Informatics

Towards Model-Driven Vertical Integration

Using IEC 62264 and REA to facilitate real-time data exchange between manufacturing and enterprise systems

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Business Informatics

eingereicht von

Philip Florian Polczer, BSc
Matrikelnummer 01260167

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Mag. Dr. Christian Huemer
Mitwirkung: Dr. Bernhard Wally, MSc

Wien, 23. April 2021

Philip Florian Polczer | Christian Huemer

Informatics

Towards Model-Driven Vertical Integration

Using IEC 62264 and REA to facilitate real-time data exchange between manufacturing and enterprise systems

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Business Informatics

by

Philip Florian Polczer, BSc
Registration Number 01260167

to the Faculty of Informatics

at the TU Wien

Advisor: Ao.Univ.Prof. Mag. Dr. Christian Huemer
Assistance: Dr. Bernhard Wally, MSc

Vienna, 23rd April, 2021

Philip Florian Polczer   Christian Huemer

Erklärung zur Verfassung der Arbeit

Philip Florian Polczer, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 23. April 2021

Philip Florian Polczer

Danksagung

Allen voran möchte ich mich bei Herrn Dr. Bernhard Wally bedanken, der mir mit seinem umfangreichem Fachwissen und unbezahlbarem Feedback stets zur Seite stand. Besonders die regelmäßigen Termine mit ihm haben mich dabei unterstützt, den Abschluss meiner Arbeit neben meiner Berufstätigkeit nicht aus den Augen zu verlieren. Ich möchte mich auch bei meinem Betreuer Herrn Dr. Christian Huemer bedanken, der diese Arbeit ermöglicht hat und mir durch sein wertvolles Feedback dabei geholfen hat, der Arbeit den letzten Schliff zu verleihen.

Ich möchte mich auch bei meiner Familie und besonders meinen Eltern aufrichtig bedanken, da sie es mir ermöglicht haben zu studieren und mich während meiner Studienzeit und darüber hinaus stets unterstützt haben.

Dankbar bin ich auch für die Freundschaften die im Laufe meines Studiums entstanden sind und die mich auch über das Studium hinaus begleiten werden.

Acknowledgements

First and foremost, I would like to thank Dr. Bernhard Wally, who has always supported me with his extensive expertise and invaluable feedback. The regular appointments with him have helped me tremendously to complete my thesis alongside my day job. I would also like to thank my supervisor, Dr. Christian Huemer, who made this work possible and helped me put the finishing touches on the thesis thanks to his valuable feedback.

I would also like to express my sincere gratitude to my family and especially my parents, as they made it possible for me to study and have always supported me during my studies and beyond.

I am also grateful for the friendships that I have made during my studies and that will be part of my life beyond my graduation.

Kurzfassung

Computer-integrated Manufacturing beschäftigt sich mit der Digitalisierung, Automatisierung und Verbindung von Fertigungsprozessen. Spätestens die vierte industrielle Revolution, die das Internet der Dinge in die Fertigung eingeführt hat, hat dazu geführt, dass die an einem Fertigungsprozess beteiligten Systeme immer häufiger und aufwendiger miteinander verbunden werden. Bei dieser Integration werden häufig proprietäre und spezifische Lösungen verwendet. Daher kann diese nicht einfach angepasst werden, wenn neue Systeme eingeführt oder bestehende Systeme angepasst werden. Eine effektive Integration ist jedoch ein wesentlicher Bestandteil von Computer-integrated Manufacturing.

Eine Möglichkeit, dieses Problem zu lösen, ist die Standardisierung der Integration von Systemen, indem man sich auf den Fertigungsprozess anstatt auf die Systeme per se konzentriert. Die Integration sollte unabhängig von den zugrunde liegenden Systemen sein. In dieser Arbeit schlagen wir zwei verschiedene Ansätze vor, um Systeme miteinander zu verbinden, indem wir auf bestehende Standards und Model-driven Engineering zurückgreifen. Insbesondere nutzen wir die Standards IEC 62264 und ISO/IEC 15944-4:2015, um eine Implementierung einer standardisierten modellgetriebenen Integration von Systemen, die auf unterschiedlichen Ebenen eines Fertigungsprozesses arbeiten, zu ermöglichen. Um die beiden Ansätze zu evaluieren, implementieren wir einen beispielhaften Fertigungsprozess und vergleichen die Ansätze miteinander. Wir zeigen, dass eine standardisierte vertikale Integration durch die Abstraktion von Systemen und die Fokussierung auf den Prozess selbst erreicht werden kann, wodurch die Integration unabhängig von den zugrunde liegenden Systemen wird.

Abstract

Computer-integrated manufacturing is about digitizing, automating and connecting manufacturing processes. At the latest, the fourth industrial revolution, introducing the Internet of Things into manufacturing, led to the increasingly widespread adoption of connecting systems involved in manufacturing processes with each other. This integration is often making use of proprietary and customized solutions. Thus, it cannot be easily adapted when new systems are introduced or existing systems are updated. However, effective integration is an essential component of computer-integrated manufacturing.

A way to solve this issue is to standardize the integration of systems by focusing on the manufacturing process instead of the systems. The integration should independent of the underlying systems. In this thesis, we propose two different approaches of connecting systems with each other by making use of existing standards and model-driven engineering. In particular, we make use of the IEC 62264 and ISO/IEC 15944-4:2015 standards in order to provide an implementation of a standardized model-driven integration of systems operating on different levels of a manufacturing process. In order to evaluate the two approaches, we implement an example manufacturing process and compare the approaches with each other. We show that a standardized vertical integration can be achieved by abstracting systems and focusing on the process itself, making the integration independent of the underlying systems.

Contents

CHAPTER 1

Introduction

Mechanization, electricity and information technology were the reason for the first three industrial revolutions. We are now amidst the fourth industrial revolution which is resulting from the introduction of the Internet of Things and Services into the manufacturing environment. Businesses are creating Cyber-Physical Systems (CPS), which incorporate various smart systems which are able to autonomously exchange information [KHHW13]. They are interested in computer-integrated manufacturing, which is a concept for integrating various levels and processes present in those businesses in an automated way [SSKD11].

In computer-integrated manufacturing, which is using computers to control production processes, the so-called automation hierarchy is often encountered. The automation hierarchy, as depicted in Figure 1.1, categorizes systems in (automated) manufacturing into various levels. At the bottom is level 0, where the actual production using hardware and personnel is happening. Level 1 is concerned with sensing and actuating in order to collect and process data. Level 2 is about controlling the production. The capabilities of a manufacturing operation, how products are made, production scheduling and data about the production performance is captured in manufacturing execution systems (MES) on level 3. Enterprise resource planning systems (ERP) are operating on the last level, level 4, which deals with similar concepts as level 3, but in the context of business planning and logistics.

In order to exchange information between the systems involved on the various levels of the automation hierarchy, the systems operating on different levels need to be somehow connected with each other. In the context of this hierarchy, there is horizontal and vertical integration. Horizontal integration is about integrating systems on the same level with each other. For example, you could connect two MES with each other, if the processes they are managing are dependent on each other, maybe in a just-in-time manufacturing environment. Vertical integration, which is the focus of this work, is about connecting systems of different levels with each other. For example, the MES could inform the ERP system about the current state of a production process. The ERP system could use this information to create appropriate business processes, for example offering the products created by the production process for

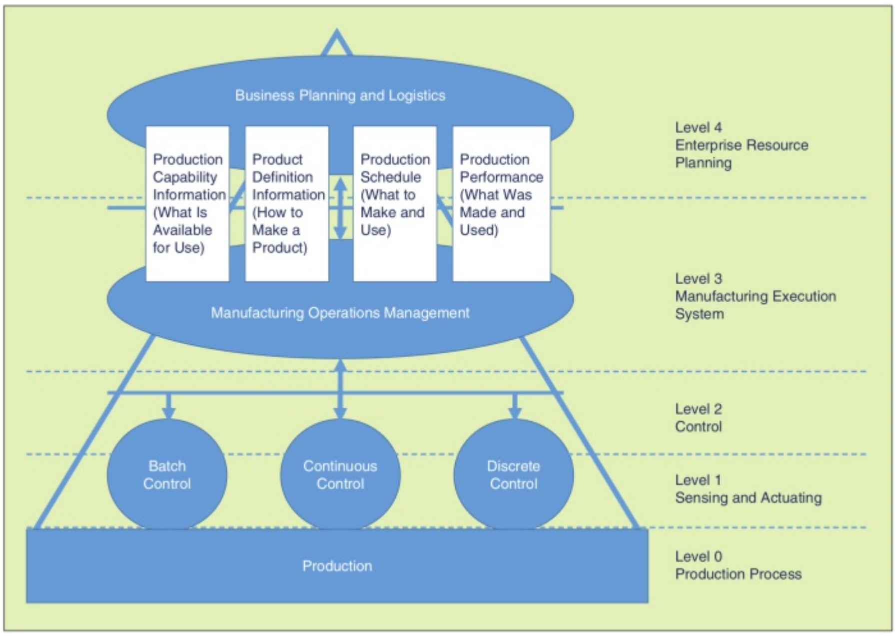

Figure 1.1: The automation hierarchy [JSW20]

sale. Another example would be that a product is ordered, and the information related to this order is first processed by an ERP system. The ERP system notices that the product is not in stock. Therefore, the system automatically requests the manufacturing of this product by communicating with the MES. In practice, this integration is often handled by proprietary or custom solutions, built specifically for certain use cases.

In our work we want to explore the possibility of abstracting processes in order to standardize the (vertical) integration of systems acting on different levels of the automation hierarchy. To set the context, we are assuming that an abstraction of concrete systems already exists and that we want to exchange data between level 3 and level 4. Throughout the thesis, we are concerned with an abstracted layer, on which we want to exchange data between these abstracted systems. We call this process mapping, in which want to define what changes in one system are of interest for another system in a standardized way. To do so, we will take a look at two models. IEC 62264, specifically part two of the IEC 62264 standard, which can be used to model manufacturing execution systems operating on level 3. On the other hand, there is the Resource Events Agents (REA) model, a popular model for representing a business ontology, that can also be used to model business processes found in ERP systems, which are operating on level 4. Figure 1.2 shows the main concepts that we are going to talk about for the remainder of this work.

A concrete system is a real system that is used in the real word to implement processes related to manufacturing. In our work, we are concerned with abstracted versions of these systems, which we group into an abstraction layer. We have made these abstractions using the REA model in order to abstract parts of an ERP system and we have used the IEC 62264 model to

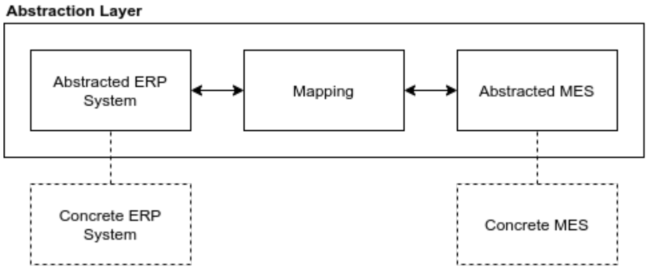

Figure 1.2: Concepts discussed throughout this thesis

abstract parts of a MES. These abstractions are meant to represent the processes that these concrete systems implement, meaning the implementation of the concrete system can change without having to change the abstracted systems if the process interfaces stay the same. The concept of mapping is about connecting the abstracted systems in order to exchange data between them and ultimately between the concrete systems, i.e., it is the abstraction of how data is exchanged between systems. This is represented by a mapping model or the mapping in the code of the application running the abstracted systems which we are going to discuss in Chapter 5.

CHAPTER 2

Scientific Approach

In this chapter, we first state the problem we tackle, second we outline the research questions derived from that problem statement and finally we discuss the methods we used to tackle the problem.

2.1 Problem Statement

While IEC 62264 is great for modeling (automated) manufacturing systems and their integration on the various levels of the automation hierarchy, it is not the most suitable tool to model ERP systems. For such systems, the Resource-Event-Agent (REA) enterprise model may prove more appropriate. However, there is no satisfying linkage between Level 3 and Level 4 available if the systems operating on each level are based on different standards. **There has been a first alignment of the IEC 62264 model and the REA model [WHM17a], but that is still untested, as there is no concrete implementation available.** This is a relevant problem as modern industrial manufacturing relies on the integration of various systems and is already integrating systems. But the various tools used to do so are often not able to be integrated out of the box. A standards-driven approach, e.g. utilizing IEC 62264, the REA model and model-driven engineering, would facilitate an integration of manufacturing systems and ERP systems on an abstract level. A model-driven approach may also speed up development, reduce complexity of systems and increase quality of the resulting software. These improvements also help to reduce costs in the long run [ABB+07]. This approach is of course not limited to systems based on IEC 62264 and REA. IEC 62264 is one of the three main standards used in the Reference Architecture Model Industrie 4.0 (RAMI4.0), which is a DIN specification about industry 4.0 [DIN16]. Therefore, this standard will probably get even more widely adopted in the future. REA will be used as it is a well-known ontology for the (abstract) definition of business models [II07]. Meta models are available for both of these standards, which we will use for our own implementation in this work.

2.2 Research Questions

Based on the problem statement above, we want to answer the following research questions:

RQ 1: To what extent do IEC 62264 and REA overlap? This question is answered by an analysis of an existing alignment of the elements of the two models. The alignment determines the rules for transforming elements of one model into elements of the other model. This analysis will serve as a knowledge base for the further development of the mapping meta model or alternative approaches.

RQ 2: What are appropriate ways to realize a model-driven vertical integration of MES and enterprise systems? Appropriate in this context means that an approach to model-driven vertical integration should be independent of the concrete systems and flexible in regards to changes of the concrete systems. It should not matter for the abstracted systems if the underlying hardware or personnel of a process change. As long as the process stays the same, it should not be necessary to update the model-driven vertical integration.

We are proposing two approaches, one where we are going to use model-driven engineering to do the mapping using an additional meta mapping model and a convention-driven approach. In the convention-driven approach, the conventions of how elements of different models are mapped to each other are expressed in the application code itself. In order to compare them with each other and to support the initial research question, we also answer the following questions:

RQ 2.1: What are advantages and disadvantages of a model-driven mapping?

RQ 2.2: What are advantages and disadvantages of a convention-driven mapping?

RQ 2.3: Which of the two approaches outperforms the other one?

2.3 Research Methods

In information systems research, there are two main paradigms: Natural science and design science. Natural science on the one hand is about understanding reality. Design science on the other hand is concerned with the creation of new artifacts. Therefore, these two sciences also complement each other: Design science creates artifacts which in turn can be observed by natural science [MS95]. As the main goal of this master thesis is to create new artifacts, namely a mapping meta model and accompanying run time systems, we think that a design science approach will be a good fit for our research. Therefore, we make use of the information systems research framework proposed by Hevner et al. [HMPR04].

Figure 2.1 gives an overview of how we applied the framework in the context of our work. The framework is made up of three main parts: the environment, the research itself and the knowledge base. In a later work, Hevner also describes three inherent cycles of design science in more detail: the relevance cycle, the design cycle and the rigor cycle. The relevance cycle is a bridge between the environment and research, the design cycle provides a connection between developing and evaluating artifacts and the rigor cycle is about linking the research with the knowledge base [Hev07]. The environment is about the context in which a problem should be solved. In our case, this environment is computer integrated manufacturing. The business need we are trying to solve with our research is the need for vertical integration based on existing standards. For our research, we have developed and analyzed multiple artifacts. We provide an evaluation of the existing alignment of IEC 62264 and REA. We have also implemented and evaluated two proof of concept applications for the two proposed approaches to model driven vertical integration. The knowledge base provides the basics for the research. For our thesis, this is mainly the existing alignment of IEC 62264 and REA, as well as the standards defining and IEC 62264 and REA models. In the end we also add to the knowledge base by contributing the findings of our work in regards to model-driven vertical integration.

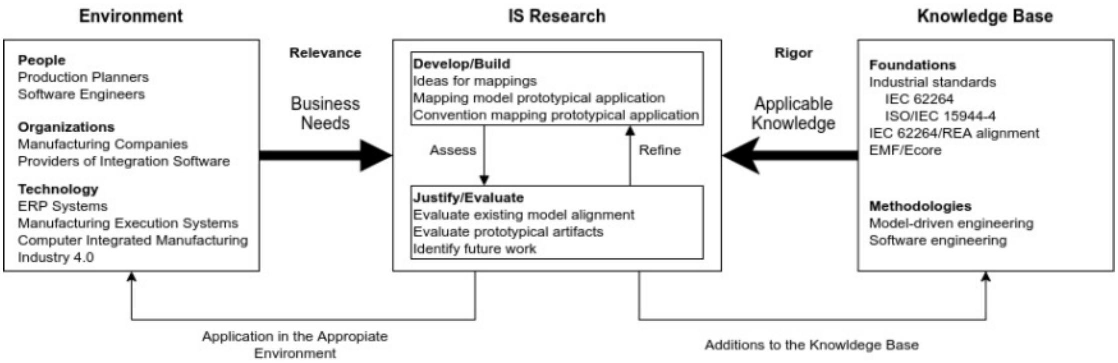

Figure 2.1: Our research in the conxtext of the research framework presented by Hevner et al. [HMPR04].

The information systems research framework provides seven guidelines for conducting design science research. In the following, we shortly discuss these proposed guidelines and how we adhere to them in order to follow a design science approach.

Guideline 1: Design as an Artifact The result of design-science research must be viable artifact. The goal of this thesis is to conduct an analysis of the overlap of IEC 62264 and REA in order to produce proof of concept applications for a model-driven mapping and a convention-driven mapping approach to demonstrate the possibility of a standards and model-driven approach in vertical integration in computer integrated manufacturing. We are providing an evaluation of an existing alignment for IEC 62264 and REA and we implement proof of concept applications for a model-driven approach and a convention-driven approach. Therefore, we create new artifacts.

Guideline 2: Problem Relevance The goal of design-science research is to solve business problems. We want to solve the problem that there is currently no model-driven implementation of data exchange between computer integrated manufacturing systems and ERP systems. However, such a model-driven approach is interesting because it can speed up development, reduce complexity of systems and increases quality of the resulting software. These improvements also help to reduce costs in the long run [ABB+07]. With our work, we are providing a first step towards a model-driven vertical integration, solving the business need for a standardized guideline of how two connect different levels of the automation hierarchy with each other.

Guideline 3: Design Evaluation The created artifact must be evaluated. After having created our artifacts, we evaluate them. We provide an analysis of the existing alignment of IEC 62264 and REA and we compare our implementation of the two proposed approaches to model-driven vertical integration with each other and evaluate the two approaches based on our implementation.

Guideline 4: Research Contributions Design science must contribute artifacts, foundations or methodologies. Through this thesis we contribute a model-driven approach to vertical integration in computer integrated manufacturing. We contribute artifacts, i.e., a model-driven mapping, a convention-driven mapping and applications to execute the mapping and foundations in the form of an in-depth analysis on the alignment of the IEC 62264 and REA meta models.

Guideline 5: Research Rigor State-of-the art model engineering and software engineering tools are used to construct our artifacts. We use the Eclipse Modeling Framework (EMF) to create our example models and generate code based from the existing IEC 62264 and REA meta models. To implement our applications, we use the Kotlin programming language[1], which is a modern, concise and safe programming language which is 100% compatible with the Java Virtual Machine. Therefore it can make use of the generated model code, since the EMF generates Java code.

Guideline 6: Design as a Search Process The design process follows a generator-test cycle. This means that after generating one or more alternatives, these alternatives should be tested against requirements and constraints. This can happen in either a single or multiple cycles and helps to determine whether the generated artifacts provide a satisfying solution to the problem at hand[Sim19]. We follow such an iterative approach. First, we explore existing literature and artifacts, e.g. the existing alignment of IEC 62264 and REA, existing meta models, etc. Based on this exploration, we create our artifacts. We implement and test multiple iterations of a model-driven mapping approach. Based on the experiences gained, we have implement a convention-driven mapping approach. In the end, the results are described, evaluated and possible future work is discussed.

[1] https://kotlinlang.org/

Guideline 7: Communication of Research The results of design science research should not only be presented in a way that it appeals to a technical audience, but also in a way that is is of interest to managers in an organization. A standards-driven approach will be beneficial to the planning of computer integrated manufacturing as well as to its technical implementation. Therefore, we not only describe the benefits at a technological level, but also by outlining the business benefits. It is envisioned to present this work to the scientific community in an appropriate conference. Communicating the results is also a way to get feedback from domain experts on our own work. Such feedback is valuable in the design science cycle.

CHAPTER 3

Foundations

The aim of the thesis is to solve a problem in the context of horizontal and vertical integration in computer-integrated manufacturing by making use of model-driven (software) engineering, and involving the IEC 62264 standard and the REA accounting model. In this chapter, we take a deeper look at each of these topics.

3.1 Horizontal and Vertical Integration

Horizontal and vertical integration both support the intent of delivering an end-to-end solution in computer-integrated manufacturing and Cyber-Physical Systems. In the context of combining production with automation and information technology, horizontal integration is about the integration of smart systems used during various manufacturing stages to exchange materials, energy and information. This exchange can happen both between companies and within a company. Vertical integration in the same context is about the integration of smart systems used in the different hierarchical levels, from the sensors used deep down in the production systems up to the enterprise resource planning systems [KHHW13].

3.2 Model-Driven (Software) Engineering

Model-driven software engineering (MDSE) is about bringing the advantages of modeling into software engineering processes. There are two main concepts: the models themselves and transformations. Brambilla et al. propose a simple formula *Models* + *Transformations* = *Software*. To describe models and transformations, modeling languages are used. A modeling language itself can also be considered as a model, the process of describing a modeling language is also known as meta modeling [BCW17].

3.2.1 Models

According to Selic, a model provides an abstraction of a concrete problem and its solutions and should adhere to five key characteristics [Sel03]:

Abstraction A model should always represent a simplified version of reality.

Under–standability A simple abstraction of reality is not enough, the model should enhance the understand–ability of the real world system it is abstracting.

Accuracy A model should truthfully represent the system it abstracts.

Predictiveness It needs to be possible to use the model to correctly predict a modeled systems behavior and properties.

Inexpensiveness The model should always be more inexpensive to construct than the abstracted system.

An even more generic but similar definition of models is provided earlier by Stachowiak [Sta73]. Models should always be models of something, i.e., abstractions, but they do not need to be associated with the original object they are abstracting. They do not capture all properties and attributes of the object they are representing, but only those relevant to the user of the model.

In computer science however, there are different perceptions of the term "model". Different authors have different definitions of models, if they define models at all [Kas99]. Therefore, models are not explicit to model-driven engineering, since we are always abstracting reality in order to create mental models, but model-driven engineering is a way of explicitly representing the abstractions we make [Oli04].

3.2.2 Model Transformations

Model transformations take one or more models as input and transform them according to transformation rules into one or more output models [SK03]. These transformations, which can happen autonomously, are necessary to merge, align, refactor, refine and translate models. A transformation is possible from model to model (M2M), model to text (M2T) or text to model (T2M) [BCW17].

3.2.3 The OMG four-layer meta modeling stack

The Object Management Group[1] (OMG) provides a four layer meta modeling stack as architecture for meta modeling, known as the MetaObject Facility MOF specification [MOF16]. It consists of the following four levels: M3, which provides meta meta models for defining meta modeling languages for the specification of meta models. Since models on this level are often designed in such a way that they are capable of describing themselves, no additional level is needed on top. An example for such a meta meta model is Ecore[2], the (partial) MOF implementation by the Eclipse foundation. On M2, the modeling language is defined. In

[1] https://www.omg.org
[2] https://wiki.eclipse.org/Ecore

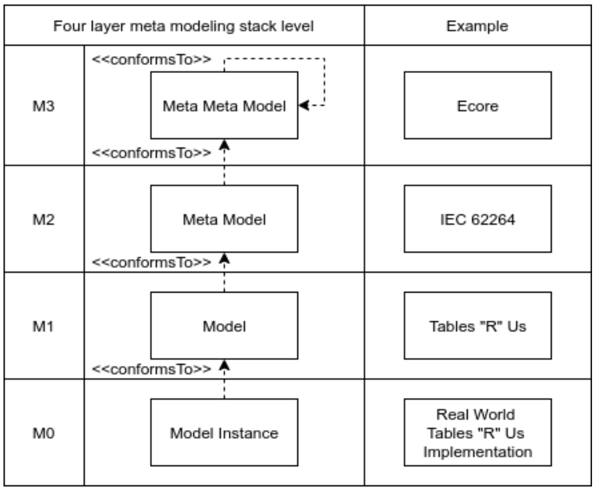

Figure 3.1: An overview of the OMG four-layer meta modeling stack with examples

our case, this would be the IEC 62264 and REA standards. This meta model conforms to the meta meta model defined in M3. M1 instantiates the meta models from M2, i.e., these models conform to the meta models defined there. This could be a IEC 62264 model, describing a production process. M0 contains the concrete instances of models, for example the actual personnel and manufacturing hardware implementing a production process described in an IEC 62264 model. An overview of this architecture is given in Figure 3.1. The *Tables "R" Us* example model describes a process of manufacturing tables. M1 would describe the abstract process of manufacturing a table object, whereas M0 is about the real world implementation of this abstract description, i.e. the actual resources like wooden plates, hardware like saws for cutting table tops and personnel to use the hardware and assemble the tables in the end.

3.2.4 Eclipse Modeling Framework

The Eclipse Modeling Framework[3] (EMF) is a project of the Eclipse foundation to support model driven engineering. It includes the Ecore meta model which allows to create and manipulate other (meta) models. An editor framework is available, which provides the ability of generating editors to create and edit Ecore model instances. EMF also includes code generation capabilities in order to generate Java code ready to use within other applications for manipulating model elements. In addition, additional classes like adapters, which can for example observe a model instance for changes and act on these notifications. It is in widespread use in academia. Another, similar tool, is e.g. JetBrains' Meta Programming System[4].

[3] https://www.eclipse.org/modeling/emf/
[4] https://www.jetbrains.com/mps/

3.3 IEC 62264

IEC 62264 is an international standard by the International Electrotechnical Commission based on the ANSI/ISA-95 specification by the American National Standards Institute (ANSI) and the International Society of Automation (ISA) and is a standard for enterprise control system integration [LMF16]. IEC 62264 consists of six parts, where each part is concerned with a different aspect of enterprise control system integration.

Part 1, **IEC 62264-1**, introduces IEC 62264 and provides the basic models, concepts and terminology necessary to describe interfaces between manufacturing systems and business systems [IEC13a].

Part 2, **IEC 62264-2**, is concerned with providing objects for the communication between manufacturing systems and business systems [IEC13b].

Part 3, **IEC 62264-3**, provides models for activities and data flows which enable enterprise-control system integration and manufacturing operations management [IEC16a].

Part 4, **IEC 62264-4**, extends on part 3 and defines the models and attributes to integrate manufacturing operations management and enterprise-control systems [IEC15].

Part 5, **IEC 62264-5,** builds on part 2 and part 4 in order to enable transactions between systems, more specifically the exchange of information between applications performing business and manufacturing activities [IEC16b].

Part 6, **IEC 62264-6**, defines a model for exchanging transaction messages based on the transaction model of part 5. This model is intended for interoperability between manufacturing operations domain applications and applications in other domains [IEC20].

The IEC 62264 standards aims to provide good practices for manufacturing operations, can be used to improve existing systems and can be applied regardless of the degree of automation. Some benefits are that new product production process can be introduced faster, and that internal and external partners can easier integrate with systems and provide tools for them. By standardizing manufacturing and enterprise operations, costs can be reduced and supply chains can be optimized.

Based on part 1 of the IEC 62264 standard, Figure 3.2 shows the areas of production operations management operations and the four pillars supporting them.

Product definition information is about what must be defined to be able to produce a product. In order to know what resources (personnel, equipment and materials) are available, systems need to provide information about their production capabilities which is captured as production capability information. Capacities can fluctuate over time. Production schedule information and production performance information depict what actual production will be executed (or is planned to be executed, whereas production performance information reports what actual production has been achieved in the end.

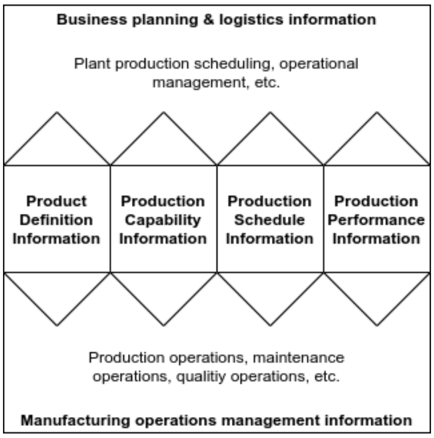

Figure 3.2: Areas of production operations management operations based on [IEC13a]

3.3.1 IEC 62264-2

This section explains part 2 of the IEC 62264 ([IEC13b] in greater detail, since this part of the standard is modeling the exchange of information between Level 4 and Level 3, which is the main concern in the context of this work. We give an overview of the models and objects of the standard and show how they are related to each other. Thereby, we focus on the models and objects related to our work.

Personnel Model

Information about specific personnel, classes of personnel, and their classifications is stored in the personnel model.

Person In contrast to personnel classes, a person is a unique specified individual. This person may have zero or more personnel classes. Persons can have properties, described by *person properties*.

Personnel Class A personnel class classifies groups of persons having similar characteristics and capabilities. For example, persons could be classified as Operator, where Operator would define a class of persons which are capable of operating machines in a manufacturing environment. Properties of a personnel class can be described using *personnel class properties*.

Equipment Model

Information about specific equipment, classes of equipment, its classifications and properties is stored in the equipment model.

Equipment Equipment represents a piece of equipment, which is necessary for manufacturing products. Equipment may have zero or more equipment classes and can be made up of other equipment. Equipment can have properties, described by *equipment properties*.

Equipment Class An equipment class classifies a group of equipment having similar characteristics and capabilities. Properties of an equipment class can be described using *equipment class properties*.

Physical Asset Model

Information about specific physical assets, classes of physical assets, its classifications and properties is stored in the physical asset model. It provides more concrete information about the actual physical equipment used.

Physical Asset Physical assets represent physical pieces of equipment. Physical assets may have zero or more physical asset classes and can be made up of other physical assets. Physical assets can have properties, described by *physical asset properties*.

Physical Asset Class Physical asset classes represent groupings of physical assets having similar characteristics and capabilities. Properties of an physical asset class can be described using *physical asset class properties*.

Equipment Asset Mapping An equipment asset mappings represents the relation between a physical asset and an equipment. It provides information about how the need for an equipment is realized in the physical production environment. The equipment model may for example specify the need for one or more circular saws. In the physical asset model information about the actual physical circular saws which have been purchased in order to use them in the production environment is stored . These are real pieces of equipment, having a serial number uniquely identifying them.

Material Model

The material model defines information about the actual materials, material definitions and classes of material definitions. Materials can be raw materials, finished products, intermediate products or materials that are consumed during the production process.

Material Lot A uniquely identify-able planned or actual material with a specified amount is depicted as material lot, describing its current state. *Material class properties* can be used to specify a material lot's properties. Material lots can be made up of material sub lots.

Material Definition To represent materials with similar name characteristics a material definition can be used.

Material Class Material classes group material definitions and are used to organize materials. *Material class properties* define properties of a material class.

Process Segment Model

The process segment model is a hierarchical model containing process segments. A process segment describes something that occurs during a manufacturing process and is the smallest element of manufacturing activities. It is a collection of the personnel classes, equipment classes, physical asset classes and material classes that are required by the activity. Dependencies of one process segment on another can be defined as well.

Operations Definition Model

The resources required to perform a specific operation are defined in an operations definition. Operations definitions may contain an operations material bill with operations material bill items, which is a collection of all material used in the operation definition. To quantify the segment for a specific operation, operations segments are used. They reference a process segment. An operations segment specifies what personnel, equipment, physical asset, and material is required for an operation.

Operations Schedule Model

To represent the request for an operation which needs to be performed, operations schedules are used. Operations requests are contained in schedules and represent the segments that are required for the scheduled operation. These segments requirements, similar to operation segments, specify what personnel, equipment, physical asset, and material is required for the requested operations.

Operations Performance Model

After a requested operation is performed, it is reported by the operations performance model in the form of operations responses, which are responses to operations requests. These responses contain segment responses corresponding to segment requests. These segment responses contain the actual personnel, equipment, physical asset and material used by a performed manufacturing operation.

Figure 3.3 shows how process segments, operations definitions, operations schedules and operations performances are related to each other. The connections from top to bottom show which elements contain which other elements, for example the process segment model contains process segments, the operations definitions contain operations segments and so on. A connection from right to left shows which elements correspond to each other. For example, an operations response corresponds to an operations request, which in turn corresponds to an operations definition. Resources describe the personnel, equipment, physical asset and material specification, requirements, or actuals used by the various elements.

Units of measurement

Many IEC 62264 objects allow the definition of their unit of measurement. To specify these units in the models created during our work, we make use of the codes for units of measure used in international trade as proposed by UN/CEFACT. More specifically, we use the code **H87** which is defined as "a unit of count defining the number of pieces (piece: a single item, article or exemplar)" for material lots to specify the unit of measure current quantity of the material represented. For personnel specifications we use the code **HEA**, which is defined as

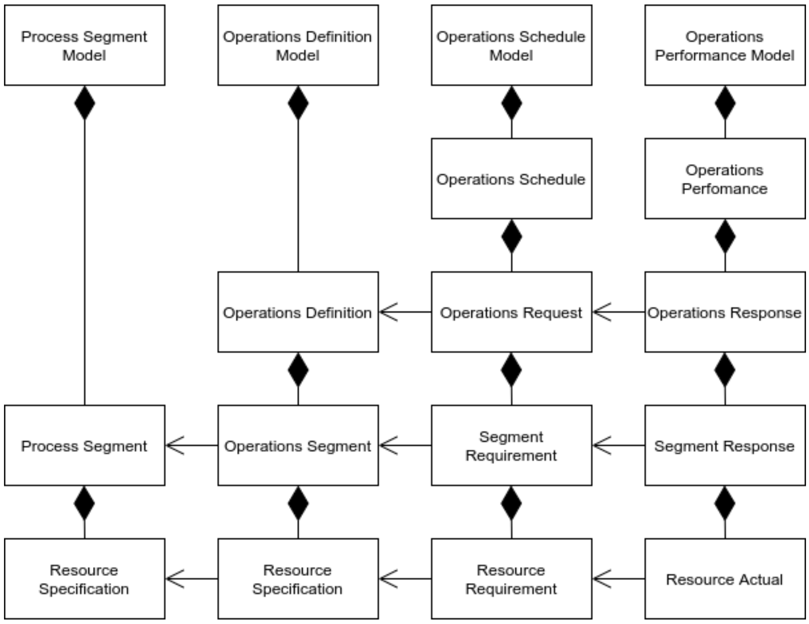

Figure 3.3: An overview of the relation of process segments, operations definitions, operations schedules and operations performances based on [IEC13a]

"a unit of count defining the number of heads (head: a person or animal considered as one of a number)" [UN/05].

3.4 The Resource Event Agent Business Model Language

The Resource-Event-Agent (REA) business model language was proposed by McCarthy in 1982 and is a framework for the conceptual design of an accountability system which describes the flow of resources inside and between companies. It characterizes the relationship of economic resources, economic events and economic agents. An important concept is that events are linked by dualities, meaning that a reduce of a resource on one side means an increase of an resource on the other side [McC82]. For example, if an item is sold for money by business A to business B, business A looses the item, but gains money. Business B on the other hand looses money, but gains the item bought. Figure 3.4 shows this fundamental relationship. Resources are linked with events by stockflows, an event either decreases or increases the stock of a resource. The duality link of events with themselves implies that the increase of a resource leads to the decrease of another and vice versa. Agents control events, and they either gain or loose something from them.

Over time, this framework has been revisited and adapted. In 1997, an object-oriented approach to systems implemented based on the REA model was discussed to increase re-usability and interoperability in more complex systems [GM97]. Later, a commitment entity has been added. A commitment is an important economic concept which describes the agreement of

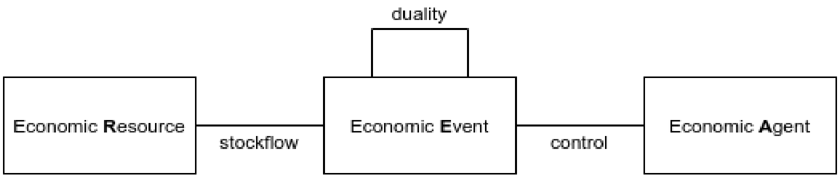

Figure 3.4: The relationship of resources, events and agents in the REA model

two partners about an exchange in the future. REA itself differentiates if a commitment is part of a contract or a schedule [GM00]. One of the latest additions were policy-level definitions, which allow to describe company policies in REA [GM06]. The REA enterprise model also serves as a basis for the ISO/IEC 15944-4:2015 standard, which offers an ontology for accounting and economic business transactions [ISO15]. For most REA elements in the ontological model, there is a type object and an instance object. A type is a concept that applies to multiple instances, i.e. concrete objects [GM00]. For example, there could be a resource type describing a basic table, which is made from a table top and table legs. An instance of this resource type would be a concrete table made from wood which is made from a specific table top and specific table legs. In the following, we will describe the various elements of REA which are relevant to this work based on the work of McCarthy et al. and the ISO/IEC 15944-4:2015 standard.

Attribute Attributes are used to further characterize REA elements. Attributes can be nested within each other.

Resource Economic resources are one of the main elements of the REA accounting model. They are objects that are scarce, have utility and are under control of an enterprise. Resources are something of value. Resources are for example money, raw materials, finished goods etc.

Event Another main element of the REA model, an economic event is an happening that changes the scarcity and control of resources. In accounting theory itself, an economic event leads to a change in the general ledger, e.g. buying raw materials or equipment, or selling manufactured goods.

Transformation Duality Transformation dualities represent the transformation of resources. They specify three types of events: produce events, consume events and use events. Produce events depict the resources that are produced, consume events depict the resources that need to be consumed in order to produce resources, and use events also depict resources that are necessary to produce another resource, but are not consumed in the process.

Transfer Duality A transfer duality represents the exchange of goods: A resource is given (give event) in order to take (take event) another resource. A typical example would be the sale of a finished good for money.

Commitment A commitment is essentially a plan to execute an event in the future. It already includes information about the resources and agents that will be part of the (future) event.

Transformation Reciprocity In contrast to transformation dualities, transformation reciprocities are used to plan the transformation of resources in the future.

Transfer Reciprocity In contrast to transfer dualities, transfer reciprocities are used to plan the exchange of resources in the future.

Schedule A schedule is the agreement to transform resources at a planned point of time in the future by specifying transformation reciprocities.

Contract A contract is the agreement to exchange resources at a planned point of time in the future by specifying transfer reciprocities.

Fulfillment A fulfillment represents the execution of a planned event after a commitment to do so has been made.

Linkage Linkages can be used to represent parent-child relationships of resources.

Stock Flow Stock flows quantify the increase or decrease of resources when an event happens. Stock flows can also be planned as part of a commitment.

Participation A participation represents an agent taking part in an event. Participation can also be planned as part of a commitment.

Agent Economic agents are persons or entities which take part in economic events. They make use of economic resources.

Although the REA model is a well-known theoretical approach for building enterprise accounting systems or enterprise resource planning systems, it is not seeing much use in industrial applications [MHR13]. Studies compare REA and SAP, a popular enterprise resource planning system, and find that there are a lot of similarities in their models, but due to the high need of customized implementations, there are also a lot of anomalies [O'L04][FP13].

Following a model-driven engineering approach, there are also efforts to explore the possibilities of using the REA model in enterprise resource planning systems in order to make customization easier [WMKH15]. REAList is a proposal for a flexible ERP system which focuses on providing for each individual tenant by leveraging a generic data structure based on REA [MMW+14].

CHAPTER 4

Related Work

In this chapter we want to take a look at other existing work and research related to (model-driven) horizontal and vertical driven integration.

The history of horizontal and vertical integration started with the concept of computer-integrated manufacturing (CIM) in the 1970s. With the automation hierarchy it was shown that the exchange of information between different layers in manufacturing is crucial for efficient automation, but it took a long time to get the necessary focus [SSKD11]. In [JSW20] the need for models in automation is discussed. An important point is that up until now, there are a lot of proprietary and customized solutions used, which cannot be reused for other use cases. Standardized concepts may require more initial work, but thanks to modern data processing solutions, models should no longer be considered to be just pretty diagrams.

The importance of integration across layers is recognized by the Reference Architectural Model Industrie 4.0 (RAMI 4.0).

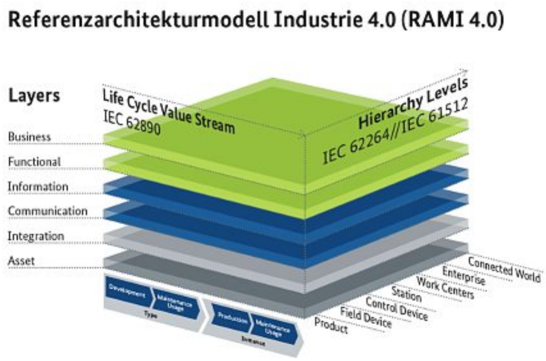

Figure 4.1: The Reference Architectural Model Industrie 4.0 as presented in [HR15]

Figure 4.1 shows the architectural model and its life cycle and value stream axis and its hierarchy levels axis. The hierarchy levels axis is of particular interest, since these levels are the levels of the automation hierarchy adapted from the IEC 62264 standard [DIN16][HR15].

A Brief Look at the State of Industry 4.0

Industry 4.0, which is about the trend of using the internet of things in the manufacturing industry, was introduced in 2011 [XXL18]. These technologies, when applied on all levels of the automation hierarchy, allow the communication of hard- and software in all aspects of manufacturing, therefore also providing a basis for horizontal and vertical integration. In the final report of the Industrie 4.0 Working group, horizontal and vertical integration are defined and discussed in the context of Industry 4.0 [For13]. A study of secondary data discussed the expected impact of Industry 4.0 on the Brazilian Industry in [DBAF18]. There are not only positive but also negative effects, and a lot of companies are still in the early stages of adopting an Industry 4.0 approach. In 2016, there were still (research) gaps between current manufacturing systems and Industry 4.0 requirements [QLG16]. Therefore, a multi-layered framework has been proposed in [QLG16] which assists in understanding the requirements of Industry 4.0 and how they can be achieved.

The readiness of Industry 4.0 in 24 German companies has been evaluated in [BDI+18] in 2018. The study ranked companies using a maturity scoring model according to the size of the enterprise. Companies are ranked from level 0 to 6, where companies on level zero are “outsiders” and companies on level six are “top-performers”. The data was collected by conducting personal interviews or conference calls. The conclusion of the study was that on average, companies are on level 2, representing an intermediate readiness for Industry 4.0.

While Industry 4.0 is still not in full swing, discussions about Industry 5.0 have already started. According to [SB17], the convergence of technologies like the Internet of Things and artificial and emergent intelligence will transform Industry 4.0 to Industry 5.0. There is a shift from the focus of technology to also consider the human factor in an increasingly automated world as well as the impact on society and the environment in general [Nah19].

ISA-95

The International Society of Automation (ISA) is a global non-profit organization dealing with various topics in industrial automation. Their 95th standard, ISA-95, has the objective of simplifying the implementation of enterprise-control systems and reducing the cost, risks and errors involved in their integration and is used across the industry [Sch07]. Throughout our work, we make use of the IEC/ISO 62264 standard, which is the international version of the ISA-95 standard. Therefore, solutions involving ISA-95 are also applicable to solutions that make use of IEC 62264.

Model Alignments

An important aspect to the integration of different models is their alignment and the task of keeping them consistent with each other. These alignments are necessary to determine the feasibility of integrating two different models. On one hand, if it is possible to express the exact same information in each of the models, no integration would be necessary: simply one model could be chosen to represent the system and all integration could be done internally. On the other hand, if there is not a big enough overlap between the models, crucial information required by either system may not be able to be integrated, drastically reducing the utility of the integration or making additional work necessary to increase the overlap.

A precursor of such an alignment of REA to ISA-95 is presented in [MH15a], where transformation rules are provided in order to transform REA models into ISA-95 models. Based on this work, an alignment of REA and ISA-95 has been proposed and evaluated in [WHM17a]. In this work the authors highlight the benefits of a generic alignment, namely that it enables systems provided by different vendors to integrate with much less effort. Instead of defining $2 \times (n \times m)$ mappings between n ERP tools and m MES tools, only $2 \times (n+m)$ mappings are required Integrating ISA-95 elements into REA seems to be a promising approach: in [WHM17c] a proposal is made to use low-level ISA-95 elements in order to model the tasks leading to a production step, where REA elements are too generic to do so.

Model transformations are a way of transforming a source model into a target model by applying rules. These rules could have resulted from an alignment of the source and target meta models. Due to an abundance of (meta) models, a generic approach transformation is of great interest to avoid increasing structural heterogeneities. In [WKK+10b] an approach to resolve heterogeneities is presented: making use of a set of pre-defined mapping operators (MOps), which are abstract from concrete meta model types. Using so-called kernel MOps, basic mappings can be created. To solve more complex mappings, it is possible to assemble the kernel MOps to composite MOps. The work provides a prototypical example and proposes an extensible library of MOps to use. Before being able to resolve heterogeneities, they need to be discovered and classified. Heterogeneities can be classified into two main classes: syntactic and semantic heterogeneities [WKK+10a]. The difference in how different meta models define the same or semantically similar concepts result in syntactic heterogeneities. Semantic heterogeneities arise from the different interpretations of meta models or their instances. [WKK+10a] also evaluates modeling tools, showing a lack of support for dealing with heterogeneity.

Another approach to align different models is to provide a independent linking model, as described in [FWKVH16]. This model allows to define rules which need to be adhered by the model instances linked with each other. Inspired by this idea we have implemented our mapping model approach, which also introduces another model to manage the integration between two models.

B2MML

XML[1], the Extensible Markup Language, is a text-based format that can be used to exchange data in a simple and flexible way. The Business To Manufacturing Markup Language (B2MML) is a complete XML implementation of the ISA-95 standard [WPL17]. B2MML is offered by the

[1] https://www.w3.org/XML/

Manufacturing Enterprise Solutions Association (MESA) and can be used free of charge if MESA is credited[2]. Since XML is a widely adopted standard, using B2MML would also be a promising candiate for modeling vertical integration. In [KNH16] B2MML is used in conjunction with Subject-oriented business process management (S-BPM) to create an integration of level 4 and level 3 of the automation hierarchy. S-BPM focuses on subjects involved in a business process. These subjects (or actors) define individual behaviour which is coordinated by exchanging messages the subjects are able to send and receive [FSS+12].

AutomationML

In the design and engineering of manufacturing plants there is a strong phase separation and there are specialized engineering tools used in each phase. AutomationML (AML) is a neutral data format which is also based on XML that was introduced in 2008 [DLPH]. In order to describe plant topology AML makes use of the IEC 62424:2016 standard, also known as Computer Aided Engineering Exchange (CAEX), which is another neutral format to store hierarchical object information [IEC16c][FD05]. AutomationML is situated on a lower level of the automation hierarchy then the main focus of this work. Nonetheless, it is a promising candidate to consider for future work, since it has also been aligned with ISA-95 by combining AML and B2MML [WHM17b]. Since REA and ISA-95 are aligned as well, this allows for the integration of more than just level 4 and level 3 of the automation hierarchy. A mapping for AutomationML and IEC 62264/B2MML is presented in [Wal94], which is endorsed by the AutomationML technical advisory committee.

HoVer: A Modeling Framework for Horizontal and Vertical Integration

Another model-driven engineering approach to horizontal and vertical integration is presented in [MH15b]. The work builds upon the Resource Event Agent (REA) business ontology defined by the ISO/IEC 15944-4 standard [II07] to do horizontal integration. In order to support vertical integration, they apply concepts of the ISA-95 standard (which is the basis for IEC 62264), to the REA model. More specifically, the rather generic concepts of resources and agents have been extended with the corresponding, but more detailed ISA-95 elements like equipment, physical assets, material and personnel.

Modeling Tools

Considering a potential wider use of modeling in integration, tools other then the Eclipse Modeling Framework (EMF) should be considered. EMF is a rather technical set of tools having dependencies and requiring some setup, which is not ideal for non-technical end-users. The ISA-95 designer is a graphical tool kit which allows end-users with coding skills to model and design automated manufacturing systems [Lan20]. REA-DSL is a domain specific language for the REA model, which can be used by business experts and IT professionals in order to easily create conceptual REA models [MH12].

[2] http://www.mesa.org/en/B2MML.asp

CHAPTER 5

Implementation

In order to evaluate the feasibility of our proposal of connecting ERP systems and manufacturing systems with each other on an abstract layer using model engineering, we provide an example implementation of this idea. We use the REA standard to model the abstraction of a simple ERP system and model the abstraction of a simple manufacturing system using the IEC 62264 standard. Based on these models, we build an application to connect our abstracted systems in order to allow them to exchange data with one another.

As a proof of concept, we build a simple application that simulates the production process based on the IEC 62264 standard. There, the mapping between MES and ERP is done in code only. In a first approach to standardize the concept of a model driven vertical integration, we build a monolithic application which uses a self-made mapping model in order to exchange data between the two systems. In a second approach, we specify the mapping of the model elements in the code of the applications which are running the model instances. We build two small applications which simulate the abstracted ERP system and the abstracted manufacturing execution system respectively. These two applications communicate with each other using Kafka[1] event streams.

In the following sections we describe our implementation in greater detail. First, we lay out the example process on which we base our implementation. Second, we describe the example models which we create to represent an ERP system and a MES. Finally, we outline the implementation of the approaches used to facilitate the communication between the two systems. Specifically we provide an initial proof of concept and the first approach in Section 5.4, and the second approach in Section 5.5.

[1] https://kafka.apache.org/

5.1 Example Process

In order to provide a practical example, we came up with a simplified process of manufacturing a wooden table. This process is depicted in Figure 5.1.

We assume a simple factory which is capable of producing wooden tables. A table is made of a wooden veneered table top and four wooden table legs. The wooden veneered table top is built by veneering a wooden plate. The wooden plate is cut from a bigger plate of wood. The wooden table legs are turned from a block of wood.

The process of manufacturing a table is comprised of a set of task which are:

- Cut a wooden table top from a plate of wood using a circular saw.
- Turn four table legs from a block of wood using a lathe.
- Veneer the table top by gluing veneer onto the table top and compressing it.
- Assemble the table by mounting the table legs to the table top.

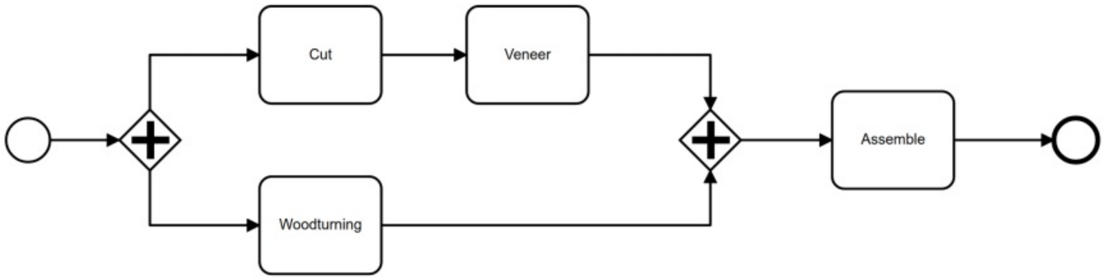

Figure 5.1: The process the example models are based upon

In this process, the act of assembling the table depends on the *Wood-turning* and *Cut* task. The *Veneer* task is dependent on the *Cut* task. *Wood-turning* can happen independently of *Cut* and *Veneer* but needs to happen before *Assemble*. The raw materials (i.e. wooden plates, wooden blocks, and veneer) are assumed to be bought elsewhere and to be in stock. If any of the other intermediate materials are in stock (e.g. wooden table legs) the respective task can be skipped before assembling the table.

The process also involves personnel and machines. Each task before assembly involves a machine to perform the task and an operator that is operating the involved machine. The final assembly does not involve a machine but involves an assembler who is responsible for putting the various pieces together.

Please keep in mind that the process is not an accurate depiction of a real-world process but rather a simplified approximation.

Task	Involved Machinery	Involved Personnel
Wood-turning	Lathe	Operator
Cut	Circular Saw	Operator
Veneer	Compressing Machine	Operator
Assemble	-	Assembler

Table 5.1: Machinery and personnel involved in each task

5.2 Example Models

A cornerstone of our implementation are our example models. In order to create our example models, we use the meta model language implementations for REA and IEC 62264 presented in [WHM17a]. Using these implementations, we build a model representing an ERP system and a model representing a MES. The meta model languages are provided as Eclipse Modeling Framework (EMF) Ecore Modeling Projects. Therefore, we utilize the Eclipse Modeling Framework to create our example models. The example models themselves are based on the process outlined in the preceding section.

For our example models, we make the following assumption: Before being used by an application, the models need to be modeled to a certain extent in order to contain the necessary information to populate and manipulate the models at run time. To do so, we define static and dynamic model elements. Static model elements exist before the models are manipulated by an application. No new static elements are created during run time, but their attributes can be changed. Dynamic model elements are created at run time. Static model elements can be seen as type classes, whereas dynamic model elements are usually instances of said type classes. The differentiation of specific model elements is described in Section 5.2.2 for the MES example model and in Section 5.2.3 for the ERP example model.

In short, we prepare the models to be in the state they are in before any ERP or manufacturing process starts.

Another assumption is that all model elements have an ID attribute which uniquely identifies them. The ID is used to track various elements across the two systems.

5.2.1 Adaptions to the meta modeling languages

We make use of the meta model implementations presented in [WHM17a], but we include some slight adaptions and bug fixes:

IEC 62264

- Fixed generated code in regards to an error due to the use of ZonedDateTime[2].
- Added a parent class (*IEC62264Element*) for all IEC 62264 elements.

- Fixed containment of equipment actuals in segment responses.

REA

- Fixed generated code in regards to an error due to the use of ZonedDateTime[2].
- Added a parent class (*ReaElement*) for all REA elements.
- Defined an ID attribute for all elements.
- Added a type model and an instance model to contain type classes and instance classes respectively.
- Fixed the cardinality of the schedule types reciprocity type containment from *1-to-1* to *1-to-n*.
- Fixed the creation of files with a .rea extension. The build.properties file needed to be updated in order to be able to use the generated EMF editors. We had to add the line *source.. = src-gen/*.
- Defined a quantity attribute for resources.
- Defined attributes for start and end times of events.

In addition, we converted the projects contaning the meta model implementations for REA and IEC 62264 into Maven[3] projects in order to be able to include them in our implementation projects using dependency management.

5.2.2 The MES example model

The MES example model is build using a meta-modeling language based on the IEC 62264 standard. It is based on the process outlined in the preceding section.

Static model elements

Everything but the operations schedule model and the operations performance model are static model elements. We do not use process segment capability model, hierarchy scope, resource relationship model, work definition model, and workflow definition model elements.

Personnel Model

We model personnel classes and persons as depicted in Table 5.2. We use this model to depict the personnel working in our manufacturing environment. Operators are used in tasks involving equipment, i.e. *Cut*, *Veneer*, and *Woodturning*. Assemblers are used in the *Assemble* task. In our example we have one instance of each class: Alice is an instance of the assembler class, Bob is an instance of the operator class.

[2]https://docs.oracle.com/javase/8/docs/api/java/time/ZonedDateTime.html
[3]https://maven.apache.org/

Personnel Class	Person
Operator	Alice
Assembler	Bob

Table 5.2: Two classes and two instances make up our personnel model.

Equipment Model

We provide models for equipment classes and equipment. We use these classes to depict the equipment necessary in our manufacturing environment. We need two circular saws to cut wood, two lathes to turn wood blocks into table legs and two compressing machines that support the veneering of table tops.

Equipment Class	Equipment
CircularSaw	CircularSaw-1 CircularSaw-2
Lathe	Lathe-1 Lathe-2
CompressingMachine	CompressingMachine-1 CompressingMachine-2

Table 5.3: The equipment necessary to manufacture a table when applying our process.

Physical Asset Model

We have modeled physical asset classes, physical assets and equipment asset mappings. We use this model to depict the actual equipment used in our manufacturing environment. Each physical asset is mapped to an equipment to show the actual implementation of the specific equipment need. This mapping may be valid only for a certain period of time and allows to model the possibility of exchanging one equipment with another fulfilling the same functionality.

Physical Asset Class	Physical Asset	Mapped to Equipment
GTS-10-XC	GTS-10-XC-1	CircularSaw-1
	GTS-10-XC-2	CircularSaw-2
VeneerMaster-3000	VeneerMaster-3000-1	CompressingMachine-1
	VeneerMaster-3000-2	CompressingMachine-2
WoodTurner-3000	WoodTurner-3000-1	Lathe-1
	WoodTurner-3000-2	Lathe-2

Table 5.4: The definition of the actual hardware being used in our process.

Material Model

We model material classes, material definitions and material lots. We use this model to depict the materials necessary to produce our table. For our example, we produce a wooden veneered table, made from birch wood and ashen veneer. Material classes define the parts of an table is made out of in the context of our production line. Material definitions give more information about the material our table are made out of. Material lots are the actual material used in order to assemble a table.

Material Class	Material Definition	Material Lot
WoodenBlock	BirchBlock	BirchBlock-1
WoodenPlate	BirchPlate	BirchPlate-1
TableTop	BirchTableTop	BirchTableTop-1
Veneer	AshVeneer	AshVeneer-1
VeneeredTableTop	AshVeneeredTableTop	AshVeneeredTableTop-1
TableLeg	BirchTableLeg	BirchTableLeg-1
Table	BirchTable	BirchTable-1

Table 5.5: The materials processed and produced while manufacturing a table.

Process Segment Model

We model process segments, personnel segment specifications, equipment segment specifications, and material segment specifications. We use this model to depict the parts involved in the process of the production of our table. Each process segment represents a task of our example process. The various segment specifications specify which personnel classes, equipment classes and material classes are required for each task. In addition to a textual description, we provide a visual presentation of process segments similar to the graphical syntax of process segments in [LWH+20]. In our visual representation, a grey rectangle represents the process segment itself. Green rectangles show the material classes used as input and output for the process segment, as defined in material segment specifications. Specified inputs are depicted using an arrow pointing at the process segment, specified outputs are depicted as an arrow pointing from the process segment to the produced output. A blue rectangle is used to show the involved personnel classes defined by personnel segment specifications. Finally, a yellow rectangle represents the Equipment Class needed, which is modeled in the Equipment Segment Specifications.

Process Segment Woodturning This segment represents the *Woodturning* task as depicted in Figure 5.2. It requires an operator, uses a lathe, consumes one wooden block, and produces four table legs.

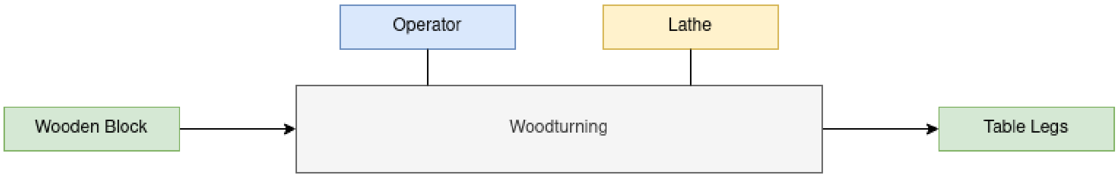

Figure 5.2: Process segment *Woodturning* and its required resource specifications.

Process Segment Cut This segment represents the Cut task as depicted in Figure 5.3. It requires an operator, uses a circular saw, consumes one wooden plate, and produces a table top.

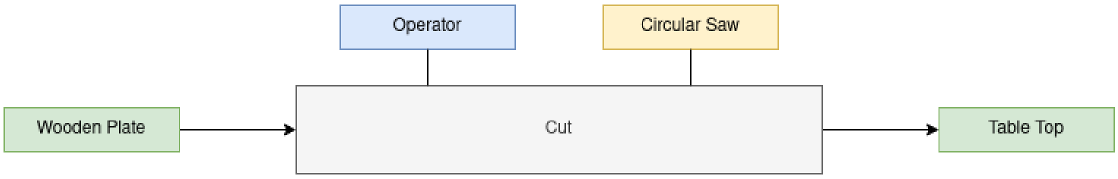

Figure 5.3: Process segment *Cut* and its required resource specifications.

Process Segment Veneer This segment represents the Veneer task as depicted in Figure 5.4. It requires an operator, uses a compressing machine, consumes one table top and one veneer, and produces a veneered table top.

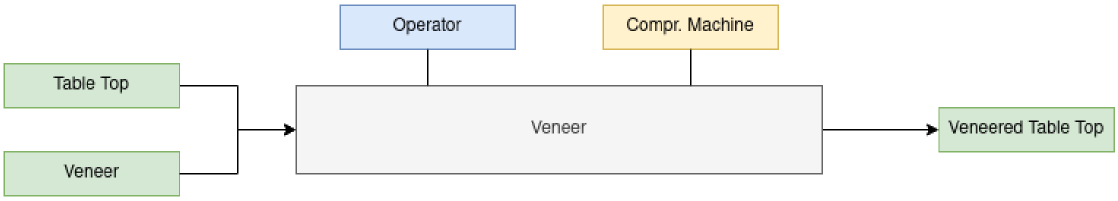

Figure 5.4: Process segment *Veneer* and its required resource specifications.

Process Segment Assemble This segment represents the Assemble task as depicted in Figure 5.5. It requires an assembler, consumes one veneered table top and four table legs, and produces a table.

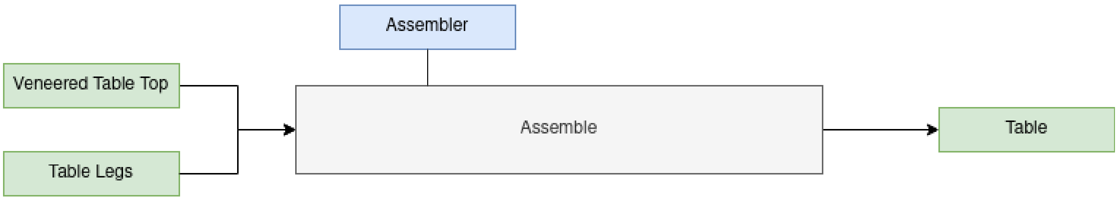

Figure 5.5: Process segment *Assemble* and its required resource specifications.

Operations Definition Model

We model operations definitions, operations segments, personnel specifications, equipment specifications, material specifications, and operations material bills. We use this model to define the whole operation of producing a table. Similar to the process segment model it defines what steps and resources are necessary in order to create a table.

Operations Segment Woodturning This segment represents the Woodturning task. It requires an operator, uses a lathe, consumes one birch block, and produces four birch table legs.

Operations Segment Cut This segment defines the Cut task. It requires an operator, uses a circular saw, consumes one birch plate, and produces a birch table top.

Operations Segment Veneer This segment defines the Veneer task. It requires an operator, uses a compressing machine, consumes one birch table top and one ash veneer, and produces an ash veneered table top.

Operations Segment Assemble This segment defines the Assemble task. It requires an assembler, consumes one ash veneered table top and four birch table legs, and produces a birch table.

Operations Definition ProduceTable This definition groups the various operations segments involved in the production of a table together. It contains the operations material bill BirchTable describing the parts a birch table is made of.

Operations Material Bill BirchTable In general, a Bill of Material is a structure used to describe the component structure of a product [OST97]. In our case, we make use of the material bill element provided by IEC 62264. Our material bill contains multiple material bill items declaring the various parts necessary to create a birch table. It is possible to nest these items, in order to declare which items are assembled from other items. This information is used when scheduling a process to produce a table. In Figure 5.6 you can see that for example, a birch table is assembled from birch table legs and an ash veneered table top.

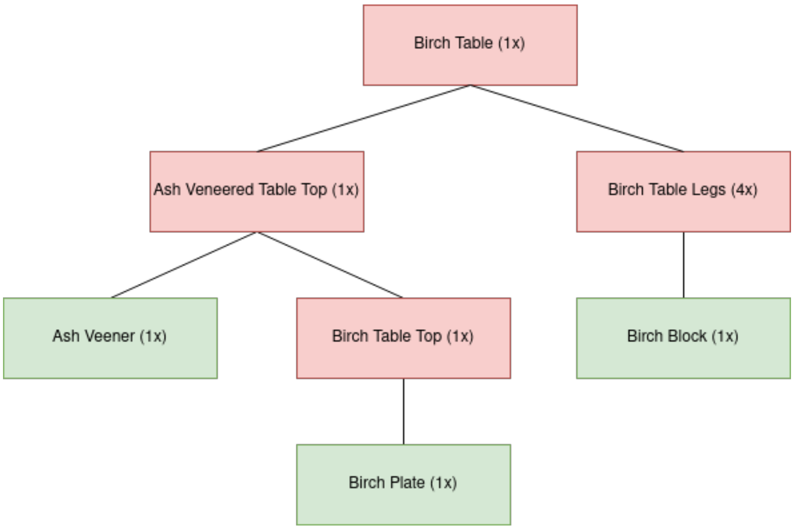

Figure 5.6: The bill of material of a birch table.

Figure 5.6 shows the operation material bill items we have defined. Items in green are raw materials that are sourced externally and are assumed to be in stock. Items in red are resources that we are able to manufacture during the process if they are not in stock. The items reference the material specifications of operation segments. Each item also defines how many instances of it are necessary to create their parent item. Using these two pieces of information, the system decides at run time which operation segments need to be actually scheduled in order to produce all intermediate materials required to produce a table. For example, if four table legs are in stock, the wood turning operation segment is not considered when scheduling the production of a table.

Dynamic model elements

Dynamic model elements are all elements belonging to the operations schedule model and the operations performance model. They are created based on the information received from events published by the ERP system and the information provided from the static model elements at run time. The creation happens in a timely manner and is determined by our implementation of the run-time system simulating our ERP and MES systems. A more detailed description of how these elements are created can be found in Section 5.4 and Section 5.5.

Operations Schedule Model

New operation schedules are created in the MES model when an (REA) schedule event is published by the ERP system and consumed by the MES.

Operations Performance Model

Elements in the operations performance model are created based on the elements contained in the operations schedule model when the scheduled production is executed. In our case this happens when we simulate a production process by executing the IEC 62264 model. The simulation of the IEC 62264 model also includes the execution of the Operations Performances which changes attributes in the static and dynamic parts of the model. Details about the simulation can be found in Section 5.3.

5.2.3 The ERP example model

The ERP example model has been built using the provided meta-modeling language for REA, which is based on the ISO/IEC 15944-4:2015 standard [ISO15]. Using the alignment of REA and IEC 62264 models described in [WHM17a], the model has been largely derived from our IEC 62264 model of the manufacturing execution system.

In this section, we make use of diagrams to show examples of how IEC 62264 elements are mapped to REA elements. The diagrams are inspired by UML Object diagrams of the UML standard [OMG17], but follow a customized syntax and semantics.

- We use UML's InstanceSpecification to depict our model elements including their name and their classification. Attributes of elements may be depicted as well.
- A simple (straight) line shows a relation between elements. This relation may be explained in more detail in the textual description surrounding the diagram figure.
- A line with an arrow at each end shows that the element on the one end of the line is mapped to the element on the other end of the line and vice versa. You can think of each diagram presenting two Object diagrams, one for REA elements and one for IEC 62264 elements. The lines with arrows on each end show which elements of the two diagrams are related to each other.

Static model elements

Static model elements are all elements belonging to the type model. In addition, we define resources, agents and linkages in the instance model as static model elements.

In the REA meta model implementation we are using, type elements are not generalizations of instance elements on a syntactic level. They are separate elements and are only associated with each other using the "Types" property of an instance element. We use type elements to represent a generalization of various instance elements in a semantic way in our model. Type elements can be generalized by other type elements as well. We will provide an example of this when explaining the first static REA model element.

Resource Types and Resources

In REA, there are no separate classes to represent equipment, physical assets and materials. These concepts can all be depicted as resource types (on the type level) and resources (on the instance level). In Figure 5.7 the mapping of the final table product serves as an example. On the type level, material class Table and material definition BirchTable each correspond to a resource type of the same name. On the instance level, a material lot BirchTable-1 corresponds to a resource of the same name. We omit the id fields in later diagrams if their values are not different from the name attribute of the element. The BirchTable-1 resource is a concrete implementation of a generalized table made of birch that our manufacturing process is capable of producing. This general birch table is further generalized by the Table resource type. Using this generalization, we can easily model different types of tables, similar to the semantic generalization offered by IEC 62264's various material model elements.

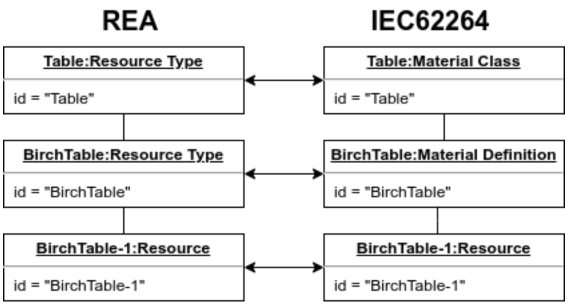

Figure 5.7: Mapping of the REA and IEC 62264 elements representing a table.

The attribute representing the quantity of material lots and resources is not depicted. In IEC66224 quantity is an inherent property of a material lot. In the original REA model, quantity needs to be modeled as an attribute element which is contained in its corresponding resource. We adapted the REA meta model in order to make quantity an inherent property of the Resource element. This quantity attribute is updated when the quantity of the mapped element changes. For example, the quantity of the resource element BirchTable-1 in the REA model is increased by one if the quantity of the material lot BirchTable-1 has been increased by one in the IEC 62264 model.

The REA model does not offer a concept similar to equipment (classes) and physical asset(s) (classes). Therefore, these elements are all mapped to resource types and resources accordingly. In Figure 5.8 the mapping of a circular saw used in the production of a table is depicted as an example. On the type level, equipment class CircularSaw and physical asset class GTS-10-XC each correspond to a resource type of the same name. On the instance level, equipment CircularSaw-1 and physical asset GTS-10-XC-1 each correspond to a resource of the same name.

Since there is no concept of Equipment and Physical Assets, we have not recreated IEC 62264's equipment asset mappings in our ERP model. This can be observed in Figure 5.8: There is an Equipment Asset Mapping GTS-10-XC-1-implements-CircularSaw-1 specifying that GTS-10-XC-1 is fulfilling the equipment requirement proposed by CircularSaw-1. No element containing this information exists on the REA side. If there is a need of representing this relation in the ERP system model as well, one could consider to make use of REA's linkage element.

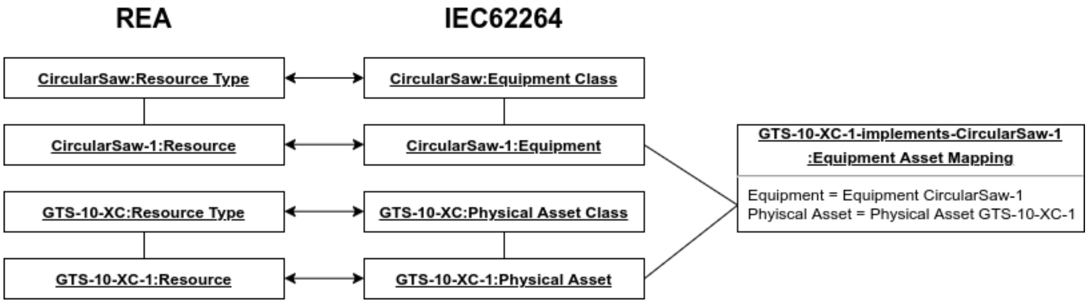

Figure 5.8: Mapping of the REA and IEC 62264 elements representing a circular saw.

An additional resource and resource type has been defined in our ERP model: Money. This resource is a simplified way of representing an increase in capital in our ERP system when we simulate the sale of a produced table through horizontal integration.

Agent Types and Agents

Personnel classes can be modeled as agent types in REA, whereas (IEC 62264) persons can be modeled as agents. Personnel class Assembler and personnel class Operator are created as agent type Assembler and agent type Operator in our ERP model. For example, Figure 5.9 shows how an operator named Alice, which is represented as person in our IEC 62264 model, is created and mapped from our IEC 62264 model to our ERP model.

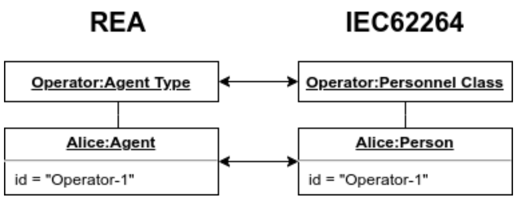

Figure 5.9: Mapping of elements representing a person operating machines.

Person Bob and person Alice are created as an agent element named Bob and an agent element named Alice in our ERP model. Their agent type resembles their respective personnel class assigned to them in the IEC 62264 model.

Linkage Types and Linkages

The “Assembled from Classes” relation of material classes, the “Assembled from Definitions” relation of material definitions, and the “Assembled from Lots” relation of material lots, which represent the concept of one element being made up of other elements, are modeled as linkage type and linkages. Figure 5.10 shows an example of how we represent this relations in our REA model. In the IEC 62264 model, we define that a table is made up of table legs and a veneered table top. This is represented by referencing the material classes for table legs and veneered table tops in the “Assembled from Classes” property. To model this relation in REA, we create two Linkage Types. The “Parent Type” property refers to the resource type mapped to the material class defining the “Assembled from Classes” relation. The “Child Type” property refers to a resource type mapped to an element from the “Assembled from Classes” property. The name of the linkage type provides additional information about the nature of the relation.

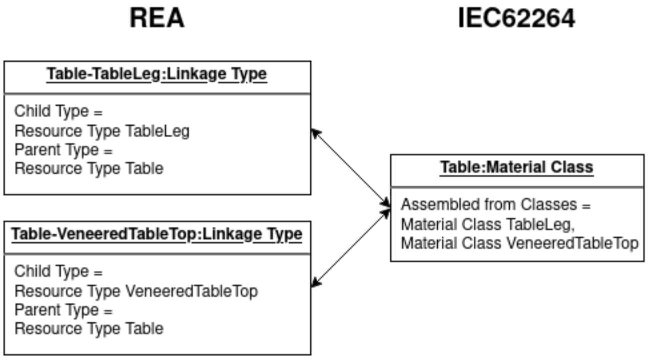

Figure 5.10: Example mapping of an IEC 62264 assembled from relation to REA.

Other “assembled from” relations are modeled in a similar fashion. “Assembled from Lots” relations of material lots are mapped to linkages referencing resources instead of linkage types, but follow the same mapping concept.

Transformation Duality Types and Event Types

Transformation Duality Types are aligned with Process Segments, but Event Types are not aligned with any IEC 62264 element. The Event Types of our example model are based on the Segment Specifications of Process Segments, since Process Segments are aligned with Transformation Duality Types. In Figure 5.11 and example is shown. The Process Segment Cut is modeled as Transformation Duality Type Cut. There is no mapping for an Event Type, but the Personnel Segment Specification OperateSaw, which is contained in the Process Segment, can be mapped to a Participation Type named OperateSaw. We are also able to keep the information of the Personnel Classes property by making use of the Agent Type property of the Participation Type. Since Participation Types need to be contained in Event Types, we need to create an Event Type as well. This Event Type is referenced by the the initial Transformation Duality Type we created in the beginning, completing the mapping.

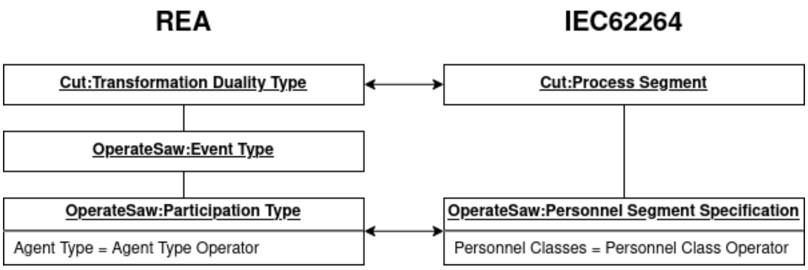

Figure 5.11: Partial example mapping of a Process Segment to a Transformation Duality Type and its Event Types.

However, Figure 5.11 does not show the complete mapping of the Cut Process Segment. The segments also contain Equipment Segment Specifications and Material Segment Specifications, which can be mapped in a similar way. The only difference is that these specifications are mapped to Stockflow Types instead of Participation Types. The Stockflow Types are contained in Event Types as well, which are then referenced by the initially created Transformation Duality Type.

Transformation Reciprocity Types and Commitment Types

Transformation Reciprocity Types are aligned with Process Segments, but Commitment Types are not aligned with any IEC 62264 element. The Commitment Types of our example model are based on the Segment Specifications of Process Segments, since Process Segments are aligned with Transformation Reciprocity Types. The mapping process is similar to the process of mapping Transformation Duality Types and Event Types. In Figure 5.12 we choose to map a Material Segment Specification in order to complement the example shown above.

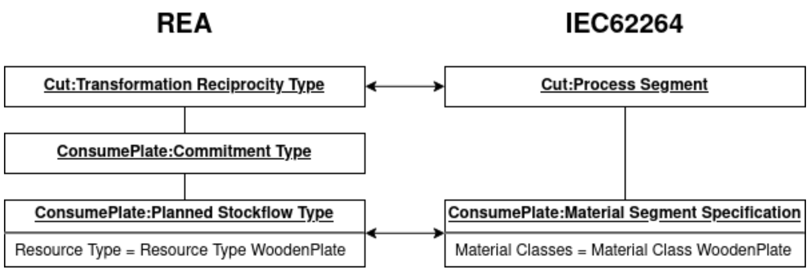

Figure 5.12: Partial example mapping of a Process Segment to a Transformation Reciprocity Type and its Commitment Types.

These mappings follow the same logic as the Transformation Duality Type mappings, but Transformation Reciprocity Types, Commitment Types and Plannend Stockflow/Participation Types are used instead of Transformation Duality Type, Event Types and Stockflow/Participation Types.

Schedule Types

Schedule Types are aligned with Operations Definitions. Operations Definitions are referencing Operation Segments, which are not aligned with any REA element. We can make use of the Process Segment defined in each Operations Segment in order to reference the correct Transformation Reciprocity Types with a Schedule Type created based on an Operations Definition. For example, in our IEC 62264 model we have just one Operations Definition ProduceTable. This definition references multiple Operations Segments, one of which is Operations Segment Cut. This segment has a reference to the Process Segment Cut, which we used to create Transformation Reciprocity Types earlier. Thus, we can create a Schedule Type ProduceTable and easily reference the Transformation Reciprocity Types it requires.

Fulfillment Types

Fulfillment types are not aligned with any IEC 62264 element. We have modeled them based on our event types and commitment types. For example, the TableProduced fulfillment type states that the commitment type ProduceTable is fulfilled by the event type TableProduced.

Dynamic model elements

Dynamic model elements are all elements that are contained in the Instance Model, with the exception of resources, agents and linkages, which are created before the system runs and are therefore static model elements. They are created at run time based on the information received from events published by the MES system and the information provided in the type model of the ERP model. Therefore, the details of their creation depend on the implementation of our run-time system. In this section, we just give an overview about the various dynamic model elements. A more detailed description of how these elements are created can be found in Section 5.4 and Section 5.5.

Contracts

A contract is created when a table is ordered. Based on its contract type, its transfer reciprocities and their commitments are created.

Schedules

A schedule is also created when a table has been ordered. Based on its schedule type, its transformation reciprocities and their commitments are created. The creation of a schedule triggers an event in order to inform the MES.

Transfer Reciprocities

Transfer reciprocities are created when a contract element is created, because they represent the agreement that a table will be exchanged for money in the future based on the contract element.

Transfer Dualities

Transfer dualities are created when a contract element is created and a table is in stock to be sold. If no table is in stock, a new table will be planned to be manufactured by creating a corresponding schedule element.

Transformation Reciprocities

Transformation reciprocities are created when segment requirements are created in the IEC 62264 model of the MES, because the MES is responsible for scheduling the necessary production steps. Transformation reciprocities are the container for the commitments, which represent the planned events that need to be executed in order to manufacture a table.

Transformation Dualities

Transformation dualities are created when any part of the process of creating a table is performed. For example, a transformation duality "Assemble" is created when the MES informs the ERP system that a table has been assembled.

Commitments

Commitments are created when reciprocities are created. They are based on the commitment types defined in the reciprocity's type.

Events

Events are created when Dualities are created. They are based on the Event Type's defined in the Duality's Type class.

Fulfillments

Fulfillments are created when events are created. They try to map the event that has triggered their creation with a matching commitment, based on the ID of the event.

5.3 Simulating the Production Process

In order to simulate the process of production, we chose to leverage the concepts provided by IEC 62264 and its meta model. To do so, we include a simple simulator in our implementations in order to simulate the process of producing a table by creating and updating various IEC 62264 model elements. The simulator evolves and changes alongside the iterations of our implementations. Both our implementation approaches make use of similar simulation logic, although the first approach described in Section 5.4 uses a less sophisticated variant. The simulation is split into three parts: creating operations schedules, creating operations performances and then performing the production process.

Creating operations schedules Operations schedules are created based on the operations definitions of the loaded example model. For each operations definition, an operations schedule is created in the following way: First, an operations schedule element is created in the operations schedule model. Then, an operations request is created and added to the operations schedule. Based on the operations definition, the request's segment requirements are created: For each operations segment, a corresponding segment requirement is created. The equipment, personnel and material requirements of the segment requirement are created based on the equipment, personnel and material specifications of the operations segment. **Note:** This step is skipped in the implementation described in Section 5.5 as the operations schedules in that approach are created when a schedule is created in the REA model.

Initializing operations performances Operations performances are created based on the created operations schedules from the previous step. For each operations schedule having the request state FORECAST, the schedule's request state is set to RELEASED and an operations performance is created and added to the operations performance model. Then, an operations response is created for each operations request contained in the schedule. Next, segment responses are created for each segment requirement of the operations request. Finally, the equipment, personnel and material actuals of the segment responses are created based on the equipment, personnel and material requirements of the operations request.

Performing the production process In order to simulate that a production process has happened, the application simply iterates over the created operations performances and updates various properties in the static and dynamic parts of the model. The start time and end time properties of the performances and their contained elements are set accordingly. The quantities of Material Lots are increased (when the material actuals "use" property is MATERIAL_PRODUCED) and reduced (when the material actuals "use" property is MATERIAL_CONSUMED) by the quantities defined in the material actuals of the segment responses. In the end, the performance state of the executed performances is set to COMPLETED in order to ignore them in the next execution cycle of the simulation.

5.4 Approach 1: Mapping Model and Monolithic Application

Both of our approaches use the Eclipse Modeling Framework in order to access model information. We also make use of the code generated by the framework in order to manipulate our models and model elements.

5.4.1 Conceptual Overview

In Figure 5.13 a conceptual overview of the idea for the first implementation of our model-driven vertical integration application is shown. Diagram elements shaded in grey are not implemented end to end because we decided to switch to another approach during the development of the first approach.

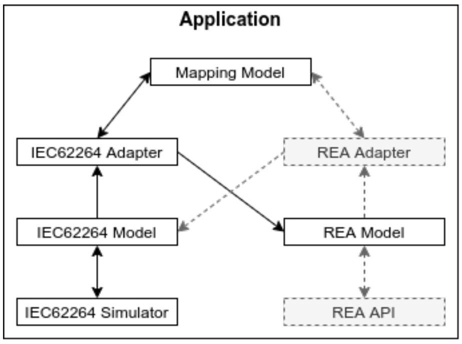

Figure 5.13: Overview of the initially implemented application

The general idea of this concept is that the models get changed by external events. In the real world, the change is triggered when production systems report data. For ERP systems, changes would be triggered when the system processes data because of business operations. These changes are simulated by a simulator on the MES side and an API on the ERP side. The simulator simulates a production process based on the IEC 62264 meta modeling language, the ERP API offers endpoints that are called in order to simulate the event of a customer ordering a product by making use of the REA meta modeling language. These changes notify an adapter listening for changes in the meta model instances. The adapter consults a mapping model, which defines what changes need to be communicated from the MES to the ERP system and vice versa and updates the corresponding meta model instance. Our first implementation is a single monolithic application, written in Java. Accessing and manipulating meta model instances is done via the code generated by the Eclipse Modeling Framework and other libraries included in the framework.

5.4.2 Proof of Concept

For a first feasibility check of our idea, we create a simple Java application. This application does the following tasks:

1. Load the REA and IEC 62264 example models from disk
2. Start the simulation of the IEC 62264 model
3. Listen to changes in the IEC 62264 model and propagate them to the REA model
4. Save the updated models to disk in new files (i.e. different from the ones the models are loaded from)

Loading and saving the models is done using code provided by the Eclipse Modeling Framework. For simulating the IEC 62264 model, the logic described in Section 5.3 is used.

The Eclipse Modeling Framework offers a *EContentAdapter* class which can be extended in order to listen to changes in a model. We create an adapter to get notified about changes in our IEC 62264 example model. For the proof of concept, we implement a simple mapping in code: When the quantity property of a material lot changes, we update the corresponding resource (having the same ID as the material lot) in our REA example model. After the first cycle of the simulation, the updated models are saved to disk and the application terminates. If the (updated) example models are opened in a model editor, the changes made by our application can be observed.

5.4.3 Using a Mapping Model

In our first attempt to create a system for model-driven vertical integration, we utilize model engineering in all aspects. Therefore, we start out to create a mapping model, which should contain the (meta) information necessary to transfer information between two systems. The mapping model should be used instead of the hard coded mapping of the proof of concept. Using the Eclipse Modeling Framework, we create multiple iterations of a mapping meta model language. For all iterations, the REA and IEC 62264 meta modeling languages are imported into our mapping meta model language in order to be able to re-use the elements defined by these languages.

Iteration 1 In this iteration, we only define that an REA element needs to be mapped to a IEC 62264 element, without any further restrictions. This allows for a very generic mapping model, but a lot of the actual mapping still needs to be done in the code. For example, resource BirchTable-1 is mapped to material lot BirchTable-1. When the application code detects a change in material lot BirchTable-1 (e.g. quantity property gets updated), the application looks in the mapping model and determines that resource BirchTable-1 needs to be updated as well. Issues with this approach are:

- The specifics of what needs to be updated are still integrated into the code itself, i.e. the code defines that when the quantity property of a material lot changes, the quantity property of the corresponding resource needs to be updated as well.
- This iteration only allows to map already existing elements, i.e. it is not possible to define that elements need to be created in one model if certain elements in the other model are created.

Iteration 2 In this iteration, we define the mapping model based on the meta model alignment used throughout this thesis. Using this model, it is only possible to map these elements to each other that are aligned with each other. Using this approach, less mapping in code is needed. This approach still allows to map only existing (static) elements but was lacking mapping capabilities for dynamic model elements.

Iteration 3 To address shortcomings of the other iterations, this iteration has is on the definition of static and dynamic model elements in our example models. We introduce this differentiation in our mapping model as well. In this version, it is possible to define generic static mappings. These static mappings are very simple: Per mapping, you need to reference one IEC 62264 element and one REA element. The application code determines what should happen if elements included in the mapping change. The dynamic part is concerned with the creation of new elements and is more complex: It allows to define a source and a target element. The source element is used to define a pattern of an element which can be created in an REA or IEC 62264 model instance. The target element defines a sort of skeleton of the element which should be created in the target REA or IEC 62264 model. If an element is created in either model and it matches the source element, then the corresponding target element will be created as an instance in the target model. Figure 5.14 shows the syntax of an example for a dynamic element mapping.

This syntax allows to define that a segment requirement should be matched to a transformation reciprocity defined in an REA instance model. Due to the nature of the REA meta modeling language we are not able to directly define the transformation reciprocity as a target element. Therefore, this approach technically allows to map specific IEC 62264 elements to any REA element that can be defined in the instance model.

Figure 5.15 shows an example instance of a mapping model. In this mapping model, we define a mapping for a segment requirement to a transformation reciprocity. For the IEC 62264 part

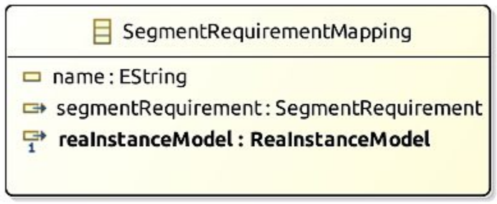

Figure 5.14: EMF Ecore representation of a dynamic mapping model element of our mapping meta model

of the mapping a segment requirement has been defined with certain properties: Having a ProduceTable operations definition, operations type PRODUCTION, process segment Cut and segment state FORECAST. If a new segment requirement is created in the IEC 62264 model, the application tries to find a matching one in the mapping model. If a match is found, the application would create a transformation reciprocity with the properties defined in the mapping model. In this case, a transformation reciprocity with the name Cut and four commitments would be created in the REA model.

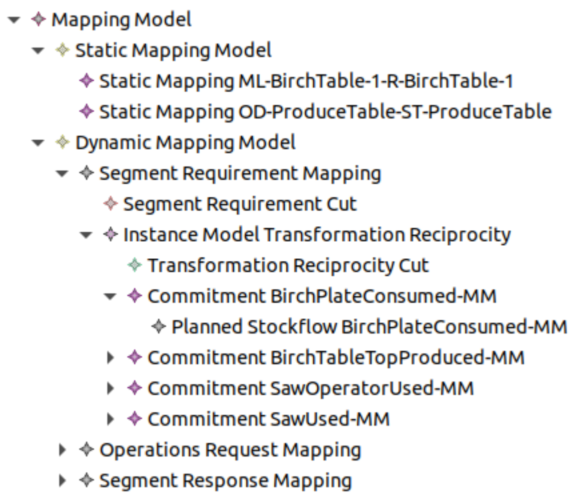

Figure 5.15: An example of a mapping model instance viewed with an EMF model editor

Figure 5.15 shows instances of static mappings. For example, static mapping ML-BirchTable-1-R-BirchTable-1 would define that material lot BirchTable-1 should be mapped to resource BirchTable-1. The application code than defines if one of these elements changes, it would also apply this change to the mapped element, i.e. update the quantity property. The logic that the quantity attribute should be updated is still defined in the application itself.

The basic application is working in the same way as described in Section 5.4.2. Example models are still loaded from disk and saved after the simulation ends. The simulation is also working in the same way. The major differences are that the mapping model is now loaded as well on application startup and that the mapping model is used in the adapter listening to IEC 62264 model changes in order to update the loaded REA model. We also took an additional step to initialize the mapping model. Using the EMF editors, one is able to reference elements of the example models, but if the mapping model is loaded from disk, the references are only valid in the loaded mapping model itself. Using the ids of the elements in the mapping model, it is possible to override the actual run time instances of the elements with the ones from the REA and IEC 62264 model. This allows us to directly manipulate the elements when working with an instance of a mapping model element in the code. For example, when a mapping of resource and material lot is encountered, one can directly access the references of this mapping element and update the quantity object of one of the referenced REA or IEC elements without the need of first retrieving this element from either model instance. This process is only relevant for static mappings referencing and mapping already existing elements. Elements contained in dynamic mappings are created as a new instance in the respective model anyway.

The model adapter for this approach is also more advanced than in the proof of concept. In addition to listening for material lot quantity changes, it is listening for the creation of new operations requests, segment requirements and segment responses. If an operations request is encountered, the adapter looks for a dynamic mapping in the mapping model where an operations request element with the same operations definition id has been mapped to a schedule. The information contained in the REA instance model of the mapping is taken and used to create the corresponding REA schedule element in the REA meta model instance. The process is similar for segment requirements, where the mapping is used to retrieve a transformation reciprocity skeleton to from the mapping model and for segment responses, where a transformation duality skeleton is retrieved. Due to the nature of the mapping model mentioned above, it is possible to make use of functionality provided by the Eclipse Modeling Framework (EcoreUtil.copy()) to create a copy of the skeleton elements, fill them with any additional data necessary and add this copy to an REA meta model instance. This eliminates some code that would otherwise be needed in order to create the various elements.

5.5 Approach 2: Event Streams and Distributed Applications

Since code is always needed to complete the mappings we also try a different approach: all of the mapping logic is in the application code itself to skip the complexity overhead introduced by a mapping model.

5.5.1 Conceptual Overview

In Figure 5.16 a conceptual overview of the idea for the second approach of implementing our model-driven vertical integration application is shown.

The idea of this concept is to mimic modern distributed system architectures. In the real world the applications for handling MES and ERP systems are likely more different and more distributed. Therefore, we decided to use separate applications to simulate our ERP system

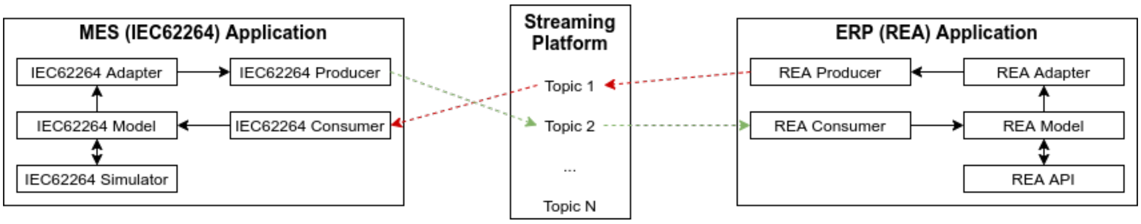

Figure 5.16: Overview of the second approach of the implemented application

and MES. These applications run standalone and are not directly connected with each other. In order to exchange data, we introduce a stream processing system. A stream processing system is a system where the channels used to exchange data are streams, which are an essentially infinite sequences of data [Ste97]. Our applications produce events when certain elements are added to or updated in their meta model instances. For example, if the MES reports that a table has been produced, the quantity of the respective material lot is updated in its IEC 62264 model and this information is published as an event into a stream. The ERP system is notified about the event and consumes it in order to update the quantity of the respective resource element in its REA model. The streams these events are published to are separated by topics. A topic contains all events related to a specific IEC 62264-to-REA or vice versa mapping. For example, there could be a topic which contains all events that are related to material lot changes. In order to create events, we simulate a IEC 62264 model (as described in Section 5.3), which changes the model loaded into the production application. These changes are picked up by an adapter listening to model changes in order to produce events and publish them into a stream from where the event will be consumed by the ERP application. On the REA side, we offer an API that simulates the possibility of ordering a product. When a product is ordered, the REA model is updated and the adapter listening for model changes publishes an event which is then consumed by the production application.

To implement this concept, we use multiple technologies: We use Kotlin[4] as a programming language. We decided to switch from Java to Kotlin, because Kotlin allows to reduce boilerplate code needed when working with collections. We spend a lot of time iterating and manipulating collections due to the nature of the models we are working with. Using Kotlin allows us to keep existing logic but reduce the necessary lines of code.

As a framework for building our applications we chose Ktor, as it allows us to run each of our applications as a separate server application. Thus, we are able to implement simple REST APIs, which we control our applications. For example, we can expose an endpoint of our MES application which would start a simulation of the applications IEC 62264 meta model.

The applications do not communicate directly with each other, but publish the data they produce into a stream. In order to stream this data, we chose Kafka[5]. Kafka is a popular open-source streaming platform. At the time of writing, the Kafka website claims that 10 out of the top 10 manufacturing companies make use of Kafka. Major benefits of Kafka are that

[4] https://kotlinlang.org
[5] https://kafka.apache.org/

the platform is scalable and capable of high throughput. Libraries are offered to interact with the platform, and there is a vast amount of documentation and examples available.

In the following sections we discuss how we have made use of the technologies mentioned above.

5.5.2 The Streaming Platform

We use Kafka in order to store the events produced by our applications into streams. We organize these streams into topics. For each alignment of elements of the IEC 62264 and REA meta models we introduce a separate topic. We implement the following topics:

Material Lots Events are published to this topic when the quantity of a material lot changes during the performance of the production process in the IEC 62264 simulator. The information published is the ID of the changed material lot and the quantity of the material lot after the update of the meta model. The ERP system application consumes the events of this topic and updates the quantity property of the affected resources by using the provided ID.

Segment Responses There are two ways events get published to this topic. The first is when a new segment response gets created in the IEC 62264 model, the second is when an existing segment responses' end time property is updated. Information published are the segment responses' ID, the ID of its process segment, its start time and its end time. Segment responses are usually created during the creation of operations performances and they are updated during the performing step of the IEC 62264 simulation. The ERP application consumes the events and either creates or updates the respective transformation dualities in the REA model.

Schedules Schedule events are published from the ERP application when a new schedule is created in the REA model. This usually happens if there is an external request to produce a product (e.g. a table is ordered using the API exposed by the ERP application). The ID of the schedule, the ID of the schedule's type and the IDs of the schedule's events are information published to the stream. The production application consume the event and creates a respective operations request element in the IEC 62264 model.

Resources When the quantity property of a Resource changes in the ERP application's REA model, an event is published to this topic. The resource's ID and current quantity after the change are provided as information. Based on the ID of the consumed resource event, the MES application updates the quantity of the respective material lot element in the IEC model.

All information is published to the streams as JSON[6] objects. We have also tried to publish the actual model elements into the event streams. This does basically work and has the advantage that one does not need to map the model elements and properties to a JSON object, and that one can make use of the code generated by the Eclipse Modeling Framework. The issue is however, that if one serializes only parts of a model, these parts get actually removed from

[6] JavaScript Object Notation, cf. https://www.json.org/json-en.html

the model. References are serialized as well, but one also looses their information if one dos not serialize the referenced objects as well. Therefore, if one is are only interested in little information of a single element, one will have a lot of overhead that one needs to send around. In the end we kept serializing and deserializing to and from JSON. Maybe the further exploration of directly serializing and deserializing the model element could be explored in future work.

Details of how the elements of the meta model instances are created and updated based on the consumed events can be found in Section 5.5.3 and Section 5.5.4.

5.5.3 Manufacturing Execution System

The MES application is built as a simple representation of a system reporting production data and scheduling production. This is achieved mainly by manipulating an IEC 62264 model at run time. In the following, we discuss the implementation of the various parts of the MES application.

Application Framework

When the application is run, the application framework is started locally as a application server listening on port 8080. It offers a REST endpoint (GET localhost:8080/simulate) which starts a simulation run of the IEC 62264 model which is loaded by the application at this moment. The IEC 62264 simulator, which has been described in Section 5.3, is executed when this endpoint is called.

Model Load and Store Functionality

Using the Eclipse Modeling Framework, an IEC 62264 model is loaded on application startup. Our implementation loads our IEC 62264 example model which we described earlier. The state of the currently loaded model is saved into a separate file every time after a simulation run or when a change has been detected by the model adapter. If the application is restarted, the initial IEC 62264 example model is loaded again.

IEC 62264 Simulator

The IEC 62264 simulation logic which has been described in Section 5.3 is part of the MES application. Operations schedules are created when REA schedule events published by the ERP system are consumed by the MES application.

Model Adapter

The model adapter listens for new segment responses and when the end time of a segment response changes. In both cases a segment response event is published. Material lot quantity changes are handled directly in the simulator in order to prevent a ping pong effect when resource quantity change events are received from the ERP application, which also change the IEC 62264 model and would therefore trigger the model adapter.

Event Consumers and Producers

The implementation offers two consumers, one consuming resource events and one consuming schedule events.

Resource Event Consumer This consumer consumes resource events published by the ERP system. The event contains an ID and a quantity. The ID is used to look up a material lot with the same id in order to set its quantity to the quantity of the event.

Schedule Event Consumer The events consumed by this consumer contain the ID and the schedule type of the schedule created in the REA model of the ERP system. First, an operations request is created. Its ID is set to the same as the ID of the corresponding REA schedule. Based on the (REA) schedule type of the (REA) schedule the corresponding operations definition is retrieved from the operations definition model. This definition is set as the created operations request's operation definition. Based on the material bill of the definition, the operations request's segment requirements are created. For each material bill item, a segment requirement is created if the quantity in stock of the material to be used in the scheduled production step is lower than the required quantity. The attributes and containment (i.e. personnel, equipment and material requirements) of the segment requirement are based upon its corresponding operations segment which is determined by the material bill item. The material bill items are iterated recursively since they can also contain other material bill items. The created segment requirements are added to the operations request. Then an operations schedule is created, using the schedule id from the schedule event, the operations request is added to the operations schedule and the schedule is added to the IEC 62264 meta model.

There are two producers present in the implementation, one responsible to produce events when the quantity of a material lot event changes and another one to produce events when a segment response is created or updated.

Material Lot Event Producer These events are produced when the IEC 62264 simulator changes the quantity property of a material lot. The produced event contains the ID of the material lot and the value of the quantity property after the update.

Segment Requirement Event Producer This producer triggers events when segment requirements are created as part of an operations request. The events produced by this producer contain the id of the segment requirement, the id of the process segment of the segment requirement and the ID of the operations request the segment requirement is contained in.

Segment Response Event Producer Segment response events are produced on two occasions: First, when a segment response is created and the creation is picked up by the model adapter. Second, when a segment response's end time property is updated, which is picked up by the model adapter as well. The produced event contains the ID of the segment response, the ID of its process segment, the value of its start time property and the value of its end time property.

5.5.4 Enterprise Resource Planning System

The ERP application has been built as a simple representation of a system reporting business operation data. This includes being able to report the order of a product by a customer. This is achieved mainly by manipulating an REA model at run time. In the following, we will discuss the implementation of the various parts of the ERP application.

Application Framework

When the application is run, the application framework is started locally as a application server listening on port 8081.

Model Load and Store Functionality

Using the Eclipse Modeling Framework, an REA model is loaded on application startup. In our implementation this is the REA example model described earlier. The loaded model is saved in a separate file every time after a simulation run. This behavior results in having our application start from the initial REA example model every time it is executed. This is identical to the logic used in the MES application, the only difference being that REA model instances are loaded and stored.

API

The ERP application offers two REST endpoints. To simulate the "sale" of a product, we are able to call GET localhost:8081/sellTable. If we want to "order" a product, we call GET localhost:8081/orderTable. In our example implementation, the product we offer is a simple wooden table. When a table is sold, a transfer reciprocity including commitments and a transfer duality including events is created. These elements are based on type elements that are defined at design time. If there is no table in stock, a table will be ordered instead. To represent this in the REA model, a schedule is created, including transformation reciprocities and their commitments. The elements are created based on type elements modeled at design time.

Model Adapter

The REA model adapter listens to changes in the loaded REA model instance. Specifically, the adapter is triggered if an event or a schedule is created. If an event named "GiveTable" is encountered, the quantity of the table resource is reduced by one. When observing an event named "TakeMoney", the money resource is increased by one. In both cases a resource event is published to the resource event stream in order to be consumed by the MES application. If a new schedule has been added to the REA model instance, a schedule event is produced and published to the schedule event stream.

Event Consumers and Producers

In the ERP application, there are two consumers: One responsible for handling material lot events and another one responsible for handling segment response events.

Material Lot Event Consumer Based on the material lot ID of the received event, the consumer retrieves the corresponding REA resource element from the loaded REA model instance and sets the quantity property of the element to the value of the quantity received in the event.

Segment Requirement Event Consumer Since the segment requirements are created as part of the creation of an operations request in the IEC model after receiving a schedule event, the operations request id contained in the segment requirement event is used to find the schedule which triggered the initial event. Based on this schedule and the process segment id from the received event, transformation reciprocities are created for this schedule. The schedule and the transformation reciprocities resemble the scheduled production process, making the MES the source of truth for production schedules rather then the REA model's schedule types.

Segment Response Event Consumer First the consumer uses the ID of the segment response contained in the event and tries to retrieve a corresponding transformation duality element from the loaded REA model instance. If a transformation duality with this ID exists, the start time and end time properties of its events are updated to the start time and end time received with the event. If no transformation duality with this id exists in the REA instance model, a new transformation duality is created. The id of the duality is set to the segment response id contained in the event, and the name property of the duality is set to the process segment id that is present in the event as well. Based on the process segment id, a corresponding transformation duality type is retrieved from the REA instance model. This type is added as type of the created duality and the various events necessary for the transformation duality are created based on this type. The ID of the segment response is included in the id of the (REA) events in order to be able to track which (REA) events are created due to events received from the stream.

Resource Event Producer Events are produced by this producer when the REA model adapter picks up the creation of an event representing that a table has been sold or that money for a sale has been received. The produced event contains the id of the resource and the value of its quantity property.

Schedule Event Producer When a new schedule is created in the REA model, the schedule event producer produces an event that contains the id of the schedule and the id of its type. We have also experimented a bit in order to only produce a schedule event when a schedule has been created by the ERP system and not based on an external event. To do so, we use a boolean attribute that is set when creating a schedule to determine if a schedule is created due to an external event or not. For example, a schedule could be created based on the fact that the MES application created an operations request in its IEC 62264 model. In this case it is not necessary for the ERP system to inform the MES that it has created a new schedule.

5.5.5 Implementation Visualized

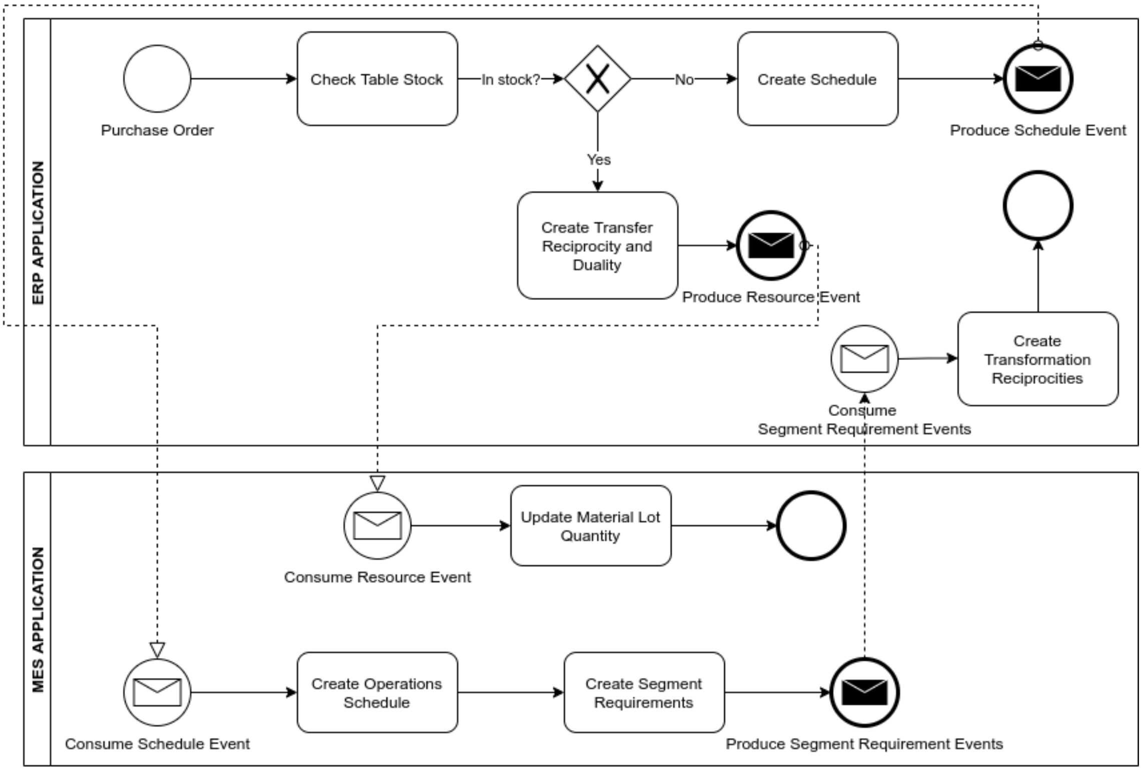

Figure 5.17: Process of a table being purchased in our implementation

Figure 5.17 shows the process of the sale of a table as it is implemented in our example applications. To simulate a purchase order, we call the API endpoint (GET localhost:8081/sellTable) provided by the ERP application. Doing so results in checking if a table is in stock by checking the quantity property of the respective resource. If a table is in stock, the respective transfer reciprocity and transfer duality elements are created. During this process, the quantity property of the table resource is reduced. This change is picked up by the model adapter, which produces a resource event. This event is picked up by the MES application, which uses the information of this event to update the quantity of the material lot representing a table.

If no table is in stock, the ERP application creates a schedule in the REA model. This change is picked up by the model adapter and a schedule event is produced. This event is consumed by the MES application, which creates an operations schedule in its IEC 62264 model. Part of creating this operations schedule is to create segment requirements representing the scheduled production steps. The information of these elements is sent to the ERP application, where transformation reciprocities are created in the REA model to reflect the scheduled production steps.

In Figure 5.18 we depict the process executed when the simulation of the IEC 62264 model is started in the MES application. First, operations performances are created and initialized. During this initialization, segment responses are created. The model adapter picks up every creation of a segment response and produces a segment response event, which is in turn consumed by the ERP system in order to create a transformation duality.

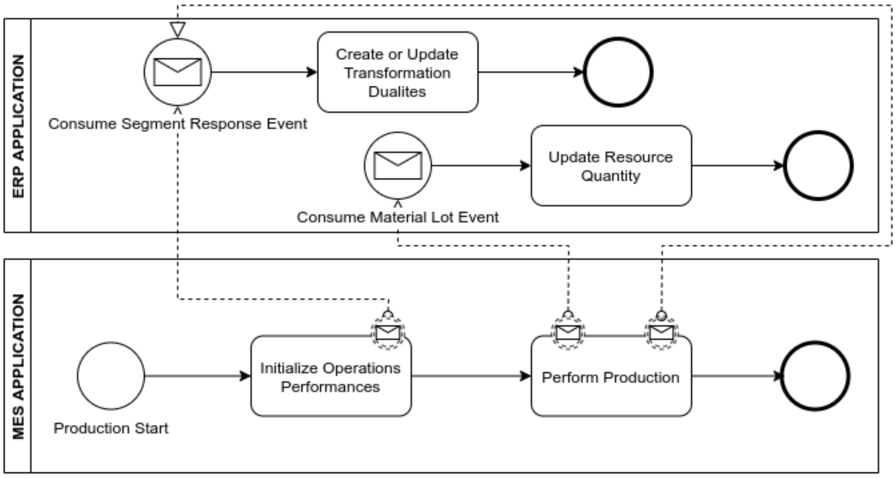

Figure 5.18: Process of running the IEC 62264 simulator in our implementation

Later, the created operations performances are performed, resulting in updates to the segment responses created earlier. This updates are again picked up by the model adapter and propagated to the ERP application's REA model using segment response events. Additionally, the performance changes the quantity of material lots and produces events to notify the ERP application.

CHAPTER 6

Evaluation

For this chapter, we first discuss the alignment of the IEC 62264 and REA meta models, since this alignment is the corner stone of our work. After that, we evaluate the two approaches of a model-driven way of vertically integrating an MES and an ERP system that we have implemented.

6.1 Alignment of the IEC 62264 and REA Meta Models

In our work, a conept for aligning the elements of the different meta models is an integral part of implementing a model-driven vertical integration. We start with quantitatively and qualitatively analyzing the alignment which was presented in [WHM17a]. In this section, we show to which extent the IEC 62264 and REA meta models overlap and discuss what we discovered using this alignment in our implementation.

6.2 Existing Alignments

In the existing alignment work, 35 REA meta model elements and 97 IEC 62264 meta model elements are considered. Table 6.1 shows how many of these elements are aligned with an element of the opposing model.

Meta Model	Aligned Elements (%)	Non-aligned Elements (%)
REA	23 (65.71%)	12 (34.29%)
IEC 62264	64 (65.98%)	33 (34.02%)
Both	87 (65.91%)	45 (34.09%)

Table 6.1: Quantified alignments

87 of the 132 considered elements (about 66%) are aligned with another element, leaving 45 (about 34%) without any alignment. Considering the direction of the alignments, 23 REA elements are aligned to 64 IEC 62264 elements. One element may be aligned to multiple other elements, for example resources can be used to depict materials, equipment and physical assets. The REA element which is aligned with the most IEC 62264 elements is the attribute element, which is aligned with 24 IEC 62264 elements.

6.2.1 Missing Alignments – REA to IEC 62264

12 out of 35 REA meta model elements are not aligned with any IEC 62264 meta model element. Out of these 12 elements, we have used 6 elements in our REA models: events and event types, commitments and commitment types, and fulfillment and fulfillment types. These elements are more or less indirectly aligned, since they are contained in elements that are aligned with IEC 62264 elements or are used to connect different REA elements. The other 6 elements are custody and custody types, claims and claim types as well as associations and association types. We do note use these elements in our example models, therefore we do not provide an assessment on the impact of their missing alignments.

Event and Event Type Events and event types are referenced by transformation dualities and transformation duality types respectively. Transformation dualities are created in the REA model when the ERP application receives a segment response event from the MES application. When the transformation duality instance is created, its type is determined by the information contained in the segment response event. Based on the transformation duality type, events are created and referenced by the newly created transformation duality. The transformation duality type is created at design time as a static model element in the REA model based on a respective process segment element of the IEC 62264 model. The specification elements of a process segment are aligned with participation and stock flow types. These types need to be contained in event type elements. Therefore, it is possible to calculate which events are to be created when a transformation duality needs to be created due to a change in the MES.

Commitment and Commitment Type Commitments and commitment types are referenced by transformation reciprocities and transformation reciprocity types respectively. In our current implementation, transformation reciprocities in an REA model are created on demand by the ERP system when the order of a product is simulated. Similar to dualities, recioprocities are created based on their transformation reciprocity type. Transformation reciprocity types are aligned with process segments as well. Therefore, their modeling and the modeling of their referenced commitment types is also based on a process segment and its contained specification elements, since commitment types also contain participation and stock flow types which are aligned with specification elements of a process segment. If transformation reciprocities need to be created based on a segment response event due to a change in the MES, the same approach will be used.

Fulfillment and Fulfillment Type Since a fulfillment is just a way of defining that a specific commitment (type) is fulfilled by a specific event (type), there is no need of an alignment to any IEC 62264 meta model element, since this concept can be modeled based only on the information available in an REA meta model instance.

6.2.2 Missing Alignments – IEC 62264 to REA

33 out of 97 IEC 62264 meta model elements are not aligned with any REA meta model element. Out of these 33 elements, we have used 12 elements in our IEC 62264 models. The other 21 elements are not used in the scope of our example models.

Equipment Asset Mapping Equipment asset mappings are used to show which physical assets are implementing the requirements specified by a equipment. Since there is no distinction between equipment and physical assets in REA, a mapping is not necessary. However, if the need arises to specify this connection in an REA model as well, we propose that a policy element will be used to specify that two resources are in such a relation with each other.

Operations Material Bill and Operations Material Bill Item We use these elements to state from which materials a product defined in an operations definition is assembled from, including the information which operations segments are responsible for producing the respective material if applicable. This information is used to determine which operations segments need to be scheduled in order to produce the product specified in the operations material bill. Therefore, this information is not necessary in our REA model, since it is only used by the MES application to create operations schedules and their contained elements in an IEC 62264 model.

Operations Segment and Specifications Operations Segments are referenced by Operations Definitions, which are aligned with Schedule Types. The various specification elements which are contained in operations segments are also not aligned with any REA element. In our implementation, we define operations definitions at design time, therefore no mapping is necessary. There are also no events produced by the ERP application that need to trigger the creation of an operations segment.

Operations Schedule, Operations Performance and Operations Response In our implementation, all of these elements are created by the MES itself during the simulation of an IEC 62264 model. They do not have to be created during any event published by the ERP system. Therefore, the missing alignment is not an issue. However, if the need of a mapping would arise, the logic to create these elements can be based on the alignments of their child elements, namely segment requirements and segment responses. These are aligned with transformation reciprocities and transformation dualities respectively and are used to create their parent elements.

In the end, there were no missing alignments that caused any serious issues when developing a concept to map and transfer data from one model to the other model. To summarize, we would answer our research question **"To what extent do IEC 62264 and REA overlap?"** as follows: Overall, the two models overlap conceptually by about two thirds. The models don't exactly overlap in the sense of that it is possible to represent the exact same information with the overlapping parts. A great amount of their elements can be aligned to each other, i.e. it is possible to express similar information in both models. These aligned elements may still be different in specifics like their properties or the level of generalization of the concept they are representing. If there are no missing alignments, the questions about the necessity of having two different models would arise: There would be no need for two different models if it is possible to depict the information with just one model.

6.3 Vertical Integration Using Model-Driven Engineering

In this thesis, we explore the possibility of implementing vertical integration in a production environment by using model-driven engineering. The general idea is to have an abstraction layer on top of existing systems. This is achieved by using models which are able to depict the various systems that may be present within a production environment. The next step was to enable the transfer of information between such systems using the abstraction layer by mapping the various elements of the models to each other.

6.3.1 Abstraction of Existing Systems

For our implementation, we assume that there is a model-driven abstraction of the systems that need to exchange data. In our case, we were looking at the two top-most levels of the automation pyramid: The business planning and logistics level, which can be represented by an ERP system and the manufacturing operations management level, parts of which can be implemented by an MES. For the proof of concept presented in this work, we did not abstract complete systems but rather a single exemplary abstracted sub-system of a production environment. The REA standard is used by us to represent an abstraction of an ERP system and part 2 of the IEC 62264 standard is used to model abstracted components of a MES system.

6.3.2 Mapping

In order to exchange data between the models of the abstracted systems, we need to find a way to model the relation of the various elements present in the (abstracted) systems. To determine which creation or update of an element is of interest to other systems, we map elements from one model to elements of the other model. When an element is created or updated, this mapping is used to update the elements mapped to it. An example would be the information that a product is assembled and is now in stock. This information is stored in the abstracted model of the MES and then published to the abstracted model of the ERP system by executing the mapping and creating or updating the respective model element. To do so, we take a look at two different approaches: First, by mapping the elements by introducing another abstraction via a mapping model. Second, by mapping the elements by convention, i.e. defining the mapping in the code of the software that is used to handle the abstraction layer.

6.3.3 Mapping by Mapping Model

First we want to make use of model-driven engineering in all aspects of vertical integration. To do so, we proposed a mapping model which should contain the necessary information to allow the (abstracted) systems to exchange data. To keep the mapping model as flexible as possible, we realize the mapping on the abstraction layer. This keeps the logic independent of the actual implementation of the systems.

Our first iteration of implementing a mapping model was simple: we added a parent class in the REA and IEC 62264 meta models. Every other element in the respective model inherits from this parent class, i.e. every element is easily recognized to be a part of an REA model or part of an IEC 62264 model. The mapping meta model is created as an additional meta model. The existing meta model implementations are referenced in the new meta model in order to use their elements in the definition of the meta model. We then created a simple mapping element which connects one IEC 62264 element with an REA element. When creating an instance of the mapping meta model, existing elements from IEC 62264 and REA model instances will be referenced when creating an instance of the mapping element if their models are referenced in the mapping model instance. This approach is very generic, but still needs a lot of logic in the application itself. If a change in an element is detected by the application, it looks up the mappings including the element in the mapping model. It then applies the change to the opposing element. But the actual change has to still be defined in the application code, for example if the quantity of a material lot is changed, the application uses the mapping model to find the opposing resource element and updates its quantity property. There is also no way of defining what should happen if a new element is created in one of the models representing the abstracted systems. However, this approach allows for a very slim mapping model which is easy to use.

In our next iteration of the mapping meta model, we implement a less generic approach. Based on the meta model alignment of REA and IEC 62264, we restrict which elements may be mapped to each other. For example, a mapping model instance allows to create a resource and material lot mapping element. This element only is only allowed to reference a resource and a material lot from model instances of the abstracted systems. This reduces the logic needed in the application code, but still does not account for the creation of new model elements. However, there is the need to create a mapping element for each possible REA and IEC 62264 alignment. Creating models also requires some knowledge about which elements are aligned to each other and why this alignment is made.

Finally, we combine the ideas of the first two iterations: We make use of the definition of static and dynamic model elements, where static elements are elements that exist at design time of the abstracted systems and dynamic elements being elements that are created during run-time. This final mapping model differentiates between generic static mapping elements and specific dynamic mapping elements. Static mapping elements simply reference one REA element and one IEC 62264 element. The application code determines what should happen to the mapped elements after looking up the mapping (similar to the first iteration). The dynamic mapping model element is more complex: it defines a source element, which is used to specify a pattern of an element that is to be created in REA or IEC 62264 model instances. This pattern is used to match created elements with a mapping element. For example, if a segment requirement with a specific operations definition is created in an IEC 62264 model instance and there is a pattern defining a segment requirement with that specific operations definition, there is a match, independent of the other properties of the created element. Furthermore, the mapping element also defines a target element. This target element serves as a template for the element which is to be created in the opposing model. After a match is recognized, the template is retrieved from the mapping meta model instance and used to create an element in the respective model instance of the abstracted system. The matching and creation of elements is still defined in the application code. This approach requires to define a mapping element for every aligned REA and IEC 62264 element in the mapping meta model. The creation of the mapping model instance requires knowledge of the underlying models representing the abstracted systems, since it is necessary to be able to define valid patterns and valid target templates. However, this approach is able of handling the creation of elements, while still keeping the mapping of static elements simple.

In general, the mapping model approach allows domain experts to model the interactions between (abstracted) systems. In order to do so in a meaningful way, a mapping meta model needs to provide more than just simple mappings of one model element to another. Properties, containments and references also need to be considered. In addition, there is a difference between mapping elements that already exist and elements that will only come into existence during run time of a system. Depending on the implementation, the mapping of elements can be restricted in order to create valid mappings that can be handled by the application implementing the mapping at run-time. However, there still needs to be specific application code in order to execute the mappings. It is also not clear what the performance of looking up elements in the mapping model in big and complex models would look like. The mapping meta model and its instances are also an additional resource that needs to be maintained. Since IEC 62264 and REA are complex meta models due to the amount of elements they consist of, a more thorough mapping model could possibly be designed more easily when meta models with fewer elements are considered.

In the end, the mapping model is not able to provide the advantages we are looking for. In the iterations we have tested, we were not able to completely separate the mapping logic from the code into the mapping meta-model itself. It would be beneficial to be able to define the mappings without having to also define and write code to complete the mapping in order to allow people who are not implementing the code to also create mappings. However, all iterations of our first application still needed specific code in order to work properly. The less code we needed in the application itself, the more expressive the mapping model needs to be. This may lead to a point were the mapping model will become hard to maintain on a regular basis.

6.3.4 Mapping by Convention

In contrast to an additional meta model, we also take a look at mapping by convention. For this approach, we still utilize the abstraction layer in order to keep the data exchange logic independent of the concrete systems. The mapping logic now only resides within the applications handling the abstraction layer. We build two separate applications, one that is responsible for handling the REA model and one that is responsible for handling the IEC 62264 model. To showcase a different form of communication between the applications, we make use of Apache Kafka. The applications publish the changes happening in their model into event streams. These streams are consumed by the opposing application in order to update the model they handle.

The main convention that we define for our work is that every element in every model has an ID which is used to determine its opposing element in the opposing model. The other conventions come from the alignment of the meta models and are translated into source code. A very simple example for such an alignment based convention is the handling of the alignment of resources (REA) and material lots (IEC 62264). If the quantity of a resource changes in the REA application, the ID and the updated quantity of the resource is published into a stream. This stream is consumed by the IEC 62264 application. The IEC 62264 application uses the ID to find a material lot with the given ID and updates its quantity property to the value received from the stream. Other alignment based conventions work in similar manner, but they may publish or consume more information and use this information to create more complex model elements with containments or references.

A clear disadvantage of this approach is that the mapping logic needs to be created and maintained in the applications handling the abstraction layer. Thus, the definition of new mappings require not only domain knowledge, but also programming skills. Without a mapping model certain restrictions cannot be enforced before at design time, i.e. when model instances are created the conventions need to be kept in mind. Since the mapping model approach also relies on some code in the applications in order to complete the mappings, the mapping by convention approach introduces much less complexity overhead. The code handling of the mapping conventions can be very generic and flexible. It does not address the semantic of model instances: model instances only need to follow the conventions that were defined beforehand. If they do, the application will work with any given model instance. No look-ups need to be made in any additional model, the applications just receive or retrieve data and process it. In the context of our work, all of these advantages made the implementation of model-driven vertical integration easier than with a mapping model.

6.3.5 Model-Driven Vertical Integration

After trying out two different approaches, we have come to the following conclusion in regards to **the advantages and disadvantages of model-driven vertical integration**. In general, we think a model-driven approach to vertical integration brings a big advantage: The vertical integration becomes independent of the actual implementation of the concrete systems that need to be integrated. By abstracting the systems, it is easy to exchange information that is generated during processes without having to worry about updating the information exchange logic when the concrete systems change but the processes stay the same. This decoupling is increased even further if the information is not directly sent to the abstracted systems, but when the concrete systems publish their information somewhere, for example in event streams or data stores, and the abstracted systems consume the data from there. The price for the increased flexibility and independence is that additional systems need to be maintained and additional standards may be introduced. Concrete systems need to be abstracted and the abstracted models need to be handled by one or more applications. But the model-driven vertical integration reduces the general amount of systems that need to be integrated with each other. Instead of integrating every ERP system (or part of an ERP system) with every MES (or part of an MES), every system just needs to integrate with the abstraction layer. The abstraction layer can handle the exchange of data between the systems without the need of connecting the concrete systems with each other.

In conclusion, both approaches we implement are an **appropriate** way of realizing model-driven vertical integration. They are not concerned with the concrete systems, just with their abstracted processes and therefore are independent of the underlying implementation. We described the advantages and disadvantages of both approaches and we come to the conclusion that the convention-driven mapping outperforms the model-driven mapping, especially in terms of flexibility and usability.

6.3.6 Illustrative Application Runs

To measure the performance of our implementation, we ran various variations of our mapping by convention application implementation. All measurements in this section are approximations. The runs were performed on a personal computer running *64-bit Ubuntu 20.04.2 LTS* installed on an *solid-state drive*, equipped with an *Intel® Core™ i5-4690K CPU @ 3.50GHz × 4* processor and *16 GB DDR3 RAM*.

Orders	IEC/REA Elements	IEC/REA Model File Size (MB)
1	157/262	0.03/0.05
10	598/1117	0.15/0.29
100	5,008/9,667	1.36/2.73
1000	49,108/95,167	13.39/27.18
10000	490,108/950,167	133.72/272.28

Table 6.2: Various final model states when only raw materials are in stock

We define two different scenarios for ordering tables based on the example models described in Section 5.2. In both scenarios, the initial models contain **108 IEC 62264 elements** and **167 REA elements** respectively. The initial models for the scenarios only differ in the quantity properties of their material lot and resource elements. The first scenario assumes that only raw materials like wooden plates, wooden blocks and veneer are in stock. Every task in the process of manufacturing a table, i.e. cutting table tops, veneering table tops, turning table legs and assembling tables, needs to be scheduled and executed in order to produce the number of ordered tables. Table 6.2 shows the results of ordering various amounts of tables through our table. Every order results in the creation of **49 new IEC elements** and **95 new REA elements**.

The second scenario assumes that all materials necessary to manufacture a table are in stock, i.e. the tables that are ordered only need to be assembled. The results in Table 6.3 show that a lot less elements need to be created, resulting in smaller model file sizes and shorter execution times. Every order results in the creation of **14 new IEC elements** and **24 new REA elements**.

Orders	IEC/REA Elements	IEC/REA Model File Size (MB)
1	122/191	0.02/0.03
10	248/407	0.06/0.09
100	1,508/2,567	0.41/0.68
1000	14,108/24,167	3.95/6.60
10000	140,108/240,167	39.38/65.92

Table 6.3: Various final model states when tables only need to be assembled

Looking at the results, we see a moderate linear increase in model elements, model file size and execution time. We expect that a system running an optimized implementation executed on dedicated hardware should have no performance issues.

6.3.7 A Brief Look at Horizontal Integration

In this work, we briefly touched upon the topic of horizontal integration in a model-driven context. An abstracted system, in our case the REA model representing an ERP system, would offer a way for external parties to interact with it in a certain way. In our example, we offered to order a table using an API. This creates elements in the REA model representing the order. Using the mapping logic, this order is transferred to the IEC 62264 model representing the production system where it is processed further. The data generated in the models is used to update the concrete systems. The advantage of this approach to horizontal integration is that it is not necessary to expose the internal concrete systems to external parties and allows the horizontal integration to be independent of the implementation of the concrete systems.

CHAPTER 7

Conclusion & Future Work

In this chapter, we summarize the main findings of this thesis and discuss possible future work.

7.1 Conclusion

Thanks to the fourth industrial revolution, modern manufacturing is deeply connected. There is an abundance of systems capable of communicating in place using proprietary or customized software to exchange data with one another. This may lead to great efforts when parts of the systems are replaced with parts which are not yet capable of integrating into the existing ecosystem, and thus require further customization of existing software.

In this work we have explored a way of delivering vertical integration on an abstract layer, which is independent of the actual implementation of a manufacturing environment. To do so, we developed an model-driven engineering approach. Assuming an abstraction layer represented by industry standards, we investigated two approaches to connect two abstracted systems with each other: First, introducing a mapping model representing the connection between the abstracted systems. Second, provide the rules of the information exchange in the application code running the abstracted systems. Each approach comes with its own advantages and disadvantages. The mapping model approach allows the mappings to be defined by domain experts and less logic needs to be implemented in the applications driving the model-driven vertical integration. However, modeling of the mappings can be complex and cumbersome for large model instances, and creating mappings requires extensive knowledge of the underlying models. During the course of our work we found the approach of defining the rules of the model-driven vertical integration in application code to be more flexible in our example implementation. The code needed to describe the mappings can be very generic, therefore no knowledge of the concrete underlying model instances is necessary if they follow the conventions defined beforehand.

In the end, both approaches help to abstract the process of data exchange in an manufacturing system from the concrete implementation of such a system. Both approaches are capable of supporting the model-driven vertical integration in manufacturing environments, as well as opening the door for approaching horizontal integration in a similar fashion.

7.2 Future Work

During our work we recognized a number of challenges that were out of scope of this thesis.In the following, we outline some of these challenges which could be investigated in future research.

Real World Case Study Our work is a first theoretical approach towards model-driven vertical integration. A real world case study would be very valuable to determine the feasibility of introducing a model-driven abstraction layer into an (existing) environment. It would be interesting to explore how actual equipment would able to publish information to the abstraction layer. Modern ERP systems are highly configurable and customizable, which could lead to problems in finding a standardized way of abstracting such systems. Additionally, surveys could be conducted with domain experts in order to determine the preferred approach presented in this work, i.e. having a mapping model versus doing a mapping by convention.

Alignment-based Model Generators In the course of this work we have built various meta model instances, in our case we usually started with an IEC 62264 model. Based upon the IEC 62264 model, using the rules of an alignment of the REA and IEC 62264 standards, we built an REA model. With our simplified example process, where we did only consider about 100 elements, we were able to do this manually. But for more complex models with thousands of elements, depicting complete systems, this would be a labor intensive and error prone activity. Therefore, it would make sense to create a generator application which is able to generate an REA model based on an existing IEC 62264 model and vice versa using the alignment rules. The generated models would probably not be complete, but would be a great starting point in order to drastically reduce the time needed to model parts of the abstraction layer needed in a model-driven vertical integration approach.

Improving the REA meta model implementation The IEC 62264 meta model implementation neatly organizes its elements in various containers. For example, there is a container that contains all material related elements like material classes, material definitions and material lots. Then there is an equipment container, containing all equipment related elements like equipment classes and equipment, and so on. The REA meta model implementation on the other hand only differentiates between a container for all type elements and a container for all instance elements. After having modeled a certain amount of elements it gets hard to navigate the model manually. Therefore, a great improvement would be to organize REA model elements in a similar fashion like the IEC 62264 meta model does. First, it would improve the experience when manually working with model instances. In addition, it would also improve the way model instances are handled in application code. This means it would be possible to specifically select containers containing specific types of elements easily, rather than needing to iterate over a single container containing all types of elements.

Leveraging Stream Processing During implementing the model-driven vertical integration approach we also made use of stream processing as a way of transferring information between the abstracted systems. Event streams are a popular way of exchanging data in modern software architectures. This could be leveraged in various ways. For example, an MES might publish information into an event stream rather than communicating directly with its IEC 62264 abstraction layer. This stream could also be used as a source of information for the ERP system abstraction, instead of having the MES abstraction communicating its state to the ERP abstraction. This makes the applications even more independent of underlying implementations. The applications running the IEC 62264 and REA model instances would simply be concerned with processing the data of the streams. This would also shift the focus to the actual data model used to publish the information.

Further Exploration of the Mapping Model Approach In our work, we explored the idea of using a mapping model in order to enable information exchange between two abstracted systems. This approach could be investigated in a more complex and overarching way, i.e. provide more than simple mappings of two elements to each other in order to leverage the power of model-driven engineering. To create instances of our mapping model during our implementation, we just used the basic tools provided by the Eclipse Modeling Framework. Some of the mapping models disadvantages we discovered may be mitigated by additional tool support, like a specialized application to support the creation of mapping model instances.

Horizontal Integration We only discussed the idea of model-driven horizontal integration briefly during the course of this thesis. At a first glance it looks like a promising approach of standardizing horizontal integration. This could be explored more thoroughly.

Industry 5.0 In the beginning of 2021 the European Union presented Industry 5.0 as a complementing paradigm for Industry 4.0. Industry 5.0 focuses on a sustainable, human-centric and resilient industry where digital and green technologies are key and the evolving role of industry workers and their well-being is considered [DNBP21]. An exploration of how model-driven engineering can support the key ideas of this emerging paradigm may provide a base for further research in this area.

List of Figures

List of Tables

Bibliography

[ABB+07] Roberto Acerbis, Aldo Bongio, Marco Brambilla, Massimo Tisi, Stefano Ceri, and Emanuele Tosetti. Developing eBusiness solutions with a model driven approach: the case of acer EMEA. In *International Conference on Web Engineering*, pages 539–544. Springer, 2007.

[BCW17] Marco Brambilla, Jordi Cabot, and Manuel Wimmer. Model-driven software engineering in practice. *Synthesis Lectures on Software Engineering*, 3(1):1–207, 2017.

[BDI+18] Daniel Bittighofer, Maike Dust, Alexander Irslinger, Marina Liebich, and Ludwig Martin. State of Industry 4.0 across German companies. In *2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, pages 1–8. IEEE, 2018.

[DBAF18] Lucas Santos Dalenogare, Guilherme Brittes Benitez, Néstor Fabián Ayala, and Alejandro Germán Frank. The expected contribution of industry 4.0 technologies for industrial performance. *International Journal of Production Economics*, 204:383–394, 2018.

[DIN16] DIN SPEC 91345:2016-04, Referenzarchitekturmodell Industrie 4.0 (RAMI4.0). Standard, Deutsches Institut für Normung (DIN), 2016.

[DLPH] Rainer Drath, Arndt Luder, Jorn Peschke, and Lorenz Hundt. AutomationML-the glue for seamless automation engineering. In *2008 IEEE International Conference on Emerging Technologies and Factory Automation, pages=616–623, year=2008, organization=IEEE*.

[DNBP21] Lars De Nul, Maija Breque, and Athanasios Petridis. Industry 5.0 - Towards a sustainable, human-centric and resilient European industry. 2021.

[FD05] M Fedai and R Drath. CAEX-a neutral data exchange format for engineering data. *ATP International Automation Technology*, 1(2005):3, 2005.

[For13] acatech Forschungsunion. Recommendations for implementing the strategic initiative INDUSTRIE 4.0. Final report of the Industrie 4.0 Working Group. Standard, Forschungsunion, acatech, 2013.

[FP13] Richard Fallon and Simon Polovina. REA analysis of SAP HCM; some initial findings. 2013.

[FSS+12] Albert Fleischmann, Werner Schmidt, Christian Stary, Stefan Obermeier, and Egon Börger. *Subject-oriented business process management*. Springer Nature, 2012.

[FWKVH16] Stefan Feldmann, Manuel Wimmer, Konstantin Kernschmidt, and Birgit Vogel-Heuser. A comprehensive approach for managing inter-model inconsistencies in automated production systems engineering. In *Proceedings of the 12th IEEE International Conference on Automation Science and Engineering (CASE)*, pages 1120–1127, 2016.

[GM97] Guido L Geerts and William E McCarthy. Modeling business enterprises as value-added process hierarchies with resource-event-agent object templates. In *Business object design and implementation*, pages 94–113. Springer, 1997.

[GM00] Guido L Geerts and William E McCarthy. The ontological foundation of REA enterprise information systems. In *Annual Meeting of the American Accounting Association, Philadelphia, PA*, volume 362, pages 127–150, 2000.

[GM06] Guido L Geerts and William E McCarthy. Policy-level specifications in REA enterprise information systems. *Journal of Information Systems*, 20(2):37–63, 2006.

[Hev07] Alan R Hevner. A three cycle view of design science research. *Scandinavian journal of information systems*, 19(2):4, 2007.

[HMPR04] Alan R Hevner, Salvatore T March, Jinsoo Park, and Sudha Ram. Design science in information systems research. *MIS quarterly*, 28(1):75–105, 2004.

[HR15] Martin Hankel and Bosch Rexroth. The reference architectural model industrie 4.0 (rami 4.0). *ZVEI*, 2(2):4, 2015.

[IEC13a] IEC 62264-1:2013, Enterprise-control system integration — Part 1: Models and terminology. Standard, International Electrotechnical Commission (IEC), 2013.

[IEC13b] IEC 62264-2:2013, Enterprise-control system integration - Part 2: Object and attributes for enterprise-control system integration. Standard, International Electrotechnical Commission (IEC), 2013.

[IEC15] IEC 62264-4:2015, Enterprise-control system integration — Part 4: Objects and attributes for manufacturing operations management integration. Standard, International Electrotechnical Commission (IEC), 2015.

[IEC16a] IEC 62264-3:2016, Enterprise-control system integration — Part 3: Activity models of manufacturing operations management. Standard, International Electrotechnical Commission (IEC), 2016.

[IEC16b] IEC 62264-5:2016, Enterprise-control system integration - Part 5: Business to manufacturing transactions. Standard, International Electrotechnical Commission (IEC), 2016.

[IEC16c] IEC 62424:2016, Representation of process control engineering - Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools. Standard, International Electrotechnical Commission (IEC), 2016.

[IEC20] IEC 62264-6:2020, Enterprise-control system integration – Part 6: Messaging service model. Standard, International Electrotechnical Commission (IEC), 2020.

[II07] International Organization for Standardization and International Electrotechnical Commission. Business Transaction Scenarios – Accounting and Economic Ontology. Technical report, 2007. ISO/IEC 15944-4:2007(E).

[ISO15] ISO/IEC 15944-4:2015 Information technology — Business operational view — Part 4: Business transaction scenarios — Accounting and economic ontology. Standard, ISO/IEC JTC 1/SC 32 Data management and interchange, 2015.

[JSW20] Juergen Jasperneite, Thilo Sauter, and Martin Wollschlaeger. Why we need automation models: handling complexity in industry 4.0 and the internet of things. *IEEE Industrial Electronics Magazine*, 14(1):29–40, 2020.

[Kas99] Roland Kaschek. Was sind eigentlich Modelle? In *EMISA Forum*, volume 9, pages 31–35, 1999.

[KHHW13] Henning Kagermann, Johannes Helbig, Ariane Hellinger, and Wolfgang Wahlster. *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*. Forschungsunion, 2013.

[KNH16] Udo Kannengiesser, Matthias Neubauer, and Richard Heininger. Integrating business processes and manufacturing operations based on S-BPM and B2MML. In *Proceedings of the 8th International Conference on Subject-oriented Business Process Management*, pages 1–10, 2016.

[Lan20] Laurens Lang. A Graphical Toolkit for ISA-95. Master's thesis, Technische Universität Wien, 2020.

[LMF16] Yan Lu, Katherine C Morris, and Simon Frechette. Current standards landscape for smart manufacturing systems. *National Institute of Standards and Technology, NISTIR*, 8107:39, 2016.

[LWH+20] Laurens Lang, Bernhard Wally, Christian Huemer, Radek Šindelář, Alexandra Mazak, and Manuel Wimmer. A graphical toolkit for iec 62264-2. *Procedia CIRP*, 93:532 – 537, 2020. 53rd CIRP Conference on Manufacturing Systems 2020.

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

[MH12] Dieter Mayrhofer and Christian Huemer. REA-DSL: business model driven data-engineering. In *2012 IEEE 14th International Conference on Commerce and Enterprise Computing*, pages 9–16. IEEE, 2012.

[MH15a] Alexandra Mazak and Christian Huemer. From business functions to control functions: Transforming REA to ISA-95. In *2015 IEEE 17th Conference on Business Informatics*, volume 1, pages 33–42. IEEE, 2015.

[MH15b] Alexandra Mazak and Christian Huemer. HoVer: A modeling framework for horizontal and vertical integration. In *2015 IEEE 13th International Conference on Industrial Informatics (INDIN)*, pages 1642–1647. IEEE, 2015.

[MHR13] Dieter Mayrhofer, Christian Huemer, and Peter Regatschnig. REA-ERP: Challenges of using REA in an ERP System. In *7th International Workshop on Value Modeling and Business Ontology (VMBO 2013)*, pages 1–4, 2013.

[MMW+14] Dieter Mayrhofer, Alexandra Mazak, Bernhard Wally, Christian Huemer, and Peter Regatschnig. REAlist: Towards a business model adapting multi-tenant ERP system in the cloud. In *8th International Workshop on Value Modeling and Business Ontology (VMBO 2014)*, 2014.

[MOF16] Meta Object Facility Specification Version 2.5.1. Specification, Object Management Group (OMG), 2016.

[MS95] Salvatore T March and Gerald F Smith. Design and natural science research on information technology. *Decision support systems*, 15(4):251–266, 1995.

[Nah19] Saeid Nahavandi. Industry 5.0—A human-centric solution. *Sustainability*, 11(16):4371, 2019.

[O'L04] Daniel E O'Leary. On the relationship between REA and SAP. *International Journal of Accounting Information Systems*, 5(1):65–81, 2004.

[Oli04] Antoni Olivé. On the role of conceptual schemas in information systems development. In *International Conference on Reliable Software Technologies*, pages 16–34. Springer, 2004.

[OMG17] OMG Unified Modeling Language (OMG UML). Standard, Object Modeling Group (OMG), 2017.

[OST97] Kai A Olsen, Per Sætre, and Anders Thorstenson. A procedure-oriented generic bill of materials. *Computers & industrial engineering*, 32(1):29–45, 1997.

[QLG16] Jian Qin, Ying Liu, and Roger Grosvenor. A categorical framework of manufacturing for industry 4.0 and beyond. *Procedia cirp*, 52:173–178, 2016.

[SB17] PO Skobelev and S Yu Borovik. On the way from Industry 4.0 to Industry 5.0: from digital manufacturing to digital society. *Industry 4.0*, 2(6):307–311, 2017.

[Sch07] Bianca Scholten. *The road to integration: A guide to applying the ISA-95 standard in manufacturing*. ISA, 2007.

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

[Sim19] Herbert A Simon. *The sciences of the artificial*. MIT press, 2019.

[SK03] Shane Sendall and Wojtek Kozaczynski. Model transformation: The heart and soul of model-driven software development. *IEEE software*, 20(5):42–45, 2003.

[SSKD11] T. Sauter, S. Soucek, W. Kastner, and D. Dietrich. The Evolution of Factory and Building Automation. *IEEE Industrial Electronics Magazine*, 5(3):35–48, 2011.

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

[Ste97] Robert Stephens. A survey of stream processing. *Acta Informatica*, 34(7):491–541, 1997.

[UN/05] UN/CEFACT. CODES FOR UNITS OF MEASURE USED IN INTERNATIONAL TRADE. 2005.

[Wal94] Bernhard Wally. Provisioning for MES and ERP-Support for IEC 62264 and B2MML. 194.

[WHM17a] Bernhard Wally, Christian Huemer, and Alexandra 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. IEEE, 2017.

[WHM17b] Bernhard Wally, Christian Huemer, and Alexandra Mazak. Entwining plant engineering data and ERP information: Vertical integration with AutomationML and ISA-95. In *2017 3rd International Conference on Control, Automation and Robotics (ICCAR)*, pages 356–364. IEEE, 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.

[WKK+10a] Manuel Wimmer, Gerti Kappel, Angelika Kusel, Werner Retschitzegger, Johannes Schoenboeck, and Wieland Schwinger. From the heterogeneity jungle to systematic benchmarking. In *International Conference on Model Driven Engineering Languages and Systems*, pages 150–164. Springer, 2010.

[WKK+10b] Manuel Wimmer, Gerti Kappel, Angelika Kusel, Werner Retschitzegger, Johannes Schönböck, and Wieland Schwinger. Surviving the heterogeneity jungle with composite mapping operators. In *International Conference on Theory and Practice of Model Transformations*, pages 260–275. Springer, 2010.

[WMKH15] Bernhard Wally, Alexandra Mazak, Bernhard Kratzwald, and Christian Huemer. Model-driven retail information system based on REA business ontology and Retail-H. In *2015 IEEE 17th Conference on Business Informatics*, volume 1, pages 116–124. IEEE, 2015.

[WPL17] Feng Wang, FuCheng Pan, and Peng Li. The perspective of B2MML based on implementation. In *2017 2nd International Conference on Automation, Mechanical Control and Computational Engineering (AMCCE 2017)*, pages 947–951. Atlantis Press, 2017.

[XXL18] Li Da Xu, Eric L Xu, and Ling Li. Industry 4.0: state of the art and future trends. *International Journal of Production Research*, 56(8):2941–2962, 2018.