PLATTFORM INDUSTRIE 4.0

DISCUSSION PAPER

Usage View "Seamless and Dynamic Engineering of Plants"

In collaboration with

STANDARDIZATION COUNCIL INDUSTRIE 4.0

ロボット革命イニシアティブ協議会
Robot Revolution & Industrial IoT Initiative

VDI VDE
VDI/VDE-Gesellschaft Mess- und Automatisierungstechnik

Imprint

Publisher
Federal Ministry for
Economic Affairs and Energy (BMWi)
Public Relations
10119 Berlin
www.bmwi.de

Editorial responsibility
Geschäftsstelle Plattform Industrie 4.0
Bülowstraße 78
10783 Berlin

Design
PRpetuum GmbH, Munich

Status
April 2020

Image credits
Shutterstock – foxaon1987 (title)
Shutterstock – MiniStocker (p. 4)
Getty Images – Westend61 (p. 7, 30, 40)
Getty Images – Sompong Rattanakunchon (p. 9)

You can obtain this and other brochures from:
Federal Ministry for Economic Affairs
and Energy (BMWi)
Public Relations
Email: publikationen@bundesregierung.de
www.bmwi.de

Central ordering service:
Tel.: +49 30 182722721
Fax: +49 30 18102722721

Contents

1 Foreword

In the application scenarios of Industrie 4.0, production and logistic systems are constantly adapted to changing requirements and value chains. Therefore, engineering tasks are not only necessary in the initial design and construction of a plant but will be required throughout the whole lifetime of a plant, to adjust to the dynamics. This engineering will hopefully be “seamless” over time and among the involved organizations. A digital model – today often called “digital twin” – can support the engineering significantly. This document is intended to describe the usage of a digital plant model so that a broad community better understands the objectives of this concept to be able to generate benefits from its usage.

We like to thank the members of the VDI/VDE Society Measurement and Automatic Control (VDI/VDE-GMA) Technical Committee 6.12 “Seamless Engineering of Process Control Systems” and the “Use Case Task Force” in the International Standardization Action Group, Robot Revolution & Industrial IoT Initiative (RRI), for having taken this approach for their contribution and open discussion. These activities are part of the Germany-Japan cooperation.

The presented elaboration is an important step for completing a common view to a core concept of Industrie 4.0 and to derive requirements for necessary standardization activities. Its results are supported by the strong commitment of VDI/VDE GMA and the Robot Revolution & Industrial IoT Initiative thanks to the open minded and integrating procedure chosen.

We also would like to thank the Standardization Council Industrie 4.0 (SCI4.0) for initiating and orchestrating the activities and partners within the project GoGlobal Industrie 4.0. This project strives for global harmonization and interlinkage of German Industrie 4.0-concepts with regional partnerships and strategic standardization development organizations.

We hope that this document will foster the joint understanding of the many ways in which an integrating plant model can be used in the engineering of industrial plants, towards a seamless and dynamic engineering.

Prof. Dr. Alexander Fay
Prof. Emeritus Dr. Eng. Fumihiko Kimura

2 Introduction

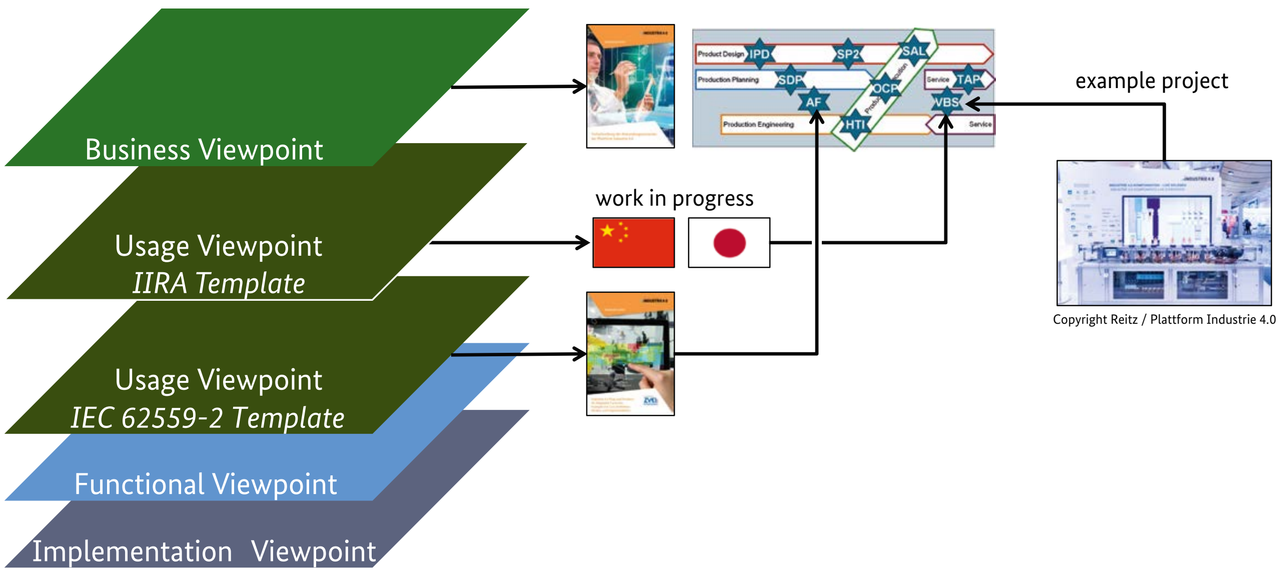
2.1 Background

The Technical Committee 6.12 “Seamless Engineering of Process Control Systems” of the VDI/VDE Society for Measurement and Automatic Control (VDI/VDE-GMA) has developed the application scenario “Seamless and dynamic engineering of plants” (in short: SDP), which is one of the application scenarios promoted by the German “Plattform Industrie 4.0”, see [1]. As of July 2017, the Technical Committee 6.12 has refined the application scenario SDP in the form of two exemplary sub-scenarios, see [2]. The description in [2] is focused on changes of business roles involved in the value network along the life of a plant and provides a business view on the sub-scenarios. This business view is based on a business model logic, illustrating the relationships between business roles such as a plant manufacturer, a system integrator, or a plant owner.

The overall principle of the application scenario SDP is that in an initial engineering process for engineering and construction of a plant, an *integrating plant model* is created, which is *maintained* and *kept consistent* throughout the entire life of the real physical plant in permanently interrelated processes between engineering, operation and service of the plant. But the integrating plant model is a *technical system* rather than a business role. Therefore, in the description of the application scenario SDP, the integrating plant model is always subordinated to a business role. In order to underline the importance of the integrating plant model, the Technical Committee 6.12 decided to add a usage view to the existing business view of the application scenario SDP. The *usage view* focuses on the integrating plant model and explains how technical roles interact with this model. The usage view is the topic of this document.

The first draft of the usage view was intensively discussed and refined together with the “Use Case Task Force” in the International Standardization Action Group “Robot Revolution & Industrial IoT Initiative”, so that these results are now available as a joint publication. These activities are part of the Germany-Japan cooperation within IEC TC65 Smart Manufacturing and were moderated by the Standardization Council Industrie 4.0.

Figure 1: Proposed structure for use cases (according to [6])



Source: Plattform Industrie 4.0

2.2 Context

In the context of Industrie 4.0, both a business view and a usage view have already been described for another application scenario (namely “Value-based Service”, in short: VBS), see [3], [4] and [5]. The knowledge gained from these activities was incorporated in the update of the German standardization roadmap, see [6], where a separate chapter on “Use Cases” was added. Figure 1 shows the core picture of this chapter.

Since the term “use case” is used in quite different contexts with different objectives, it was proposed to consider subjects under different views and then to conceptually distinguish between *business scenarios* and *use cases*, see Figure 1:

- Business scenarios primarily describe a business context, which is addressed by the business viewpoint. The basis is a value-network of business roles where each business stakeholder is characterized by its own business model. Relations between the business stakeholders within the value-network are characterized by value propositions.
- Use cases primarily describe the interaction of technical stakeholders (later called “roles” in this paper) with a technical system. They are addressed by a usage view based on a usage view-point. Thus, use cases describe the context of a technical system and high-level requirements, how the technical system interacts with the context. Use cases can be described on different level of detail. Figure 1 illustrates the possibilities which might be used for description: The IIRA template, see [6], or the more detailed template of IEC 62559-2.

A central recommendation of the German standardization roadmap in the context of “Use Cases” is to create further use case descriptions and to classify them using the structure as shown in Figure 1. Various international activities in the context of use cases, such as cooperation’s between Germany and Japan resp. China, also integrate their activities into this overarching structure.

The Technical Committee 6.12 took up this recommendation. The description of the application scenario SDP according to [2] is a business scenario according to Figure 1 and this document is a use case description according to Figure 1. For more information about the relation between the business view and usage view of the application scenario SDP, we refer to chapter “6 Relationship between Business View and Usage View”.

2.3 Objectives

Guided by the experiences from related activities, the goal is to find a suitable level of abstraction for the description of the usage view under the following boundary conditions:

- Understandable for persons outside of the author team
- Completeness (in the sense of the 80/20 rule[1]), balanced and representative with respect to the concept of an “integrating plant model”
- Manageable size (about 20 pages for the description of the usage view)

The target audience of this document are *experts* who want to better understand the concept of an integrating plant model. Our understanding of “experts” are solution and process architects having the interest to understand a technical concept in an application context. In addition, system and software architects and even technical implementers are addressed in the sense to understand the high-level requirements and usage of an integrating plant model, but not to have guidance for design and implementation concepts. To be more precise, our focus will be on *intrinsic* relations between physical objects and model objects[2], more details can be found in section “4.2 System under Consideration”. The usage view is therefore a general description that may serve as a template for specific examples. In order to illustrate this, we have prepared four examples in chapter “5 Examples for Illustration” to explain how the general description of the usage view is reflected in these examples.

2.4 Application Scope

The concepts of usage of an integrating plant model can be applied to both greenfield (where a new plant is built) and brownfield (where an existing plant is converted) projects. Also, these concepts can be applied both in discrete industries and in process industries, even if chapter “5 Examples for Illustration” is characterized by terms that are used more in process industries than in discrete industries.

In all of these cases typically models, libraries, and physical objects already exist. We want to create awareness for working with an integrating plant model from a usage perspective, but without going into details. Nevertheless, the document describes many aspects that need to be specifically defined in a specific application. In this respect, a central benefit of the document is that such critical design decisions are made explicit, which in our perception often does not happen in many discussions.

But note that in a usage view, an integrating plant model is described in terms of its *application* perspective. Although an integrating plant model contains structural, functional and behavior-based aspects, these aspects are *not* considered in a usage view, but should be considered in a functional view, what is not in the scope of this document.

1 The 80/20 rule (also known as pareto principle) states that, for many events, roughly 80% of the effects come from 20% of the causes.

2 These intrinsic relations are independent on opportunities based on evolving technologies like e.g. “Artificial Intelligence” or “Blockchain”. The possible impact of such technologies could be elaborated in a subsequent activity addressing a *functional view* of the application scenario SDP.

3 Business View

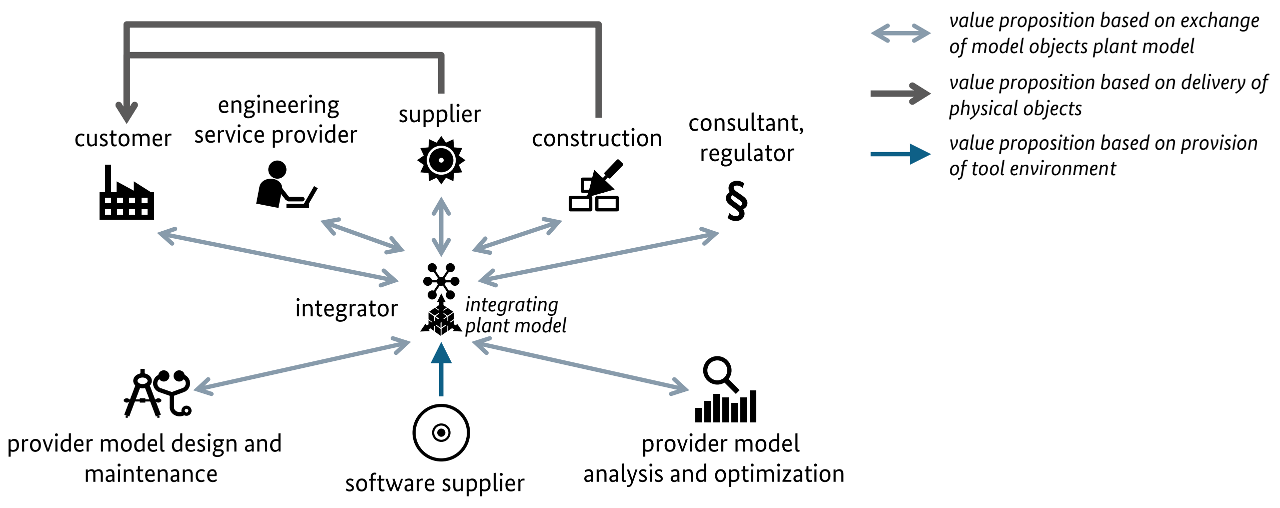
In this chapter, we give a brief introduction to the business view of the application scenario SDP, so that the present document remains comprehensible. For further information we refer to [2]. Figure 2 shows the value network[3] underlying the application scenario SDP.

The overall principle of the application scenario SDP is that in an initial engineering process for engineering and construction of a plant, an *integrating plant model* is created, which is *maintained* and *kept consistent* throughout the entire life of the real physical plant in permanently interrelated processes between engineering, operation and service of the plant. Besides a model of the real physical plant over its life (engineering and operation phase including conversions), this model also includes boundary conditions, context information, possible variants of the plant, conceivable and implemented engineering decisions as well as the impact of such decisions.

The application scenario SDP can be exemplified from a business perspective in different ways. In [2] two examples of such different exemplifications were developed, which are illustrated in Figure 3. It should be noted that green and orange color-coded roles in the value network are executed by the same company.

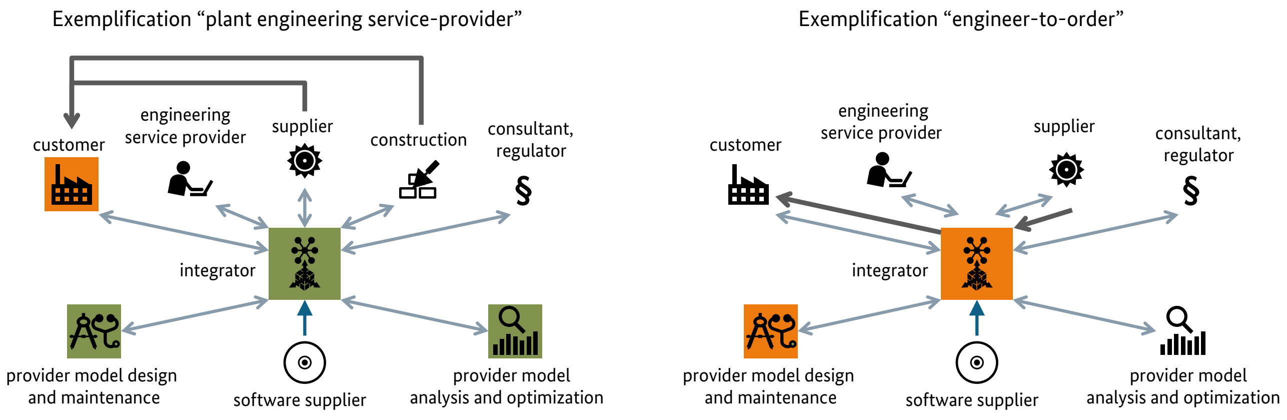
3 In addition to the value network described in [2], we have further detailed the value chains. Originally, only the value chains from the integrator's point of view were shown. We now distinguish between value chains, which are based on an exchange of models, and the value chain of the software supplier, which is based on the provision of a tool environment. Furthermore, we have added value chains between stakeholders based on the provision of physical assets.

Figure 2: Value network according to business view of application scenario SDP, see [2]

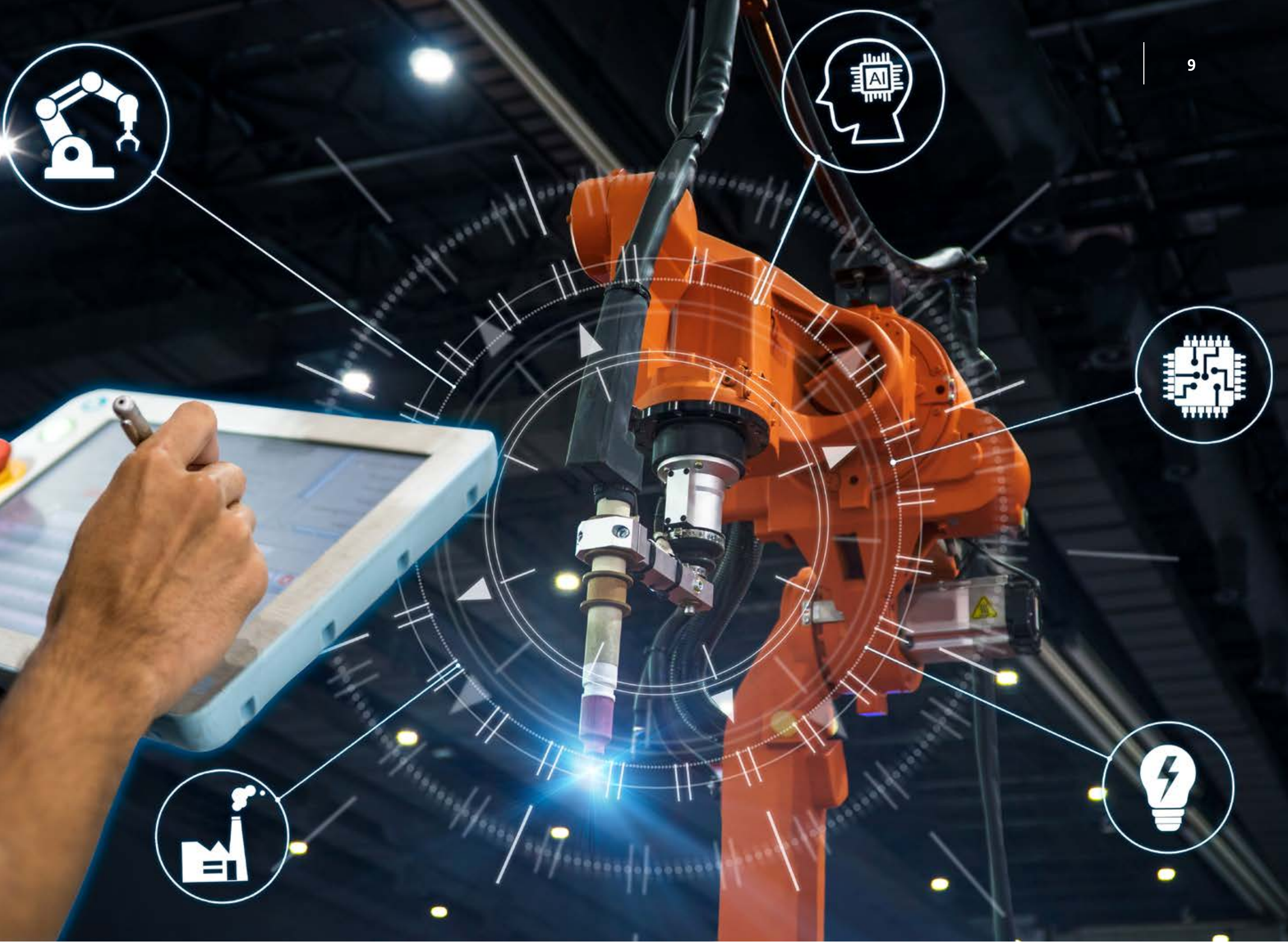

customer
engineering service provider
supplier
construction
consultant, regulator
integrator
integrating plant model
provider model design and maintenance
software supplier
provider model analysis and optimization
value proposition based on exchange of model objects plant model
value proposition based on delivery of physical objects
value proposition based on provision of tool environment

Source: Plattform Industrie 4.0

Figure 3: Different business exemplifications of application scenarios SDP, see [2]


Exemplification "plant engineering service-provider"
customer
engineering service provider
supplier
construction
consultant, regulator
integrator
provider model design and maintenance
software supplier
provider model analysis and optimization
Exemplification "engineer-to-order"
customer
engineering service provider
supplier
consultant, regulator
integrator
provider model design and maintenance
software supplier
provider model analysis and optimization

Source: Plattform Industrie 4.0



4 Usage View

To understand the Usage View, we first introduce concepts of the usage *viewpoint* proposed by the Industrial Internet Consortium, see Figure 4, for details see [7]:

- The basic unit of work is a *task*[4]. A task is **carried**[5] out by a party assuming a role.

- A *role* is a set of capacities assumed by an entity to initiate and participate in the **execution** of, or consume the outcome of, some tasks or functions in a system as required by an activity. Roles are assumed by parties.

- A *party* is an agent, human or automated, that has autonomy, interest and responsibility in the **execution** of tasks. A party **executes** a task by assuming a role that has the right capacities for the execution of the task.

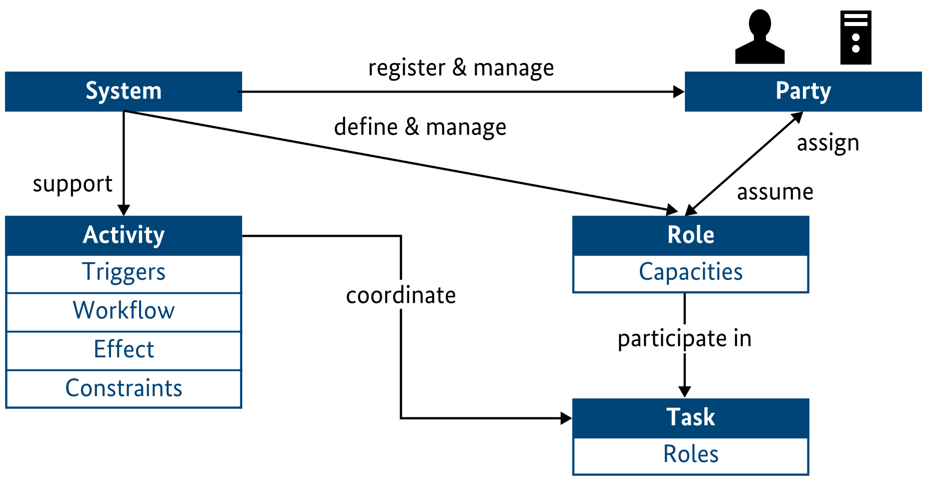
A party may assume more than one role, and a role may be fulfilled by more than one party.

- An *activity* is a specified coordination of tasks required to realize a well-defined usage or process of a system. An activity has the following elements:
  - A *trigger* is one or more condition(s) under which the activity is initiated.
  - A *workflow* consists of a sequential, parallel, conditional, iterative organization of tasks.
  - An *effect* is the difference in the state of the system after successful completion of an activity.
  - *Constraints* are system characteristics that must be preserved during execution and after the new state is achieved.

4 The tasks according to [7] include a Functional Map referring to the Functional Viewpoint and an Implementation Map to the Implementation Viewpoint. Since we are focusing on the Usage Viewpoint, we do not consider Functional resp. Implementation Maps.

5 The **bold** marked terms refine and illustrate the single term “participate-in” in Figure 4.

Figure 4: Overview of Usage Viewpoint (according to [7])



Source: Plattform Industrie 4.0

At this point we would like to point out the difference regarding the terms "viewpoint" and "view": An (architecture) view expresses the architecture of a system from the perspective of specific system concerns, whereas an (architecture) viewpoint establishes the conventions for the construction, interpretation and use of architecture views to frame these specific system concerns. For more details we refer to [7] or even to ISO/IEC/IEEE 42010.

4.1 Overview

We decided to take a more general approach to the original concept of an integrating plant model and instead to talk about a *set of model objects* in this paper. The integrating plant model then is a specific exemplification of a set of models which includes all model objects that are considered in a specific case.

Figure 5 shows a comprehensive overview of the usage view of the application scenario SDP. The system under consideration is shown in grey and the roles are shown in dark purple. Arrows in blue indicate core relations between the constituents of the system under consideration and arrows in purple indicate core interactions of the roles with the system under consideration.

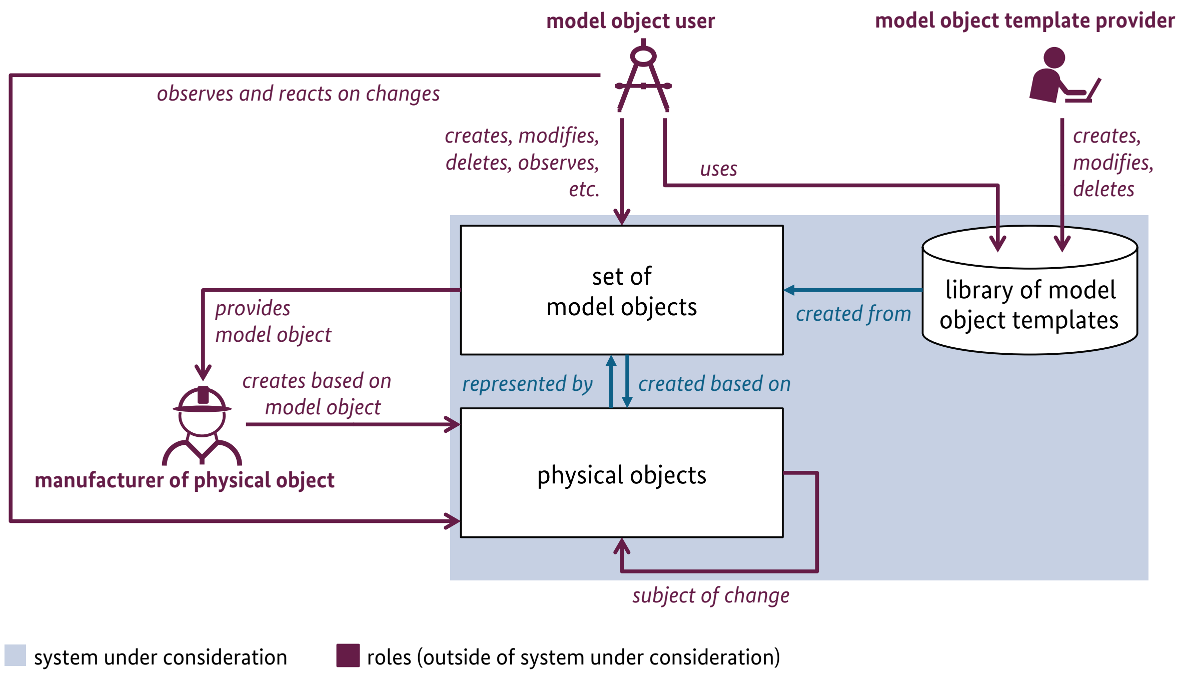
The system under consideration consists of two parts:

- On the one hand, it includes objects from the model world[6]. The objects of the model world are divided in *model objects* and *model object templates*. Model object templates are organized in a library of model object templates and can serve as templates for creating model objects, thus, model object templates represent meta-models (i.e. models describing a model). Models can be used for descriptive purposes (i.e. describing an existing object) or prescriptive purposes (i.e. describing a not yet existing object).
- On the other hand, it includes objects from the physical world[7]. These objects exist independently of the objects from the model world; however, a physical object can be represented by one or more objects from the model world, or one or several model objects can be the basis for the creation of an object of the physical world.

6 The model world is sometimes also called digital world, information world or virtual world. Note also that we do not consider physical objects representing a model of another physical object.

7 Note that we focus in on the model world and that we do not describe the concrete usage of physical objects, but only their relation to model objects.

Figure 5: Overview of usage view of "Seamless and Dynamic Engineering of Plants"


model object user
model object template provider
observes and reacts on changes
creates, modifies, deletes, observes, etc.
uses
creates, modifies, deletes
set of model objects
library of model object templates
created from
provides model object
creates based on model object
represented by
created based on
physical objects
manufacturer of physical object
subject of change
system under consideration
roles (outside of system under consideration)

Source: Plattform Industrie 4.0

Overall, the following roles are distinguished:

- The *model object template provider* is responsible for creating a library of model object templates.
- The *model object user* handles the set of model objects.
- The *manufacturer of physical object* can create a physical object based on one or more model objects.

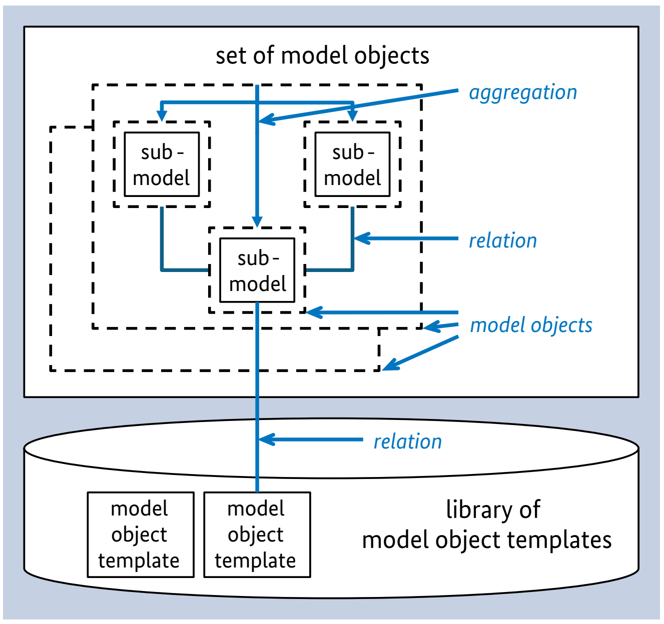
In practice, all these roles request for a powerful tool environment for their interaction with the system under consideration. We have decided to consider this tool environment merely as an aid, but not as part of the system under consideration. Nevertheless, we have included some aspects of a tool environment in sections "4.2.4 Tool environment" and "4.5.5 Management of tool environment".

4.2 System under Consideration

Regarding the system under consideration, objects are assigned – because of their fundamental being – to either the physical or the model world. Regardless of this assignment, every object exists in reality and has its own life. However, the management of the objects and their life are fundamentally different. While physical objects are permanently in an aging process due to their physical presence, objects of the model world remain unchanged in themselves. They change only because of targeted interventions from outside.

Objects of the model world are pure information objects. However, these cannot exist virtually, but need a physical carrier. But the information object itself, however, is completely independent of its physical carrier and does not participate in the life of the physical carrier. Conversely, objects of the model world are independent of the systems they might describe in the physical world.

Figure 6: System under consideration – model objects and model objects templates



Source: Plattform Industrie 4.0

Objects of the physical world, once created, are present independently of existence of objects in the model world. They have their own physical life and exist regardless of whether they are somehow considered or not.

Based on this fundamental assumption, we describe in this section the system under consideration in more detail. As already mentioned, we distinguish between objects of a model world, see Figure 6, and objects of a physical world, see Figure 8.

4.2.1 Set of model objects

The concept of *model object* forms the basis for modeling:

- A model object can be associated to another model object via a relation. A specific relation between model objects is an aggregation, where the existence of the sub-models is independent of the existence of the superior model object.
- Model objects may have attributes and attributes have values. Typically, attributes may have additional characteristics like value range, unit or default value, but these aspects are out-of-scope of our considerations in this paper.
- Each model object has an *owner* and a *life*[8]. These concepts are explained in separate sections, see “4.2.1.1 Principles for ownership for model objects” and “4.2.1.2 Principles of life of model objects”.

Be aware that in our understanding a model object is a description and *not* a concept in mind.

There are various purposes intended by models. Therefore, model objects and/or their attributes can describe requirements, solutions, assertions, measurements, value-assignment, etc. We do not distinguish here between such different purposes and descriptions. In addition, model objects can be structured according to different principles, e.g. spatial structure, logical structures, disciplines, cause-effect

8 In this context often the term “lifecycle” is chosen. We use the term “life” or “vita” according to the IEC/PAS 63088 specification (RAMI4.0).

relations, etc. Again, we do not distinguish here between such principles of structuring.

4.2.1.1 Principles for ownership for model objects

Ownership is a property of a party (i.e. stakeholder assuming a role, see Figure 4), which interacts with the system under consideration. The concept of ownership is used to manage the interactions of different parties with the system under consideration, because it is intended that some interactions are restricted to specific parties only.

Ownership can be defined for model objects, attributes, values of attributes and relations between model objects.

The following principles can be applied:

- An owner can grant ownership to another party acting in the same role. Typically, there are different levels of ownership to be able to manage also the withdrawal of ownership.

- There may be also different kinds resp. levels of ownership, e.g. hidden objects, read access to objects, write access to objects.

Management of ownership is a substantial capability to be provided by a tool environment. Nevertheless, we do not describe granting of ownership in the chapter “4.5 Activities”, because we do not want to describe the capabilities of a tool environment on such a level of detail.

4.2.1.2 Principles of life of model objects

Figure 7 illustrates the common life concept according to the IEC/PAS 63088 specification (RAMI4.0).

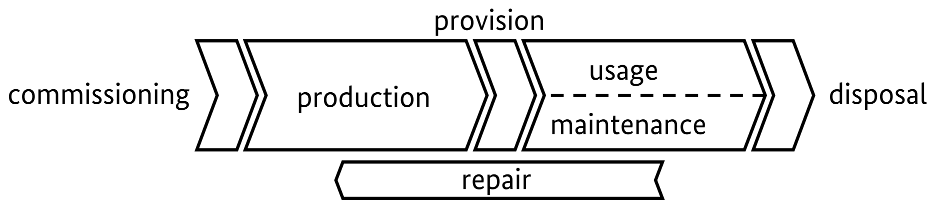
We assume that every model object knows its complete history since its creation according to this concept. When a model object is deleted and its life is ended, it continues to live in an “archive world”. The “archive world” is a subset of the model world. A model object in the “archive world” cannot be changed further, but its full history is still available.

Management of life of model objects is a substantial capability to be provided by a tool environment. This includes various aspects like versioning, tracking, audit trail, history, documentation, etc. Nevertheless, we do not describe these aspects in the chapter “4.5 Activities”, because we do not want to describe the capabilities of a tool environment on such a level of detail.

4.2.2 Library of model object templates

The concept of *model object template* as illustrated in Figure 6 is intended to be used for the creation of model objects. A model object template describes common characteristics of model objects, which are created from the model object template, thus, model object templates represent so-called meta-models. The model object templates and their lives are considered as part of the system under consideration.

Figure 7: Illustration of common life concept



Source: IEC

- If a model object template is used as the basis for the creation of a model object, this may result in a relation between model object template and the created model object. There are different kinds of relations possible, which have different effects in the case of a change of a model object template. Examples for different kinds of relations are:
  - Model object templates are organized in an "offline" library of model object templates: Here, changes to a model object template do not affect the associated model objects; only with newly created model objects will these changes take effect.
  - Model object templates are organized in an "online "library of model object templates: Here, changes to a model object template take effect immediately; thus, all associated model objects are affected and will consider these changes accordingly.
  - Hybrid concepts of both approaches with an explicit authority of the user are also possible.
  - In the activities in the chapter "4.5 Activities", we will only mention in general terms that in case of a change to a model object template an application specific reaction with regard to the consequences of related model objects is necessary, without looking at the different cases in detail.

- Model object templates can be structured by relations[9] and organized in libraries.

- Each model object template and each library of model object templates has an *owner*, which is described in the section "4.2.1.1 Principles for ownership for model objects templates", and a *life*, which is correspondingly described in the section "4.2.1.2 Principles of life of model objects".

4.2.2.1 Principles for ownership for model objects templates

The general concept and principles of ownership for model object templates and libraries are the same as described in the section "Principles for ownership for model objects".

Ownership can be defined for model objects templates, attributes, libraries, relations between model objects templates and relations between model object templates and model objects.

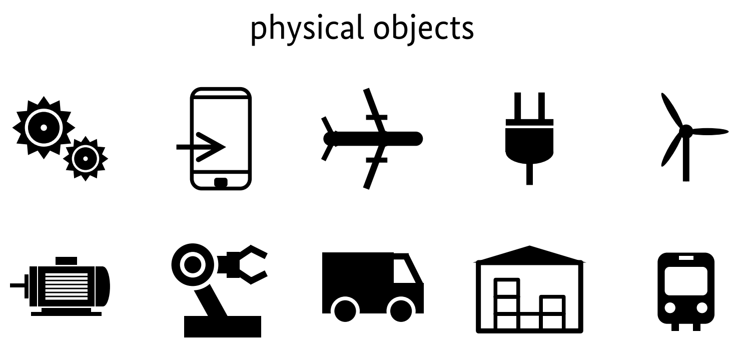
4.2.3 Physical object

Figure 8 illustrates the objects of the physical world.

Physical objects are objects in the real physical world.

- A physical object can be assigned to multiple model objects. These model objects not only represent explana-

Figure 8: System under consideration – physical objects



Source: Plattform Industrie 4.0

9 Relations between model object templates are not illustrated in Figure 6.

tions, but also, for example, requirements, descriptions or measurements[10].

- Physical objects also have a *life* as illustrated in Figure 7. In order to make the real life of a physical object accessible to the model world, an activity is necessary by explicitly assigning a model object to a physical object. This assigned model object must be distinguished from the physical objects and has its own life. But there can be physical objects whose life are not represented in the model world.

- The scope as well as the start and end of the life of a specific physical object is a design decision.

Physical objects may comprise executable software (for example the process control system of a plant is part of the physical world). Physical objects may also be "intelligent" (for example they could act autonomously or could provide "plug & produce" capabilities).

There are physical objects which are typically consistent with the associated model objects (for example, executable software), but there also are physical objects, where updates in the associated model objects must be initiated to synchronize the associated model object with the physical object in the case of changes.

4.2.4 Tool environment

A tool environment is a suite of software tools and applications including methods necessary to execute the various activities, especially if they are executed by humans. Such a tool must provide various capabilities, such as the

- creation, management and usage of model objects and model object templates including versioning, access control, etc.

- evaluation and simulation of model objects

- supporting engineering workflows and engineering decisions

- interaction with physical objects by sensing and actuating.

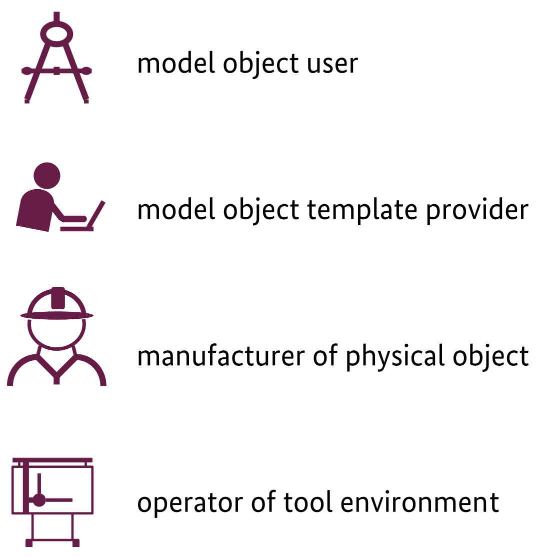
The capabilities of such a tool environment will continue to increase. Activities, which are nowadays typically executed manually, will be increasingly supported or even successively automated in the future.

The specific capabilities of such a tool environment address more a functional view and therefore are out-of-scope of this paper. The tool environment serves as a physical carrier for the model objects and model object templates according to section "4.2.1 Set of model objects" and "4.2.2 Library of model object templates".

4.3 Roles

In this section, we describe the various roles in more detail. Figure 9 shows an overview of the roles considered.

Figure 9: Overview on roles


model object user
model object template provider
manufacturer of physical object
operator of tool environment

Source: Plattform Industrie 4.0

10 In various communities, the term "digital twin" is currently being discussed intensively. According to the "Plattform Industrie 4.0", a "digital twin" is a virtual digital representation of a physical asset. In the sense of this paper, a model object associated with a physical object would then be a "digital twin" of the physical object.

- A *model object user* creates, modifies, deletes, observes, etc. one or more model objects respectively evaluates or simulates a model object. The role can be assumed by a human using a tool environment, but it can also be assumed by a software application providing appropriate capabilities.

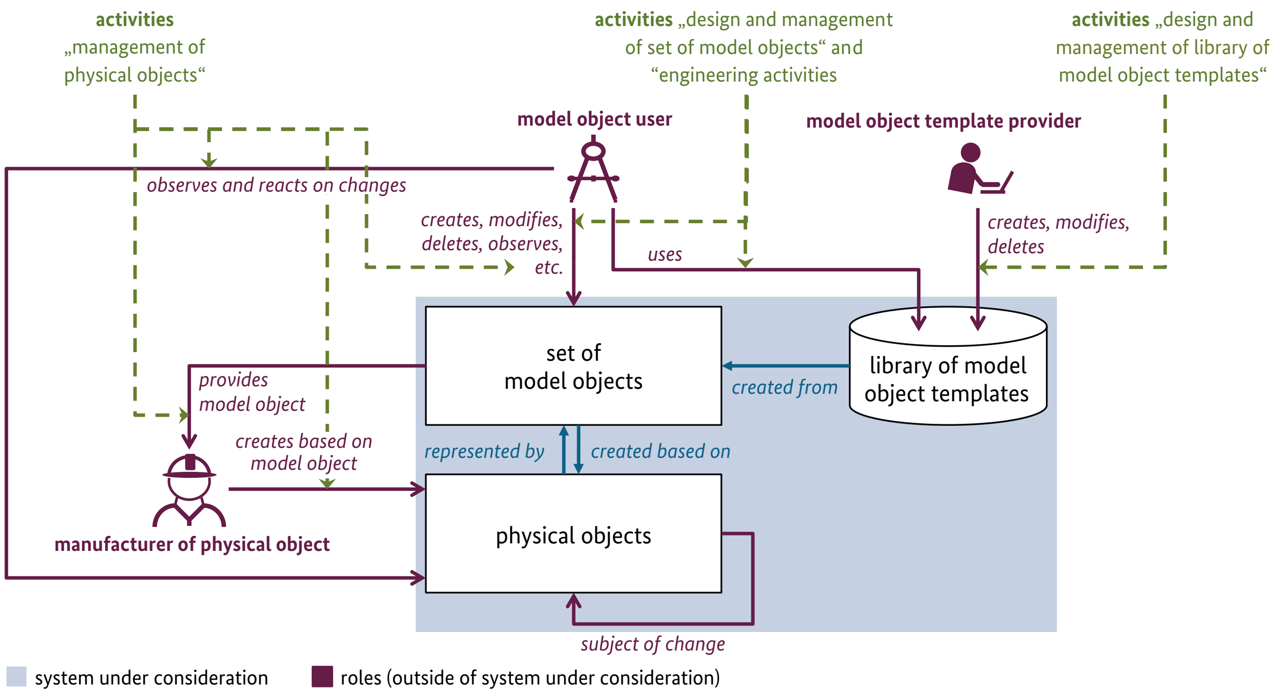
- A *model object template provider* manages a library of model object templates by creating, modifying and deleting model object templates. This role can be assumed by a human using a tool environment, but it can also be assumed by a software application providing appropriate capabilities.

- A *manufacturer of physical object* creates a physical object – often using a physical factory (or plant) – and makes it available. Typically, the process of creation (often called production or manufacturing) follows a model object. The role is typically assumed by a combination of human, tool environment and physical factory. In the case of executable software as physical object, the creation is typically a capability of a tool environment (often a so-called compiler, which creates based on a model object describing software some executable software).

It should be mentioned that a party can assume *different* roles, for example a party may assume the two roles model object user and model object template provider.

In addition, there is the role of the *operator of tool environment*, which we consider only casually for the reasons already mentioned. An operator of tool environment provides and operates a tool environment, where a tool environment may also include methods to create new model objects from existing model objects or model object templates or to set or change attributes.

Figure 10: Clusters of activities in connection with the system under consideration and roles


activities „management of physical objects“
activities „design and management of set of model objects“ and “engineering activities
activities „design and management of library of model object templates“
model object user
model object template provider
observes and reacts on changes
creates, modifies, deletes, observes, etc.
uses
creates, modifies, deletes
set of model objects
library of model object templates
created from
provides model object
creates based on model object
represented by
created based on
physical objects
manufacturer of physical object
subject of change
system under consideration
roles (outside of system under consideration)

Source: Plattform Industrie 4.0

4.4 Parties

A party is an agent executing tasks by assuming a role. Parties strongly depend on the concrete business setup of the application scenario “Seamless and Dynamic Engineering of Plants”. Therefore, we do not address the association of parties in this paper.

4.5 Activities

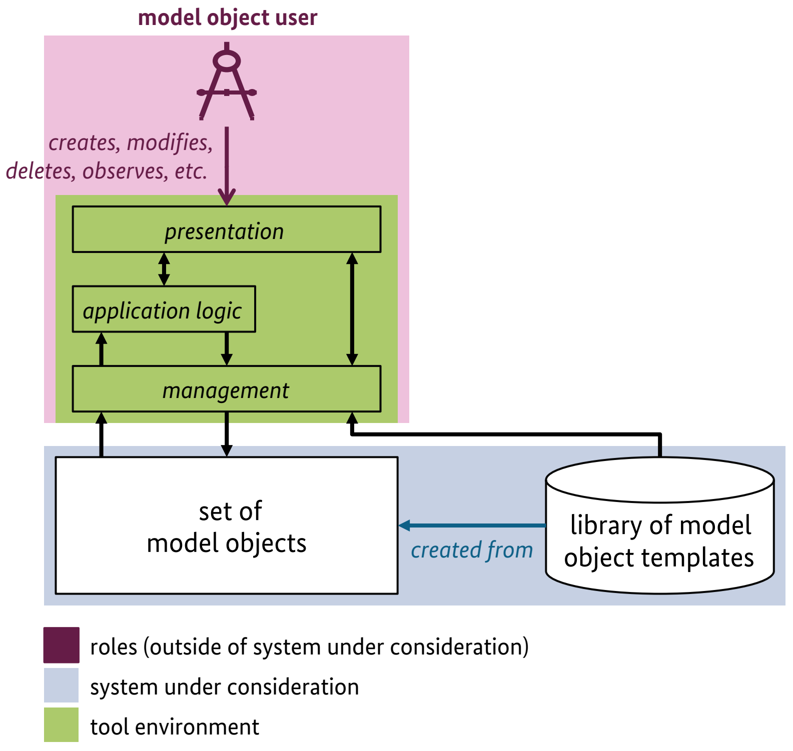
In this section, we describe the core activities as illustrated in Figure 10. We break down the various activities in the following different clusters:

- Design and management of set of model objects
- Design and management of library of model object templates
- Management of physical objects
- Engineering activities[11]
- We also added a separate cluster for the activities related to the management of a tool environment, where we sketch some activities with respect to the development and operation of a tool environment.

For the description we use the concept as proposed by the Industrial Internet Consortium, see Figure 4.

Since, as mentioned, the roles can be assumed by humans or by capabilities of a tool environment, the involved roles in the individual activities are not only shown by the icons according to Figure 9, but in a combination of humans (purple color) and capabilities of a tool environment (green color).

Figure 11: Common structure of “Design and management of set of model objects” activities


model object user
creates, modifies, deletes, observes, etc.
presentation
application logic
management
set of model objects
created from
library of model object templates
roles (outside of system under consideration)
system under consideration
tool environment

Source: Plattform Industrie 4.0

11 Engineering activities are sequences of creative, planning activities executed typically by engineers, where the results are typically documented by model objects. Examples for such creative, planning activities are the creation, modification, deletion of model objects or the observation of physical objects.

4.5.1 Design and management of set of model objects

The activities in this cluster follow a common overall structure as illustrated in Figure 11. Note, that also some activities of the cluster Engineering activities follow this structure.

The tool environment comprises the following capabilities, which follow a layered architectural approach:

- Presentation: A role *model object user* assumed by human interacts with the capabilities of the presentation layer of the tool environment.

- Application logic: This layer supports the role *model object user* assumed by humans, for example by notification, decision support, workflow automation, or bulk processing (automation of "non-creative" tasks). The application logic may also represent a role *model object user* assumed by a software application (automation of "creative" tasks).

- Management: This layer comprises capabilities like for example version management, user management, configuration management, management of life of model objects.

4.5.1.1 Activity "Creation of a model object"

Triggers: This activity will be initiated by role model object user.

Workflow
- Task 1 "Identification of the requirements to the model object": role model object user
- Task 2 "Creation of a model object": role model object user

Effects
- Beginning of the life of the model object
- Beginning of the consideration of impact of this new model object on its related model objects as well as the impact of already existing model objects, which are now related to the new model object
- Creator of the model object is the owner of the model object

Constraints: n.a.

Comments
- The requirements identified in Task 1 may be documented by some model object(s).
- Task 2 may be done by using an appropriate model object template: The relation between the created model object and the underlying model object template must be defined application-specific, e.g. no relation, created_from relation, inherits_from relation, etc.[12]
- Model objects can also be created remotely via import/data migrations capabilities of the tool environment (i.e. application logic of tool environment).
- Creating a model object requires the definition of various boundary conditions, for example the definition of model boundaries and limitations, the definition of the model depth, the definition of model interfaces, the definition of the model type like black-box, gray-box or white-box, the definition of model runtime like co-simulation, architecture of the solver or model exchange or the definition of access like visible or changeable attributes. This level of detail is not covered by the various activities.

4.5.1.2 Activity "Modification of a model object"

Triggers: This activity will be initiated by role model object user.

Workflow
- Task 1 "Modification of a model object": role model object user

Effects
- Creates a new entry in the life of the model object

Constraints
- Task 1 can be executed only by a party (i.e. stakeholder assuming the role model object user) having appropriate ownership rights
- Consequences of such a modification to related model objects should be considered

Comments
- Examples for such modifications are changes of attributes, changes of values of attributes, and changes with respect to related model objects

12 In the sense of structured modeling, an adequate reuse concept should be applied here.

- Examples from a usage perspective are for example that a physical object changed and therefore the associated model object will be changed also, or that a model object representing an implementation of a model object representing a role is replaced by another model object representing another implementation.
- This also includes automatically generated modifications through, for example, an automated engineering workflow. For more details see section 4.5.1.5 Activity “Reaction to a change in an associated model object”.

4.5.1.3 Activity “Refinement of a model object”

Triggers: This activity will be initiated by role model object user.

Workflow
- Task 1 “Refinement of a model by associating a model object representing a refinement of the model object”: role model object user

Effects: n.a.

Constraints: n.a.

Comments
- This activity is a special case of the activity “modification of a model object” because the association of additional model objects to a model objects is a modification of the model object. Nevertheless, we consider this as a separate activity, because here, from a usage perspective, creative steps are usually taken in the form of engineering decisions.
- An example from a usage perspective is, for example, the implementation of a requirement by a solution: The model object representing a requirement is associated with a model object representing a solution.
- It must be defined application-specific how modifications of a model object are handed over to an associated model object. Often, such changes require a close look at the context of the model objects and such a change can also have consequences on the context. In this respect, an adequate concept of seamlessly integrated modeling should be applied here.

4.5.1.4 Activity “Deleting a model object”

Triggers: This activity will be initiated by role model object user.

Workflow
- Task 1 “Deleting a model object”: role model object user

Effects
- Ends the life of the model object

Constraints
- Task 1 can be executed only by a party (i.e. stakeholder assuming the role model object user) having appropriate ownership rights
- Associations of other objects to the model object are now referenced to a model object in the “archive world”, which cannot be changed any longer. Especially, a model object of the “archive world” does not react on changes of associated model objects. Nevertheless, the model objects in the “archive world” exist and are accessible. Often, this is very important in terms of traceability and tamper prevention.

Comments
- It must be defined application-specific how deletion must be handed over to associated model objects. It may be that a model object can be deleted only if some preconditions are guaranteed, for example that there are no associations from other objects to this model object.

4.5.1.5 Activity “Reaction to a change in an associated model object”

This activity describes at a more detailed level than the other activities in this section the possibility of automating engineering workflows using an appropriate tool environment. It is restricted to changes in model objects, where a model object *X* is associated to a model object *Y* and there is some change of model object *Y* with implications on model object *X*, for example a model object Y describes requirements, which are implemented by a model object *X*. Changes of physical objects are addressed by the section 4.5.1.5 Activity “Reaction on physical change of a physical object”.

Triggers: This activity will be initiated by a tool environment based on the evaluation of the trigger conditions formulated by model object user

Workflow
- Task 1 "Controlling the process of changes to an associated model object that is automated by the tool environment": role model object user

Effects: n. a.

Constraints
- The reaction itself must be defined application-specific and can be executed by the owner of the model object only

Comments
- The model object user defines during the design of the model objects, how to react on changes. The tool environment has capabilities to automate the execution of these reactions.
- A reaction to a change may result in complex consequential changes. Thus, an adequate concept of seamlessly integrated modeling should be applied for this purpose.

4.5.2 Design and management of library of model object templates

The activities in this cluster follow a common overall structure as illustrated in Figure 12.

The tool environment comprises the following capabilities, which follow a layered architectural approach:

- Presentation: A role *model object template provider* assumed by human interacts with the capabilities of the presentation layer of the tool environment.

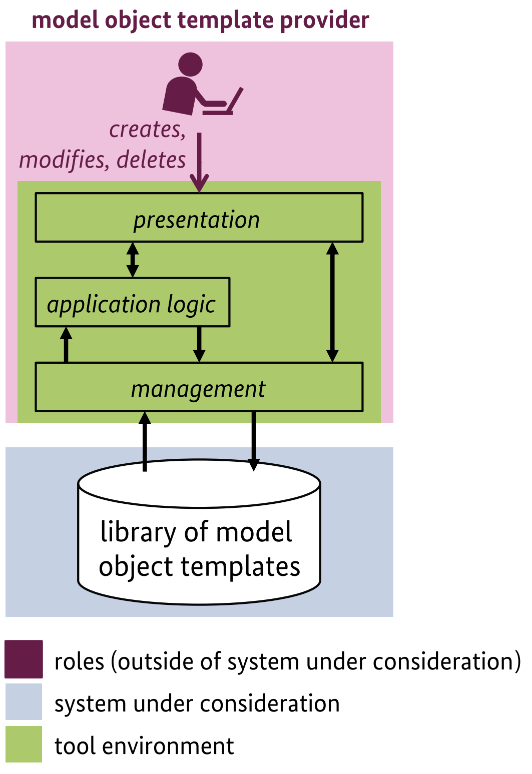
- Application logic: This layer supports the role *model object template provider* assumed by humans, for example by notification, decision support, or workflow automation (automation of "non-creative" tasks). The application logic may also represent a role *model object template provider* assumed by a software application (automation of "creative" tasks).

- Management: This layer comprises capabilities like for example version management, user management, configuration management, management of life of model object templates.

In practice there are different types of model object templates. Without any claim of completeness these are for example:

- A "white-box" sample document such as a concrete wiring diagram: this document is later copied by a model object user and the model object user then makes changes in the copy.

Figure 12: Common structure of "Design and management of library of model object templates" activities


model object template provider
creates, modifies, deletes
presentation
application logic
management
library of model object templates
roles (outside of system under consideration)
system under consideration
tool environment

Source: Plattform Industrie 4.0

- A “black-box” encapsulated function such as a function block: the model object user then later configures the function via externally accessible parameters but cannot change the original function.

- A product catalog such as a catalog for drives: based on the requirements the model object user will select and use a suitable element from this catalog.

4.5.2.1 Activity “Creation of a model object template”

Triggers: This activity will be initiated by role model object template provider and usually requires a decision from a business perspective.

Workflow
- Task 1 “Creation of a model object template”: role model object template provider
- Task 2 “Quality control[13] of the model object template”: role model object template provider
- Task 3 [optional] “Assignment of the model object template to libraries”: role model object template provider
- Task 4 [optional] “Integration of the model object template within assigned libraries, e.g. documentation, release, launch, etc.”: role model object template provider

Effects
- Beginning of the life of the model object template
- Creator of the model object template is the owner of the model object template

Constraints: n.a.

Comments
- Adequate concepts and strategies of reuse should be applied here.
- Model object templates could also be provided by external vendors.

4.5.2.2 Activity “Modification of a model object template”

Triggers: This activity will be initiated by role model object template provider.

Workflow
- Task 1 “Modification of a model object template”: role model object template provider

Effects
- Creates a new entry in the life of the model object template
- Depending on the relationship between model objects and model object templates, this may result in modifications of associated model objects

Constraints
- Task 1 can be executed only by a party (i.e. stakeholder assuming the role model object template provider) having appropriate ownership rights
- Consequences of such a modification to related model object templates should be considered

Comments:
- There are many reasons for modifying a model object template, such as fixing a bug in a model object template, conceptual advancement of a model object template, or content wise extension of a model object template.
- Modifications can be the modification of parameters, the modification of associations between model object templates or the modification with respect to an assignment of a model object template to a library of model object templates.
- If there exist relations to model objects, which were created based on the model object template, it must be defined application-specific, whether and how the associated model objects must be modified. An example is that a model object is modified according to the modifications of the model object template. Another example is that a model object is not modified, thus, the model object remains associated to an “older” entry in the life of the model object template.
- The release process for changes to model object templates must be carefully considered. This is particularly the case if a change in a model object template results in changes to the model objects that were created based on the model object template.

13 A model object template can later be used in different contexts. This must be ensured after development of the model object template by means of suitable quality control measures, for example by testing.

4.5.2.3 Activity "Deleting a model object template"

Triggers: This activity will be initiated by role model object template provider.

Workflow

- Task 1 "Deleting a model object template": role model object template provider

Effects

- Ends the life of the model object template
- Depending on the relationship between model objects and model object templates this may result in modifications or even deletions of associated model objects
- If the model object template is assigned to some library of model object templates this assignment will be deleted

Constraints

- Task 1 can be executed only by a party (i.e. stakeholder assuming the role model object template provider) having appropriate ownership rights
- Associations of other model object templates to the model object template are now referenced to a model object in the "archive world", which cannot be changed any longer. Nevertheless, the model objects templates in the "archive world" still exist and are accessible. Often this is very important in terms of traceability and tamper prevention.

Comments

- If there exist associations to model objects, which were created based on the model object template, it must be defined application-specific, whether and how the associated model objects must be modified. An example is that a model object has no longer an association to a model object template. Another example is that a model object has an association to the final entry in the life of the model object template.
- It must be defined application-specific how deletion must be handed over to associated model objects templates. It may be that a model object template can be deleted only if some preconditions are guaranteed, for example that there are no associations from other model object templates to this model object template.

4.5.2.4 Activity "Creating a model object template based on a model object"

Triggers: This activity will be initiated by role model object template provider.

Workflow

- Task 1 "Identification of a model object": role model object template provider
- Task 2 "Identification of common characteristics of the intended model object template based on the model object": role model object template provider
- Task 3 "Transfer of the model object into a model object template": role model object template provider

Effects

- Beginning of the life of the model object template
- Creator of the model object template is the owner of the model object template

Constraints: n.a.

Comments

- Common characteristic could be, for example, parameters, relations, associated objects, or associations to libraries.
- It would be possible to establish an association from the original model object to the created model object template in order to modify the created model object template in case of changes to the original model object. Such modifications have to be defined application specific.

4.5.2.5 Activity "Creation of an empty library of model object templates"

Triggers: This activity will be initiated by role model object template provider and usually requires a decision from a business perspective.

Workflow

- Task 1 "Creation of an empty library of model object templates": role model object template provider

Effects

- Beginning of the life of the library of model object templates
- Creator of the library of model object templates is the owner of the library of model object templates

Constraints: n.a.

Comments: n.a.

4.5.2.6 Activity "Deleting a library of model object templates"

Triggers: This activity will be initiated by role model object template provider.

Workflow
- Task 1 "Deleting a library of model object templates": role model object template provider

Effects
- Ends the life of the library of model object templates
- Model object templates assigned to the library of model object templates will no longer be assigned to the library of model object templates, but the model object templates will not be deleted

Constraints
- Task 1 can be executed only by a party (i.e. stakeholder assuming the role model object template provider) having appropriate ownership rights
- The library of model object templates (as an object of the model world) still exists and is accessible in the "archive world". Often this is very important in terms of traceability and tamper prevention.

Comments: n.a.

4.5.3 Management of physical objects

The activity "Creation of a physical object" follows a structure as illustrated in Figure 13.

The tool environment comprises the following capabilities, which follow a layered architectural approach:

- Presentation: A role *model object user* assumed by human interacts with the presentation capabilities of the presentation layer of the tool environment to select and provide an appropriate model object.

- Application logic: This layer supports the role *model object user* assumed by humans in the "configuration" of the model object to be provided. The application logic also provides capabilities to create executable software based on a model object (compile and deploy). Such physical objects only change by changing the corresponding model object and initiating a new creation. Note that we do not consider hardware failures.

- Management: This layer comprises capabilities like for example version management, user management, configuration management, management of life of model object.

4.5.3.1 Activity "Creation of a physical object"

Triggers: This activity will be initiated by role model object user.

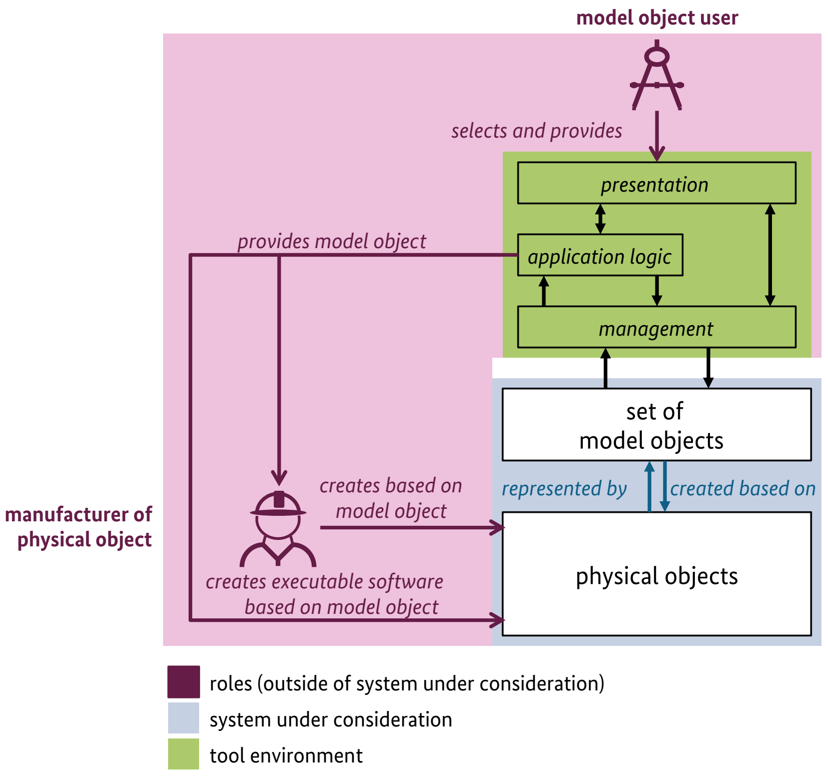
Workflow
- Task 1 "Identification of a model object (resp. set of model objects) using the capabilities of the tool environment and provision of the model object via the tool environment": role model object user
- Task 2 "Creation of physical object based on the provided model object, either using a physical factory to produce the physical object or using capabilities of a tool environment (compiler) to create executable software": role manufacturer of physical object

Effects
- Beginning of the life of the physical object in the understanding that the life can be accessed in the model world by associated model objects.

Constraints: n.a.

Comments
- Such a model object may describe requirements, assertions and (technical) solutions.
- The creation of a physical object using a physical factory may be for example by production in a factory or by a process request, engineering and erection of a plant or by production of raw materials
- In case of executable software, there is a well-defined association between the model object and the created physical object.
- The beginning of the life of a physical object must be defined application specific. This may be for example after completion of manufacturing, after delivery to customer, or after commissioning by customer.

Figure 13: Structure of "Creation of a physical object" activity


model object user
selects and provides
presentation
application logic
management
provides model object
set of model objects
represented by
created based on
manufacturer of physical object
creates based on model object
physical objects
creates executable software based on model object
roles (outside of system under consideration)
system under consideration
tool environment

Source: Plattform Industrie 4.0

The other activities in this cluster follow a common overall structure as illustrated in Figure 14. Physical objects are permanently subject of change (e.g. deforming, wearing out, rotting, etc.). Some of these changes may be observed by observing capabilities of a tool environment, but some changes must be observed by humans in an explicit way and in case of a change the human has to react accordingly. Note, that executable software is not subject of change of this kind, because we do not consider hardware failures.

The tool environment comprises the following capabilities, which follow a layered architectural approach:

- Presentation: A role *model object user* assumed by human interacts with the capabilities of the presentation layer of the tool environment.
- Application logic: This layer supports the role *model object user* assumed by humans, for example by notification, decision support, workflow automation, or bulk processing (automation of "non-creative" tasks). The application logic may also represent a role *model object user* assumed by a software application (automation of "creative" tasks[14]).
- Observing: Some physical objects or a tool environment may have the capability to perceive specific physical changes. Thus, the role *model object user* assumed by a human can be supported as an observer or even the modification of the model objects can be performed automatically by appropriate capabilities of the tool environment. For some physical objects, it will be the responsibility of the role *model object user* assumed by a human to realize such changes and then react in a suitable way.
- Management: This layer comprises capabilities like for example version management, user management, configuration management, management of life of model objects.

14 This includes applications, where it is postulated that in the future activities which are based on the crea-tive invention of humans will be successively supported or even replaced by methods of artificial intelligence.

4.5.3.2 Activity "Creation of a descriptive model of a physical object"

This activity addresses the creation of a model object that serves as a descriptive model for a physical object.

Triggers: This activity will be initiated by role model object user.

Workflow

- Task 1 "Identification of scope of physical object": role model object user
- Task 2 "Selection of an existing model object or creation of a new model object acting as a descriptive model of the identified physical object": role model object user

Effects

- There will be an associated model object (descriptive model) to a physical object

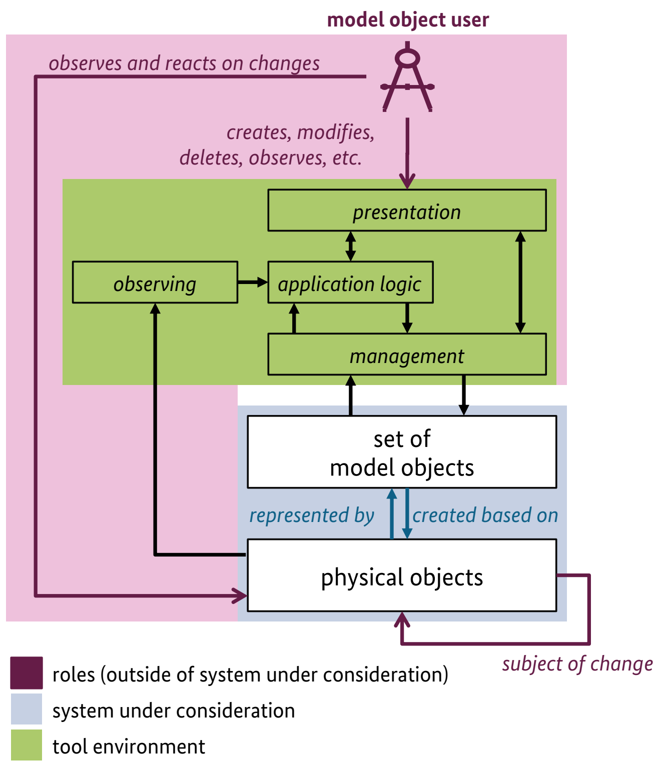
Constraints

- The descriptive model describes the real physical object (as good as necessary in the context of a specific application)
- There may be several model objects associated to the same physical object

Comments

- If in the context of the activity a new model object is created, see Task 2. This is a special case of the creation of a model object, see section 4.5.1.1 Activity "Creation of a model object"
- In case of executable software there is no need to create a descriptive model in an explicit way
- It is a design decision driven essentially by the purpose of the descriptive model which concrete content a descriptive model should contain
- Various examples for illustration are described in chapter "5 Examples for Illustration"

Figure 14: Common structure of other "Management of physical objects" activities



Source: Plattform Industrie 4.0

- There may be model objects, which are not descriptive models of a physical object, for example the description of a requirement, a constraint, a rule or a mathematical formula

4.5.3.3 Activity "Reaction on physical change of a physical object"

This activity addresses the situation that some physical object has changed (for whatever reason) and consequently model objects associated to the physical object should be adopted accordingly now.

Triggers: This activity will be initiated by role model object user. The party assuming the role model object user is responsible to decide that because of a change of a physical object the associated model objects must be adapted.

Workflow
- Task 1 "Identification of model objects being subject of change because of a physical change of a physical object": role model object user
- Task 2 "Modification of identified model objects": role model object user

Effects
- Creates a new entry in the life of the model objects associated to a physical object

Constraints
- Task 2 can be executed only by a party (i.e. stakeholder assuming the role model object user) having appropriate ownership rights

Comments
- Through this activity, the physical changes in the physical world are made accessible to the model world through associated model objects.
- This is a special case of the modification of a model object, see section 4.5.1.2 Activity "Modification of a model object".
- It must be defined application-specific how to deal with the associated descriptive models, especially the trigger. From the perspective of the model object, this can be "compared" to the reaction to changes of associated model objects.
- Tracking changes in the physical world in associated model objects typically requires a powerful tool environment.

4.5.3.4 Activity "Destroying/scrapping/dismantling of a physical object"

Triggers: Changes of physical objects may result in a destroying, scrapping or dismantling of a physical object, but destroying, scrapping or dismantling may also be initiated by role model object user. Nevertheless, it is in the responsibility of a role model object user to trigger this activity in an explicit way.

Workflow
- Task 1 "Identification of model objects being subject of change because of scrapping/dismantling of a physical object": role model object user
- Task 2 "Modification of identified model objects": role model object user

Effects
- Terminates the life of the physical object in the understanding that this life cannot be modified in the model world by associated model objects any longer.

Constraints
- The reason for the destroying, scrapping or dismantling of a physical object is out of scope. The focus of this document is on the effects to associated model objects only.
- Task 2 can be executed only by a party (i.e. stakeholder assuming the role model object user) having appropriate ownership rights.

Comments
- Dealing with the associated descriptive models of a destroyed/scrapped/dismantled physical object must be defined application-specific. From the perspective of the model object, this is "comparable" with the reaction to changes of associated model objects
- Even if associated physical objects do not exist any longer, there may be applications, where the model object still must be managed, for example for regulatory reasons or good manufacturing practice (GMP).

4.5.4 Engineering activities

The activities "Transfer a model object from a model user to another model user" and "Planning activity" in this cluster follow a common overall structure as illustrated already in Figure 11. Note that these activities can also be applied to model object templates accordingly.

4.5.4.1 Activity “Transfer a model object from a model user to another model user”

Triggers: This activity will be initiated by role model object user.

Workflow
- Task 1 “Specification of the terms of use for the transferred model in a contract between the original owner and the future owner of the model object”: role model object user (parties acting as current and future owner of the model object)
- Task 2 “Provide a copy of a model object from a model user (owner of the model object) to another model user (future owner of the model object)”: role model object user (parties acting as current and future owner of the model object)

Effects
- Transferred model object is an independent model object with an own life
- Beginning of the life of the transferred model object
- The original model object also further exists and belongs to the original owner

Constraints: n. a.

Comments
- By transferring a model object, the owner of the model object may remove selected attributes, values of attributes or relations, so that for the future owner of the transferred “copy” of the model object not all information of the original owner of the model object is available.
- This activity not only applies to model objects, but also to model object templates.
- Such a transfer requires a common understanding of the content of the transferred model object and can be very complex in a specific case. In addition to the model object, model object templates or libraries of model object templates may also have to be transferred.
- In this respect, also an adequate concept of seamlessly integrated modeling should be applied here.

4.5.4.2 Activity “Planning activity”

Triggers: This activity will be initiated by role model object user.

Workflow
- Task 1 “Copying a model object (the owner of the copy is the same as the owner of the origin), where ‘copy’ will be represented by a specific relation between a model object and its copy”: role model object user
- Task 2 “Applying changes to the copy of the model object, see activity modification of a model object”: role model object user
- Task 3 “Completion of planning activity”: role model object user
- Task 4 “The existing copy of the model object is either discarded or transferred to the original model object”: role model object user

Effects
- ‘Copy’ of a model object has its own life, the end of life is defined by discarding or transferring to the original model object

Constraints: n. a.

Comments
- It must be defined application-specific how to transfer the planning into the original model object. There may arise contradictions, which must be resolved by this transfer.
- It must be defined application-specific how changes of the original model object influence the copy of the model object. This is a specific case of a reaction to a change in an associated model object, see section 4.5.1.5 Activity “Reaction to a change in an associated model object”
- In general, this may lead to complex relationships between model objects.
- The concrete execution highly depends on the capabilities of the tool environment.
- This activity is also applicable when starting from scratch and no model object is available, but in this case an initial model object must be created and there is no need for a copy.
- In this respect, also adequate concepts of seamlessly integrated modeling as well as reuse should be applied here.

4.5.4.3 Activity “Evaluation/simulation of a model object”

The activity “Evaluation/simulation of a model object” follows a structure as illustrated in Figure 15.

The tool environment comprises the following capabilities, which follow a layered architectural approach:
- Presentation: A role *model object user* assumed by human interacts with the capabilities of the presentation layer of the tool environment.
- Evaluation/simulation: This layer provides capabilities to evaluate respectively simulate model objects.
- Actuating & sensing: This layer provides capabilities to couple a simulator to physical objects.
- Management: This layer comprises capabilities like for example version management, user management, configuration management, management of life of model objects.

Triggers: This activity will be initiated by role model object user.

Workflow
- Task 1 “Definition of purpose of evaluation/simulation of a model object”: role model object user
- Task 2 “Definition of evaluation/test cases as well as evaluation/test variables”: role model object user
- Task 3 “Evaluation/simulation of a model object (validate/verify/predict/etc.)”: role model object user

Effects
- The result of this activity is typically no model object, but the results can be stored as values of attributes of the model object(s)

Constraints: n.a.

Comments
- The evaluation/simulation can be coupled with physical objects, if the physical objects provide data to the tool environment or if the tool environment can influence the physical objects. This requires specific capabilities of the tool environment, for example realtime communication.
- Verification and validation results significantly depend on the quality of the model object specification. Test cases as well as test requirements are often rather difficult to gain.

4.5.5 Management of tool environment

4.5.5.1 Activity “Development of a tool environment”

Triggers: This activity will be initiated by role operator of tool environment.

Workflow
- Task 1 “Identifying the business targets and requirements for tool environment”: role operator of tool environment
- Task 2 “Selecting and buying tools from various suppliers of tools”: role operator of tool environment
- Task 3 “Customizing and integrating the tools to an integrated tool environment”: role operator of tool environment
- Task 4 “Migration of existing engineering data to new tool environment”: role operator of tool environment
- Task 5 “Training and coaching of users (i.e. role model object user resp. role model object template provider) in using the new tool environment”: role operator of tool environment

Effects
- Provision of ready to use tool environment

Constraints: n.a.

Comments
- In practice there are often limitations with respect to the migration of legacy data and integration due to missing openness of specific tools and applications
- An integrated tool and application environment is complex and often requires a considerable upfront investment
- The functionality and usability requirements of a tool environment will continue to increase due to the increasing complexity of the models

4.5.5.2 Activity “Operation of a tool environment”

Triggers: This is a continuous activity of role operator of tool environment

Workflow
- Task 1 “Creation of a lifecycle model for the tool environment including a baseline management based on the capabilities and roadmaps of the various tools and applications involved and the requirements of the intended

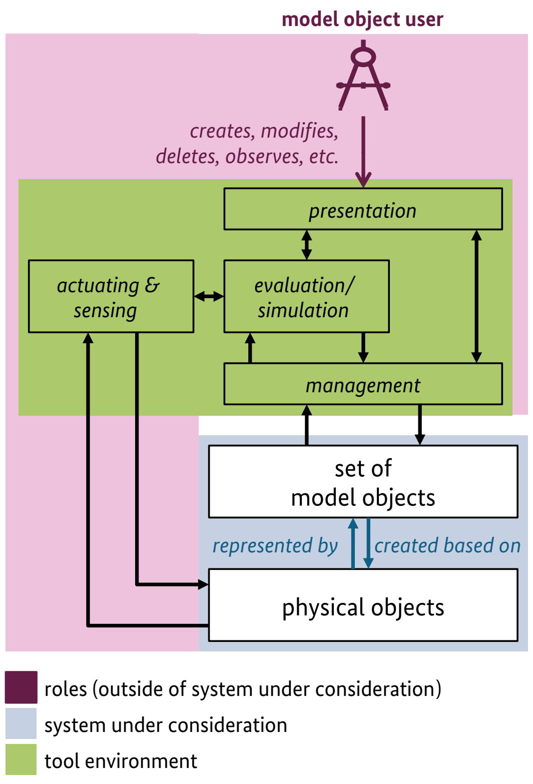
users (i.e. role model object user resp. role model object template provider) according to defined service level agreements": role operator of tool environment
- Task 2 "Provision of functionality of tool environment to users (i.e. role model object user resp. role model object template provider) according to defined service level agreements": role operator of tool environment
- Task 3 "Management of usage of tool environment, e.g. user management, engineering data backup, migration to new releases of tools (including necessary migration of engineering data), integration of new functionalities resp. tools": role operator of tool environment
- Task 4 "Systematic collection of lessons learned and documentation of best practice workflows": role operator of tool environment

Effects: n.a.

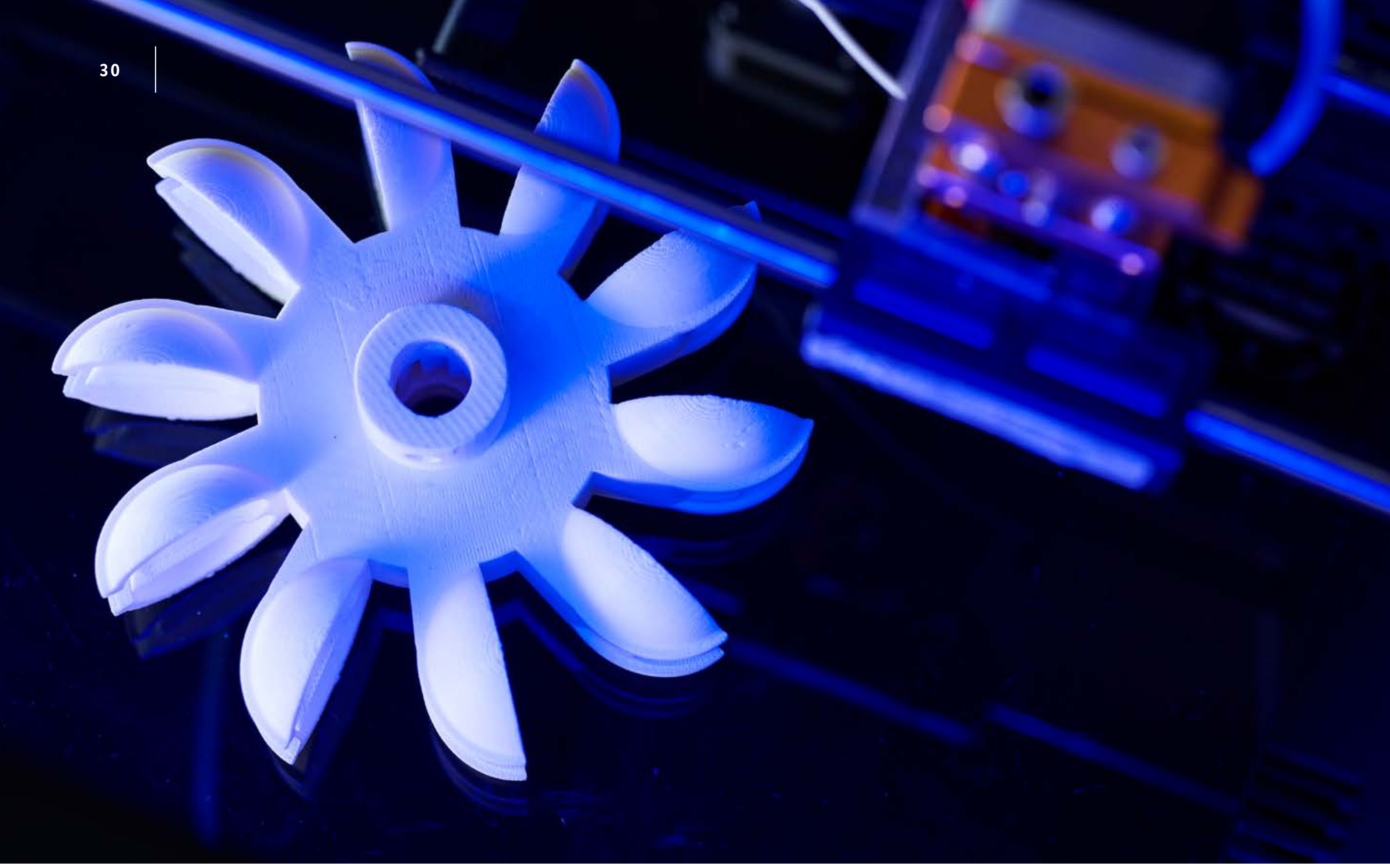
Constraints
- Consequences of fundamental changes done by a vendor of an individual tool resp. application should be considered in the whole tool environment

Comments
- This includes also the update of an existing tool environment.
- The systematic collection of lessons learned includes for example the enrichment of libraries, the generation of project templates, the provision of data and methods for design and analysis tasks (for example cost calculation of projects) or the provision of best practice for modules and modularization.
- The functionality and usability requirements of a tool environment will continue to increase due to the increasing complexity of the models.

Figure 15: Structure of "Evaluation/simulation of a model object" activity



Source: Plattform Industrie 4.0



5 Examples for Illustration

The description of the usage view "Seamless and Dynamic Engineering of Plants", as provided in the chapter 4

Usage View, is generic, but rather abstract. Therefore, the examples in this chapter are intended to illustrate how these basic concepts can be applied. By having considered these various examples, a certain representativeness of the list of activities can be postulated. However, it should be noted that these examples are neither intended to illustrate the content structure of an integrating plant model nor sufficiently consider the real complexity in practice. It should also become clear that the usage view not only addresses the aspect of an integrating plant model but can also be applied in product development and production to build an integrating product model.[15]

5.1 Example: Installation and replacement of a physical pump

In this example, we consider a physical pump that is installed in a plant and later replaced by another physical pump, and in addition, various model objects and model object templates designed for that purpose. In this example we also consider – in contrast to the other examples – the relation to the roles according to section 4.3 Roles.

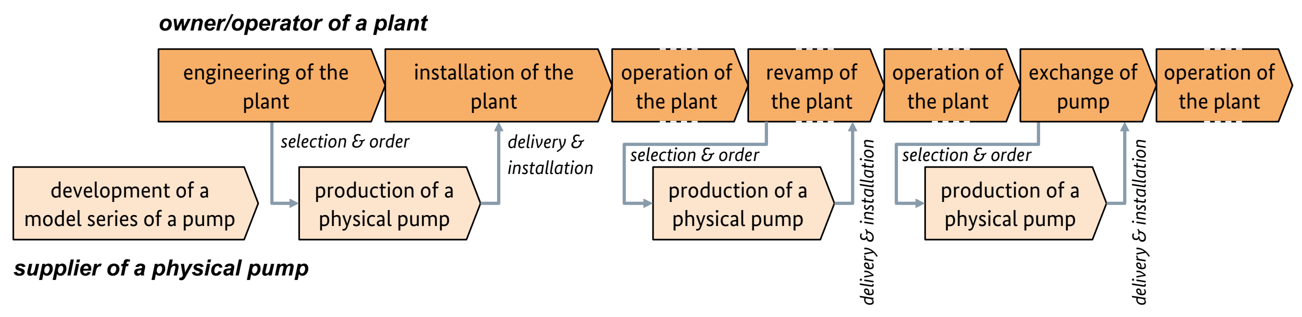
We look at two different companies, see Figure 16:

- On the one hand, the supplier of a physical pump, who develops a model series of a pump model (as part of its development activities) and, on request of a customer, produces and delivers a corresponding physical pump to the customer. Within this company the roles *model object user*, *model object template provider* and *manufacturer of physical object* are implemented.

- On the other hand, the owner/operator of a plant, who first designs a plant, in which such a pump will be installed, then orders a corresponding pump and physically installs this pump in the plant. Later the owner/operator of a plant replaces this physical pump by another pump, for example, because the operating conditions have changed and the original pump cannot exe-

15 In this respect, the application scenario "Smart Product Development for Smart Production" is also reflected in the usage view "Seamless and Dynamic Engineering of Plants". The close relationship between these two application scenarios from a technical perspective has already been pointed out in [8].

Figure 16: Companies involved in the example



Source: Plattform Industrie 4.0

cute the intended task any longer or, for example, that the pump must be replaced due to a defect by another equivalent pump. Within this company also the roles *model object user*, *model object template provider* and *manufacturer of physical* object are implemented.

Related to this context, especially the following processes should be considered:

- The development department of the supplier of a physical pump assumes the role *model object user* and will create a model object *pump_model_series* according to 4.5.1.1 Activity “Creation of a model object” and 4.5.4.2 Activity “Planning activity”. This model object describes all development and production artifacts for this model series of a pump. This model object may be associated with various model object templates and physical objects (for example physical prototypes of the pump), it may include requirements, assurances, or technical solutions and may have been created with various engineering tools. But all these aspects will not be considered in more detail in this example.
  In particular, the model object *pump_model_series* will comprise a product catalog *product_catalogue*. It is a model object template that describes the assurances that a concrete physical pump will satisfy. It is provided by the development department assuming the role *model object template provider*.

- The engineering department of the owner/operator of a plant assumes the role *model object user* and will create a piping & instrumentation diagram using some engineering tool. Typically, the engineering tool will provide in the form of a library a set of symbolic elements from which elements in the piping & instrumentation diagram can be created via a “copy & modify” or “instantiate” procedure. The library provided by the engineering tool is a library of model object templates and the generated piping & instrumentation diagram is a model object. This piping & instrumentation diagram will comprise a model object *req_pump* describing the requirements that some pump, which should later implement these requirements in the physical plant, must satisfy. At some time, the engineering department will select and order a specific pump. For this purpose, the product catalog *product_catalogue* is made available by the supplier of a pump according to 4.5.4.1 Activity “Transfer a model object from a model user to another model user” and from this catalogue a suitable pump satisfying the requirements is ordered. With the order, the engineering department creates a model object *virtual_pump0* according to 4.5.1.1 Activity “Creation of a model object”, which is associated to the model object *req_pump* according to 4.5.1.3 Activity “Refinement of a model object” and which is made available to the supplier of the pump according to 4.5.4.1 Activity “Transfer a model object from a model user to another model user” in the form of a model object *virtual_pump0_supplier*.

- The supplier of a pump will assume the role *manufacturer of physical object* and based on the model object *virtual_pump0_supplier* and with reference to the model object *pump_model_series* the supplier of a pump will manufacture a physical pump *physical_pump0* according to 4.5.1.1 Activity “Creation of a physical object”. In doing so, the supplier (assuming the role *model object*

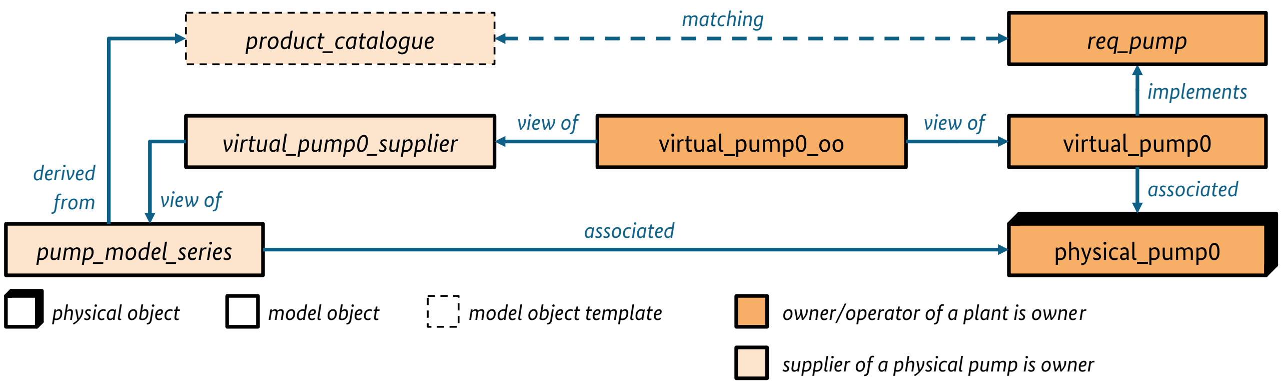
user) will enrich the model object *virtual_pump0_supplier* according to 4.5.1.2 Activity "Modification of a model object" based on information relevant to him resulting from the manufacturing of the physical pump *physical_pump0* according to 4.5.3.1 Activity "Creation of a physical object". In this context, according to 4.5.3.2 Activity "Creation of a descriptive model of a physical object" the model object *virtual_pump0_supplier* is associated with the physical object *physical_pump0*.
The supplier of a pump (assuming the role *manufacturer of physical object*) will deliver the physical pump *physical_pump0* to the owner/operator of a plant and will also provide (assuming the role *model object user*) parts of the model object *virtual_pump0_supplier* according to 4.5.4.1 Activity "Transfer a model object from a model user to another model user" in the form of a model objects *virtual_pump0_oo*.

- The owner/operator of a plant (assuming the role *manufacturer of physical object*) will install the physical pump *physical_pump0* in the plant and will modify (assuming the role *model object user*) the model object *virtual_pump0* based on the model object *virtual_pump0_oo* according to 4.5.1.2 Activity "Modification of a model object" and will associate the physical pump *physical_pump0* with the model object *virtual_pump0* according to 4.5.3.2 Activity "Creation of a descriptive model of a physical object".

- After a certain period of operating the physical pump *physical_pump0* the pump's operating conditions may have changed: In this case, the model object *req_pump* is changed according to 4.5.1.2 Activity "Modification of a model object" by the engineering department of the owner/operator of a plant assuming the role *model object user* so that *req_pump* now describes the new requirements. If the physical pump *physical_pump0* cannot meet these new requirements any longer, a new physical pump must be selected according to a model object *virtual_pump1*[16] and ordered from the supplier of a pump. The supplier of pump will then provide to the owner/operator of a plant a physical pump *physical_pump1* including a model object *virtual_pump1_oo*. The association between *req_pump* and *virtual_pump0* is deleted according to 4.5.1.2 Activity "Modification of a model object" and instead *req_pump* is associated with the model object *virtual_pump1* according to 4.5.1.2 Activity "Modification of a model object". Usually, the physical pump *physical_pump0*, including the associated model object *virtual_pump0*, will continue to exist because, maybe it is, for example, (temporarily) in a spare parts warehouse.

- And after a another certain period of operating the new physical pump *physical_pump1* must be replaced due to a defect: Because of the defect changes of model objects

Figure 17: Overview of main physical objects, model objects and model object templates



Source: Plattform Industrie 4.0

16 Typically, *virtual_pump1* is a new model object with an own life and not a modification of *virtual_pump0*, because usually the physical pump *physical_pump0*, including the associated model object *virtual_pump0*, will continue to exist, but other ways of modeling may also be conceivable.

are necessary according to 4.5.3.3 Activity “Reaction on physical change of a physical object”, which is executed by the engineering department of the owner/operator of a plant assuming the role *model object user*. If corresponding relationships were explicitly modeled during the engineering of the plant, some of the following activities can also be carried out automatically according to 4.5.1.5 Activity “Reaction to a change in an associated model object”. In this case, due to the unchanged requirements, a new pump is ordered from the supplier of a pump. Perhaps even a pump of the same design is no longer available, so that a renewed selection of a suitable pump is necessary. This means that according to 4.5.1.2 Activity “Modification of a model object” the association between *req_pump* and *virtual_pump1* is deleted and the model object *virtual_pump2* is created and associated to the model object *req_pump* according to 4.5.1.1 Activity “Creation of a model object”. It is assumed that due to the defect the physical pump *physical_pump1* is scrapped according to 4.5.3.4 Activity “Destroying/scrapping/dismantling of a physical object” and therefore does not exist any longer and the model object *virtual_pump1* is transferred to the “archive world”.

Figure 17 summarizes the main physical objects, model objects, model object templates and relationships[17] of this example.

5.2 Example: Virtual commissioning

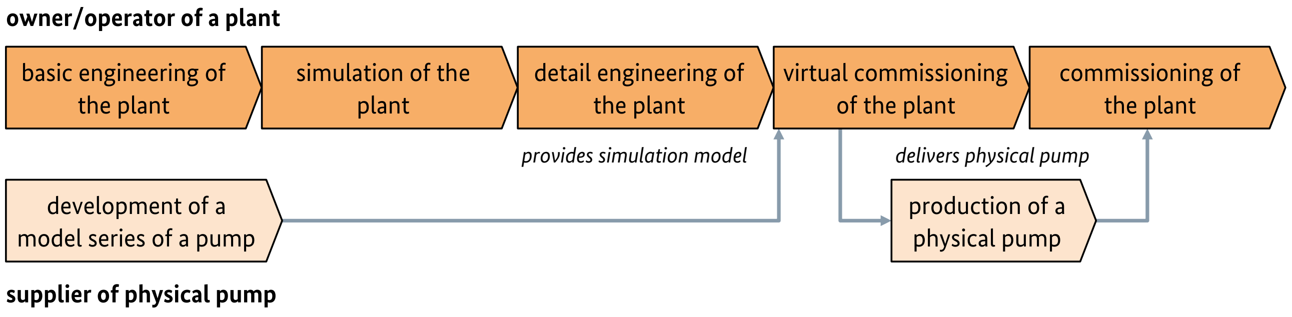
This example is an extension of the example “Installation and replacement of a physical pump”. We consider the same companies in the value network, namely the supplier of a physical pump and the owner/operator of a plant, see Figure 18, as well as the same central object, namely the pump. In this example, we assume that besides the physical pump, the supplier of a physical pump also provides to the customer a simulation model of the pump. Using this simulation model, the owner/operator of a plant can verify early certain design decisions during the engineering process of the plant with respect to the selected pump.

Virtual commissioning is the testing of application software of a process control system without physical connection to the plant, whereas during commissioning the application software of a process control system is tested with a physical connection to the plant.

Related to this context, especially the following processes should be considered:

- In the second bullet point of the example “Installation and replacement of a physical pump” above, the piping & instrumentation diagram is mentioned. This diagram comprises model objects, which represent all important

Figure 18: Companies involved in the example


owner/operator of a plant
basic engineering of the plant
simulation of the plant
detail engineering of the plant
virtual commissioning of the plant
commissioning of the plant
provides simulation model
delivers physical pump
development of a model series of a pump
production of a physical pump
supplier of physical pump

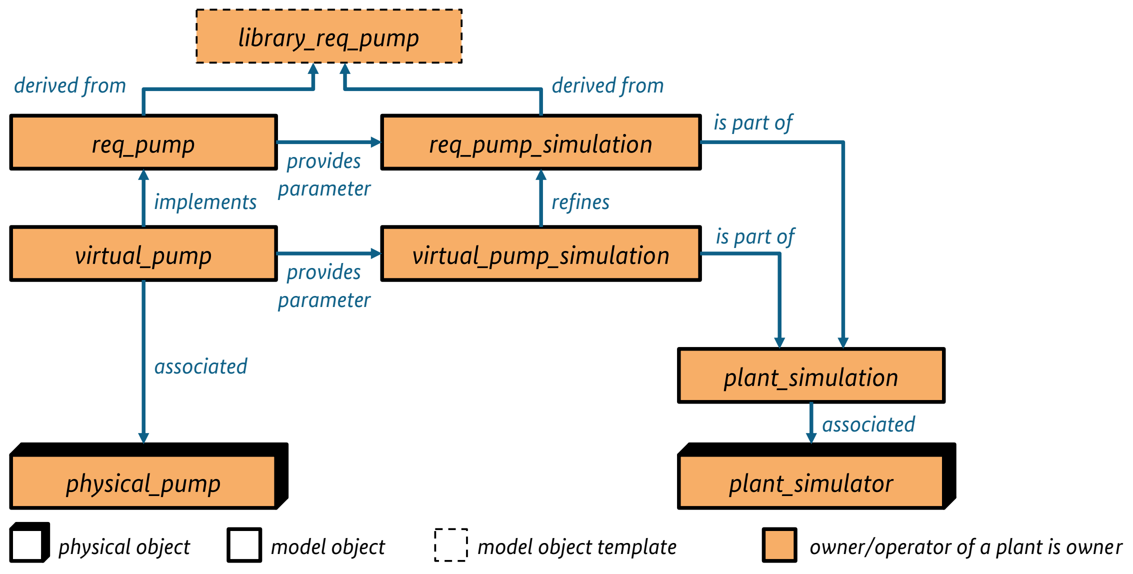
Source: Plattform Industrie 4.0

17 The relation “view of“ means that an object contains only selected aspects of another object. Nevertheless, the annotation of the relations serves as an illustration only and it is not intended to define them in a formal way.

physical objects of the later plant, for example the model object *req_pump* describing the requirements that some pump, which should later implement these requirements in the physical plant, must satisfy.
For each model object in the piping & instrumentation diagram a corresponding model object can be derived, which is described in a simulation language. These two model objects can, for example, linked together by a common model object template of some library of model object templates. This library of model object templates is typically provided by the provider of the engineering tool used by the owner/operator of the plant for the engineering of the plant Thus, in this example, there is a model object template *library_req_pump*, from which the two model objects *req_pump* and *req_pump_simulation* are derived. These two model objects must be appropriately parameterized and correlated.

- The aggregation of all these simulation model objects according to the piping & instrumentation diagram is a model object *plant_simulation* and following the 4.5.4.3 Activity "Evaluation/simulation of a model object" this model object *plant_simulation* can be simulated using a suitable simulation tool environment. The simulation based on the simulation tool environment configured by the model object *plant_simulation* is a physical object *plant_simulator*, which is then physically connected to the process control system of the plant – as physical object. By execution of *plant_simulator* connected to the process control system different simulations can be performed. These executions can reveal whether the code of the process control system of the plant has been designed properly (virtual commissioning).
  In addition to the physical connection of *plant_simulator* and the process control system of the plant, this physical communication must also be logically parameterized based on so-called tags. Tags are model objects that represent the physical input and output signals of the process control system. These tags have a correspondence in the individual model objects aggregated in the model object *plant_simulation*. For example, the model object *req_pump_simulation* includes representations of the input and output signals of the underlying pump. The tags of the process control system and their representatives in the model objects being the basis for *plant_simulator* must therefore be synchronized suitably.

Figure 19: Overview of main physical objects, model objects and model object templates



Source: Plattform Industrie 4.0

- Once a suitable pump *virtual_pump* satisfying the requirements according to *req_pump* has been selected from the product catalogue of the supplier of the pump, the model object *req_pump_simulation* can be replaced by a model object *virtual_pump_simulation* following the 4.5.4.1 Activity "Transfer a model object from a model user to another model user" and the 4.5.1.2 Activity "Modification of a model object". The model object *virtual_pump_simulation* is used to refine the model object *plant_simulation*. Note that the model object *virtual_pump_simulation* is also a refinement of the model object *req_pump_simulation*.

- Once physical objects like the pump *physical_pump* have been delivered by the various suppliers, the model objects like *virtual_pump_simulation* can be excluded[18] from the model object *plant_simulation*. Instead, the physical objects like the physical pump *physical_ pump* are connected to the process control system of the plant and the complete functionality of the plant can be tested (commissioning).

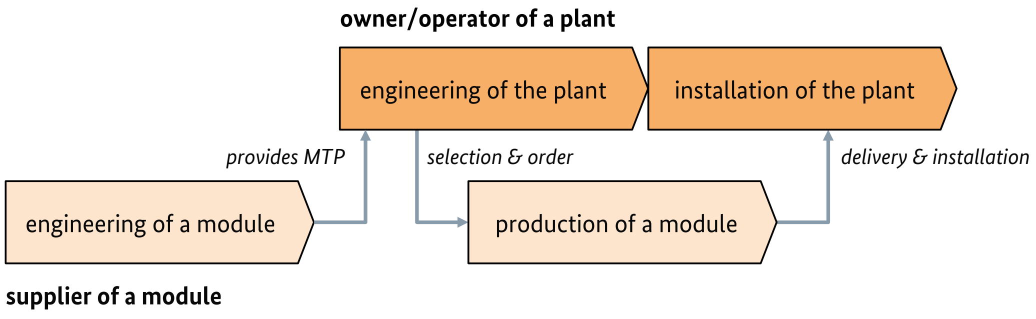
Figure 19 summarizes the main physical objects, model objects, model object templates and relationships of this example.

5.3 Example: Engineering of modular plants

In this example, we consider the engineering of modular plants as it is currently discussed in VDI, NAMUR and IEC TC65 in the context of module type packages (MTP), see [9] and [10].

We look at two different companies, see Figure 20. On the one hand, the supplier of a module, who develops a model series of a module (as part of its development activities) and, on request of a customer, produces and delivers a corresponding physical module to the customer. In addition to the internal development documents, the supplier of the module provides an interface description of the module following the specification according to MTP. Such a module could be, for example, a reactor or a mixing unit. On the other hand, the owner/operator of a plant, who first designs a plant, in which such a module will be installed, then orders a corresponding module and physically installs this module in the plant. Because the supplied module complies to standardized interfaces, the installation and replacement of such a module in the infrastructure can be done easily. The infrastructure comprises both physical aspects such as material flow and energy supply, as well as the integration into the application software of the process control system.

Figure 20: Companies involved in the example



Source: Plattform Industrie 4.0

18 The plant_simulation including the model object virtual_pump_simulation can be used after the commissioning for online simulation.

The engineering of the module and the engineering of the plant are, in its first steps, similar to "5.1 Example: Installation and replacement of a physical pump", but later show significant differences:

- The development department of the supplier of a module will create a model object *module_model_series*. This model object describes all development and production artifacts for this model series of a module.
  In particular, the model object *module_model_series* will comprise a product catalog *product_catalogue*. It is a model object template that describes the assurances that a concrete physical module will satisfy. In addition, it includes a description what the concrete physical module will require from its environment.

- The engineering department of the owner/operator of a plant will create a block flow diagram using some engineering tool. Each block in the block flow diagram is a model object *model_process_step* which represents requirements regarding a certain process step of the production process, for example, the input and output characteristics.

- At some time, the engineering department will select and order a specific module. For this purpose, the product catalog *product_catalogue* is made available by the supplier of a module according to 4.5.4.1 Activity "Transfer a model object from a model user to another model user". The selection of suitable modules for the process steps requires a comparison of respective model objects provided by the product catalogue *product_catalogue* with the various model objects mo*del_process_step*.

To ease the selection decision of an owner/operator of a plant, who wants to use the module, the supplier of a module provides a so-called "Module Type Package (MTP)" *module_model_series_MTP*, which is a model object containing a subset of the information of the model object *module_model_series*. It has been created according to 4.5.1.1 Activity "Creation of a model object" and is made available to the engineering department of the owner/operator of a plant according to 4.5.4.1 Activity "Transfer a model object from a model user to another model user" as *module_MTP* based on the selection of the owner/operator of a plant in the product catalogue. The structure and general content of a MTP is described in a model object template *MTP*, as specified jointly by NAMUR (owner/operator organization) and ZVEI (supplier organization) according to 4.5.2.1 Activity "Creation of a model object template". The engineering department of the owner/operator of a plant will combine module_MTP with the model objects representing the MTPs of the other process steps and check (by using a tool environment) for static and dynamic compatibility.

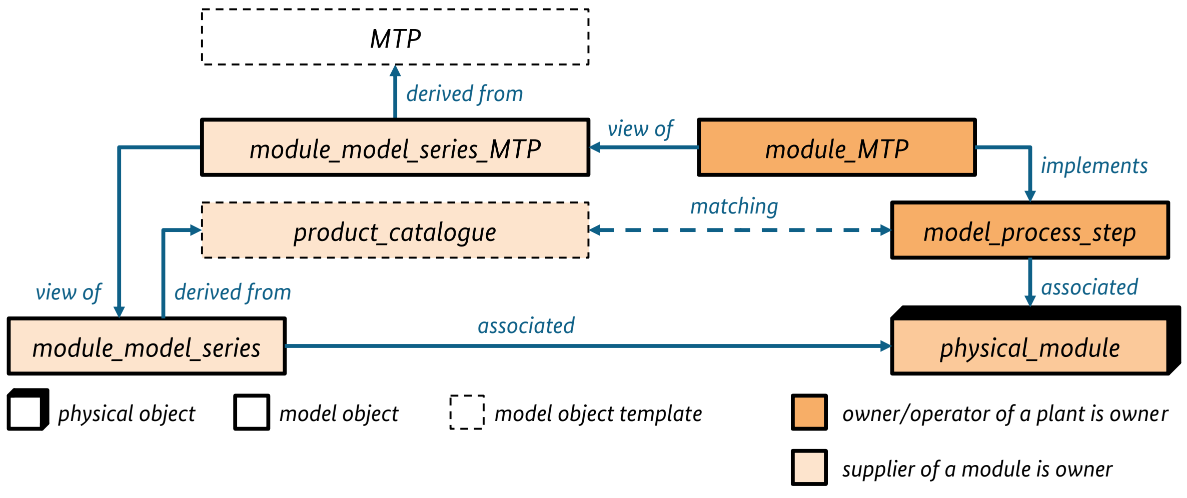
- Once the engineering department of the owner/operator of a plant has decided to order a certain module according to *module_MTP* (and probably additional information) for the plant, a physical representation *physical_module_MTP* of the model object *module_MTP* is loaded into the process control system. The creation of the physical object *physical_module_MTP* is done according to 4.5.3.1 Activity "Creation of a physical object". The result could be a physical file according to a certain file format, for example XML Schema.[19]
  The process control system – as a specific exemplification of a tool environment according to Figure 14 – recognizes this change in the physical world according to the 4.5.3.3 Activity "Reaction on physical change of a physical object" and will react correspondingly. For example, appropriate model objects are created so that on the basis of these model objects the process control system is prepared by generating according to the 4.5.3.1 Activity "Creation of a physical object" physical software in the physical process control system that after the physical plug of the physical object *physical_module* into the physical process control system communication between the *physical_module* and the physical process control system is possible.

- The physical module *physical_module* delivered by the supplier of the module is physically integrated into the plant, for example, placed at a suitable place and connected via pipes and cables with the backbone of the plant. Thus, the physical objects are connected according to the possibilities for material, energy and information flow.

Figure 21 summarizes the main physical objects, model objects, model object templates and relationships of this example.

19 The description of the file format is a model object; the underlying XML schema, which defines the file format, is a model object template.

Figure 21: Overview of main physical objects, model objects and model object templates



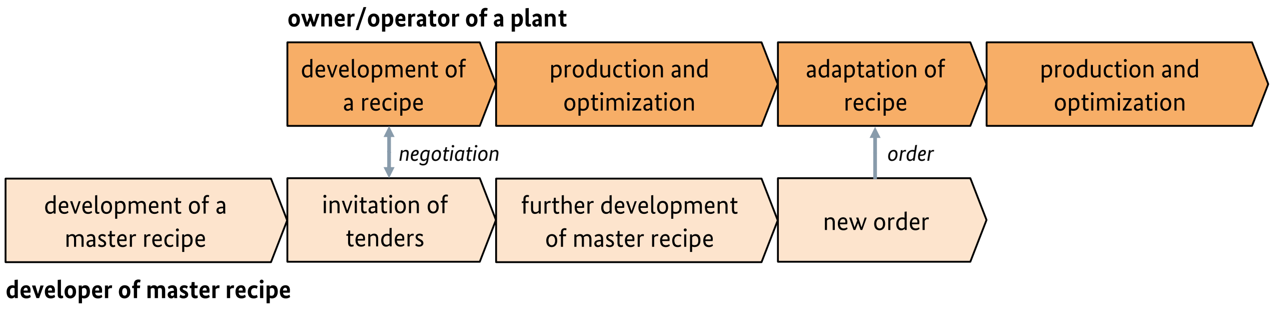
Source: Plattform Industrie 4.0

5.4 Example: Recipe-driven production of active pharmaceutical ingredients

In this example, we consider two business partners, one of whom develops a recipe for an active pharmaceutical ingredient and then provides this recipe to an owner/operator of a plant, who acts as a manufacturing service provider and produces the active pharmaceutical ingredient, see Figure 22:

- The developer of a master recipe uses a laboratory facility to find a suitable manufacturing process for the active pharmaceutical ingredient. The owner/operator of a plant executes the scale-up in order to produce the active pharmaceutical ingredient in his plant industrially. In parallel, the developer of the master recipe optimizes the production using his laboratory facility and, at a given time, provides the owner/operator of the plant an optimized master recipe. Afterwards the owner/operator of the plant produces according to the optimized master recipe.

Figure 22: Companies involved in the example


owner/operator of a plant
development of a recipe
production and optimization
adaptation of recipe
production and optimization
negotiation
order
development of a master recipe
invitation of tenders
further development of master recipe
new order
developer of master recipe

Source: Plattform Industrie 4.0

Related to this context, especially the following processes should be considered:

- The developer of a master recipe develops the master recipe for the active pharmaceutical ingredient. This is a model object template *master_recipe_lab*.
  - In a laboratory facility, which is a physical object *physical_lab*, a production process is developed until finally some suitable recipe *control_recipe_lab* is obtained. This recipe is a model object.
  - Based on the recipe *control_recipe_lab* a master recipe *master_recipe_lab* is created according to 4.5.2.4 Activity "Creating a model object template based on a model object". The core of this activity is a conversion or recalculation of the specific recipe *control_recipe_lab* based on standard units.
  - The master recipe *master_recipe_lab* is released.

- The developer of a master recipe is looking for a suitable owner/owner of a plant, who could act as manufacturing service provider to produce the active pharmaceutical ingredient according to the master recipe and requests for a proposal.
  - For that the developer of the master recipe provides to the owner/operator of a physical plant the master recipe *master_recipe_lab* according to the 4.5.4.1 Activity "Transfer a model object from a model user to another model user" and based on the provided master recipe the owner/operator of a physical plant creates a model object *site_recipe_plant* according to the 4.5.1.1 Activity "Creation of a model object".

- The owner/operator of the plant provides an offer for the developer of a master recipe to produce the active pharmaceutical ingredient according to his recipe *site_recipe_plant*.
  - Based on the recipe *site_recipe_plant*, the owner/operator of the plant creates a recipe *control_recipe_plant* according to the 4.5.1.1 Activity "Creation of a model object" for his plant *physical_plant*. The recipe *control_recipe_plant* is a model object. Typically, the owner/operator of the plant will try not to reconfigure the plant, but "just" to find the right parameters for some recipe *control_recipe_plant*. As a consequence, typically the owner/operator of a plant will copy the model object *site_recipe_plant* according to 4.5.4.1 Activity "Transfer a model object from a model user to another model user" and modify the copy according to 4.5.1.2 Activity "Modification of a model object".

  For example, in the laboratory *physical_lab* a scaling from 1 to 10 could have been developed and been included in the recipe *control_recipe_lab* and master recipe *master_recipe_lab*. In the plant *physical_plant* a scaling should be from 1 to 10 to 100 and finally to 1000. Although these scaling steps are described in principle in the master recipe *site_recipe_plant*, in practice this scaling up typically involves many challenges such as, for example, safety.
  - The owner/operator of the plant will prepare an offer based on the developed recipe *control_recipe_plant* for the developer of the master recipe.

- The developer of the master recipe accepts the offer and commissions the owner/operator of the plant to produce the active pharmaceutical ingredient.

- The owner/operator of the plant will produce the active pharmaceutical ingredient
  - The owner/operator of the plant will optimize the production of the active pharmaceutical ingredient on his plant. This is done by suitably modifying parameters of the recipe *control_recipe_plant* according to 4.5.1.2 Activity "Modification of a model object". Thereby new entries in the life of the recipe *control_recipe_plant* are created.

- The developer of the master recipe will optimize the active pharmaceutical ingredient and/or improves the production process.
  - The developer of the master recipe will continue to experiment using his laboratory *physical_lab* according to 4.5.1.2Activity "Modification of a model object" and will find at some time an improved recipe *control_recipe_lab*. Thereby new entries in the life of the recipe *control_recipe_lab* are created.
  - Then he will modify the master recipe *master_recipe_lab* according to the 4.5.2.2 Activity "Modification of a model object template". Thereby a new entry in the life of the master recipe *master_recipe_lab* is created.

- The new master recipe *master_recipe_lab* is released

- The developer of the master recipe commissions the owner/operator of the plant to now produce the active pharmaceutical ingredient according to the updated master recipe.

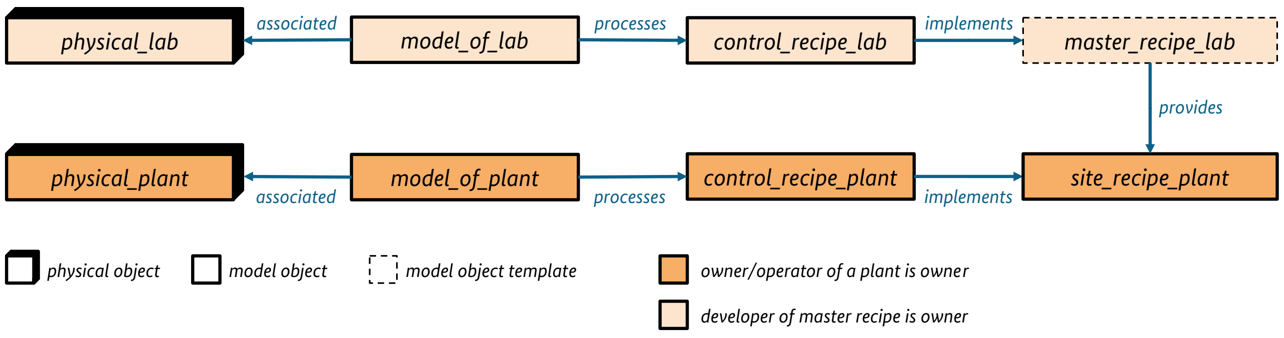
- The developer of the master recipe will provide a new master recipe *site_recipe_plant* to the owner/operator of the plant according to the 4.5.4.1 Activity "Transfer a model object from a model user to another model user". As a consequence the owner/operator of a plant will create a new entry in the life of the recipe *site_recipe_plant* according to the 4.5.1.2 Activity "Modification of a model object".
- Typically, the developer of the master recipe and the owner/operator of a plant are closely tied to each other on business and regulatory approvals, so there should be no new call for tenders by the developer of the master recipe.

- The owner/operator of the plant will modify his recipe *control_recipe_plant* based on the up-dated master recipe *site_recipe_plant* according to the 4.5.1.2 Activity "Modification of a model object" and create a new entry in the life of the recipe *control_recipe_plant*. Then he will produce according to the new recipe and further optimize this.

Figure 23 summarizes the main physical objects, model objects, model object templates and relationships of this example.

Note, that the ISA S88 standard defines meta-models for the description of master recipes, site recipes, control recipes and plant structure elements.

Figure 23: Overview of main physical objects, model objects and model object templates



Source: Plattform Industrie 4.0

6 Relationship between Business View and Usage View

This chapter explains the relationship between the business view, see chapter 3 Business View, and the usage view as described in chapter 4.

Usage View

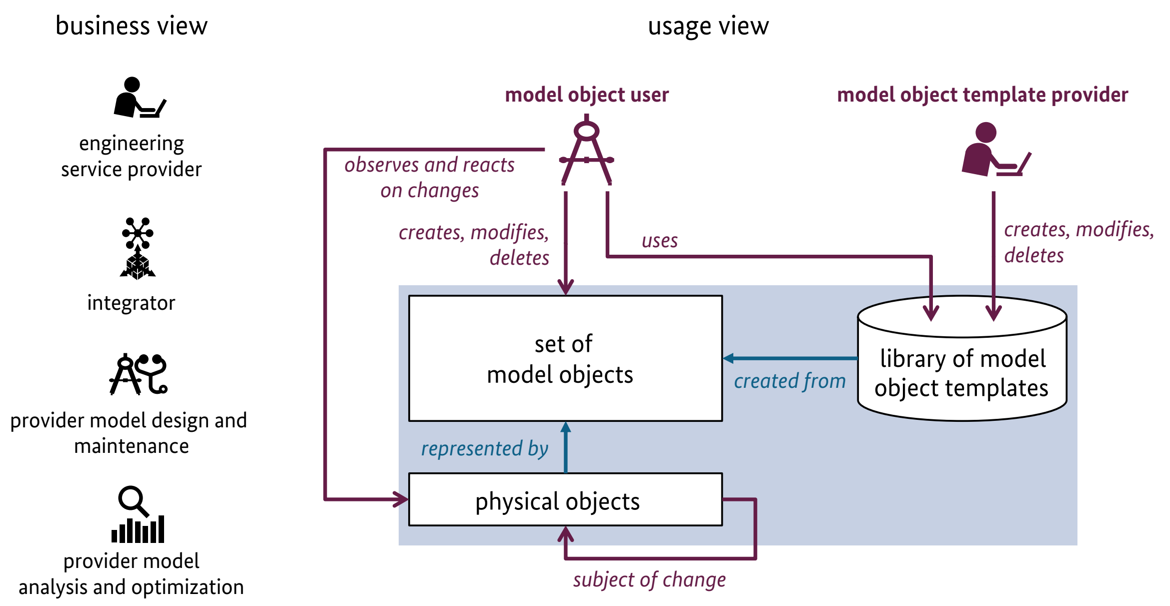
As explained in [2], the general business setup as shown in Figure 2 can be exemplified in various ways. Two possible exemplifications have been explained in [2]. In this chapter we consider another possible exemplification, namely that every stakeholder in the business view of the application scenario SDP is an independent company. The reason for this is that it allows us to simplify the description of the relationship and to focus on the essential principles of the relationship between business and usage view of the application scenario SDP.

Each of the companies shown in Figure 2 internally uses their own model objects and libraries and provides certain physical objects. Therefore, certain employees or departments of these companies assume the roles “model object user”, “model object template provider” or “manufacturer of physical object” according to the usage view. These companies share parts or aspects of their own models with other companies along the value chains shown in Figure 2, which are based on models. This exchange takes place in accordance with the activity described in section 4.5.4.1 Activity “Transfer a model object from a model user to another model user”.

However, the exemplifications regarding their own model objects, libraries and physical objects are different:

- Integrator, engineering service provider, provider model design and maintenance and provider model analysis and optimization, see Figure 24: Internally, these companies use their own models and typically their own libraries and thus assume the roles “model object user” and “model object template provider” according to the usage view. An example of a model object template of an integrator is a reference plant that has been developed based on the past plant engineering projects. Typically, these companies do not have physical objects and therefore do not assume the role “manufacturer of physical object” according to the usage view, but they usually refer to physical objects of other companies and may even create corresponding model objects of them.

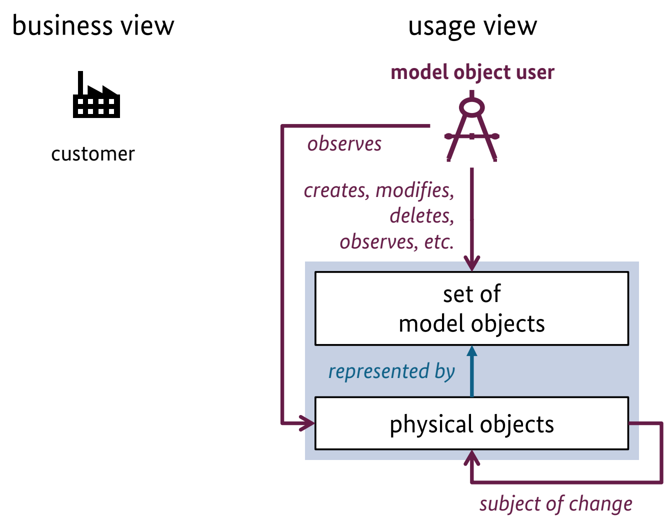
Figure 24: Comparison of business view and usage view (I)


business view
usage view
engineering service provider
integrator
provider model design and maintenance
provider model analysis and optimization
model object user
model object template provider
observes and reacts on changes
creates, modifies, deletes
uses
creates, modifies, deletes
set of model objects
created from
library of model object templates
represented by
physical objects
subject of change

Source: Plattform Industrie 4.0

- Customer, see Figure 25: The customer receives a model from the integrator and also physical objects from the stakeholder supplier and construction; typically, the customer is afterwards the owner of these physical objects. These physical objects change throughout their lives, whether through wear and tear, or through reconfiguration initiated by the customer. In the application scenario SDP, it is assumed that the customer may adjust parameters in models for minor changes, but for particularly structural changes necessary due to changes in the physical world the integrator is involved – thus, the customer initiates the activity described in section 4.5.1.5

Figure 25: Comparison of business view and usage view (II)

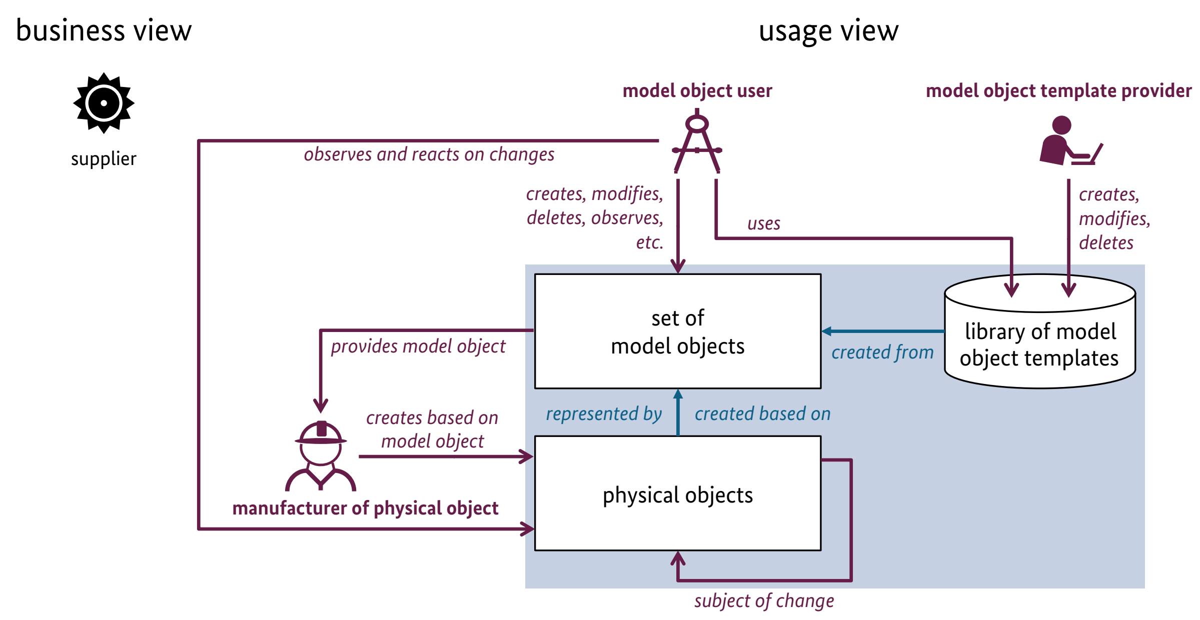


Source: Plattform Industrie 4.0

Activity “Reaction on physical change of a physical object”, but does not execute the activity itself. In this respect, the customer only assumes the role “model object user” according to the usage view and is essentially concerned with the activities described in the section 4.5.1.2 Activity “Modification of a model object” and 4.5.4.3 Activity “Evaluation/simulation of a model object”.

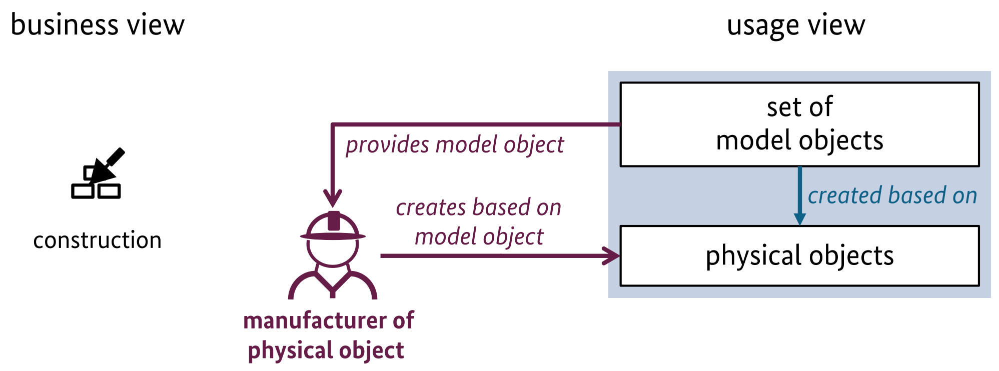
- Supplier, see Figure 26: The supplier of a physical object assumes all three roles “model object user”, “model object template provider” and “manufacturer of physical object” according to the usage view. He also performs all the activities described in section 4.5 Activities (except for the activities described the section 4.5.5 Management of tool environment). This is an example where the usage view SDP is applied not to plant engineering, but to the development and manufacturing of physical products.

Figure 26: Comparison of business view and usage view (III)


business view
usage view
supplier
model object user
model object template provider
observes and reacts on changes
creates, modifies, deletes, observes, etc.
uses
creates, modifies, deletes
set of model objects
library of model object templates
created from
provides model object
represented by
created based on
creates based on model object
physical objects
manufacturer of physical object
subject of change

Source: Plattform Industrie 4.0

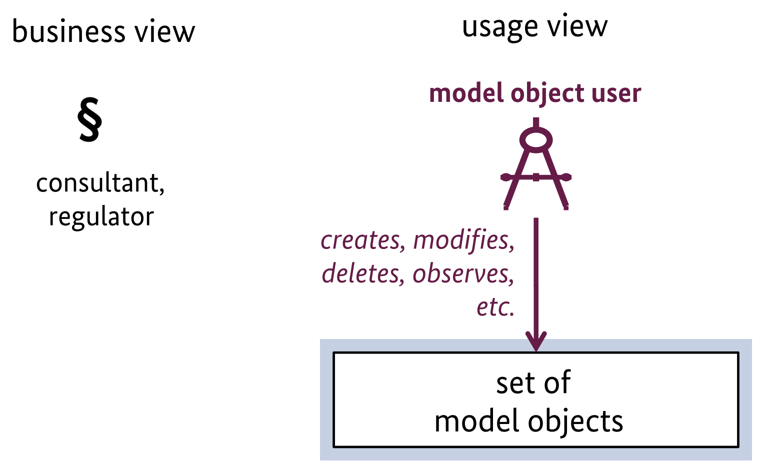
Figure 27: Comparison of business view and usage view (IV)



Source: Plattform Industrie 4.0

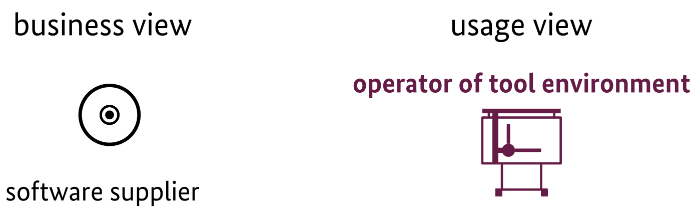
- Construction, see Figure 27: We assume that the stakeholder construction merely carries out the execution of the erection, the planning of the erection is executed by the integrator. In this respect, the stakeholder construction is given a model by the integrator. On this basis, he carries out the physical erection of the plant, so that the finished physical plant can afterwards be made available to the customer. The stakeholder construction therefore assumes the role “manufacturer of physical object” according to the usage view and executes the activity described in section 4.5.3.1 Activity “Creation of a physical object”.
- Consultant, regulator, see Figure 28: These companies are usually provided with a model and then they evaluate this model. Usually they do not create new model objects. Therefore, they assume the role “model object user” according to usage view and carry out their evaluation according to the activity described in section 4.5.4.3 Activity “Evaluation/simulation of a model object”.
- Software supplier, see Figure 29: This company assumes the role “operator of tool environment” according to the usage view.

Figure 28: Comparison of business view and usage view (V)



Source: Plattform Industrie 4.0

Figure 29: Comparison of business view and usage view (VI)



Source: Plattform Industrie 4.0

7 References

[1] Aspects of the Research Roadmap in Application Scenarios, July 2016, http://www.plattform-i40.de/I40/Redaktion/EN/Downloads/Publikation/aspects-of-the-research-roadmap.html

[2] Application Scenario SDP – Seamless and Dynamic Engineering of Plants, VDI-Statusreport, https://www.vdi.de/ueber-uns/presse/publikationen/details/vdi-status-report-seamless-and-dynamic-engineering-of-plants

[3] Proposal for a joint “scenario” of Plattform Industrie 4.0 and IIC, http://www.plattform-i40.de/I40/Redaktion/DE/Downloads/Publikation/joint-scenario.html

[4] Benefits of Application Scenario Value-Based Service, https://www.plattform-i40.de/I40/Redaktion/DE/Downloads/Publikation/benefits-application-scenario.html

[5] Usage Viewpoint of Application Scenario Value-Based Service, https://www.plattform-i40.de/I40/Redaktion/DE/Downloads/Publikation/hm-2018-usage-viewpoint.pdf?__blob=publicationFile&v=8

[6] German Standardization Roadmap Industrie 4.0, Version 3, https://www.din.de/blob/65354/57218767bd6da1927b181b9f2a0d5b39/roadmap-i4-0-e-data.pdf

[7] The Industrial Internet Reference Architecture Technical Report, https://www.iiconsortium.org/IIRA.htm

[8] A. Fay, J. Gausemeier, M. ten Hompel: Einordnung der Beispiele der Industrie 4.0-Landkarte in die Anwendungsszenarien, April 2018, https://www.plattform-i40.de/I40/Redaktion/DE/Downloads/Publikation/hm-2018-fb-landkarte.html

[9] VDI/VDE/NAMUR 2658, www.vdi.de/2658

[10] IEC 63280 ED1 Automation engineering of modular systems in the process industry – General concept and interfaces (65E/663/NP)

AUTHORS AND GUESTS

From the German side, the status report has been elaborated in the VDI/VDE-GMA Technical Committee 6.12 “Seamless Engineering of Process Control Systems”, especially by the following members and guests:

MEMBERS

Dr. Oliver Drumm; Siemens AG, Karlsruhe | Ralph J. Eckhardt; EY Advisory Services | Feras El Sakka; Helmut-Schmidt-Universität/Universität der Bundeswehr Hamburg | Prof. Dr. Alexander Fay; Helmut-Schmidt-Universität/Universität der Bundeswehr Hamburg | Freihold Hasselfeld; CIM-BASE GmbH, Sersheim | Dr. Nasser Jazdi; Universität Stuttgart | Dietmar Krumsiek; Phoenix Contact Electronics GmbH, Bad Pyrmont | Prof. Dr. Ulrich Löwen; Siemens AG, Erlangen | Thomas Makait; QPRI, Hofheim (Taunus) | Dr. Tina Mersch; Beckhoff Automation GmbH, Verl | Andreas Schertl; Siemens AG, Nürnberg | Thomas Schindler; Yokogawa Europe | Dr. Miriam Schleipen; Siemens AG, Karlsruhe | Dr. Sebastian Schröck, Robert Bosch GmbH, Renningen

CO-AUTHORS AND ADVISORS

From the Japanese side, some additional ideas were contributed by the following members of the “Use Case Task Force” in the International Standardization Action Group, Robot Revolution & Industrial IoT Initiative and related associates: Prof. Fumihiko Kimura; The University of Tokyo | Dr. Hitoshi Komoto; National Institute of Advanced Industrial Science and Technology | Kazuo Nakashima; Robot Revolution & Industrial IoT Initiative | Takuya Iijima; Toshiba Infrastructure Systems & Solutions Corporation | Nobuyuki Ogura; Hitachi, Ltd.

ADVISORS

Shinji Oda; Yokogawa Electric Corporation | Dr. Youichi Nonaka; Hitachi, Ltd.

ACKNOWLEDGEMENT

The authors would like to thank Zhiman Chen (CRRC ZIC) for her valuable comments.
The authors also thank the members of the VDI/VDE-GMA Technical Committee 6.11 “Virtual Commissioning”, especially Prof. Dr. Mike Barth; Hochschule Pforzheim, Prof. Dr. Jens Jäkel; Hochschule für Technik, Wirtschaft und Kultur Leipzig and Roland Rosen; Siemens AG, München, for the fruitful discussions.

This working paper has been elaborated in the working group “Modeling Example” VDI/VDE-GMA Technical Committee 6.12 “Seamless Engineering of Process Control Systems” together with the “Use Case Task Force” in the International Standardization Action Group, Robot Revolution Initiative in cooperation with the working group on research an innovation (Plattform Industrie 4.0). The activities and partners were initiated and orchestrated by the Standardization Council Industrie 4.0 (SCI4.0) withing the project GoGlobal Industrie 4.0.

www.plattform-i40.de