2013.

[ASE16] Hisham M. Abdelsalam, Amal R.S. Shoaeb, and Magy M. Elassal. Enhancing Decision Model Notation (DMN) for Better Use in Business Analytics (BA). In *Proceedings of the 10th International Conference on Informatics and Systems*, INFOS '16, pages 321–322, New York, NY, USA, 2016. ACM.

[BCFP19] Barbara Rita Barricelli, Fabio Cassano, Daniela Fogli, and Antonio Piccinno. End-user development, end-user programming and end-user software engineering: A systematic mapping study. *Journal of Systems and Software*, 149:101 – 137, 2019.

[BCW12] Marco Brambilla, Jordi Cabot, and Manuel Wimmer. *Model-Driven Software Engineering in Practice*. Morgan & Claypool Publishers, 1st edition, 2012.

[BLR+17] Francis Bordeleau, Grischa Liebel, Alexander Raschke, Gerald Stieglbauer, and Matthias Tichy. Challenges and research directions for successfully applying MBE tools in practice. In *MODELS*, 2017.

[Bob08] Ralph Bobrik. Konfigurierbare Visualisierung komplexer Prozessmodelle, 2008.

[Bro01] P. Brooks. Ethernet/ip-industrial protocol. In *ETFA 2001. 8th International Conference on Emerging Technologies and Factory Automation. Proceedings (Cat. No.01TH8597)*, volume 2, pages 505–514 vol.2, Oct 2001.

[Cab07] Jordi Cabot. From declarative to imperative uml/ocl operation specifications. 11 2007.

[CAGT15] Juan Cadavid, Mauricio Alférez, Sébastien Gérard, and Patrick Tessier. Conceiving the model-driven smart factory. In *Proceedings of the 2015 International Conference on Software and System Process*, ICSSP 2015, pages 72–76, New York, NY, USA, 2015. ACM.

[Cas85] Albert F. Case. Computer-aided software engineering (case): Technology for improving software development productivity. *SIGMIS Database*, 17(1):35–43, September 1985.

[Cza05] Krzysztof Czarnecki. Overview of generative software development. In *Proceedings of the 2004 International Conference on Unconventional Programming Paradigms*, UPP'04, pages 326–341, Berlin, Heidelberg, 2005. Springer-Verlag.

[Deu16] Deutsches Institut für Normung (DIN). *DIN SPEC 91345: Referenzarchitekturmodell Industrie 4.0 (RAMI4.0)*, 2016.

[DH14] R. Drath and A. Horch. Industrie 4.0: Hit or hype? [industry forum]. *IEEE Industrial Electronics Magazine*, 8(2):56–58, June 2014.

[DK15] Rainer Drath and Heiko Koziolek. Industrie 4.0: Im Spannungsfeld zwischen dem Machbaren und Sinnvollen. *atp edition – Automatisierungstechnische Praxis*, 57(1-2):28–35, 2015.

[DLPH08] R. Drath, A. Luder, J. Peschke, and L. Hundt. Automationml - the glue for seamless automation engineering. In *2008 IEEE International Conference on Emerging Technologies and Factory Automation*, pages 616–623, Sep. 2008.

[FGY+04] G. Fischer, E. Giaccardi, Y. Ye, A. G. Sutcliffe, and N. Mehandjiev. Meta-design: A manifesto for end-user development. *Commun. ACM*, 47(9):33–37, September 2004.

[FHK+15] Stefan Feldmann, Sebastian J.I. Herzig, Konstantin Kernschmidt, Thomas Wolfenstetter, Daniel Kammerl, Ahsan Qamar, Udo Lindemann, Helmut Krcmar, Christiaan J.J. Paredis, and Birgit Vogel-Heuser. Towards effective management of inconsistencies in model-based engineering of automated production systems. *IFAC-PapersOnLine*, 48(3):916 – 923, 2015. 15th IFAC Symposium onInformation Control Problems inManufacturing.

[GDD06] Dragan Gasevic, Dragan Djuric, and Vladan Devedzic. *Model Driven Architecture and Ontology Development*. 01 2006.

[GF08] Elisa Giaccardi and Gerhard Fischer. Creativity and evolution: A metadesign perspective. *Digital Creativity - DIGIT CREAT*, 19:19–32, 03 2008.

[Gro00] IEEE Architecture Working Group. IEEE Std 1471-2000, Recommended practice for architectural description of software-intensive systems. Technical report, IEEE, 2000.

[Hei15] Heinz Nixdorf Institut der Universität Paderborn, Werkzeugmaschinenlabor WZL der Rheinisch-Westfälischen Technischen Hochschule Aachen.

Handlungsempfehlungen – Zukünftige Rahmenbedingungen für die Industrie 4.0-Wirtschaft in Deutschland. Technical report, 2015.

[HLML19] Dazhuang He, Andrei Lobov, and Jose Luis Martinez Lastra. ISA-95 tool for enterprise modeling. 01 2019.

[HMPR04] Alan R. Hevner, Salvatore T. March, Jinsoo Park, and Suda Ram. Design science in information systems research. *MIS Quarterly*, 28(1):75–106, 2004.

[Hof79] Douglas R. Hofstadter. *Godel, Escher, Bach: An Eternal Golden Braid*. Basic Books, Inc., New York, NY, USA, 1979.

[HPO15] Mario Hermann, Tobias Pentek, and Boris Otto. Design principles for industrie 4.0 scenarios: A literature review, 01 2015.

[HR15] Martin Hankel and Bosch Rexroth. The reference architectural model industrie 4.0 (rami 4.0). *ZVEI*, 2:2, 2015.

[Int] Manufacturing Enterprise Solutions Association International. Business to manufacturing markup language (b2mml).

[Int13a] International Electrotechnical Commission (IEC). Enterprise-control system integration—part 1: Models and terminology, 2013. IEC 62264-1:2013.

[Int13b] International Electrotechnical Commission (IEC). Enterprise-control system integration – part 2: Objects and attributes for enterprise-control system integration, 2013. IEC 62264-2:2013.

[Int16] International Electrotechnical Commission (IEC). Iec 62424:2016: Representation of process control engineering - requests in p&i diagrams and data exchange between p&id tools and pce-cae tools, 2016. IEC 62424.

[Int18] International Electrotechnical Commission (IEC). Iec 62714-1:2018 Engineering data exchange format for use in industrial automation systems engineering - Automation Markup Language - part 1: Architecture and general requirements, 2018. IEC 62714-1:2018.

[ISO05] Enterprise modelling and architecture – Requirements for enterprise-reference architectures and methodologies. Standard, International Organization for Standardization, June 2005.

[ISO06a] ISO. Ergonomics of human-system interaction — Part 110: Dialogue principles, 2006.

[ISO06b] Enterprise integration – Framework for enterprise modelling. Standard, International Organization for Standardization, April 2006.

[ISO07] Enterprise integration – Constructs for enterprise modelling. Standard, International Organization for Standardization, April 2007.

[ISO13] Automation systems and integration – Evaluating energy efficiency and other factors of manufacturing systems that influence the environment – Part 1: Overview and general principles. Standard, International Organization for Standardization, May 2013.

[ISO14] ISO. Automation systems and integration — Key performance indicators (KPIs) for manufacturing operations management — Part 2: Definitions and descriptions, 2014.

[JMJ+16] S. Jäger, R. Maschotta, T. Jungebloud, A. Wichmann, and A. Zimmermann. Creation of domain-specific languages for executable system models with the eclipse modeling project. In *2016 Annual IEEE Systems Conference (SysCon)*, pages 1–8, April 2016.

[Jun16] Michael Junker. Flexible graphical editors for extensible modular meta models. Master's thesis, Karlsruher Institut für Technologie (KIT), 2016.

[KD10] Yildiray Kabak and Asuman Dogac. A survey and analysis of electronic business document standards. *ACM Comput. Surv.*, 42(3):11:1–11:31, March 2010.

[KNH16] Udo Kannengiesser, Matthias Neubauer, and Richard Heininger. Integrating business processes and manufacturing operations based on s-bpm and b2mml, 2016.

[KSS16] Serope Kalpakjian, Steven R. Schmid, and K. S. Vijay Sekar. *Manufacturing engineering and technology*. Pearson Education South Asia Singapore, seventh edition in SI units. edition, 2016 2016.

[KWH13] Henning Kagermann, Wolfgang Wahlster, and Johannes Helbig. Recommendations for implementing the strategic initiative industrie 4.0 – securing the future of german manufacturing industry. Final report of the industrie 4.0 working group, acatech – National Academy of Science and Engineering, München, 4 2013.

[Lan13] Laurens Lang. Entwicklung eines usability-konzepts für ein prozessorientiertes informationssystem zum austausch von nachhaltigkeitsdaten. October 2013.

[LM06] Stefan-Helmut Leitner and Wolfgang Mahnke. Opc ua - service-oriented architecture for industrial applications. *Softwaretechnik-Trends*, 26, 2006.

[LMF16] Yan Lu, Kc Morris, and Simon Frechette. Current standards landscape for smart manufacturing systems. *Nat. Inst. Stand. Technol.*, 01 2016.

[LMHM10] Daniel L. Moody, Patrick Heymans, and Raimundas Matulevičius. Visual syntax does matter: Improving the cognitive effectiveness of the i* visual notation. *Requirements Engineering*, 15:141–175, 06 2010.

[LPW06] Henry Lieberman, Fabio Paternò, and Volker Wulf. *End User Development (Human-Computer Interaction Series)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[LS87] Jill H. Larkin and Herbert A. Simon. Why a diagram is (sometimes) worth ten thousand words. *Cognitive Science*, 11(1):65 – 100, 1987.

[LWH+] Laurens Lang, Bernhard Wally, Christian Huemer, Radek Šindelár, Alexandra Mazak, and Manuel Wimmer. A graphical toolkit for IEC 62264-2. *Procedia CIRP*. accepted for publication.

[Mag14] Fabrizio Maria Maggi. Discovering metric temporal business constraints from event logs. In Björn Johansson, Bo Andersson, and Nicklas Holmberg, editors, *Perspectives in Business Informatics Research*, pages 261–275, Cham, 2014. Springer International Publishing.

[May99] Deborah J. Mayhew. *The Usability Engineering Lifecycle: A Practitioner's Handbook for User Interface Design*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 1999.

[McC82] William E. McCarthy. The rea accounting model: A generalized framework for accounting systems in a shared data environment. *The Accounting Review*, 57(3):554–578, 1982.

[MH12] D. Mayrhofer and C. Huemer. REA-DSL: Business model driven data-engineering. In *2012 IEEE 14th International Conference on Commerce and Enterprise Computing(CEC)*, volume 00, pages 9–16, 09 2012.

[MH15] A. Mazak and C. Huemer. Hover: A modeling framework for horizontal and vertical integration. In *2015 IEEE 13th International Conference on Industrial Informatics (INDIN)*, pages 1642–1647, July 2015.

[Moo09] D. Moody. The "physics" of notations: Toward a scientific basis for constructing visual notations in software engineering. *IEEE Transactions on Software Engineering*, 35(6):756–779, Nov 2009.

[Mor15] S. Morson. *Designing for iOS with Sketch*. 2015.

[MP15] Frédéric Madiot and Marc Paganelli. Eclipse sirius demonstration. In *P&D@MoDELS*, volume 1554 of *CEUR Workshop Proceedings*, pages 9–11. CEUR-WS.org, 2015.

[MRvdA10] Jan Mendling, Hajo A. Reijers, and Wil M. P. van der Aalst. Seven process modeling guidelines (7pmg). *Information & Software Technology*, 52(2):127–136, 2010.

[MWBD18] Tanja Mayerhofer, Manuel Wimmer, Luca Berardinelli, and Rainer Drath. A model-driven engineering workbench for CAEX supporting language - customization and evolution. *IEEE Trans. Industrial Informatics*, 14(6):2770–2779, 2018.

[MWH+19] Alexandra Mazak, Manuel Wimmer, Christian Huemer, Bernhard Wally, Thomas Frühwirth, and Wolfgang Kastner. *Rahmenwerk zur modellbasierten horizontalen und vertikalen Integration von Standards für Industrie 4.0*. Springer, 2019. submitted for review.

[NKW17] P. Novak, P. Kadera, and M. Wimmer. Model-based engineering and virtual commissioning of cyber-physical manufacturing systems — transportation system case study. In *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–4, Sep. 2017.

[OAHB13] Toshio Ono, Shahzad Ali, Paul Hunkar, and Dennis Brandl. *OPC Unified Architecture for ISA-95 Common Object Model Companion Specification*. OPC Foundation, 10 2013. Rev. 3.

[OMG11] OMG. Business Process Model and Notation (BPMN), Version 2.0, January 2011.

[Par98] H. Partsch. *Requirements-Engineering Systematisch*. Springer, 1998.

[Pec09] Rick Pechter. What's PMML and what's new in PMML 4.0? *SIGKDD Explor. Newsl.*, 11(1):19–25, November 2009.

[Pic10] Roman Pichler. *Agile Product Management with Scrum: Creating Products That Customers Love*. Addison-Wesley Professional, 1st edition, 2010.

[PK02] Risto Pohjonen and Steven Kelly. Domain-specific modeling. *Dr. Dobb's Journal*, 27, 08 2002.

[PSv07] M. Pesic, H. Schonenberg, and W. M. P. van der Aalst. Declare: Full support for loosely-structured processes. In *11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007)*, pages 287–287, Oct 2007.

[REIWC18] Roberto Rodriguez-Echeverria, Javier Luis Cánovas Izquierdo, Manuel Wimmer, and Jordi Cabot. Towards a Language Server Protocol infrastructure for graphical modeling. In *Proceedings of the 21th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*, MODELS '18, pages 370–380, New York, NY, USA, 2018. ACM.

[RF16] Michael Richter and Markus D. Flückiger. *Usability und UX kompakt*. IT kompakt. Springer Vieweg, Berlin, Heidelberg, 4. aufl. 2016 edition, 2016.

[RHF10] Kerry Rodden, Hilary Hutchinson, and Xin Fu. Measuring the user experience on a large scale: User-centered metrics for web applications. In *Proceedings of CHI 2010*, 2010.

[Rit14] Tom Ritchey. General morphological analysis - a general method for non-quantified modelling. 2014.

[Row07] Jennifer Rowley. The wisdom hierarchy: representations of the DIKW hierarchy. *J. Inf. Sci.*, 33:163–180, 2007.

[Sch06] Douglas C. Schmidt. Guest editor's introduction: Model-driven engineering. *Computer*, 39(2):25–31, February 2006.

[Sch07] B. Scholten. *The Road to Integration: A Guide to Applying the ISA-95 Standard in Manufacturing*. ISA, 2007.

[Sch13] A.W. Scheer. *CIM Computer Integrated Manufacturing: Der computergesteuerte Industriebetrieb*. Springer Berlin Heidelberg, 2013.

[Sel03] Bran Selic. The pragmatics of model-driven development. ieee softw. 20(5), 19-25. *Software, IEEE*, 20:19 – 25, 10 2003.

[SH10] Thomas Stober and Uwe Hansmann. *The Flaw in the Plan*, pages 1–14. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[SK03] S. Sendall and W. Kozaczynski. Model transformation: the heart and soul of model-driven software development. *IEEE Software*, 20(5):42–45, Sep. 2003.

[SSM05] C. Scaffidi, M. Shaw, and B. Myers. Estimating the numbers of end users and end user programmers. In *2005 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05)*, pages 207–214, Sep. 2005.

[Sta73] Herbert Stachowiak. *Allgemeine Modelltheorie*. 1973.

[Sta98] International Organization For Standardization. *ISO 9241-11 - Definition and Concepts*. Freuburg i.B., 1998.

[Sta15] I.O.F. Standardization. *ISO/IEC 15944-4: Information technology – Business operational view – Part 4: Business transaction scenarios – Accounting and economic ontology*. 2015.

[SVEH07] Thomas Stahl, Markus Völter, Sven Efftinge, and Arno Haase. *Modellgetriebene Softwareentwicklung: Techniken, Engineering, Management*. dpunkt, Heidelberg, 2 edition, 2007.

[Tho04a] D. Thomas. Mda: revenge of the modelers or uml utopia? *IEEE Software*, 21(3):15–17, May 2004.

[Tho04b] Dave Thomas. Mda: revenge of the modelers or uml utopia? *Software, IEEE*, 21:15 – 17, 06 2004.

[TWW17] Klaus-Dieter Thoben, Stefan Wiesner, and Thorsten Wuest. Industrie 4.0 and smart manufacturing – a review of research issues and application examples. *International Journal of Automation Technology*, 11:4–19, 01 2017.

[VMP14] Vladimir Vujovic, Mirjana Maksimovic, and Branko Perisic. Sirius: A rapid development of dsm graphical editor. 07 2014.

[Wal18] Bernhard Wally. Provisioning for MES and ERP. Application recommendation, TU Wien, 2018.

[WHM17a] B. Wally, C. Huemer, and A. Mazak. Aligning business services with production services: The case of rea and isa-95. In *2017 IEEE 10th Conference on Service-Oriented Computing and Applications (SOCA)*, pages 9–17, Nov 2017.

[WHM17b] B. Wally, C. Huemer, and A. Mazak. A view on model-driven vertical integration: Alignment of production facility models and business models. In *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, pages 1012–1018, Aug 2017.

[WHM17c] Bernhard Wally, Christian Huemer, and Alexandra Mazak. Isa-95 based task specification layer for rea in production environments. In *Proceedings of the 11th International Workshop on Value Modeling and Business Ontologies (VMBO 2017)*, 2017.

[WHMW18a] Bernhard Wally, Christian Huemer, Alexandra Mazak, and Manuel Wimmer. AutomationML, ISA-95 and others: Rendezvous in the OPC UA universe. In *Proceedings of the 14th IEEE International Conference on Automation Science and Engineering (CASE 2018)*, 2018.

[WHMW18b] Bernhard Wally, Christian Huemer, Alexandra Mazak, and Manuel Wimmer. Iec 62264-2 for automationml. 10 2018.

[Wir78] Niklaus Wirth. *Algorithms + Data Structures = Programs*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1978.

[WLW+19] Bernhard Wally, Laurens Lang, Rafal Wlodarski, Radek Sindelar, Christian Huemer, Alexandra Mazak, and Manuel Wimmer. Generating structured automationml models from iec 62264 information. In *Proceedings of the 5th AutomationML PlugFest 2019*, 2019.

[WSH+17] B. Wally, M. Schleipen, R. Henßen, N. Schmidt, N. D'Agostino, and Y. Hua. Automationml auf höheren automatisierungsebenen. Eine Auswahl relevanter Anwendungsfälle. In *Beitrag vom 18. Leitkongress der Mess- und Automatisierungstechnik, Baden-Baden*, 2017.

[ZMVS16] F. Zezulka, P. Marcon, I. Vesely, and O. Sajdl. Industry 4.0 – an introduction in the phenomenon. *IFAC-PapersOnLine*, 49(25):8 – 12, 2016. 14th IFAC Conference on Programmable Devices and Embedded Systems PDES 2016.

[Zü10] Detlef Zühlke. Smartfactory-towards a factory-of-things. *Annual Reviews in Control*, 34:129–138, 04 2010.

Appendix User Study Documents

In this chapter, the documents of the study are attached. First, the Consent form is shown. Second, the first questionnaire . Third, the two exercise sheets A and B follow. Third, the second questionnaire follows. Fourth, the cheat sheet that users had as help to complete the study.

Fifth, the a table shows the numbers of errors of the participants of the usability study. Sixth, the the table of the times of the performed task are documented in a table.

Sehr geehrte(r) Teilnehmer(in),

vielen Dank für Ihre Bereitschaft an dem von der Technischen Universität durchgeführten Software-Studie teilzunehmen.
Die Studie dient der Evaluierung einer graphischen domänenspezifische Sprache für ISA-95.

Vor Beginn des Interviews möchten wir Sie über die Verwendung der Daten informieren.

Weiterverwendung der Daten
Die Daten, die Sie bei diesem Interview angeben, werden von uns anonymisiert und nur im Kontext der Masterarbeit von Laurens Lang ausgewertet. Ihre Angaben sind für diese Arbeit von großer Bedeutung und deshalb äußerst wertvoll.

Ich bin damit einverstanden, dass meine Daten wie oben beschrieben aufgezeichnet und verwendet werden.

Dear Participant,

Thank you for your willingness to participate in the software study conducted by the Technical University.
The study is designed to evaluate a graphical domain-specific language for ISA-95.

Before starting the interview, we would like to inform you about the use of the data.

Data reuse
The data you provide in this interview will be anonymized by us and will only be evaluated in the context of the master's thesis by Laurens Lang. Your information is very important for this work and therefore extremely valuable.

I agree that my data will be recorded and used as described above.

_____ _____

Datum
/ Date

Unterschrift des Teilnehmers
signature of the participant

Level 1

Demographic Questionnaire:

Thank you for your willingness to participate in the software study conducted by Laurens Lang. The study is designed to evaluate a graphical domain-specific language for ISA-95.

* Erforderlich

1. **Define your unique ID, which connects your questionnaires with your models.** *

2. **Age?** *
Markieren Sie nur ein Oval.
- 18-24
- 25-34
- 34-50
- 50+

3. **Highest degree?** *
Markieren Sie nur ein Oval.
- < A Level
- A Level
- Bachelor
- Master
- PhD
- > PhD

4. **How many hours do you use a computer daily?** *
Beispiel: 8:30 Uhr

5. **Gender**
Markieren Sie nur ein Oval.
- male
- female
- Prefer not to say
- other

6. **What's your field of expertise? (Software engineering, Automation engineering, control engineering, etc.)**

Domain Knowledge

7. **Automatization** *
Markieren Sie nur ein Oval.
- Never heard
- I know the context
- I tried it out
- Work with it on a regular basis
- Expert and nuanced view

8. **Have you experience with ISA 95?** *
Markieren Sie nur ein Oval.
- Never heard
- I know the context
- I tried it out
- Work with it on a regular basis
- Expert and nuanced view

9. **If yes: Why? What have you developed? In which context(es)?**

Modeling Skills

10. **Do you know UML?** *
Markieren Sie nur ein Oval.
- Never heard
- I know the context
- I tried it out
- Work with it on a regular basis
- Expert and nuanced view

11. **Do you know BPMN?** *
Markieren Sie nur ein Oval.
- Never heard
- I know the context
- I tried it out
- Work with it on a regular basis
- Expert and nuanced view

12. **Do you know DECLARE?** *
Markieren Sie nur ein Oval.

- Never heard
- I know the context
- I tried it out
- Work with it on a regular basis
- Expert and nuanced view

13. **Do you know DECLARE time?** *
Markieren Sie nur ein Oval.

- Never heard
- I know the context
- I tried it out
- Work with it on a regular basis
- Expert and nuanced view

14. **Other graphical DSLs?**

Tooling

15. **Have you used Ecore/EMF?** *
Markieren Sie nur ein Oval.

- Never heard
- I know the context
- I tried it out
- Work with it on a regular basis
- Expert and nuanced view

16. **Have you used sirius based Editors (ECore Editor, ArchiMate, Capella)?** *
Markieren Sie nur ein Oval.

- Never heard
- Played with it
- I tried it out
- Work with it on a regular basis
- I'm an expert

17. **Have you developed a graphical DSL with Sirius?** *
Markieren Sie nur ein Oval.

- No
- Played with it
- I tried it out
- Work with it on a regular basis
- Expert and nuanced view

ISA95 Designer Evaluation: Level 2

Congratulations! You passed Level 1. You **unlocked Level 2:**

The CIIRC (Czech Institute of Informatics, Robotics and Cybernetics) has a Industrie 4.0 Testbed.

They want to connect their MES now with an ERP System. In order to make this possible, you have to model the resources and the processes in the paradigm of ISA-95.

1. Kickstart:

Get used to the tools. Explore the tool on your own. Click around. You can't do anything wrong.

2. The Joy of Painting: Processes

The following tasks exist:

A) Grab LEGO brick

B) Assemble LEGO

C) Transport finished master piece to the inventory

Sketch the following process sequence on the paper.
Grab a LEGO brick. Assemble it on top of a LEGO base plate. Repeat steps **A** and **B** eventually or transport it to the inventory **C**.

B

3. Physical Asset Model (ISA-95 Designer)

4 Robots are in your factory of the 2 Robottypes:
i) *KUKA LBR iiwa*
ii) *KUKA Agilus*

1. Create two PhysicalAssetClasses:
   i) *KUKA LBR iiwa*
   ii) *KUKA Agilus*

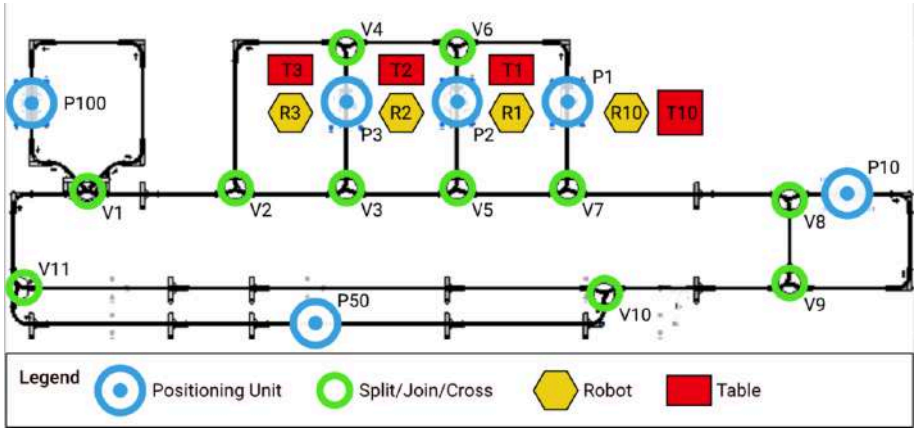
2. Create the properties. The property view can be opened via right click:
   i) *KUKA LBR iiwa*: Precision: 0.06mm
   ii) *KUKA Agilus*: Precision: 0.09 mm

3. Ensure that 3 instances of *Agilus were created*: **KUKA Agilus 1, KUKA Agilus 2** and **KUKA Agilus 3.**

4. Create an instance of *LBR iiwa* class called **KUKA LBR iiwa 1**.

4. Equipment Model (Generic)

Equipment Model describes an organisational structure.



1. Equipment Model. Model the following organizational Structure:
The enterprise Czech Technical University **CTU** has a *site* in **Prague**. One *area* of this *site*, is the **CIIRC** (Czech Institute of Informatics, Robotics and Cybernetics). The **CIIRC** has a *production line*, the Industry 4.0 **Testbed.** Create this organisational hierarchy using **instances** of type *equipment*.

2. The **Testbed** consists of a railway transport system and four robot cells. The transport system is highly complex, therefore we want to sum it up with a single piece of equipment called, **Transportsystem.** In addition to that, you should create the four **Robotcells**.

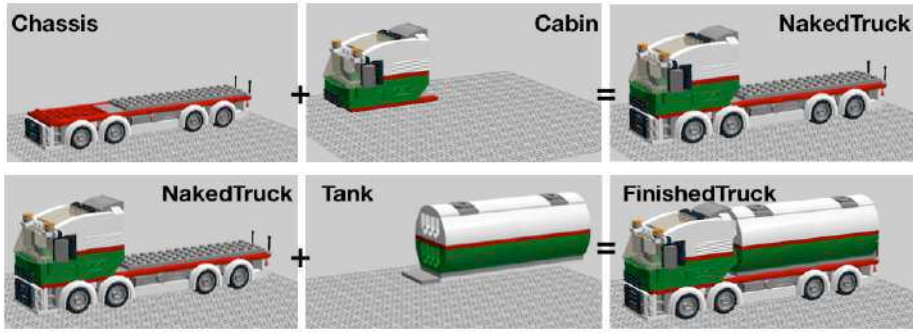
3. Now fill up these **Robotcells**: Each Robotcell consists of a **Table** (*Unit*) and a **Robot** (*Unit*); **T1** and **R1**, **T2** and **R2**, **T3** and **R3**, and **T10** and **R10**.

4. Now map the three *KUKA Agilus* instances to **R1**, **R2**, **R3** and *KUKA LBR iiwa* instance to **R10**. Therefore you have to create an EquipmentAssetMapping, which is contained in the PhysicalAssetModel.

B

5. Material Model (ISA-95 Designer)

Now, the materials have to be created.



1. Create a MaterialClass called *Truck Component*.
2. Now create the following MaterialDefinitions, which belong to the MaterialClass *Truck Component*:
   - **i) Chassis**
   - **ii) Cabin**
   - **iii) NakedTruck**
   - **iv) Tank**
   - **v) FinishedTruck**

3. Create MaterialLots for the following MaterialDefinitions: **Chassis**, **Cabin** and **Tank**. Each MaterialLot represents a quantity of 1000 pieces.

6. Process Segment (Generic)

Process Segments are logical connectors of resources, a scaffold for real production steps.
In the case of the ISA-95
Create four hierarchical structured Process Segments:

- **Transport:** A Truck component has to be transported through the transport system via the shuttle
- **Pick and Place:** A robot, sometimes also a Warehouse Worker, picks a component with a robot and place it on the shuttle.
  - **Pick**
  - **Place**

1. Create the **Transport** Process Segment
2. Create the **Pick and Place** Process Segment
3. Now connect the resources to the Process Segments:

Transport
- Material: *Truck Components*
- Equipment: **Montratec Shuttle**

Pick'n'Place
- Physical Asset Class: KUKA *Augilus*
- Personnel class: *Warehouse Worker*
- Equipment: **Platform**
- Material: *Truck Components*

4. Create Subprocess Segments in Pick and Place: **Pick**, and **Place.** Therefore you have to create them first inside the Process Segment Model and then define in the parent process segment, that they are subprocesses.

Connect them with a Material Class Connection (*Truck Component*):
Pick:
- *Truck Component* as Material Segment Specification with the attribute MaterialUse=Produced.
Place:
- *Truck Component* as Material Segment Specification with the attribute MaterialUse=Consumed.

B

7. Product Overview (ISA-95 Designer)

You are almost finished. Just 2 small tasks and you will win a candy. Browse through the Operation Overview and answer the following questions.

There can be multiple answers.

Which Material Bill Item is missing if you compare it with the Material exercise?

☐ Chassis
☐ Cabin
☐ Naked truck
☐ Tank

Which are valid Operation sequences? (multiple possible)

☐ Place Chassis, Place Cabin, Transport, QA
☐ Place Chassis, Place Tank,Transport, QA
☐ Place Chassis, Place Trank, Place Cabin, QA, Transport

Which is the Starting Operation?

☐ Place Chassis
☐ Place Cabin
☐ Place Tank
☐ Transport

Which is the ending Operation?

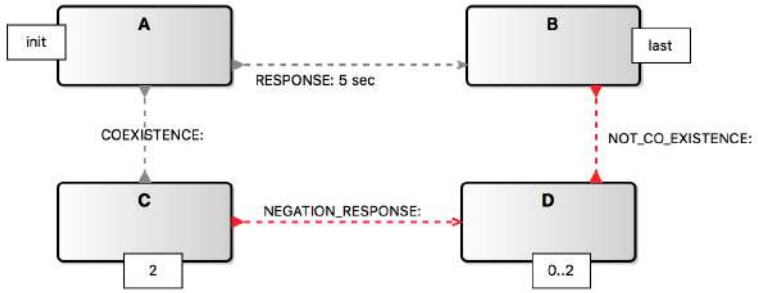
☐ Place Chassis
☐ Place Cabin
☐ Place Tank
☐ Transport
☐ QA

Which Person is involved in the Place Tank?

☐ Laurens Lang
☐ Bernhard Wally
☐ Max Mustermann

8. Operation Dependencies

However there can be constraints in the production, which have logical or temporal dependencies between production steps. The following diagram shows these dependencies.



There is always just one correct answer.

Which OperationsSegment cannot coexist?
□A and B
□ A and C
□ B and C
□ C and D
□ B and D

Which is the starting OperationsSegment?
□ A
□ B
□ C
□ D

Which OperationsSegment can coexist?
□ A and B
□ A and C
□ B and C
□ C and D
□ B and D

Which is the ending OperationsSegment?
□ A
□ B
□ C
□ D

Which OperationsSegment cannot follow?
□ A -> B
□ B -> C
□ C -> D
□ B -> D

How often can C be exectued?
□ 1
□ 2
□ 3

Which OperationsSegment can follow after some seconds?
□ B after A
□ C after B
□ D after C
□ D after B

Which is a valid number of executions for D?
□ 1
□ 3
□ 4

ISA95 Designer Evaluation: Level 2

Congratulations! You passed Level 1. You **unlocked Level 2:**

The CIIRC (Czech Institute of Informatics, Robotics and Cybernetics) has a Industrie 4.0 Testbed.

They want to connect their MES now with an ERP System. In order to make this possible, you have to model the resources and the processes in the paradigm of ISA-95.

1. Kickstart:

Get used to the tools. Explore the tool on your own. Click around. You can't do anything wrong.

2. The Joy of Painting: Processes

The following tasks exist:

A) Grab LEGO brick

B) Assemble LEGO

C) Transport finished master piece to the inventory

Sketch the following process sequence on the paper.
Grab a LEGO brick. Assemble it on top of a LEGO base plate. Repeat steps **A** and **B** eventually or transport it to the inventory **C**.

A

3. Physical Asset Model (Generic)

4 Robots are in your factory of 2 Robottypes:
i) *KUKA LBR iiwa*
ii) *KUKA Agilus*

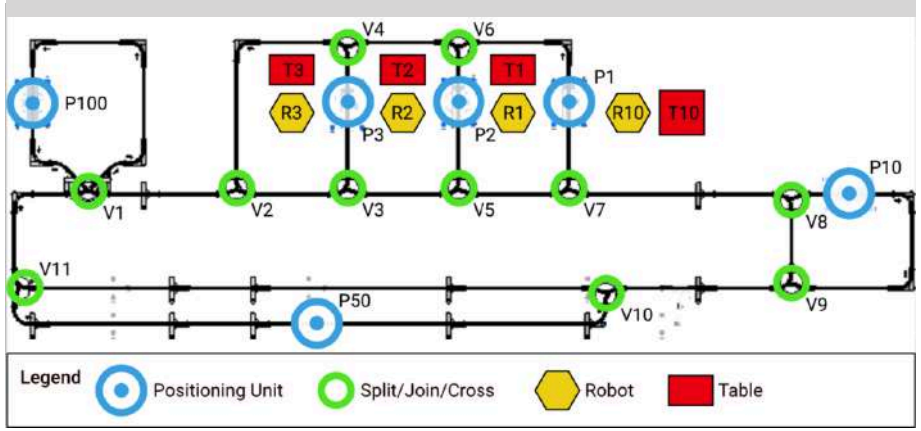
1. Create two Physical Asset Classes:
   i) *KUKA LBR iiwa*:
   ii) *KUKA Agilus*

2. Create the Physical Asset Class Properties:
   i) *KUKA LBR iiwa*: Precision: 0.6mm
   ii) *KUKA Agilus*: Precision: 0.9 mm

3. Now create 3 instances of *KUKA Agilus*: **KUKA Agilus 1, KUKA Agilus 2** and **KUKA Agilus 3**.
4. Create 1 instance of *KUKA LBR iiwa* called **KUKA LBR iiwa 1**.

4. Equipment Model (ISA 95 Designer)

Equipment Model describes an organisational structure.



1. Model the following organizational Structure:
The *enterprise* Czech Technical University **CTU** has a *site* in **Prague**. One *area* of this *site*, is the **CIIRC** (Czech Institute of Informatics, Robotics and Cybernetics). The **CIIRC** has a *production line*, the Industry 4.0 **Testbed.** Create the instances of the appropriate classes: Create this organisational hierarchy using **instances** of type *equipment*.

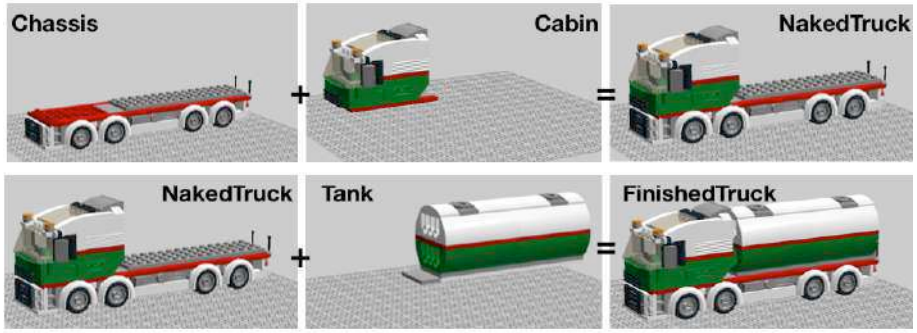
2. The **Testbed** consists of a railway transport system and four robot cells. The transport system is highly complex, therefore we want to sum it up with a single piece of equipment called, **Transportsystem.** In addition to that, you should create the four **Robotcells**.

3. Now fill up these **Robotcells**: Each Robotcell consists of a **Table** (*Unit*) and a **Robot** (*Unit*); **T1** and **R1**, **T2** and **R2**, **T3** and **R3**, and **T10** and **R10**.

4. Now map the three **Agilus** instances to R1, R2, R3 and the **LBR iiwa** instance to R10.

5. Material Model (Generic)

Now, the materials have to be created.



1. Create a MaterialClass called *TruckComponent*. Now the following MaterialDefinitions, which belong to this MaterialClass:

   i) **Chassis**

   ii) **Cabin**

   iii) **NakedTruck**

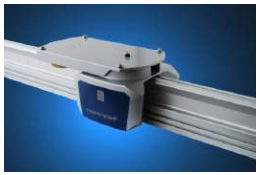
   iv) **Tank**

   v) **FinishedTruck**

2. Now create MaterialLots for the following MaterialDefinitions: **Chassis**, **Cabin** and **Tank**. Each MaterialLot represents a quantity of 1000 pieces.

6. Process Segment (ISA-95 Designer)

Process Segments are logical connectors of resources, a scaffold for real production steps. Create four hierarchically structured Process Segments:



- **Transport:** A *TruckComponent* has to be transported through the transport system via the equipment **Shuttle. —>**

- **Pick and Place:** A robot, sometimes also a Warehouse Worker, picks a *TruckComponent* and places it on the shuttle.
  - **Pick**
  - **Place**

1. Create the Process Segment **Transport**

2. Create the Process Segment **Pick and Place**

3. Now connect the resources to the Process Segments:

Transport
- Material: *TruckComponent*
- Equipment: **Montratec Shuttle**

Pick'n'Place
- Physical Asset Class: KUKA *Agilus*
- Personnel class: *Warehouse Worker*
- Equipment: **Platform**
- Material: *TruckComponent*

4. Create Subprocess Segments within Pick and Place: **Pick**, and **Place.** Connect them with a Material Class Connection (*TruckComponent*)

A

7. Product Overview (Generic)

You are almost finished. Just 2 small tasks and you will win a candy. Browse through the Operations Model and answer the following questions.

There can be multiple answers.

Which Material Bill Item is missing if you compare it with the Material exercise?

☐ Chassis
☐ Cabin
☐ NakedTruck
☐ Tank

Which are valid Operation sequences? (multiple possible)

☐ Place Chassis, Place Cabin, Transport, QA
☐ Place Chassis, Place Tank,Transport, QA
☐ Place Chassis, Place Trank, Place Cabin, QA, Transport

Which is the Starting Operation?

☐ Place Chassis
☐ Place Cabin
☐ Place Tank
☐ Transport

Which is the ending Operation?

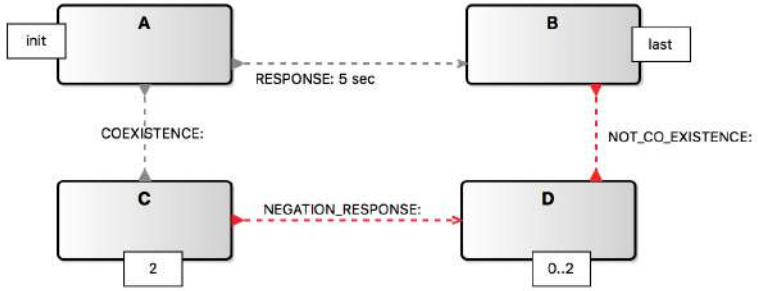
☐ Place Chassis
☐ Place Cabin
☐ Place Tank
☐ Transport
☐ QA

Which Person is involved in the Place Tank?

☐ Laurens Lang
☐ Bernhard Wally
☐ Max Mustermann

8. Operations Dependencies

However, there can be constraints in the production, such as logical or temporal dependencies between production steps. The following diagram shows these dependencies.



There is always just one correct answer.

Which OperationsSegment cannot coexist? □ A and B □ A and C □ B and C □ C and D □ B and D	**Which is the starting OperationsSegment?** □A □ B □ C □ D
Which OperationsSegment can coexist? □ A and B □ A and C □ B and C □ C and D □ B and D	**Which is the ending OperationsSegment?** □ A □ B □ C □ D
Which OperationsSegment cannot follow? □ A -> B □ B -> C □ C -> D □ B -> D	**How often can C be exectued?** □ 1 □ 2 □ 3
Which OperationsSegment can follow after some seconds? □ B after A □ C after B □ D after C □ D after B	**Which is a valid number of executions for D?** □ 1 □ 3 □ 4

Level 3

Second questionaire

1. **Unique Identifier**

Happiness

2. **What was more fun?**
Markieren Sie nur ein Oval.

	1	2	3	4	5	
ISA 95 Designer	◯	◯	◯	◯	◯	Generic Editor

3. **Which was easier to use?**
Markieren Sie nur ein Oval.

	1	2	3	4	5	
ISA 95 Designer	◯	◯	◯	◯	◯	Generic Editor

Engagement

4. **Will you check out the website for updates and upcoming features of ISA-95 at isa95.info?**
Markieren Sie nur ein Oval.

- ◯ Yes
- ◯ No

Adoption & Retention

5. **If you would have to build ISA-95 Models, would you use ISA 95 Designer?**
Markieren Sie nur ein Oval.

- ◯ Yes
- ◯ No

6. **Improvement suggestions**

Table 1

ID	D3	G3	D4	G4	D5	G5	D6	G6	D7	G7			
1		1	1			0	0			1			
2	2			3	1			2	0				
3	0			1	0			3	0				
4		0	1			2	0			0			
5		0	0			1	0			2			
6		2	0			1	1			1			
7		1	0			0	0			0			
8		3	2			3	0			1			
9	0			2	1			2	0				
10		0	0			0	0			1			
11	0			1	0			1	0				
12	0			1	0			0	0				
13	1			0	0			0	0				
13	0,5	1	0,666666666666667	1,33333333333333	0,333333333333333	1	0,142857142857143	1,33333333333333	0	0,857142857142857			

ID	D3	G3	D4	G4	D5	G5	D6	G6	D7	G7		
1	432			1172	1101			991	412			
2	419			969	295			801	206			
3	580			1195	262			634	230			
4	331			1092	630			478	285			
5		537	1028			374	890			666		
6		317	930			356	278			439		
7		431	707			394	468			493		
8		273	713			295	501			872		
9		274	671			277	421			635		
10		539	1424			558	602			575		
11		966	1620			949	1045			960		
12	358			840	202			466	171			
13	225			735	206			498	135			
	390,8333333	476,71	1013,2	1000,5	449,33	457,57	600,71	644,66	239,83333	662,8571428		
	6:31	**7:56**	**16:53**	**16:40**	**7:29**	**7:37**	**10:01**	**10:45**	**3:59**	**11:03**		