



for efficient and iterative  
development of smart factories



ITEA 3 – 15015

**Work package 2**  
Formalized Requirements Engineering

**Deliverable 2.2**  
Language for formalized requirements engineering

Document type	: Deliverable
Document version	: 0.1
Document Preparation Date	: 09.07.2018
Classification	: public
Contract Start Date	: 01.09.2016
Contract End Date	: 31.08.2019



	<p style="text-align: center;"> <b>ENTOC</b>        Engineering tool chain for efficient and iterative        development of smart factories        ITEA 3, 15015        Project Coordinator: Thomas Bär, Daimler AG     </p>	
--	---	---

Final approval	Name	Partner
Review Task Level	Andreas Schlag	Daimler AG
Review WP Level	Mario Thron	ifak e.V.
Review Board Level	Johann Vallhagen	Volvo Trucks

## Executive Summary

Verbal statements with references to images and tables are currently used to describe requirements for production equipment in industrial contexts. Those specifications are sometimes ambiguous and are not suited to be evaluated by computer programs. An approach of formalization of requirements is presented within this document.

Several concepts and language constructs have been developed, which can be used independently from each other, but are developed with the intention to use them complementarily. All language constructs have been described by use of lower level syntax elements provided by the AutomationML standard [IEC62714:2018], which makes it possible to use one generic basic parsing algorithm for usage of all the concepts within a respective tool. The developed language constructs make it possible to describe important aspects of products, production processes, production resources and projects for creation of production systems.

Simple hand-written usage examples prove the theoretical usability of the concepts. Tools for editing more complex examples will be developed within task 2.3. More complex requirement specification scenarios will be evaluated by using those tools within working package 6 of the ENTOC project.

## List of Authors

Chapter 1	Introduction	Andreas Schlag, Daimler AG Mario Thron, ifak e.V.
Chapter 2	Project Planning	Mario Thron, ifak e.V.
Chapter 3	Product Variability Handling	Knut Akesson, Chalmers Sabino Roselli, Chalmers
Chapter 4	Process-Driven Production Planning	Andreas Schlag, Daimler AG
Chapter 5	Production Resource Planning	Klaus Hanisch, Tarakos GmbH
Chapter 6	Production Facility Requirement Specification as a Result of PPR-Model-Based Production Planning	Andreas Schlag, Daimler AG
Chapter 7	Requirement Specification based on Natural Language Boilerplates	Ireneus Wior, TWT GmbH
Chapter 8	Conclusion and Outlook	Andreas Schlag, Daimler AG Mario Thron, ifak e.V.

## Contents

Executive Summary .....	3
1 Introduction.....	6
1.1 Formalized Requirements Engineering .....	6
1.2 Chapter Overview.....	7
1.3 General Structure of Chapters .....	8
1.4 UML to AutomationML Mapping Strategy .....	8
2 Project Planning.....	10
2.1 Motivation .....	10
2.2 Approach .....	12
2.3 Overview of UML-Model .....	13
2.4 Example of General Functionality .....	13
3 Product Variability Handling.....	15
3.1 Motivation .....	15
3.2 Approach .....	16
3.3 Overview of UML-Model .....	18
3.4 Example of General Functionality .....	19
4 Process-Driven Production Planning .....	21
4.1 Motivation .....	21
4.2 Approach .....	22
4.2.1 Manufacturing Processes .....	22
4.2.2 Handling Processes.....	25
4.3 Overview of UML-Model .....	26
4.4 Example of General Functionality .....	26
5 Production Resource Planning .....	28
5.1 Motivation .....	28
5.2 Approach .....	28
5.2.1 Basic Concept .....	28
5.2.2 Module Level .....	29
5.2.3 Port Level.....	30
5.3 Overview of UML-Model .....	31
6 Production Facility Requirement Specifications as a Result of PPR-Model-Based Production Planning.....	32

6.1	Motivation .....	32
6.1.1	Requirements Engineering .....	33
6.1.2	The Structure of Requirement Specifications .....	34
6.2	Approach .....	35
6.2.1	Methodology for Optimization of the Planning and Engineering Phase.....	35
6.2.2	Base Concept of Auto-Generation of Requirements.....	36
6.2.3	Extended Concept of auto-Generation of Requirement Specifications.....	37
6.2.4	Extended Boilerplates and Templates of Requirements.....	38
6.2.5	Benefit of PPR-Model-Based Production Facility Requirement Specification.....	39
6.3	Overview of UML-Model .....	40
6.4	Example of General Functionality .....	42
7	Requirement Specification based on Natural Language Boilerplates .....	44
7.1	Motivation .....	44
7.2	Approach .....	45
7.2.1	Specification Structure .....	45
7.2.2	Notation of the Extended Backus-Naur form (EBNF).....	46
7.2.3	Preliminary Definitions.....	46
7.2.4	Set of Boilerplates for the Production Domain .....	48
7.3	Overview of UML-Model .....	50
7.4	Example of General Functionality .....	52
8	Conclusion and Outlook .....	54
9	References.....	55
10	Appendix.....	57
10.1	Tables of UML to AutomationML Mapping.....	57
10.1.1	Chapter 2 – Project Planning .....	57
10.1.2	Chapter 3 – Product Variability Handling .....	65
10.1.3	Chapter 4 – Process-Driven Production Planning.....	70
10.1.4	Chapter 5 – Production Resource Planning.....	78
10.1.5	Chapter 6 – Production Facility Requirement Specifications as a Result of PPR-Model-Based Production Planning.....	108
10.1.6	Requirement Specification based on Natural Language Boilerplates .....	115

## 1 Introduction

### 1.1 Formalized Requirements Engineering

The main goal of the ENTOC project is an engineering tool chain for efficient and iterative development of smart factories, which provides the possibility to create and use valuable engineering information through the whole development process of those factories. This tool chain realises the design methodology for mechatronic systems in the style of the V-Model of VDI 2206 (see Figure 1).

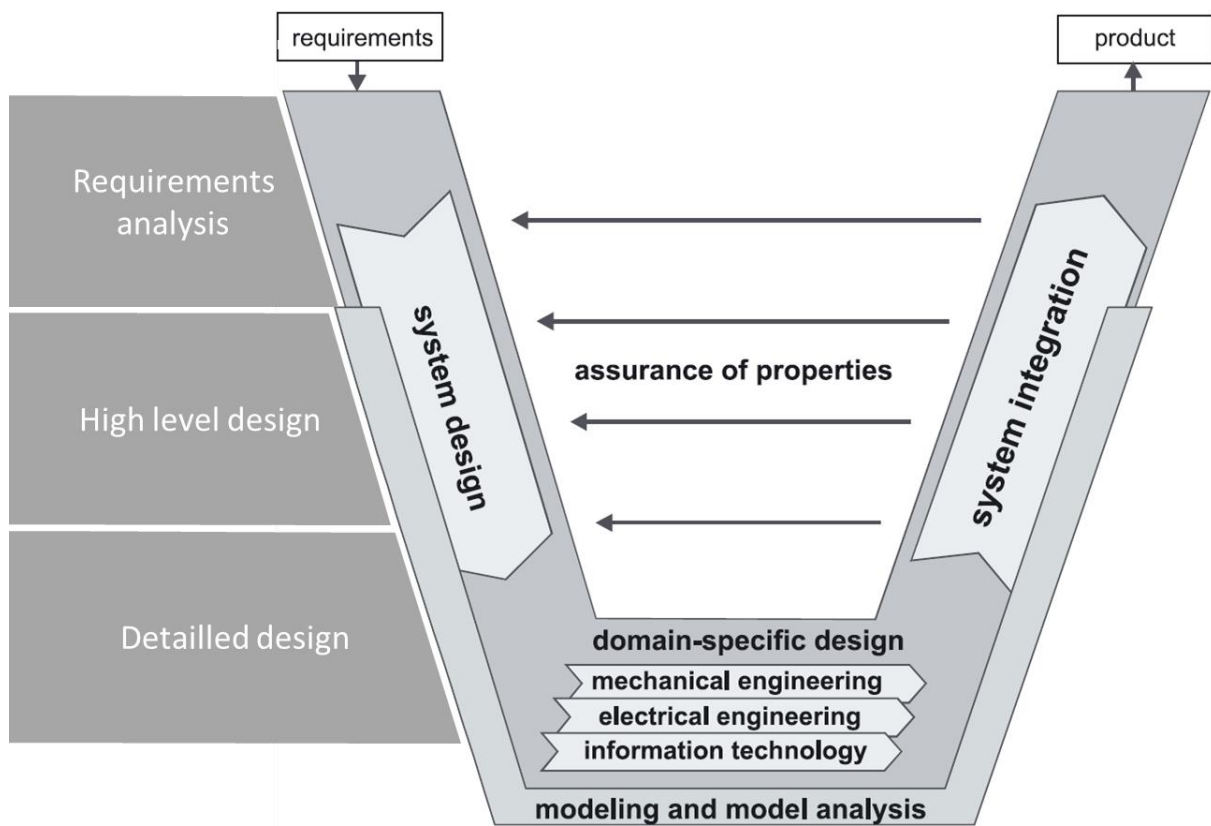




Figure 1: The V-Model of VDI 2206

Working package 2 “Formalized Requirements Engineering” of the ENTOC project, represents the first phase of the V-Model with respect to the steps in requirements analysis and high level design.

Requirements engineering is a cooperative, interactive and incremental process which obtains the development of requirements out of abstract specifications with the goal to determine, analyze, understand and establish requirements [Gil2014].

Requirements are differentiated into functional and non-functional requirements and have the ability to be varying in abstraction and detail. They are mostly written out of the authors’ technical understanding and perspective, especially in industrial practice.

Requirements are transferred into concepts, functions or modules during high level design. It is a common method to sketch concepts on paper and refine them until a certain degree of maturity is

	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---

reached. During this phase of work, often much of the “Know How” of the refinement process gets lost for the community, as it is difficult to capture in any documentation.

## 1.2 Chapter Overview

Within this chapter the structure of this document is introduced for a better readability and general understanding. The chronological order of the chapters follows the described development process in the previous chapter.

### Project Planning

A project is described in the context of planning to fulfill a task and reach a specific goal. General project information is related to premises, budgets, time, human resources, etc. in order to create or modify a production system. In WP2 we define data items, which often are exchanged between customers and vendors of technical equipment. A project requires a plan agreed between customer and vendor in order to succeed in a defined timespan. Therefore a time plan is one of the first elements which will be created at the start of a project. A time plan in production facility planning contains activities, phases, milestones and related results. In context of this document chapter 2 will introduce a model-based approach for project time planning in a production facility planning project and demonstrates the exchange via AutomationML.

### Product Variability

In general the result of a development process is the release of a product, which is identifiable by the “what to produce” question out of a planning context. With the rising demand on custom tailoring of products and flexibility of production facilities the handling of the amount of product variants appears as a more and more complex process in production. Chapter 3 introduces a model-based approach to describe product variants with logical connectors, Boolean expression and a storage mechanism with AutomationML.



### Process-Driven Production Planning

Out of the production facility planning perspective the production process planning is the root element to adapt methods of optimization. Required operations sequences, joining technologies and other methods are assigned during the production process planning. The joining order is derived by these technologies, stable states of the product and a given cycle time. This results in a description of required technologies and an amount of manufacturing steps or stations to meet the demands for production capacity and quality.

The production process planning gets exponentially complex by adding product variants into the PPR-Model. The essential relations between products, processes and required resources are created within the process driven production planning phase. A model-based approach to break down the complexity is introduced in Chapter 4. This approach is illustrated with the formalized process description language.

### Production Resource Planning

The result of production facility planning is a resource structure with the ability to fulfill the required capabilities of technologies, manufacturing processes and cycle time. The chronological order of production resource planning is in general from abstract to concrete. In the automotive industry it is a common standard to limit the list of allowed resources to reduce stocks. In chapter 5 a model-based library approach is introduced to plan production resources from an abstract level down to concrete resources from suppliers based on a flexible library concept.

	<p style="text-align: center;">ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p style="text-align: center;">Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

### **Production Facility Requirement Specifications as a Result of PPR-Model-Based Production Planning**

Contracts between customers and suppliers are legally binding. They are based on requirement specifications in which a customer describes the need of his request in a varying level of details. In general a requirement specification describes solution neutral requirements.

It is common use in the automotive industry to describe some parts of a requirement specification in a concrete way, where technologies or already existing elements have to be used, re-used or re-tooled. In application a production planner is merging the information of project and product premises, process and resource requirements into a document, the requirement specification document. This document is often written in prose grammar and carries the signature of the planners' technical understanding. This offers a potential for error rates.

In chapter 6 a model-based approach is introduced to extract the related information for a requirement specification out of the PPR-Model and to merge them in a human readable form using requirement template pattern, so called boilerplates. The result is a PPR-Model of the planning context with relations between Product, Process, Resource and Requirement.

#### **Requirement Specifications based on Natural Language Boilerplates**

With the goal of 100% coverage of requirements, functional and non-functional, a further technique is introduced in chapter 7. To create all required types of requirements in a natural, human readable language the ability to create them manual by the use of boilerplates is necessary. Chapter 7 describes a technique to create requirement specifications based on natural language boilerplates.

### **1.3 General Structure of Chapters**

A general structure for chapters is used to increase the readability and recognition of this document. Each chapter consists of four sections to lead the reader from the problem statement via an approach description to an application example.

#### **Motivation**

Within the motivation section the reader should get an overview about the area of related contribution for this topic. Further-on, the why and the estimated benefit are explained next to description of stakeholders.

#### **Approach**

The approach or technique is introduced in this section. Relevant elements, tools, techniques are described and build a context for this topic.

#### **Overview of UML-Model**

The data model to be used within the approach is demonstrated in an UML-diagram to get a better understanding.



#### **Example of General Functionality**

With an exemplary application of the approach and model the general functionality is demonstrated.

### **1.4 UML to AutomationML Mapping Strategy**

A general goal of ENTOC is to create a chain of tools. The basic necessity is to provide an opportunity to exchange data from tool to tool. Thus the data models depicted as UML class diagrams within the



	<p style="text-align: center;"> <b>ENTOC</b>        Engineering tool chain for efficient and iterative        development of smart factories        ITEA 3, 15015          Project Coordinator: Thomas Bär, Daimler AG     </p>	
--	---	---

several chapters have to be transformed into a data exchange format. There is a variety of basic formats, which could have been chosen, like JSON, XML, STEP EXPRESS and more. We decided to use AutomationML, which appeared as whitepaper [Aml2016a] and is standardized as IEC standard [IEC62714:2018]. It is an XML-based data format, which is compatible with [IEC6242:2008]. The decision for AutomationML has been made, because it is a well-known, standardized and easy to use data format, which additionally was known and used by various project partners. Thus this specification describes formally how to use AutomationML for data exchange within an engineering tool chain.

For transformation of UML class diagrams appearing in chapters 2 to 7 into AutomationML elements, the following rules were specified for this working package:

- UML aggregation relations are mapped to InternalElement sub-hierarchies. Thus the container-IE is the direct parent IE of all the aggregated IEs.
- UML association relations are mapped to InternalLinks. For that purpose related InternalElements will provide ExternalInterfaces of specialized InterfaceClasses as direct sub-elements. Those InterfaceClasses are named at the left and right end of the UML association.
- IDs are expected to be provided implicitly by the ID attribute of AutomationML InternalElements.
- The following libraries are expected to be imported into the AutomationML project:
  - AutomationMLBaseRoleClassLib (Version 2.2.2)
  - AutomationMLInterfaceClassLib (Version 2.2.2)

ENTOC specific UML Packages were colored in light yellow and the usage of AutomationML standardized Interfaces are marked with the prefix “AML:”<InterfaceName>.

## 2 Project Planning

### 2.1 Motivation

The engineering and commissioning of a production system is called a project in the following. It involves time, resources and efforts. The time, when the system should be ready for production is a key non-functional requirement of a customer of such a system. Customer and supplier of the production system often define intermediate time points to monitor the project progress especially within large projects. Those time points are called mile-stones of the project.

Figure 2 provides an overview about a typical mile-stone plan. It provides a mapping of expected key results to activity time spans and mile-stones to involved staff (resources).

Key results	2016				2017				Involved staff
	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	
Product definition	■								Product designer
Process definition		■							Process designer
Rough layout of site		■							Plant designer
Planning Milestone (M1)			▲ 30.6.						Steering Committee
Supplier identification			■						Procurement, sales management
Detailed engineering			■						Engineers
Virtual commissioning				■					Engineers, IT experts
Engineering Milestone (M2)				▲ 4.1.					Steering Committee
Commissioning				■					Engineers, IT experts
Tuning						■			Engineers
Documentation (as built)							■		Documentation experts
Acceptance test								■	Production dept.
Handover Milestone (M3)								▲ 30.9.	Steering Committee, Production dept.
First month production								■	Production dept., Engineers
Fine tuning								■	Production dept., Engineers
Final test								■	Production dept.
Final Milestone (M4)								1.12.▲	Steering Committee, Production dept.

Figure 2: Example of a mile-stone plan

The assignment of staff and resources to be used are internal processes of the customer and supplier of the production system. Thus this assignment will not be part of the requirements specification. If the relations of expected results to mile-stones are done by using formal languages, then this data may be used as import data for the planning tools of the customer and supplier.

Within large projects the supplier engages sub-suppliers for creation of partial results and even further sub-contract relations may appear (Figure 3 **Fehler! Verweisquelle konnte nicht gefunden werden.**). Project timing information has to be distributed and processed via all sub-contractor relationships. It will be much easier to manage for each of the related companies, when formal descriptions of time constraints are used, which may be imported into and exported out of the planning tools of the customers and suppliers.

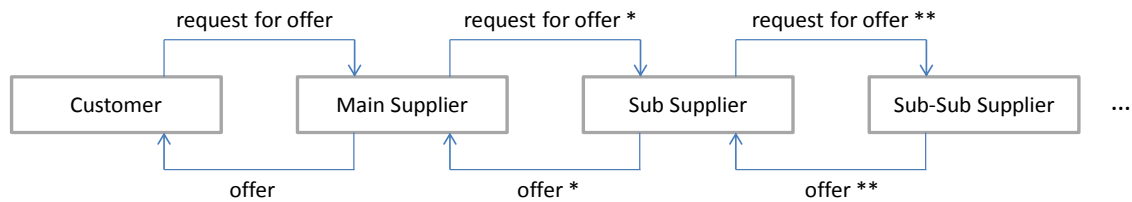


Figure 3: A supplier may be customer in relation to sub-suppliers (number of stars indicate level in the customer relationship hierarchy).

In early stages of the project planning customers try to find the technically and monetary best promising solution to satisfy the customers' needs. Different sub-suppliers are therefore requested in many cases for offers regarding the same set of results and the resulting offers have to be managed by the customer (see Figure 4 **Fehler! Verweisquelle konnte nicht gefunden werden.**). Thus a machine readable data exchange format will reduce the overall effort for planning activities.

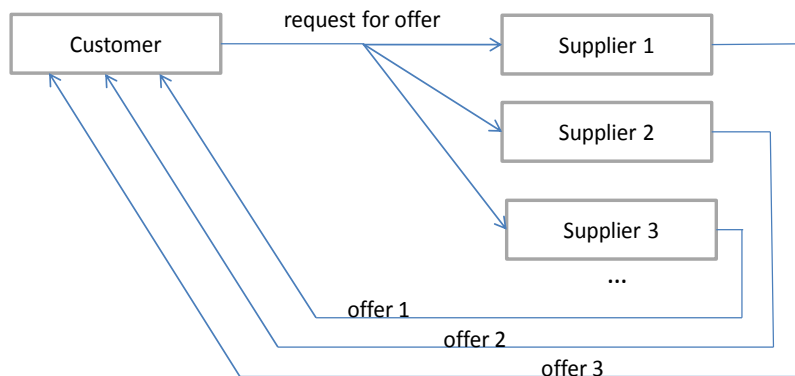


Figure 4: Multiple possible suppliers are involved in first stages of planning

Formal representations of mile-stone plans would provide following benefits:

- They can be integrated easily into suppliers and sub-suppliers resource planning tools.
- The expectations of customers become crystal clear, so there is no ambiguity for delivery of equipment and payment.
- The mile-stone plans can be passed easily as a whole or in parts in a sub-supplier chain.
- If there is a mixture of absolute time definitions and durations of activities, then inconsistencies (false expectations) can be detected by algorithms.
- Tracking of finished mile-stones enables recalculation of due dates of dependent later mile-stones.

Thus the direct goal of this mile-stone planning approach is to provide formal representations of mile-stone plans as non-technical requirements to be included into final offer as base for contract designs. A derived goal is to provide possibilities to track intended results at mile-stone dates during the project.

## 2.2 Approach

The formal representation of a mile-stone plan involves different data as depicted informally by Figure 5

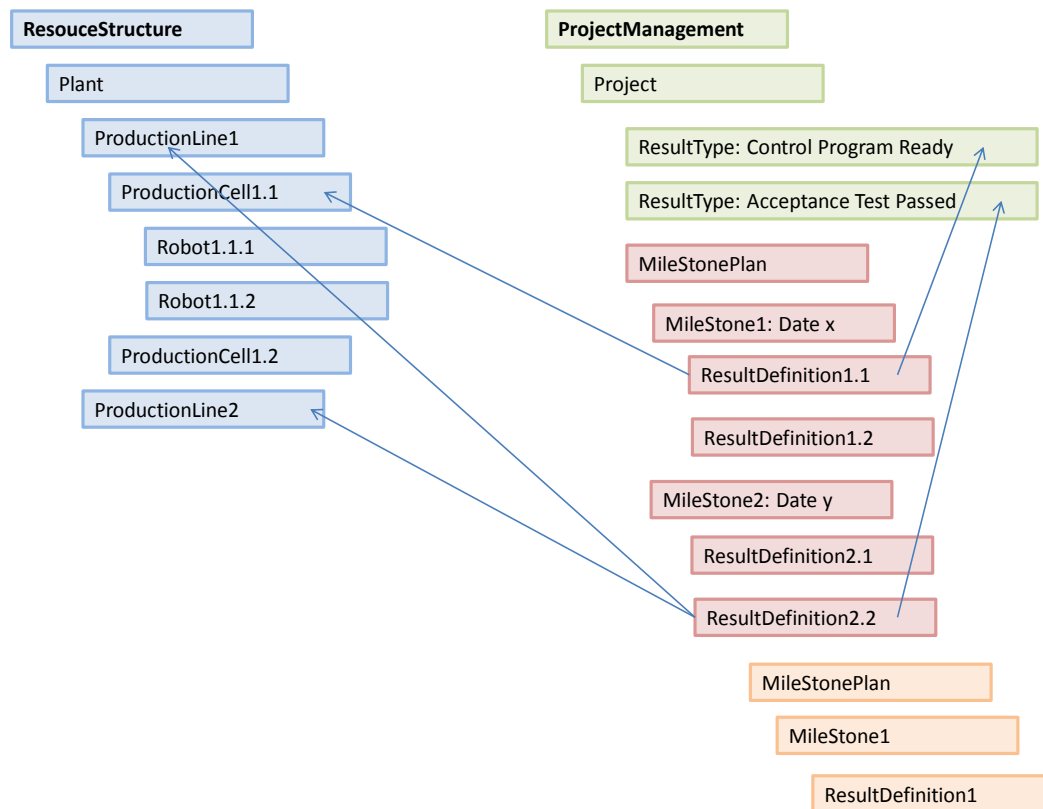


Figure 5: Milestones are mappings from due dates to resources and intended results

The resource structure (marked blue in Figure 5 **Fehler! Verweisquelle konnte nicht gefunden werden.**) is defined by a hierarchical tree of equipment. It is assumed to be available for the mile-stone planning. Such a resource structure can be created for example by using 3D modelling tools, which map the virtual layout of the production system to a scene graph, which indeed is in most cases a hierarchical tree of equipment parts.

An additional tree of information defines project management information. The project declaration part is colored green in Figure 5. It contains definitions of result types, for example 'the control program is ready for download'. This list of result types can easily be extended per project. Those result types have to be related to parts of or the whole production system. Thus special nodes called "ResultDefinition" interlink the result types with elements of the resource structure within the first-level mile-stone plan (colored red). Any kind of those result definitions are attached to mile-stones, which carry information about due date and time. Multiple milestones are assembled to form complete mile-stone plans.

An advanced and optional concept is to define sub-mile-stone plans (e.g. the orange colored sub-mile-stone plan in Figure 5) to define mile-stone activities in more detail. It can be used to coordinate supplier - sub-supplier relations (see Figure 3).

## 2.3 Overview of UML-Model

Figure 6 provides a more formalized representation of the approach as a UML class diagram. Therein white classes depict AutomationML RoleClasses, which can be interrelated according to the UML association types. Role classes inside the yellow highlighted namespace "MilestoneRSL" are classes developed within the ENTOC project. This role class library is contained in the annex (section 10.1.1).

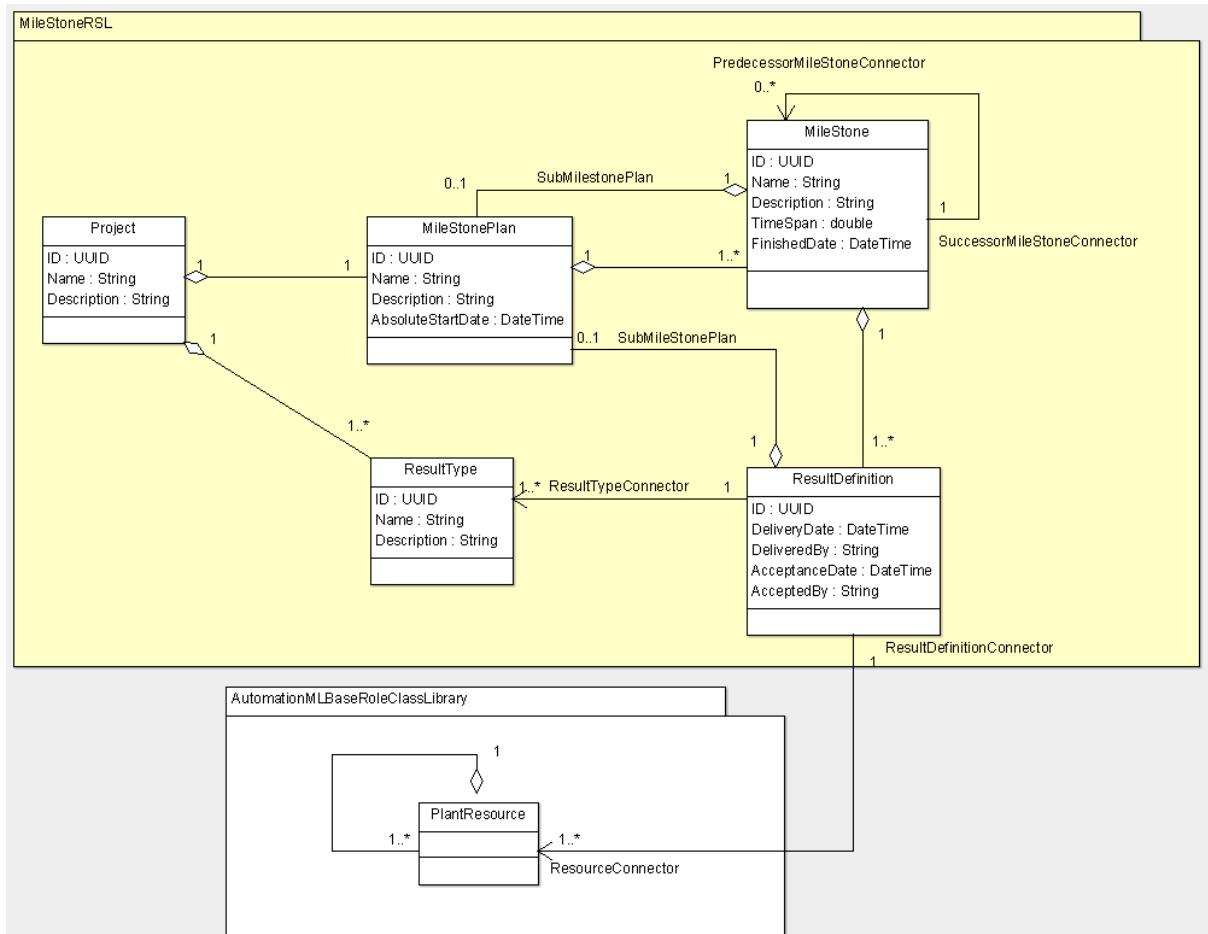


Figure 6: Overview of model elements used for mile-stone planning (UML class diagram)

## 2.4 Example of General Functionality

Figure 7 depicts how to use the concepts introduced here. The resource structure consists of equipment for cockpit installation in a car production context. The raw design contains three stations for part supply, body positioning and cockpit mounting. This resource structure is referenced by a kind of project plan, which is implemented as mile-stone plan. The current plan contains definitions of "control program is ready" and "acceptance test has passed" as result types for milestones. The example shows how to use them for a result definition of mile-stone 001. This result definition contains references from resource "station 1" to "control program is ready", which translates to the definition "when mile-stone 001 is due, then the control program for station 001 has to be ready."

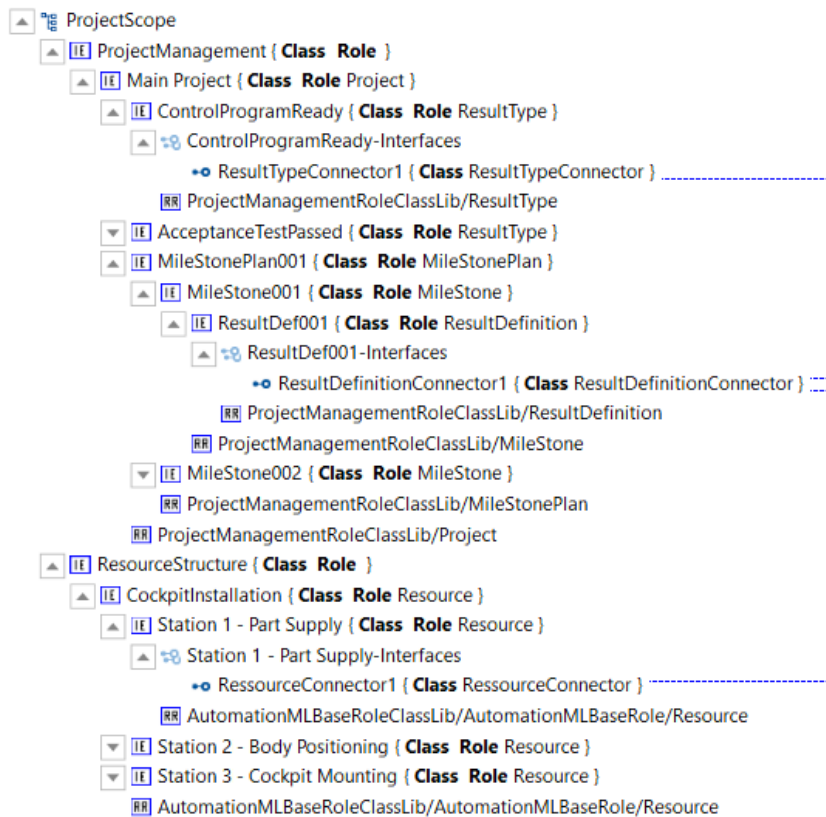


Figure 7: Example of mile-stone definition

Figure 8 provides an overview of MileStone001 with internal links to the result type and to the related resource. The duration of work for the results of this mile-stone is expected to be 8'640'000 seconds, which is around 100 days.

```

<InternalElement Name="MileStonePlan001" ID="d53e177f-1241-41be-8e15-38c30ac8f0f3" >
  <InternalElement Name="MileStone001" ID="83a9a714-d4f0-4590-9215-f9d697203a46">
    <Attribute Name="Name" AttributeDataType="xg:string">
      <Value>Milestone 001</Value>
    </Attribute>
    <Attribute Name="Description" AttributeDataType="xg:string">
      <Value>The first mile-stone ...</Value>
    </Attribute>
    <Attribute Name="TimeSpan" AttributeDataType="xg:double">
      <Value>8640000.0</Value>
    </Attribute>
    <Attribute Name="FinishedDate" AttributeDataType="xg:dateTime"/>
    <InternalElement Name="ResultDef001" ID="92007ea3-fc0b-4bb7-ad02-0c27d912db5d">
      <ExternalInterface Name="ResultDefinitionConnector1"
        RefBaseClassPath="ProjectManagementInterfaceClassLib/ResultDefinitionConnector"
        ID="fc7ab3fb-ch13-4bbd-b4f6-4dc94db4eb7f"/>
      <InternalLink Name="ResourceLink001"
        RefPartnerSideA="92007ea3-fc0b-4bb7-ad02-0c27d912db5d:ResultDefinitionconnector1"
        RefPartnerSideB="ded4ef8c-d4cc-466b-885d-017dde0d64a5:ResourceConnector1"/>
      <InternalLink Name="ResultTypeLink001"
        RefPartnerSideA="92007ea3-fc0b-4bb7-ad02-0c27d912db5d:ResultDefinitionconnector1"
        RefPartnerSideB="28c97fbc-6e0b-40ef-a340-e0ca466b026a:ResultTypeConnector1"/>
      <RoleRequirements RefBaseRoleClassPath="ProjectManagementRoleClassLib/ResultDefinition" />
    </InternalElement>
    <RoleRequirements RefBaseRoleClassPath="ProjectManagementRoleClassLib/MileStone" />
  </InternalElement>
  <InternalElement Name="MileStone002" ID="736d8d41-0d7a-4eeb-95ca-9d4f3313442e">
    <RoleRequirements RefBaseRoleClassPath="ProjectManagementRoleClassLib/MileStone" />
  </InternalElement>
  <RoleRequirements RefBaseRoleClassPath="ProjectManagementRoleClassLib/MileStonePlan" />
</InternalElement>

```

Figure 8: Mile-stone 1 of the project example

### 3 Product Variability Handling

#### 3.1 Motivation

Modern production systems are becoming more and more flexible both in terms of range of products (and variants within the same product) and in terms of resources employed to carry out the manufacturing process. It is therefore necessary to generate a standard and efficient representation of the system elements so that it is possible to manage the process, no matter how complex it might be and both in the design phase and when the facility is up and running.

Taking into account variability when planning for production equipment and layout is an essential task since nowadays products are highly customized and even a slight change in the order can lead to a complete reorganization of the production schedule.

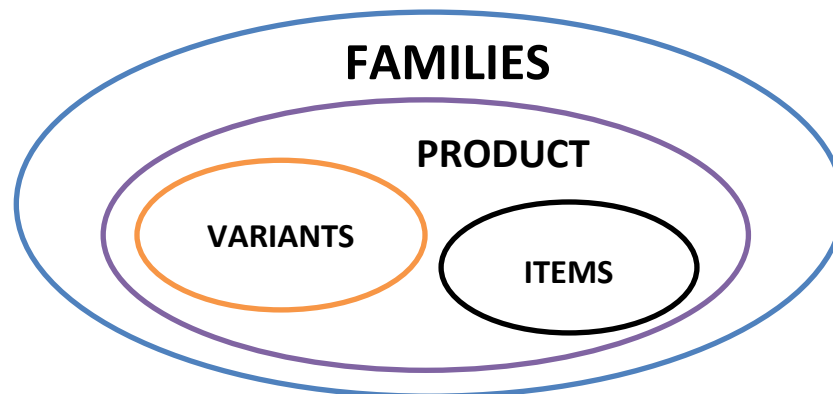


Figure 9: Product structure

A family is a group containing products that share common features and can therefore undergo similar production processes. Each product is composed by main modules, henceforth called *variants*, and secondary components called *item*.



Variants are main elements in a product that can be either chosen directly by the customer for performance or design reasons or included as a consequence of a previous choice. On the other hand, items are elements of the product that do not get chosen directly due to their minor importance in terms of customer requirements.

For instance, a variant could be the engine that the customer wants to have installed on their vehicle and as a consequence of this choice, another variant is chosen that matches the requirements generated by choosing the first one. An example could be a precise type of gear box that **MUST** be chosen since a precise type of engine has been selected.

When choosing a variant, there is a set of auxiliary items that must be selected (screws, washers, cables) for the variant assembly but that are of no interest for the customer.

A rule-based model containing all the constraints regarding variants as well as parts defines what decisions can be taken, given the current state of the system and what actions are forbidden.

Whether the target is to design a new facility, or to plan the production for an existing one, the task of finding an optimal schedule to execute operations according to specific parameters is essential. In case the scope is to design a new workstation, or cell (or even the entire plant) the independent

	<p style="text-align: center;"> <b>ENTOC</b>        Engineering tool chain for efficient and iterative        development of smart factories        ITEA 3, 15015          Project Coordinator: Thomas Bär, Daimler AG     </p>	
--	---	---

variable is the number of resources to install. The more resources one has, the faster he can manufacture the product. There are layout and budget constraints though and therefore one wants to find the minimum number of resources necessary to meet a certain capacity.

When instead, a facility happens to already exist and there are no renovation plans to increase the capacity, an efficient schedule is required to meet due dates for production components and optimize the allocation of resources. In order to be able to calculate a schedule (possibly the optimal one) some information is required:

- Precedence relations among operations;
- Operation Duration;
- Mapping Resource-Operation.

In a flexible environment the last information is not precisely defined, since there could be more resources eligible to execute a certain operation. This feature is handled by stating what abilities are required to execute and then a check is made to spot the resources that match such requirements.

There are also constraints regarding the sequence in which operations must take place. For each operation the set of operations being its predecessors must be defined.

### 3.2 Approach

The approach introduced here tackles the problems presented in the previous section and offers a solution to store the information regarding variability and scheduling problems within an AutomationML file so that it can be used to generate a complete model of the real system.

Having such model allows to calculate optimal (or at least feasible) solutions to manufacture products. We look for optimality when the problem is to maximize or minimize some cost function, very often the production time (but it could be energy consumption, tools usage...). The modern tools to calculate such solutions are growing faster and faster but the calculation time increases exponentially with the system size, therefore it is not always possible to find the true optimal in a reasonable time. Nevertheless specific-purpose algorithms and heuristics have been developed to provide “good-enough” solution in a short time.

Sometimes though, the problem is about finding out whether certain configurations can be manufactured or certain operations executed in that order without causing a dead-lock. In this case we look for satisfiability, in other terms we want to know if there exists a solution at all.

For example, if two variants of a product are mutually exclusive, there should be a rule to define it so that it is not possible to select both of them for the same product. Also, if there is a precedence constraint between two operations there should be a rule setting the starting time of the second one to be greater than the completion time of the first one.

By knowing the relations among the variants that compose a product, it is possible to figure out what configurations are feasible and what are not. It means that when a new order is about to be placed it is possible to spot inconsistencies and conflicts.

The information regarding sequence, duration and resource requirements for the operations, allows to schedule the production efficiently and safely once the product is configured. It means that it is possible to spot and avoid “dead locks” (the unfortunate situation of two or more operations that cannot be completed because each of them needs the resource that is held by the other) and to set optimization and deadline constraints, e.g. minimization of make span.



Such model is defined through a Product-Process-Resource framework, where each of the three elements contains all the data required to handle multiple configurations for the products as well as flexibility in the process in terms of resource allocation.

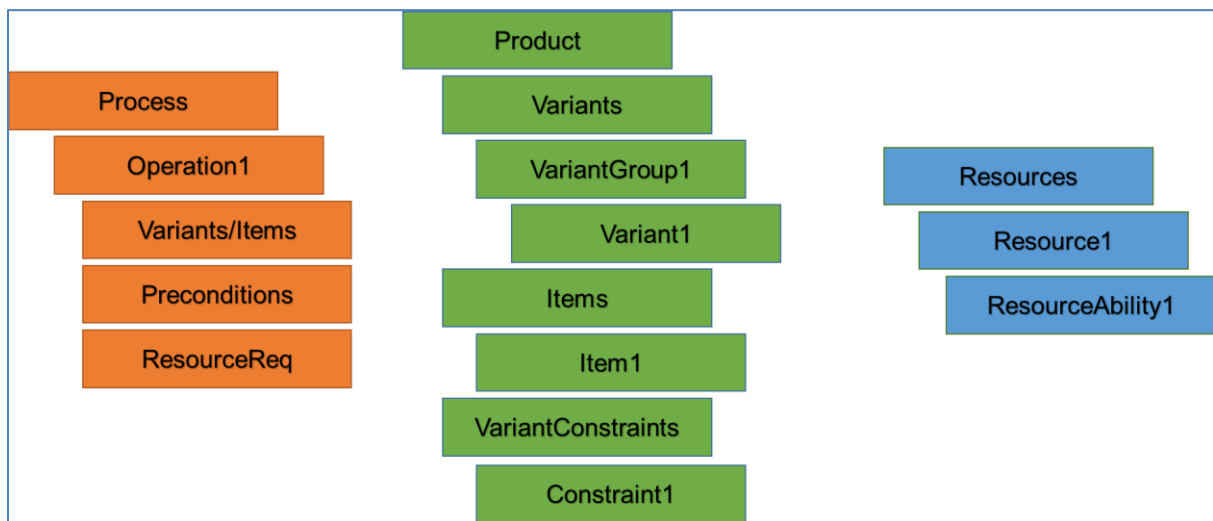


Figure 10: Product-Process-Resource structure

The AutomationML file of a product (or sub-product) contains all the alternative variants that might be chosen and therefore all the configurations of such product. These variants are grouped into lists according to their common features (i.e. a list for the engines, a list for the gear-boxes, etc.). Each of these lists have an attribute called *CARDINALITY* that indicates how many variants must/can be chosen from it. For instance, if the cardinality is EXACTLY ONE, not more nor less than one variant must be picked from that list. A separate, unique list contains all the items needed for the product to be assembled once the variants are chosen. The list VARIANT CONSTRAINTS contains all the logical conditions regarding variants and items. Whenever a variant is chosen, one or more constraint will tell whether another variant (or item) is required or forbidden.

On the process side, the overall execution (manufacturing or assembly) is split into single operations where each of them must contain information about:

- Variants and items involved in the operation;
- Preconditions - set of operations to be executed before;
- Requirements on the resources.

Finally, resources are listed in the corresponding group, and attributes are specified for each of them. Such attributes are the ones that have to match the resource requirements for a resource to be eligible to execute an operation.

To sum up, with this approach it is possible to automatically keep track of all the dependencies existing among the different components of a product as well as the ones regarding operations and resources required to manufacture such products or, in other words, of the variability within the process. Once the information is stored in the AML file, it can be retrieved to:

- Execute configuration analysis to detect conflicts
- Used as guide by customers when choosing variants

- Generate a model to feed an optimizing/scheduling solver for the resource allocation.

Besides, such approach can also be useful during the designing phase of a new cell (even the whole plant) to decide upon the number of resource required or, according to the outcome of the scheduling problem, their layout.

### 3.3 Overview of UML-Model

Figure 11 provides a formalized representation of the previously mentioned concepts as UML class diagram. The white namespaces "SchedulingRoleClassLib" and "VariabilityRoleClassLib" are later on modeled as AutomationML RoleClassLibraries, while the grey and blue colored namespaces contain class definitions assumed to be existing and authored by either the AutomationML standardisation group ("AMLRoleClassLib") or by other ENTOC partners ("ENTOCSpecRoleClassLib").

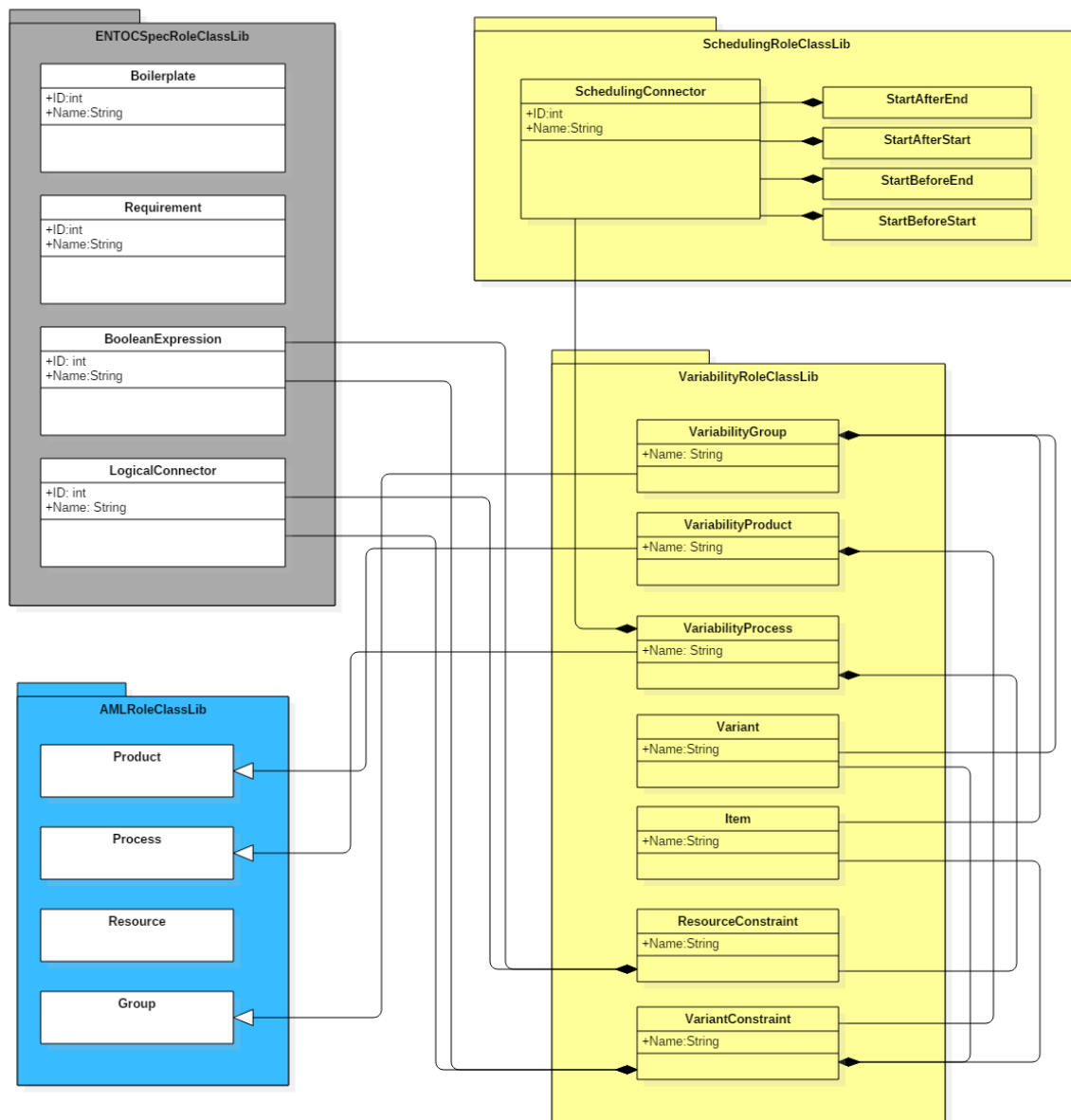


Figure 11: Overall view of the class relations

### 3.4 Example of General Functionality

The following pictures show how the frame Product-Process-Resource presented in the previous section is employed to implement the assembly of engines and gear boxes for a truck (it could be part A and B on product C, common components and product are used merely to make the concept more understandable).

In the example there are two variants for the category “Engines” and two variants for the category “Gear Boxes”, as well as a list of items to assemble them. In the process section the operations to assemble each variant are listed and the resources available to execute the operations are listed in the corresponding section.

```

▲ [IE] Truck { Class Role ProductStructure }
  ▲ [IE] Variants { Class Role Group }
    ▲ [IE] Engines { Class Role Group }
      ▼ [IE] 6_cy { Class Role Variant }
      ▼ [IE] 4_cy { Class Role Variant }
      [RR] AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Group
    ▲ [IE] Gear_Boxes { Class Role Group }
      ▼ [IE] strong_gear_box { Class Role Variant }
      ▼ [IE] weak_gear_box { Class Role Variant }
      [RR] AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Group
      [RR] AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Group
    ▲ [IE] Items { Class Role Group }
      ▼ [IE] screws_set_for_6cy { Class Role Item }
      ▼ [IE] screws_set_for_4cy { Class Role Item }
      ▼ [IE] screws_set_for_strong_gear_box { Class Role Item }
      ▼ [IE] screws_set_for_weak_gear_box { Class Role Item }
      [RR] AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Group
    ▲ [IE] VariantConstraints { Class Role Group }
      ▲ [IE] strong_engine_requirement { Class Role VariantConstarint }
        ▲ [IE] Implies { Class Role Implication }
          [IE] 6_cy { Class 4b95918b-eef8-4ea0-a784-8bd95e131c5e Role }
          [IE] strong_gear_box { Class 582de45a-1ed3-43a0-b8b7-e3de20548b35 Role }
          [RR] EntocSpecRCL/LogicalConnector/Implication
  
```

Variants and items are listed as well as the constrain implying that a “strong” gear box is required if a “six-cylinders” engine is chosen.



- ▲ IE Assembly { **Class Role** ProcessStructure }
- ▲ IE mount\_6\_cy { **Class Role** Process }
- ▲ IE Variants/Items { **Class Role** Group }
- IE 6\_cy { **Class** 4b95918b-eef8-4ea0-a784-8bd95e131c5e **Role** }
- IE screws\_set\_for\_6cy { **Class** f72ed64d-323a-442f-bff5-e277966629c9 **Role** }
- RR AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Group
- ▲ IE Preconditions { **Class Role** Group }
- ▲ IE StartAfterEnd { **Class Role** SchedulingConnector }
- IE mount\_strong\_gear\_box { **Class** 57d56614-f0f6-4089-bd6f-e40de526e54e **Role** }
- RR SchedulingRoleClass/SchedulingConnector
- RR AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Group
- ▲ IE ResourceRequirements { **Class Role** Group }
- ▲ IE XOR { **Class Role** ExclusiveDisjunction }
- IE working\_station\_2 { **Class** 10bba69e-4b49-43f3-ac7d-4b3fb0adeaa3 **Role** }
- ▲ IE LiftingCapacity { **Class Role** ResourceConstraint }
- ▼ IE GREATER\_THAN\_EQUAL\_TO { **Class Role** BooleanExpression }
- RR VariabilityRoleClassLib/ResourceConstraint
- RR EntocSpecRCL/LogicalConnector/ExclusiveDisjunction
- RR AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Group
- RR AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process
- ▲ IE mount\_4\_cy { **Class Role** Process }
- ▼ IE Variants/items { **Class Role** Group }
- ▼ IE Preconditions { **Class Role** Group }
- ▼ IE ResourceRequirements { **Class Role** Group }

Operations are listed in the section assembly and they reference to variants, required parts, resources and other operations as preconditions. The operation “mount\_6\_cy” is enabled if the variant “6\_cy” and the item “screw\_set\_for\_6cy” are chosen. It has to start after the operation “mount\_strong\_gear\_box” and it can be executed either by the resource “work\_station\_2” or by any resource having the attribute “LiftingCapacity” greater than or equal to the integer value 5.

- ▲ IE Resources { **Class Role** ResourceStructure }
- ▲ IE working\_station\_1 { **Class** working\_station\_1 **Role** Resource }
- IE Robot { **Class Role** }
- IE Tool { **Class Role** }
- IE Device { **Class Role** }
- RR AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Resource
- ▼ IE working\_station\_2 { **Class Role** Resource }
- RR AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Structure/  
ResourceStructure

Resources are listed in the corresponding section, including their specifications and subcomponents.

## 4 Process-Driven Production Planning

### 4.1 Motivation

A core task in production planning of automotive engineering is the process planning. The goal of this task is to find the most efficient way to produce a product. In this context efficiency is defined as the fastest, most value adding and well-balanced sequence of process steps over a production line with respect to the project premises like cycle time, amount of variants, requirements of technologies and environmental requirements.

As a result of the product analysis a joining sequence depending on joining technologies and stable product states, to ensure the ability to handle the product, is usual created by a product development team. All steps of this joining sequence are direct value adding process steps to reach the feasible structure of the product by creating fasteners like weld points. Next to this process steps additional steps are required to create further joining features to reach the intended stiffness and optimal crash behaviour of the product. These additional steps serving the potential to optimize and balance the production process steps by analysing the various possibilities of joining the product.

Until this point no processes of product handling (e.g. from carriers to fixtures) are planned or defined. These process steps do not increase the state of the product, because of that this process steps are defined as non-value adding process steps.

As an essential part of production planning, the production process is the link between the goal of production, the product, and the required tools of production, the resources (see Figure 12).

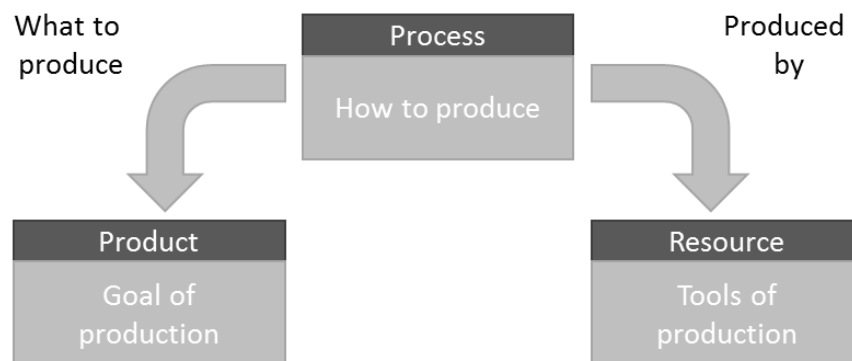


Figure 12: Process as link between product and resources

In the popular used tools for production process planning a process step is mapped by a generic object to create the link between product and resource via the process by simply referencing the affected objects out of the scoped model without differentiation of the references. Some tools provide the ability to select types of technologies for process steps (e.g. spot welding, generic process, human operation, etc.) with the same disadvantage of no distinction of referenced objects (e.g. input element, joining element, output element or used resource).

In context of ENTOC this gap in model-based production process planning is closed by an approach of extended process models to enable more detailed process planning, automated assignments of resources by required capabilities and centralized access of engineering information of product, process and resource with the ability to auto generate requirement specification of the described process.

## 4.2 Approach

Following the definition of IEC 60050-351 a process is a set of interacting actions in a system that transforms, transports or stores material, energy or information.

With the VDI/VDE 3682 guideline a concept and graphical representation of formalised process description is released and represents the definition of a process as shown in Figure 13.

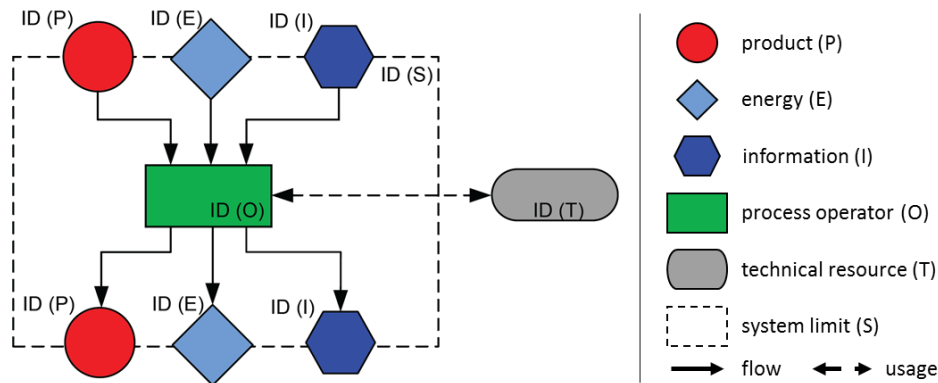


Figure 13: Formalized process description

In general a process step is described by a start point, a duration, an end point and one or more successors.

To reach the goal of work package 2 in ENTOC, a formalized requirements language, the introduced definitions do not match the level of detail to generate useful information about the described process. With DIN 8580 a hierarchy of classified manufacturing processes and by VDI 2860 a classification of assembly and handling processes are published. In the next two sections the essential differentiation of these are described and extended to get the base for a formalized technique of process description.

### 4.2.1 Manufacturing Processes

As mentioned in the previous section the DIN 8580 – manufacturing processes classifies manufacturing processes by their characteristic. The hierarchy reaches up to four levels in addition of the detailed process description of the classified element.

This section introduces the six most characterizing classifications of manufacturing steps. Concrete manufacturing steps like gluing, screwing, spot welding, etc. are derived for example from the manufacturing process joining and have more detailed characteristics. For a better readability the description is limited to the first level with the most elemental characteristics and focused on the product dependent process information. The elements information and energy are not used in this context, because of convention that the system limit describes product changes and no information or energy transformation.

**Primary Shaping**

Primary shaping is defined as manufacturing a solid body out of shapeless fabric by creating cohesion. Figure 14 depicts the concept of primary shaping in the graphical representation of formalized process description.

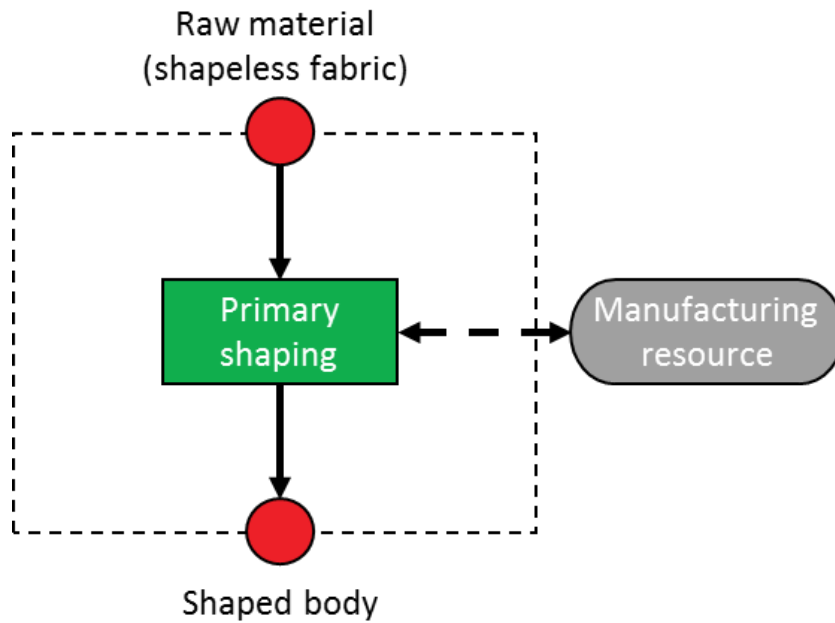


Figure 14: Concept description of primary shaping

**Forming**

Forming is defined as manufacturing a solid body by plastically changing the shape of the origin solid body while maintaining the mass and cohesion. Figure 15 depicts the concept of forming in the graphical representation of formalized process description.

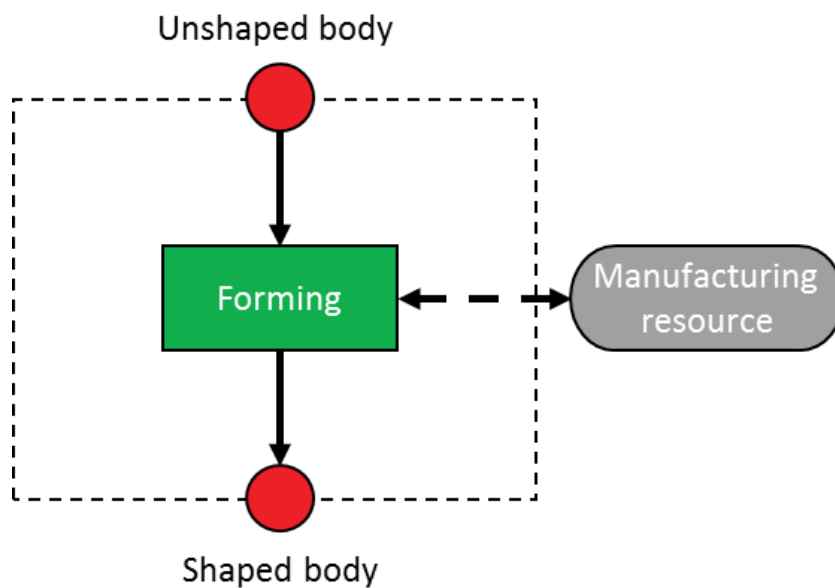


Figure 15: Concept description of forming

### Cutting

Cutting is defined as manufacturing by resolving cohesion of bodies. Thereby the cohesion of the bodies is partly or in total reduced. Figure 16 depicts the concept of cutting in the graphical representation of formalized process description.

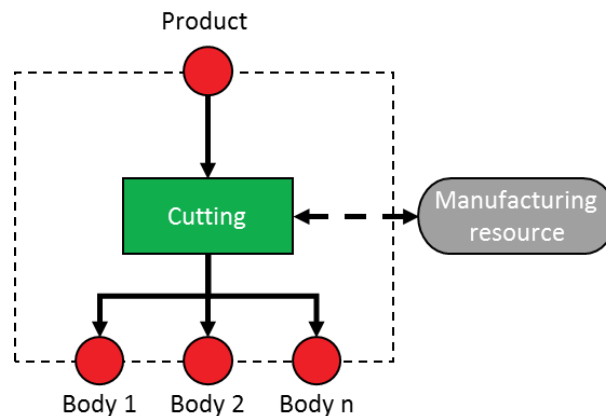


Figure 16: Concept description of cutting

### Joining

Joining is defined as the creation of a local connection of two or more geometrical defined solid or shapeless bodies by increasing the cohesion of the workpiece. Figure 17 depicts the concept of joining in the graphical representation of formalized process description.

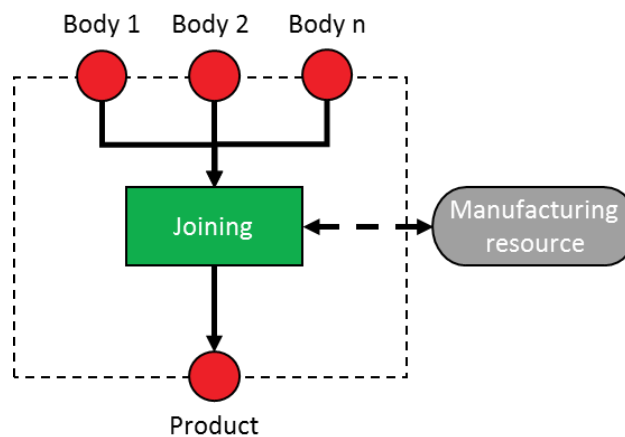


Figure 17: Concept description of joining

### Coating

Coating is defined as applying of firmly adhering layers of shapeless fabrics onto a workpiece. Characterizing is the state of the layer immediate before application. Figure 18 depicts the concept of primary coating in the graphical representation of formalized process description.



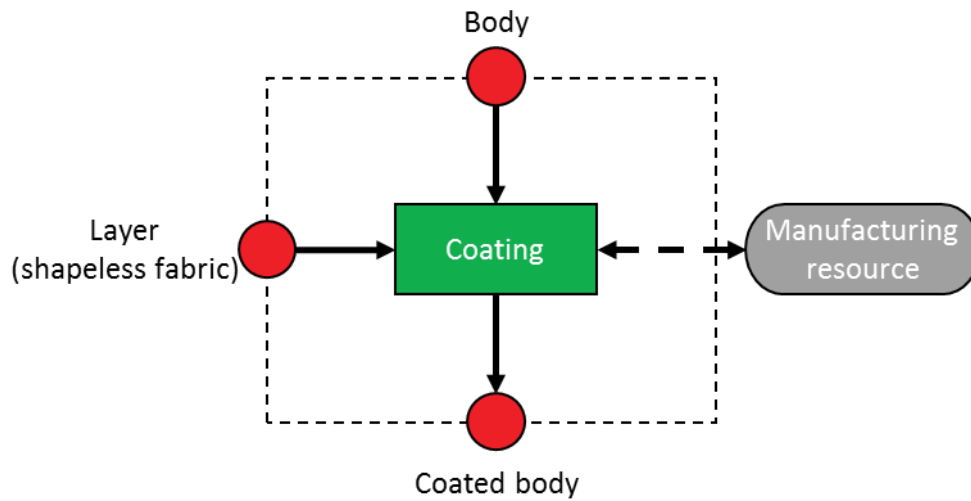


Figure 18: Concept description of coating

### Property Changing

Property changing is defined as manufacturing by changing the material properties of a workpiece. This is applied on sub microscopic and atomic level by diffusion of atoms, dislocations in atomic lattice or chemical reactions. Figure 19 depicts the concept of primary shaping in the graphical representation of formalized process description.

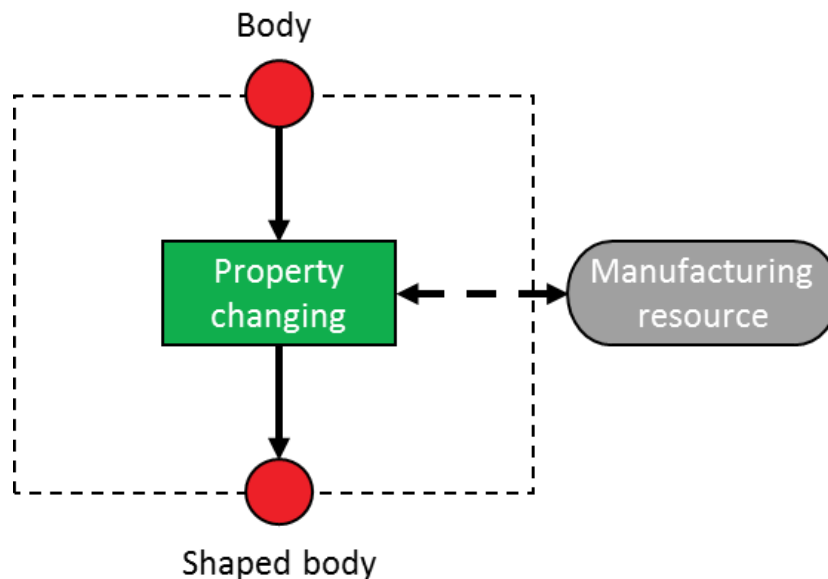


Figure 19: Concept description of property changing

### 4.2.2 Handling Processes

As defined in the guideline VDI 2860 – Assembly and handling, handling is the creation, defined changing or temporarily maintaining of spatial arrangement of geometrically defined bodies related to a coordinate system. Conditions like time, amount or trajectories can be pretended. In general handling processes are required to describe material flows. The guideline differentiates between five functions: storing, amount changing, moving, securing and verification. The most formative characteristic is the description of the carried objects' location depending on the definition of the degrees of freedom.

### 4.3 Overview of UML-Model

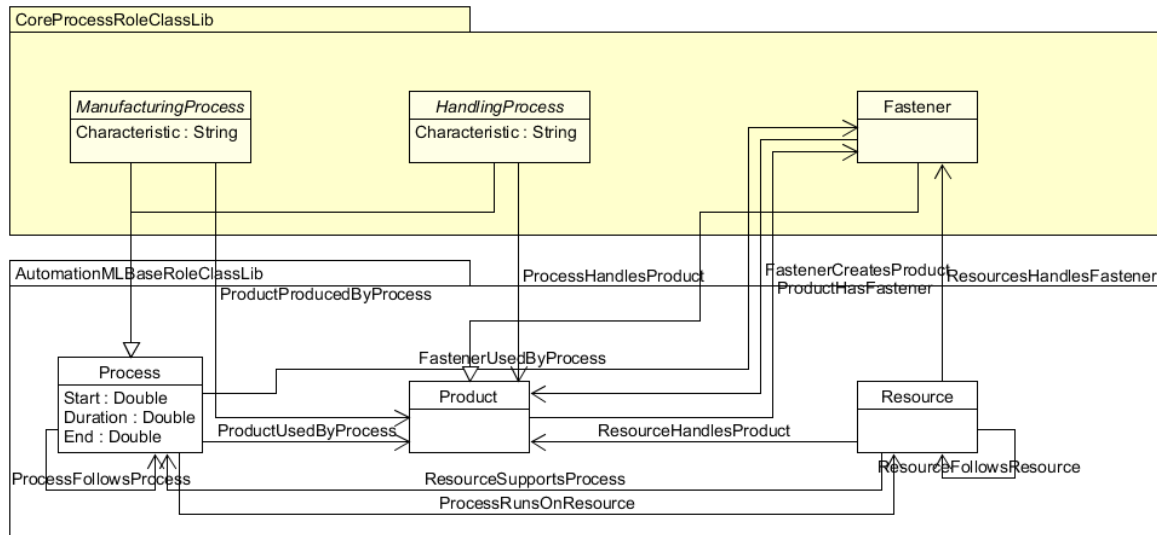
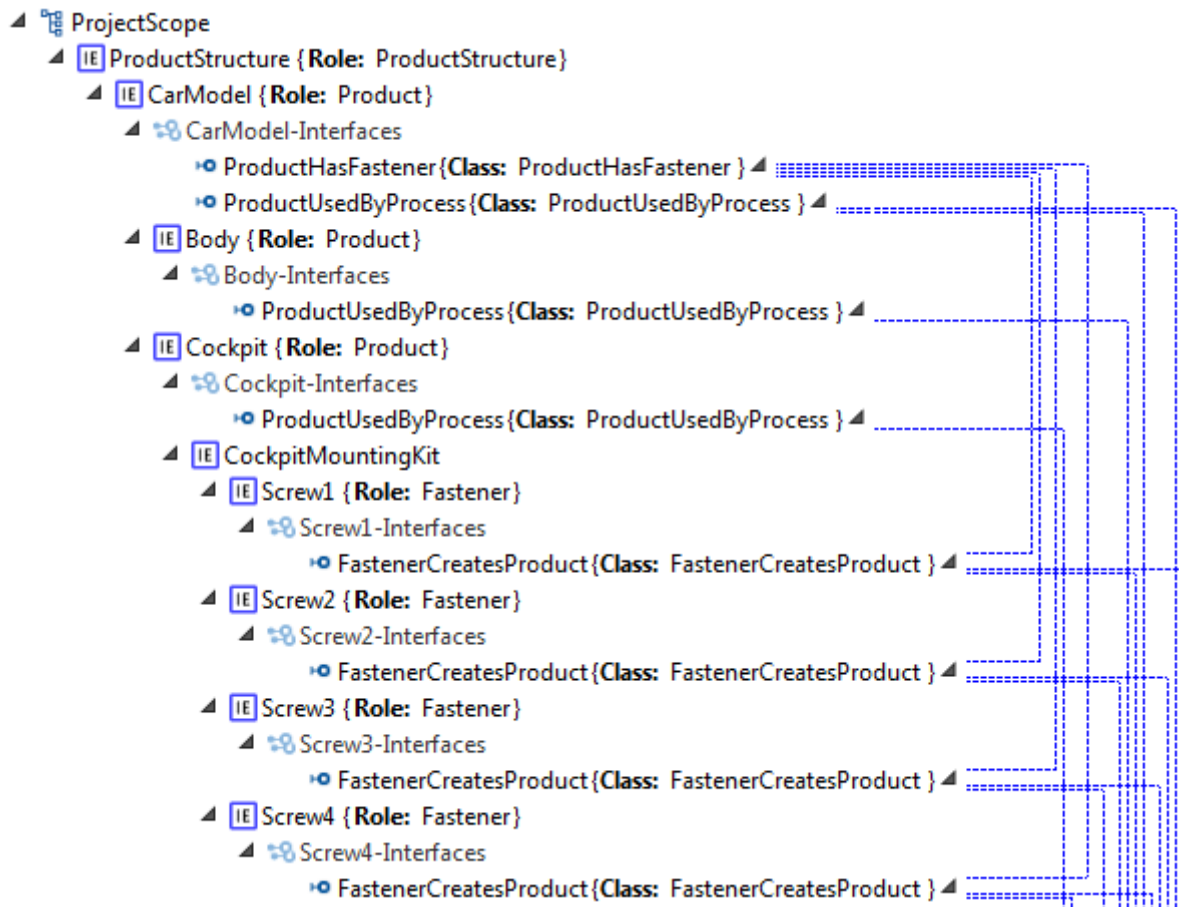


Figure 20: UML overview of the model elements

### 4.4 Example of General Functionality



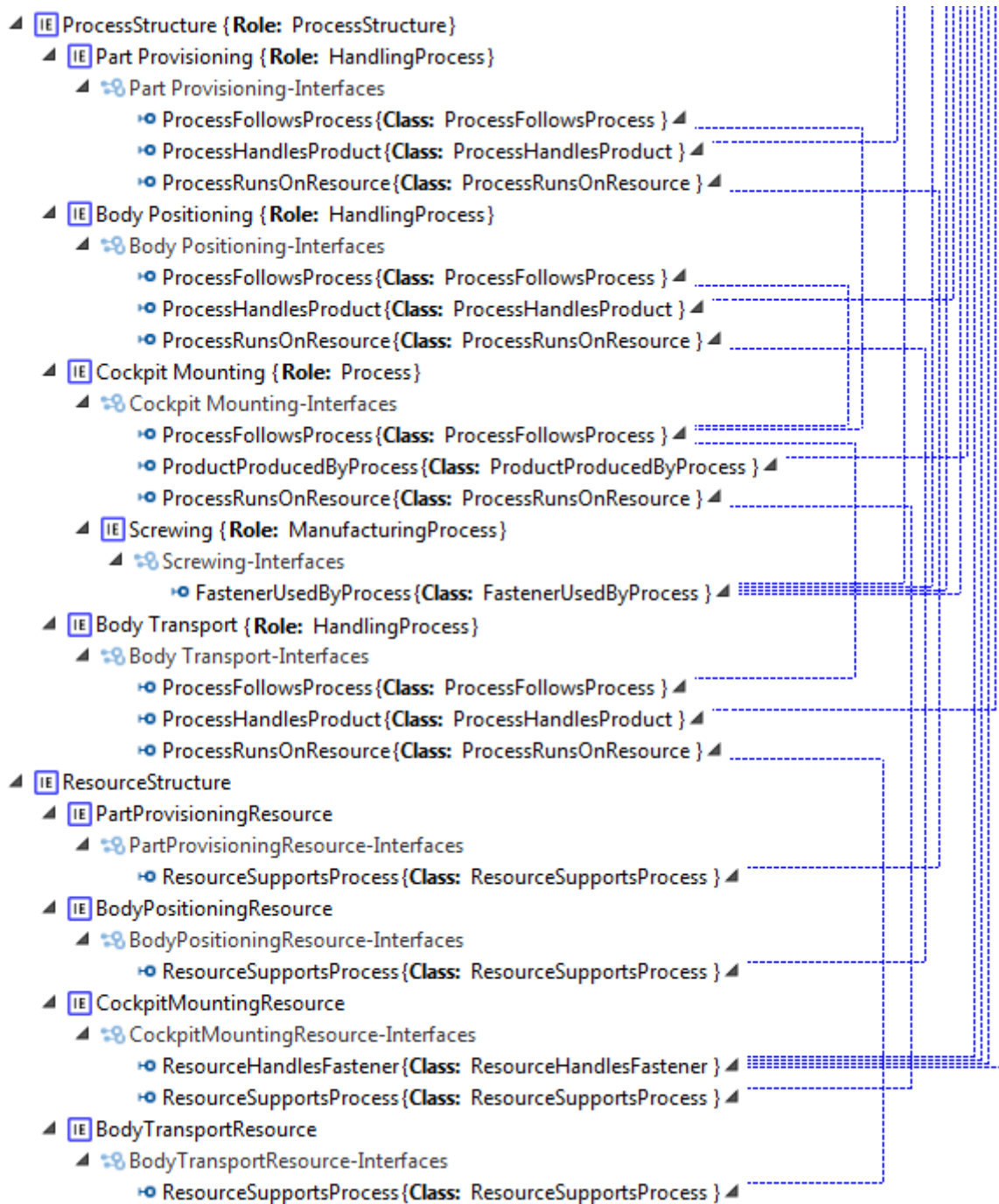




Figure 21: Example of formalized process description

	<p style="text-align: center;">ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p style="text-align: center;">Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

## 5 Production Resource Planning

### 5.1 Motivation

Planners of material handling systems have to consider aspects of:

- Layout design
- Material flow characteristics
- Construction characteristics
- Process description

Depending on the current level of detail and planning phase different data are necessary and tools are used.

During the requirements specification phase, different levels of details are available and need to be represented in the requirements documents. For some parts of a future factory layout, basic requirements can be as abstract as defining that a material transport has to be realized, without further requirements concerning the transportation means. Requirements specification may also include requirements describing some characteristics of the good being handled. This ranges from general requirements defining i.e. the mass to be transported and the required transportation time or speed, up to detailed requirements specifying load carriers to be used or specific conveying technologies like automated guided vehicles (AGV). Other requirements necessary to adequately describe, and later simulate, a production system's material handling resources and processes include logical attributes like material flow predecessor-successor-relations.

AutomationML offers generic possibilities to model production resources. Various model variants are possible, desired and necessary for different use cases. To be able to model these requirements using a common approach for modelling material handling related resources and their connections based on an AutomationML RoleClassLibrary has been developed.

### 5.2 Approach

#### 5.2.1 Basic Concept

The primary approach is to define reusable requirements by categorizing and dissecting the material handling components according to their functionality. Classifying a component according to its mechanical properties is especially not adequate for early design phases because the technical implementation might not be considered yet. The role classes should accompany the planning of material handling from layout-requirement-planning phase to on production phase. An abstract function defining role can be assigned to a requirement component in the concept design and refined later during the detailed planning. An example of this approach: Conveyors are not categorized as belt conveyors, chain conveyors, band conveyors, etc. but the conveyor is considered as a functional role *MaterialHandlingConveyor* and the load carrier is implemented as a second role *ConveyorTechnology*, which ascertain if it is a belt, chain or band conveyor. An element, which perform the conveyor role, can contain a child element to define the required conveyor technology. More examples of the concept are given in section 0. The core modelling concepts consider two different levels of detail. The module level focuses on the resource module descriptions and the possible transport directions and routes excluding the technical details of the linking. The port level focuses on the connection points between the requirement modules. An AutomationML document may include only one or both of the modelling levels.

### 5.2.2 Module Level

Figure 22 depicts a small example requirements scene. There are three conveyors required, one shall be the product source and the other two sinks. In between is a turntable, which is required to divide the material flow.

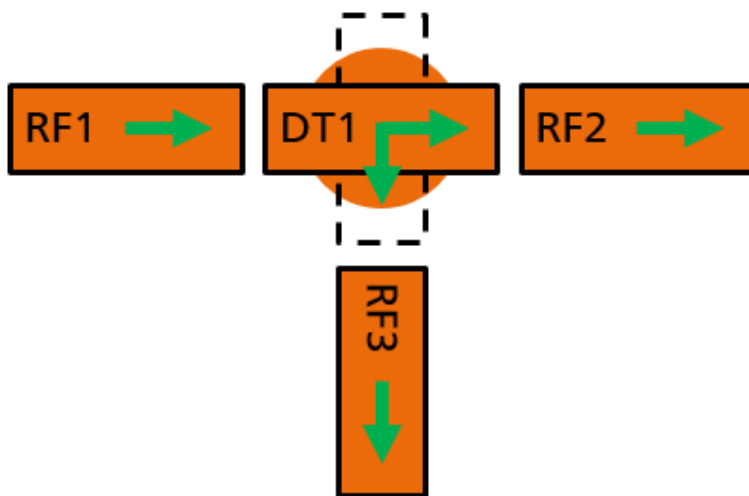


Figure 22: Example scene with four conveyors

Figure 23 depicts the module level. All conveyors are modelled as *InternalElements* representing requirements and have a direct or indirect reference to any role of the *MaterialHandlingRoleClassLib*. The required transport directions are modelled with a predecessor-successor semantic. Therefore, the resources shall have *ExternalInterfaces* of the type *MaterialHandlingConnector*. It is not defined if there is a one-to-one relation between these interfaces or not.

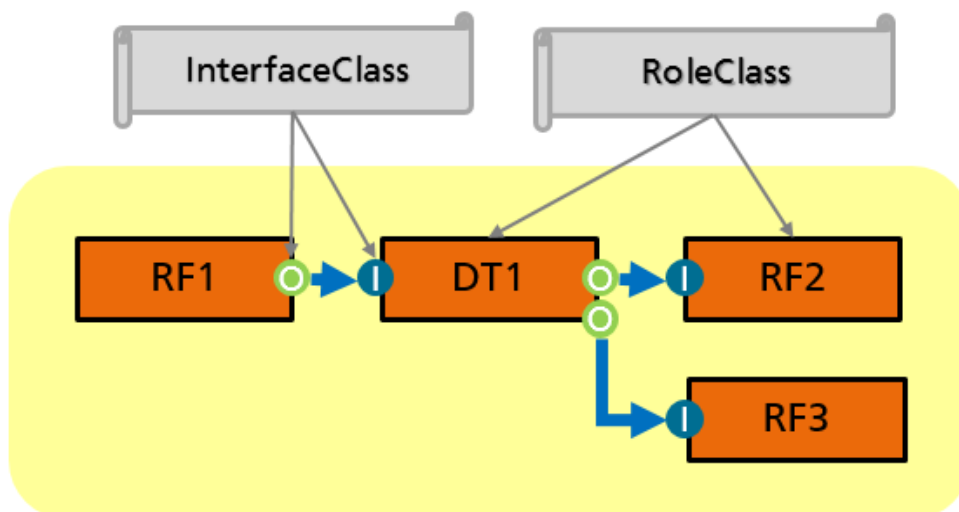




Figure 23: Modeling of Level 1 (Module level)

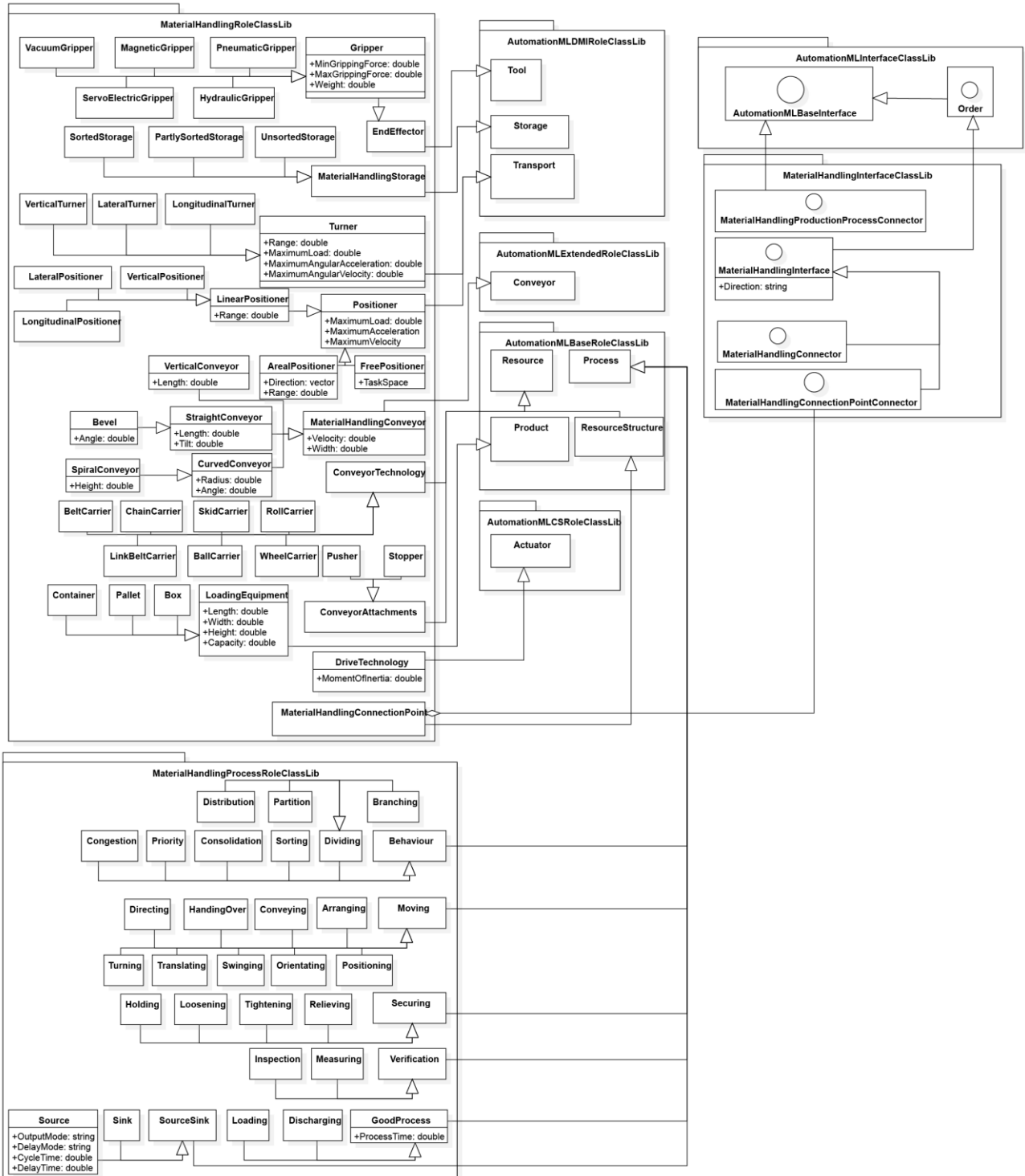
	<p style="text-align: center;"> <b>ENTOC</b>        Engineering tool chain for efficient and iterative        development of smart factories        ITEA 3, 15015        Project Coordinator: Thomas Bär, Daimler AG     </p>	
--	---	---

### 5.2.3 Port Level

The port level requirements design, depicted in Figure 23, offers more modelling possibilities. It adds connection port descriptions to each material handling resource and offers a more detailed specification of the required connections between the modules. In the example, the conveyor objects get child `InternalElements` of type `MaterialHandlingConnectionPoint`. These connection points shall include each an `ExternalInterface` `MaterialHandlingConnection` of the type `MaterialHandlingConnectionPointConnector`.

The connection points may e.g. contain a `Frame` attribute, defined in IEC62714 part 3, to specify the geometric position relative to the parent resource. It is possible to extend the geometric information for example for required handover areas. Other possible extensions may be links to logic, communication, wiring or other requirements.

### 5.3 Overview of UML-Model



## 6 Production Facility Requirement Specifications as a Result of PPR-Model-Based Production Planning

### 6.1 Motivation

With the rising complexity, increasing variant diversity and the goal of shorter time-to-market the importance of simultaneous engineering is as high as it never was. This aspect implicates challenges in every phase and discipline of the product development process.

Due to the gap of seamless engineering data provision in the engineering process of production facilities [Sch2017] the three most time-consuming challenges of a production planner are managing changes of premises and products, transfer data from concepts into requirement specifications and clarify misunderstandings and missing information (see Figure 24).

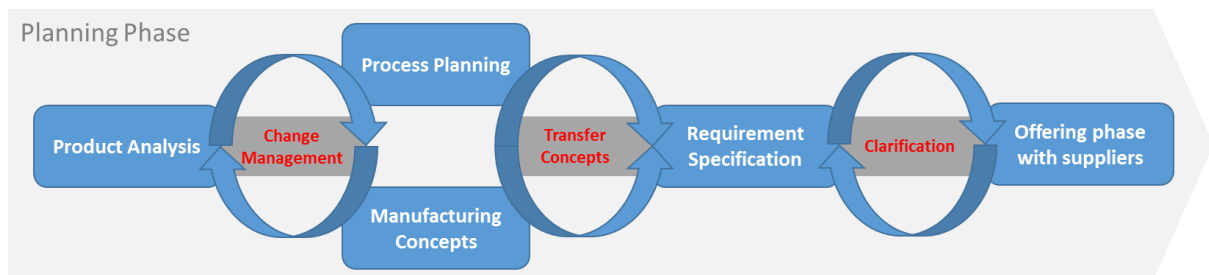


Figure 24: Today's production planners pain points

Through the front loading of phases in the product development process the planning phase of production facilities starts with the initial project kick-off. As a consequence of this the initial information about the product is in a very abstract state. This turns into that the product analysis and the process planning are running in parallel with the disadvantage that product changes will affect the recent process plan. The process plan is a strategic point of deriving manufacturing concepts. Finally, a product or project premises change could affect the whole planning phase. At this point the importance of an established change management is the key to success.

The second time-consuming challenge is the transfer of manufacturing concepts into requirement specifications. The discipline of writing requirement specifications for production facilities is also affected by changes in the product development process like the process planning and derivation of manufacturing concepts discipline. The requirement specification is the total description of the production process with required production facilities to produce a product.

Another, mostly underrated, phase is the offering phase between customer and supplier. In this phase several rounds of clarification are needed to ensure a common understanding and agreement of the requirement specifications.

All involved disciplines have dependencies to the others. Process planning needs information about the product, to derive manufacturing concepts and for resource planning information about the product and the process are required. With this chain of engineering data flow product-process-resource-model is addressed and has the ability for automation of workloads in the planning phase. In chapter 6.2 an approach is introduced which focusses the challenge of transfer concepts into requirement specifications depending on PPR-model-based production planning.



### 6.1.1 Requirements Engineering

Requirements engineering is a cooperative, interactive and incremental process which obtains the development of requirements out of abstract specifications with the goal to determine, analyze, understand and establish requirements [Gil2014].

The notion of Requirements engineering originates in software engineering where the methodology is now established for over two decades. With the increasingly spreading of the methodology tools were developed to support the creation process and validation of requirements so that there are common models of requirements [Man1995, Rup2014], like the core of a requirement (see Figure 25) from [Rup2014].

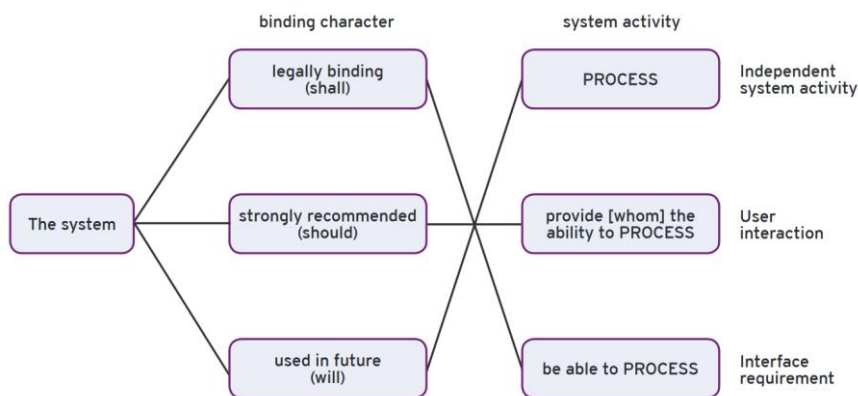


Figure 25: The core of requirements [Rup2014]

The core model of a requirement describes three significant artefacts of a requirement: subject, binding character and the system activity. The subject represents the system on which the requirements are addressed. With the binding character the necessity has to be described in one out of the three possibilities. The System “shall” is the strongest type of necessity and describes a legally binding character of the requirement. The both other types “should” and “will” are weaker types and imply that this requirement could be prioritized against other requirements. These characters are not recommended to use in requirement specifications for production facilities. The third artefact of the core requirement model is the system activity which allows to characterize the requirement depending on the activity to fulfill.

Another wider requirement model is the universal requirement template of Schwärzler. The model extends the core model of a requirement with object, action and condition and supports a generic way to generate requirements [Sch2008].

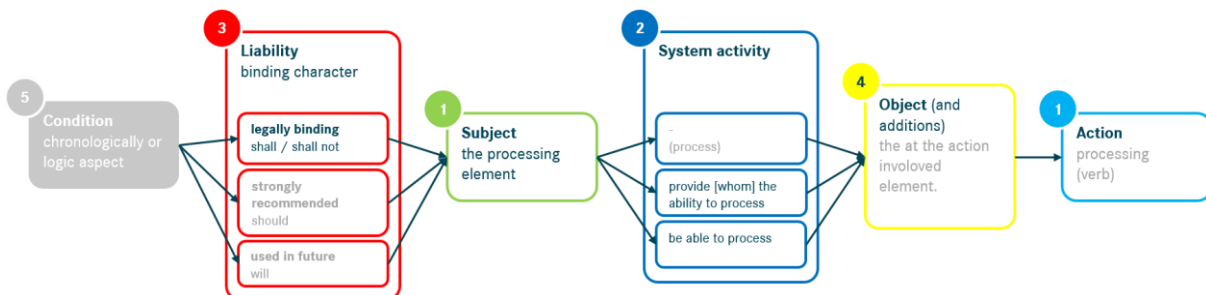


Figure 26: The universal requirement template [Sch2008]

In addition to the introduction of the core model of a requirement the generic artefacts of a well-formed requirement are:

**Action**

The action addresses the process characteristic. This element has the dominant effect of the character of a requirement.

**Subject**

The subject addresses the processing element in context of the requirement.

**SystemActivity**

The system activity addresses the dependency of the requirement.

**Liability**

The liability allows to prioritize a requirement by the strength of it.

**Object**

The object addresses elements which were involved to process.

**Condition**

The condition enables a requirement to describe a required state related to properties.

**6.1.2 The Structure of Requirement Specifications**

As mentioned in the introduction a requirement specification is the total description of the production process with required production facilities to produce a product, which means that four essential elements have to be described (see Figure 27):

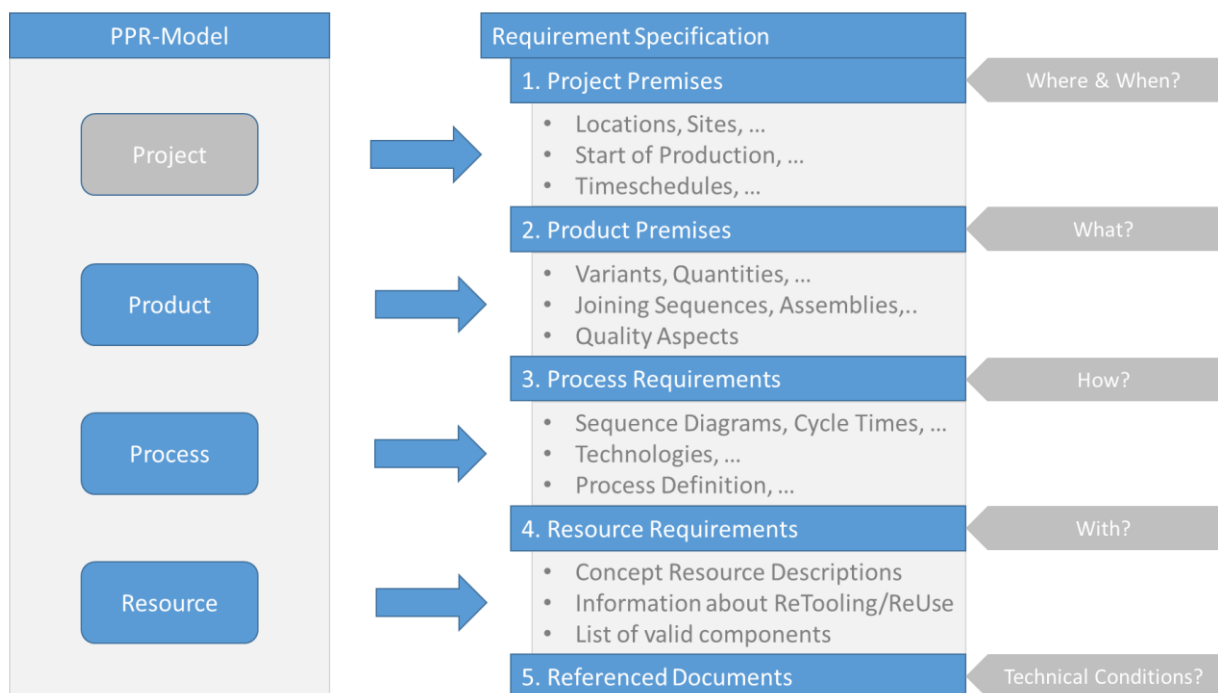


Figure 27: Comparison PPR-Model with requirement specification

In general the project is not part of the PPR-Model, but it addresses elements of it. Same like the requirement specification does. A more integrated model between PPR, Projects and Requirements enables the capability of improvement to automation and validation in the planning phase.

The project describes premises concerning locations, sites, the start of production (SOP) and a time schedule with milestones and related deliverables. The product element of the PPR-model describes the hierarchical structure of the products with technical information properties. A conception of a sequence diagram with cycle times and required technologies is described with the process element of the PPR-model. Resource requirements conceptions of production facilities are described within this chapter and information about ReTooling and ReUse is provided.

## 6.2 Approach

### 6.2.1 Methodology for Optimization of the Planning and Engineering Phase

The approach of auto-generation of requirement specifications requires initial data. Therefore a methodology was developed in [Sch2017]. The methodology consists of six essential steps, which could have an iterative behavior.

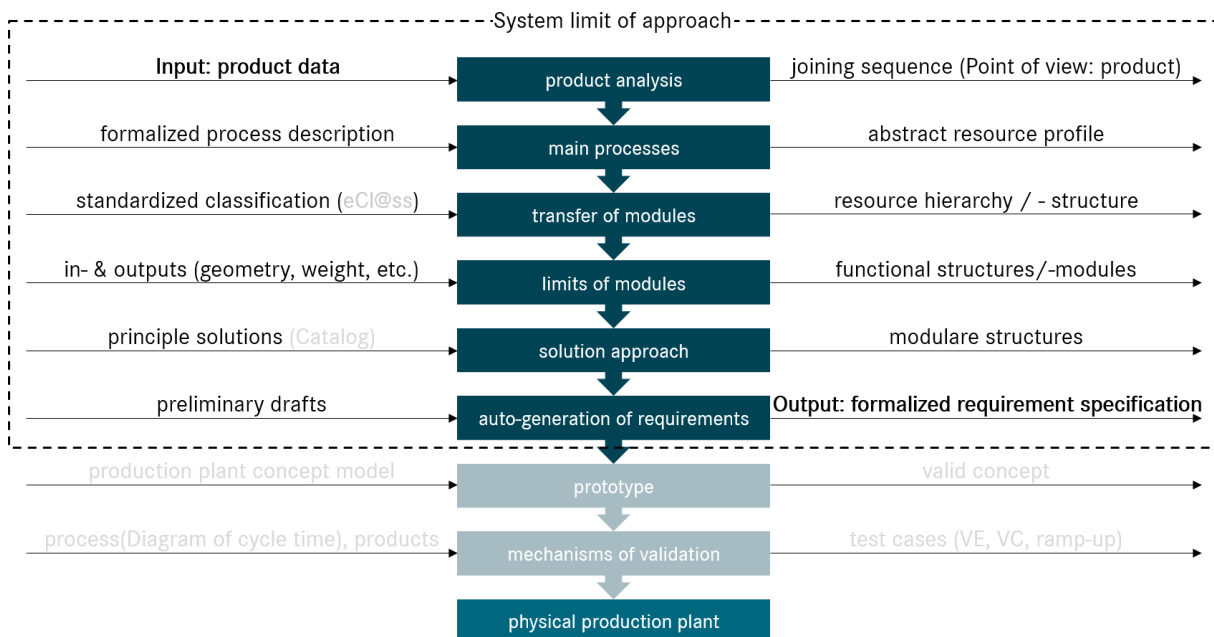


Figure 28: Methodology of production planning for auto-generation of requirement specification [Sch2017]

In general, the production planning phase starts with the product analysis. This step needs product data as input. Typically this data is in the early phase alphanumeric data or references to predecessor products and will be more sharpened in detail over the product development process. The product analysis results in having information about the joining sequence with joining technologies and striking information about parts of the product.

With the second step, the product based on the joining sequence is described as an abstract joining or manufacturing process related on empirically determined plan cycle times of manufacturing processes, e.g. a weld spot needs six seconds for welding. This process has to be described in a formalized format. The result of step two is a required abstract profile of a production facility resource. The abstract resource profile will be transferred into resource modules.

As input for the third step, a standardized classification of resource modules is required. With the example of welding weld-spots a weld gun is required, which could not fulfill the necessary process step as single resource. Therefore a robot weld module is required. By getting a more detailed resource the step three results in a hierarchical and structural resource profile. The now more detailed required resource profile needs information about limits of the modules. As limits product information, e.g. dimensions or weights, or resource standards and restrictions are possible inputs, also outputs of the previous steps can define depending limits. The output of the step limits of modules is an extended resource structure with a functional view.

The last planning relevant step of the methodology is the solution approach. With this step already used or existing resource modules can be used to replace required functional modules. The result of this step is a validated modular structure of the required resources which matches the general structure and format of the documentation, which has to be returned in detail at the acceptance test of the physical production facilities at the end of the engineering phase.

The step solution approach is optional to reach the last step of the methodology, because of the restriction of the creativity of the suppliers. With carrying out all the steps a well-formed PPR-Model is created.

The last step of the methodology is the auto-generation of requirements. Based on the well-formed PPR-Model formalized requirements (base concept) [Sch2017] and formalized requirement specifications (extended concept) can be derived. In the next two chapters these both concepts are introduced.

### 6.2.2 Base Concept of Auto-Generation of Requirements

In [Sch2017] an approach to auto-generate requirements is introduced. The approach focusses on the perspective of process description as the source of information to derive well-formed requirements. For the schematic representation the recommended form of graphical representation of VDI/VDE 3682 [VDI2015-1, VDI2015-2] is used.

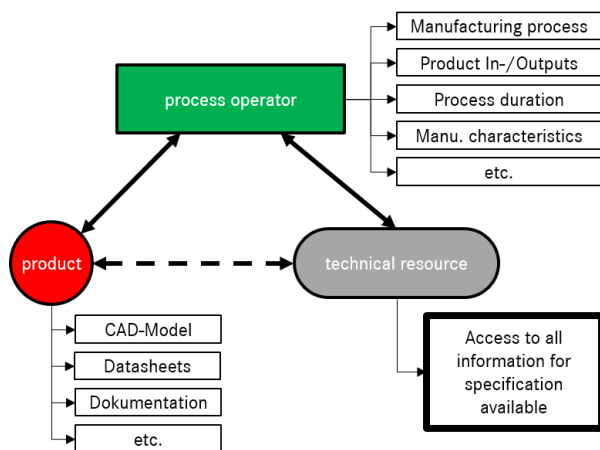


Figure 29: Concept of PPR-Relation Model [Sch2017]

As shown in Figure 29 the process operator is the essential element of the PPR-model, which builds the bridge between products and resources. The technical resource and product is only able to get information via the process operator. This behavior is matching the method of production planning. The product should be isolated from the resource as much as possible, because of the probability of product and premises changes. This behavior supports the ability of a validation method for change

management. The procedure of the introduced methodology in short can be described in a sequential form, see Figure 30:

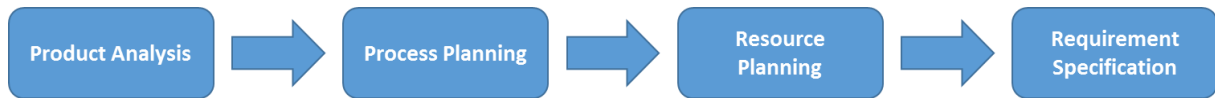


Figure 30: Simplified sequential diagram of creation of PPR-Model

The process operator is the accessor for information of a scope in the PPR-Model. With the relation to the products and the characteristic of the process operator engineering information of product and process is accessible from the technical resource.

The clue of deriving/auto-generated requirements is the combination of scoping the PPR-Model and transfer information into a configured requirement model template (see 6.1.1). The general functionality is illustrated in Figure 31. Within this use case a gluing process operation is scoped as the system limit for deriving requirements. As input products the base product, on which additional products shall be glued on. In this case the additional product is called “part\_to\_glue”. To glue products together glue is as necessary as the products. In this case glue is the connecting element and also a product with specific information. Via the process operator all products are referenced which results in a set of information which can be transferred into a set of requirements, see the tables on the right side of Figure 31.

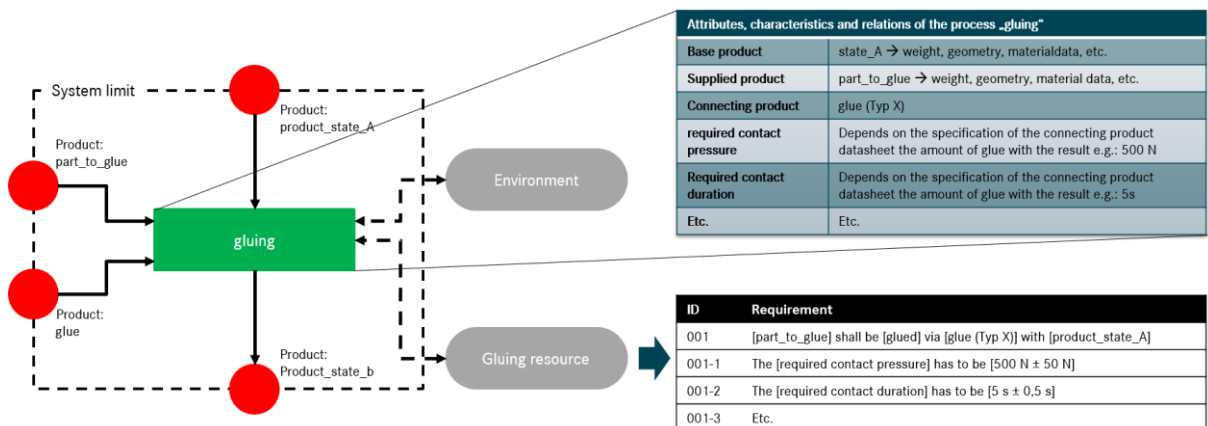


Figure 31: PPR-Relation for auto-generated requirements [Sch2017]

The set of requirements can be uniformed into a structured set of requirements in order to create a pre-configured requirement specification template, see table on the right bottom of Figure 31.

### 6.2.3 Extended Concept of auto-Generation of Requirement Specifications

The specification process operator is not capturing all perspectives of the PPR-Model by using the base concept (see 6.2.2) and the structure of a requirement specification (see 6.1.2). Thus description of project, product and quality premises should not be neglected beside the process and resource requirements descriptions. For this further information the base concept requires the capability to switch the perspectives between project, product, process and resource. Quality is in this case secondary information dependent on the four perspectives. The method of getting information via accessors can be adapted.

Another challenge next to extending the base concept by a perspective switch is the extension of the requirement model template in a more flexible and generic form. This extension is introduced in the next chapter, see 6.2.4.

### 6.2.4 Extended Boilerplates and Templates of Requirements

A much wider established term of a requirement template in requirements engineering is the so called boilerplate, which enables pre configuration of requirements. Demonstrated on Figure 25 the universal requirement template is able to be configured in 9 (3<sup>2</sup>) ways via the liability and system activity only.

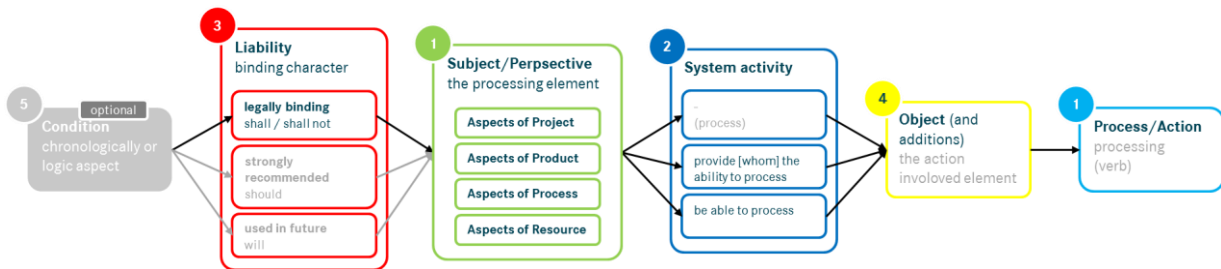


Figure 32: Extended requirement model template

To match the needs of the previous chapters the universal template has to be modified with the ability to provide a perspective. The perspective in context of a requirement affects the subject, which aligns the scope of the requirement.

In addition to the extensions of the concept and the requirement model template, a general characteristic identifier has to be considered to identify the shaping of the requirement. As an example the optionality of the condition or the cardinality of subjects, objects and conditions has to be considered. Therefore Figure 33 illustrates the UML diagram of the extended concept.

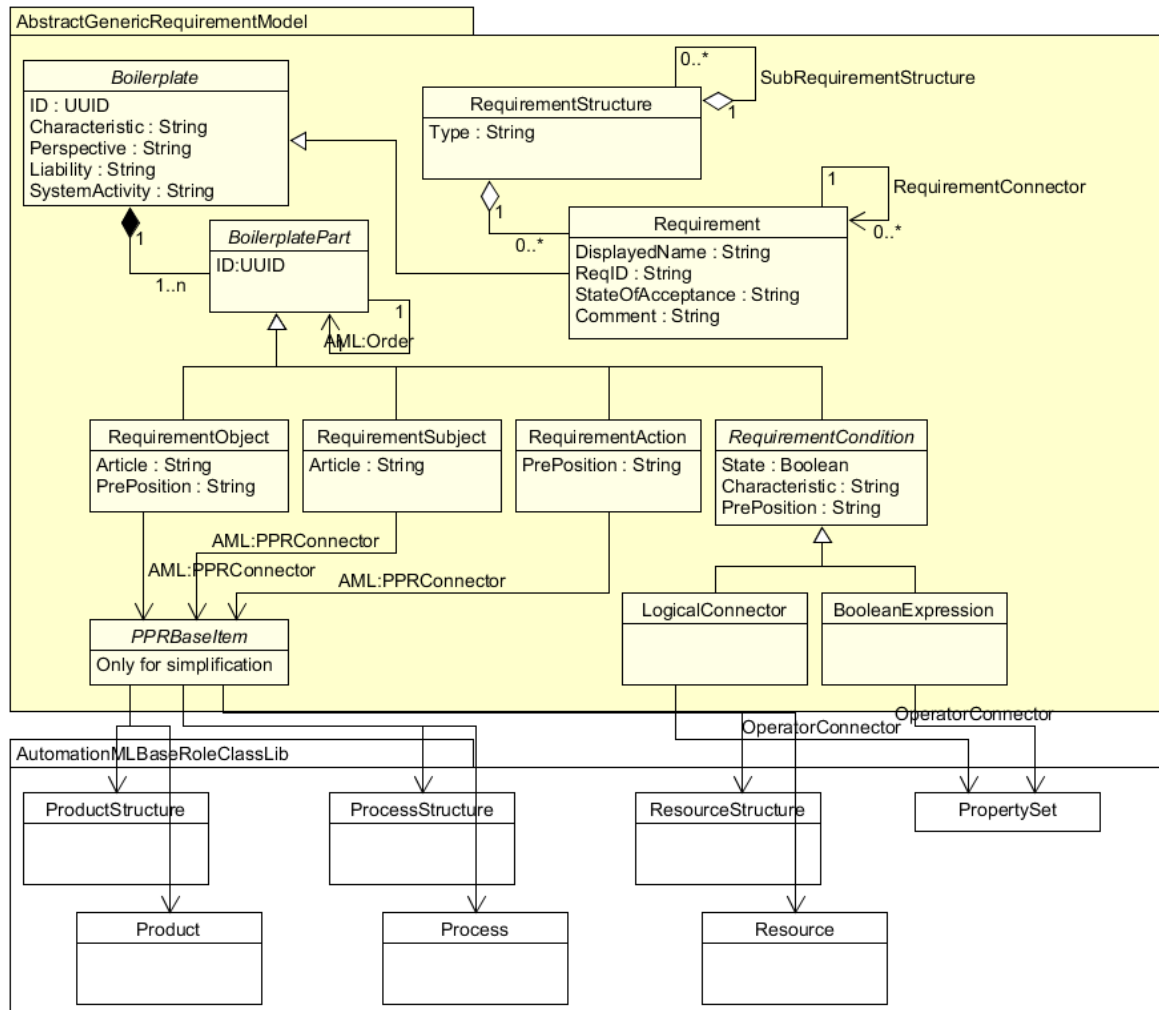




Figure 33: UML Model of the extended requirement model boilerplate

The illustrated model in Figure 33 contains several special features which are worth for mentioning. With the object “RequirementStructure” the structure elements of a requirement specification will be depicted. To generate a requirement as a human readable sentence articles and prepositions are necessary parts, which are depending on the related noun or, in case of the action, verb. The human readable requirement representation is stored in the attribute “DisplayName” of the object “Requirement”. For a better readability user friendly IDs were generated and stored in the attribute “ReqID”. At this point it is also worth of mentioning that there is a standardized requirements interchange format “ReqIF”, which also will be supported by the implementation of the approach. There are two further attributes in the object “Requirement”, “StateOfAcceptance” and “Comment”. These attributes are enabling the model to support the exchange of the requirement specification between customer and supplier. The following chapter contains a further description of the benefit of this approach in the planning phase and development process of resources.

### 6.2.5 Benefit of PPR-Model-Based Production Facility Requirement Specification

The introduced methodology including the extended concept and boilerplate focuses on a seamless and efficient workflow for production planning, which closes the gap of information provision in the

	<p style="text-align: center;"> <b>ENTOC</b>        Engineering tool chain for efficient and iterative        development of smart factories        ITEA 3, 15015        Project Coordinator: Thomas Bär, Daimler AG     </p>	
--	---	---

development process of production facilities. The goal is the elimination of the three addressed pain points in production planning mentioned in the introduction.

**Pain Point 1 – Change Management of Project and Product Premises**

Through the passive referencing of product data with process operators and resources, the implementation of validation mechanisms support change management related to the degree of PPR description, which results in an overview of tasks to get a valid model. An overview of required tasks to get a valid model, enables the adaption of validation automatisms.

**Pain Point 2 – Transfer of concepts into Requirement Specifications**

With the extended concept, an approach was introduced to derive/auto-generate or extract requirements out of the PPR-Model. The structure of the specification document can be generated out of a generic model or out of the hierarchies of the PPR-Model.

**Pain Point 3 – Clarification during the Offering Phase between Customer and Supplier**

The clarification of misunderstandings or missing information is owed by the manual creation of requirement specifications. Today's established standard for requirement specifications is a text document written in prose with copied screenshots. With this approach the requirement specification is interchanged as an integrated model. With formalized functions, methods and objects, technical misunderstandings and dialects in prose text are avoided. In the application of the model on the supplier side, the both mentioned attributes "StateOfAcceptance" and "Comment" enables the capability to the supplier to directly edit the model on the requirement aspects. This supports focusing on requirements which were not accepted and leads to shorter and efficient clarification rounds. By accepting all requirements a common understanding is ensured.

**6.3 Overview of UML-Model**

In dependence of the model description in 6.2.4 the "GenericAMLRequirementModel" in Figure 34 is reduced to the raw exchange data which should be contained in AutomationML.



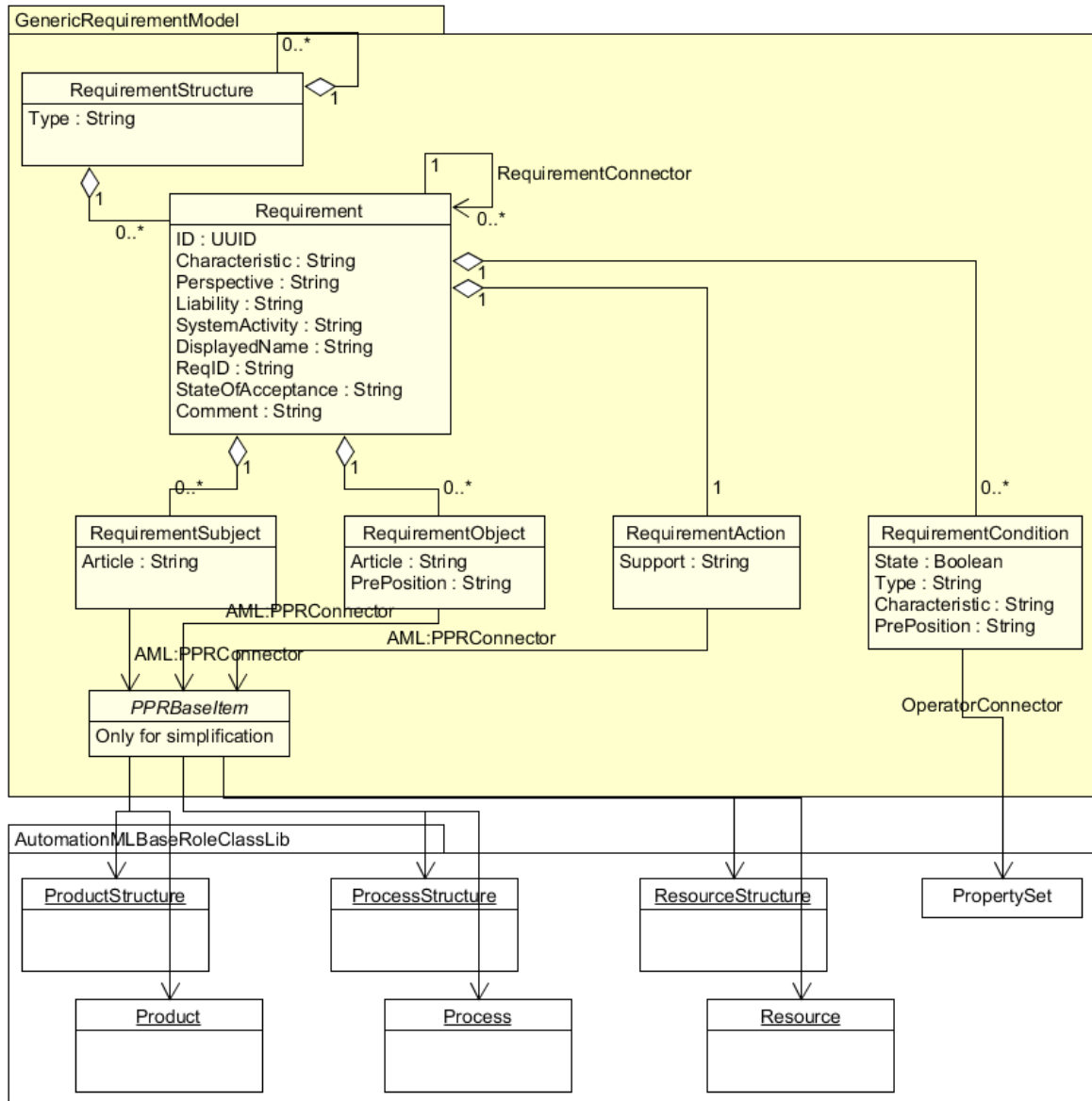
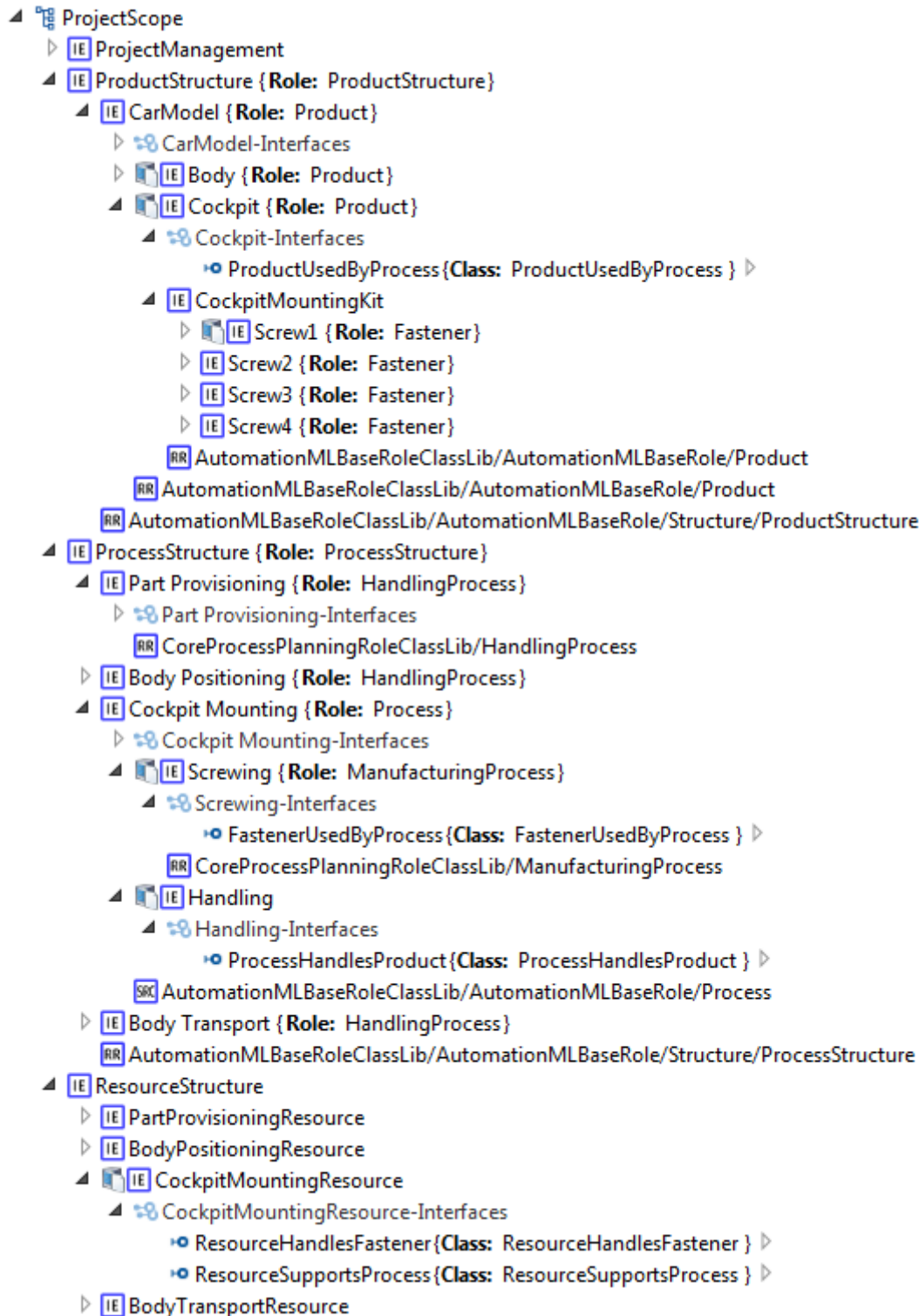


Figure 34: UML overview of the model elements



## 6.4 Example of General Functionality

Picture 35 introduces a simplified example of the specific requirements.



- ▲ [IE] RequirementsSpecification { **Role:** RequirementStructure }
  - ▲ [IE] ProjectPremises { **Role:** RequirementStructure }
    - ▶ [IE] MileStonePlan { **Role:** Requirement }
      - [IE] Site
      - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure
  - ▲ [IE] ProductPremises { **Role:** RequirementStructure }
    - ▲ [IE] The body has a weight equal to 780 kg { **Role:** Requirement }
      - ▲ [IE] The body
        - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure/Requirement/RequirementSubject
      - ▲ [IE] has { **Role:** RequirementAction }
        - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure/Requirement/RequirementAction
      - ▲ [IE] a weight { **Role:** RequirementObject }
        - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure/Requirement/RequirementObject
      - ▲ [IE] equal to 780 kg
        - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure/Requirement/RequirementCondition
        - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure/Requirement
        - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure
    - ▲ [IE] ProcessRequirements { **Role:** RequirementStructure }
      - ▲ [IE] The Screw1 has to be screwed with a torque equal to 75 Nm { **Role:** Requirement }
        - ▲ [IE] The Screw1
          - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure/Requirement/RequirementSubject
        - ▲ [IE] has to be screwed
          - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure/Requirement/RequirementAction
        - ▲ [IE] with a torque { **Role:** RequirementObject }
          - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure/Requirement/RequirementObject
        - ▶ [IE] equal to 75 Nm
          - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure/Requirement
          - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure
      - ▲ [IE] ResourceRequirements { **Role:** RequirementStructure }
        - ▲ [IE] The CockpitMountingResource has to be able to carry the cockpit { **Role:** Requirement }
          - ▲ [IE] The CockpitMountingResource
            - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure/Requirement/RequirementSubject
          - ▲ [IE] has to be able to handle
            - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure/Requirement/RequirementAction
          - ▲ [IE] the cockpit
            - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure/Requirement/RequirementObject
            - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure/Requirement
            - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure
        - ▲ [IE] ReferencedDocuments { **Role:** RequirementStructure }
          - ▲ [IE] ReferencedDocuments-Interfaces
            - ↳ Document\_XYZ
              - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure
              - [RR] DaimlerRequirementsRoleClassLib/RequirementStructure

Figure 35: General Functionality of Auto-generated requirements

	<p style="text-align: center;"> <b>ENTOC</b>        Engineering tool chain for efficient and iterative        development of smart factories        ITEA 3, 15015        Project Coordinator: Thomas Bär, Daimler AG     </p>	
--	---	---

## 7 Requirement Specification based on Natural Language Boilerplates

### 7.1 Motivation

The growing complexity within the development, construction, and conversion of production plants makes a uniform and complete requirements specification process resulting in unambiguous and precise requirements indispensable. Requirements based on uncontrolled natural language are human readable but often ambiguous and hence interpreted differently by different stakeholders [Poh2010]. The main problem is this ambiguity, which often results in misunderstandings between the stakeholders on the customer side that define and extend requirements (customer, OEM, general contractor, ...) and the stakeholders on the contractor side that develop solutions for the production plant based on the understanding they get from the requirements (general contractor, contractor, provider of material handling systems, PLC programmer, ...). This misunderstandings result in the development of production plants, which do not satisfy all of the customer's intended requirements, because these production plants are built to satisfy wrongly understood requirements.

One way to reduce ambiguity is to constrain or standardise natural language by so called boilerplates [HJD2011] (in [Poh2010] called "syntactic requirements patterns"). Boilerplates structure requirements, by defining what kind of boilerplate elements are available and in which sequence these can be arranged. Hence with boilerplates human readable requirement sentences are generated, of which the natural language expressiveness is as much constrained as necessary to result in unambiguous requirements. ([HJD2011] and [Poh2010])

Reducing the ambiguity will result in less misunderstandings, will reduce many sources for handover errors, and will improve the communication and collaboration between the stakeholders along the engineering process chain to make it more reliable for all sides. This will increase the efficiency of the engineering process, because firstly, stakeholders on the contractor side will have less enquiries to the customers and will need less iterations and time to clarify unclear or inconsistent aspects of the requirements. Secondly, production plants will be planned, developed, and built that better fulfil the intended requirements of the customer, hence the quality of the final product is improved.

Furthermore, for direct integration of the collected requirements with advanced analysis and optimization methods (e.g. automatic testing approaches, model checking, or schedule optimization) the requirements have to be machine interpretable and hence have to be available in a formalized form. The boilerplate approach is very flexible and allows finding a formalized representation for many requirements.

In the following sections a set of domain-specific boilerplates is presented covering requirements from the production domain. In 7.2 the approach is described. This includes at first the specification structure together with different phase dependent representation forms of the requirements. Additionally, the Extended Backus-Naur form (EBNF, see for example [ISO14977:1996]) is presented with which the syntax of the boilerplates will be described. Finally, after some preliminary definitions and description of some boilerplate elements and possible element instances, the different boilerplates are presented. In 7.3 an UML model of the data structure together with a mapping to an AML format is presented. At the end, in 7.4, an example is provided demonstrating how the boilerplate based requirement specification approach is used.

## 7.2 Approach

Natural language boilerplates are the core of the requirement specification approach presented within this chapter. A boilerplate consists of boilerplate elements arranged in a specific sequence. Both, the boilerplate elements and the allowed sequence, are predefined, and constrain the expressiveness of natural language to result in unambiguous requirements. The boilerplate element “Subject”, for example, can be a resource or a process of the considered production plant and is hence constrained to elements available in the system. A boilerplate’s specific sequence of specific boilerplate elements carries a specific semantic meaning. The semantic meaning follows the conventions of a spoken language, like English (and is hence easily understood by human users). But because each boilerplate has only one specific semantic meaning it is uniquely determined.

For example, a requirement of the form

***Robot 1 transports the left side door to the welding table 3.***

fits the structure of a subject-verb-object-boilerplate (later called Ba1) and is broken down to the boilerplate elements subject, verb, object, and additional information. This requirement is represented by the subject-verb-object-boilerplate as:

***robot 1 = subject***

***transport = verb***

***left side door = object***

***to welding table 3 = additional information***

Note that the additional information is of type (\*Where to?\*) and can be further broken down to the fixed text part “to” together with a location:

***to = fixed text part***

***welding table 3 = location***



Mapping the requirement to the structure of the subject-verb-object-boilerplate specifies its semantic meaning unambiguously. Hence automated enquires can determine what for a process is considered (verb: transportation process), who or what is performing that transportation process, what is transported, and additionally the information where to.

### 7.2.1 Specification Structure

To be able to put each requirement into a context, to be able to express the relationships between different requirements, to be able to group requirements, and to avoid repetitions, the requirements are structured by means of the specification structure and its sections. Statements linked to a section have to be applied to all requirements in this section and all of its subsections. Out of the specification structure and its sections a specification document with its chapters will be generated.

To express different development phases of the engineering process the requirements can be formulated differently. Following different forms of a single boilerplate for three different development phases are presented:

Phase A)	Given/actual state	Subject	-	Verb	Object
Phase B)	Intended state/problem domain/announcement		,shall'		
Phase C)	Solution domain/ offer		,will'		

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

The separation into three development phases follows the engineering process. Different process or development steps are grouped and result in the three phases A) Actual state, B) Announcement, and C) Offer:

Phase	Engineering process
A) Actual state	Capturing the actual state → Documenting the actual state → Milestone to complete the capturing of the actual state
B) Announcement	Capturing the requirements → Requirements analysis → Specification preparation → Specification approval → Announcement preparation → Announcement distribution
C) Offer	Analysis of the customer's requirements → Offer preparation → Offering → Offer revision → Offer approval

Examples:

A) Actual state	A worker	-	transport	product	from table A to table B
B) Announcement	A robot	,shall'	transport	product	from table A to table B
C) Offer	The robot from type 1102 with max. 50kg permitted load	,will'	transport	product	from table A to table B

For boilerplates presented in the following sections the three phase versions are used. Later in the AML the placeholder DevelopmentPhase is used to describe the development phases.

### 7.2.2 Notation of the Extended Backus-Naur form (EBNF)

The Extended Backus–Naur form (EBNF) is a meta-syntax notation and commonly used to formally describe a formal language. In the next subsections EBNF is used to describe boilerplates. The EBNF notation is summarized in the following table.

Usage	Notation	Usage	Notation
definition	=	grouping	( ... )
concatenation	,	terminal string	" ... "
termination	;	terminal string	' ... '
alternation		comment	(* ... *)
optional	[ ... ]	special sequence	? ... ?
repetition	{ ... }	exception	-

### 7.2.3 Preliminary Definitions

In this subsection some preliminary definitions are presented. Artefact sets are introduced to specify boilerplate elements further. Examples are provided for some of the boilerplate elements. In the next subsection these boilerplate elements are used to form the different boilerplates.

So called artefact sets are introduced to be able to represent the boilerplates in a compact and readable form for some boilerplate elements. Artefact sets describe how artefacts can be combined to form a valid boilerplate element. Such an artefact set could be for example the SubjectSet that describes how the boilerplate element that represents the subject part of a sentence can be formed. For example, the SubjectSet defined as

$$\text{SubjectSet} = ([N] [Adjective] Subject) | (SubjectSet \text{'AND'} SubjectSet) | (SubjectSet \text{'OR'} SubjectSet);$$

allows to combine different SubjectSets by a logical AND connection (SubjectSet 'AND' SubjectSet), or a logical OR connection (SubjectSet 'OR' SubjectSet). Furthermore, a SubjectSet is a subject optionally with one or several adjectives and optionally with a number N expressing the considered amount of the subject ([N] [{Adjective}] Subject).

Boilerplate element	Definition or explanation (following the EBNF notation)
BaX	(*An atomic boilerplate with the atomic boilerplate number X. Atomic boilerplates are boilerplates that do not include other boilerplates. Atomic boilerplates are further described in subsection 7.2.4*)
BmX	(*A main boilerplate with the main boilerplate number X. Main boilerplates are boilerplates that give a frame into which other boilerplates are inserted. Main boilerplates are further described in subsection 7.2.4*)
BaSet	= Ba1   Ba2   Ba3   ...; (*One boilerplate from the group of all atomic boilerplates*)
BmSet	= Bm1   Bm2   Bm3   ...; (*One boilerplate from the group of all main boilerplates *)
BSet	= BaSet   {BaSet}   BaSet 'AND' BaSet   BaSet 'OR' BaSet;
Condition	= BaSet   BmSet; (*One boilerplate from the group of all atomic and main boilerplates*)
Statement	= BaSet   BmSet; (*One boilerplate from the group of all atomic and main boilerplates*)
Product, Process, and Resource	Objects from the PPR-model
Subject	= Process   Resource;
SubjectSet	= ([N] [{Adjective}] Subject)   (SubjectSet 'AND' SubjectSet)   (SubjectSet 'OR' SubjectSet);
Object	= Product   Process   Resource;
ObjectSet	= ([N] [{Adjective}] Object)   (ObjectSet 'AND' ObjectSet)   (ObjectSet 'OR' ObjectSet);
Location	= Resource   "Assembly"   "stacking area"   "GEO table"   "vehicle carrier"   ...;
LocationSet	= ([N] [{Adjective}] Location)   (LocationSet 'AND' LocationSet)   (LocationSet 'OR' LocationSet);
Verb	= "transport"   "process"   "support"   "secure"   "stack up"   "stack down"   ...;
VerbSet	= ['does not/do not'], Verb, [Adverb]   (VerbSet 'AND' VerbSet)   (VerbSet 'OR' VerbSet);
State	= "idle"   "occupied"   "broken";
Orientation	= "with glass side facing upwards"   "in installation position"   ...;
Tool	= Resource   "gripper"   ...;
Qualifying expression	= "="   "<"   ">="   "at least"   "more than"   "not less than"   ...;
Event	= "power cut"   "delivery"   ...;
Value	A real number.
Unit	= "second"   "year"   "kg"   "N"   "dB"   "%"   "°C"   ...;
N	= ...; (*the amount N of the subject or object*)

Various AddOnSets	
(*Where?*) AddOnSet	= “in”   “behind”   “on”   “between”   “at”, LocationSet   ObjectSet;
(*Where from?*) AddOnSet	= “from”, LocationSet   ObjectSet;
(*Where to?*) AddOnSet	= “to”   “in”   “behind”   “between”, LocationSet   ObjectSet;
(*With what?*) AddOnSet	= “with”   “via”, Tool   ObjectSet;
(*To whom?*) AddOnSet (a dative object)	= “to”   “for”, ObjectSet;
(*Why?*) AddOnSet (a reasoning)	= “for”, Text/comment;
(*Orientation?*) AddOnSet	= Orientation;

### 7.2.4 Set of Boilerplates for the Production Domain

The set of boilerplates collected within ENTOC consists of atomic and main boilerplates which are presented in this section. Atomic boilerplates (abbreviated by Ba) are boilerplates that do not include other boilerplates. Main boilerplates (abbreviated with Bm), on the other hand, are boilerplates that give a frame into which other boilerplates are inserted. Each boilerplate is first described in the EBNF notation, followed by some examples.

#### Atomic Boilerplates – Ba

Atomic boilerplates do not include other boilerplates. The boilerplate Ba7 (Ba7 = SubjectSet, ‘of’, ObjectSet;) is an exception and can be part of another atomic boilerplate, because it refines the boilerplate element’s subject or object.

- **Ba1 = SubjectSet, ‘- / shall / will’, VerbSet, [{ObjectSet}], [{AddOnSet}];**
  - Examples for Ba1:

SubjectSet	VerbSet	ObjectSet	AddOnSet 1	AddOnSet 2
Stacking equipment	stacks down	carriers	-	-
Support robot	supports	inner part of roof	(*With what?*) with gripper	-
Mounting	secures	carrier stack	(*Where?*) at GEO-table	-
Industry robot	transports	roof	(*Where from?*) from carrier location	(*Where to?*) to the GEO-table

- **Ba2 = SubjectSet, ‘is/shall be/will be’, Qualifying expression, [Value, Unit], [ObjectSet], [AddOnSet];**
  - Examples for Ba2:

SubjectSet	is	Qualifying expression	Value	Unit
Availability	is	bigger than	98,5	%
Temperature	is	smaller than	45	°C

- **Ba3 = SubjectSet, ‘is / shall / will’, ‘in the state / be in the state’, State;**
  - Examples for Ba3:

SubjectSet	is	in the state	State
Stacking area	is	in the state	occupied
Industry robot	is	in the state	broken



- **Ba4 = SubjectSet, 'is / shall / will', ['be'], [ObjectSet], [Adjective], ['on the' Date], [AddOnSet];**

- Examples for Ba4:

SubjectSet	is	ObjectSet	Adjective	on the Date
Robot	is	handling device	-	-
Level of communication	is	PROFINET level 41 PN	-	-
Additional remote client	is	-	necessary	-
Beginning of assembly	is	-	-	on the 03.12.2012

- **Ba5 = SubjectSet, 'consists of / shall consist of / will consist of', ObjectSet, [AddOnSet];**
- **Ba5' = ProcessSet, 'consists of / shall consist of / will consist of', ProcessSet, [AddOnSet];**

- Examples for Ba5:

SubjectSet	consists of	ObjectSet
Stacking equipment	consists of	two transport robots, a welding robot, and a gripper.
Process at front lid cell	consists of	transport 1, welding, turning over, and transport 2.

- **Ba6 = 'The event', Event, 'happens';**

- Examples for Ba6:

The event	Event	happens
The event	power cut	happens
The event	delivery	happens

- **Ba7 = SubjectSet, 'of', ObjectSet;**

- Examples for Ba7:

SubjectSet	of	ObjectSet
Subprocess	of	process
Attribute	of	PPR-object

- **Ba8 = 'After', ProcessSet, 'follows / shall follow / will follow', ProcessSet;**
- **Ba8' = 'After', Location, 'follows / shall follow / will follow', Location;**

- Examples for Ba8:

After	ProcessSet   Location	follows	ProcessSet   Location
After	drilling	follows	milling and welding
After	drilling station	follows	carrier location

- **Ba9 = SubjectSet, 'is allowed with', ObjectSet;**

- Examples for Ba9:

SubjectSet	is allowed with	ObjectSet
Motor x	is allowed with	gear y and gear z
Vehicle series x, y, and z	is allowed with	roof window x

- **Ba10 = Information, comment, or description;**

- Information, comment, or description given in plain text. Ba10 will encapsulate also requirements that can't be covered by the other boilerplates.

- **Ba11 = Graphics;**
  - A graphic can be a sketch, an illustration, a table, or any other graphical depiction.

### Main Boilerplates – Bm

Main boilerplates include atomic boilerplates as building blocks. Main boilerplates consist of four elements in the following fixed sequence

1. Signal word
2. Condition
3. Signal word
4. statement.

The boilerplate elements Condition and Statement are filled by atomic boilerplates.

- **Bm1 = 'If', Condition, 'then', Statement;**
  - Examples for Bm1:

If	Condition	then	Statement
If	gluing robot is in the state disturbed	then	gluing robot transports roof to storing table
If	person places isolating bridges between cables	then	person uses prepared cable channels

- **Bm2 = 'As soon as', Condition, ' then', Statement;**
  - Examples for Bm2:

As soon as	Condition	then	Statement
As soon as	glue drying time is smaller than time since glue applied on roof	then	material handling equipment transports roof to exit
As soon as	resource removes ASD	then	resource removes empty carrier

- **Bm3 = 'As long as', Condition, ',', Statement;**
  - Examples for Bm3:

As long as	Condition	,	Statement
As long as	glue drying time is bigger than time since glue applied on roof	,	roof can be processed

## 7.3 Overview of UML-Model

Next for the set of boilerplates defined in subsection 7.2.4 a data model in form of an UML diagram is presented. First the mapping of boilerplate elements onto UML objects is defined in following table.

Boilerplate element	UML object	Boilerplate definition	UML object
Subject	Subject	Tool	Resource
Verb	Verb	Qualifying expression	BooleanExpression
Object	Object		
Product	Product	AddOn	FixedText
Process	Process	N	Value
Resource	Resource	Comments	FixedText
Condition	Condition	Orientation	FixedText
Statement	Statement	SubjectSet	Subject, Adjective, Value, LogicalConnector
State	State	VerbSet	Verb, Adjective, Value, LogicalConnector
Event	Event	ObjectSet	Object, Adjective, Value, LogicalConnector
Value	Value	LocationSet	Location, Adjective, Value, LogicalConnector
Unit	Unit	AddOnSet	AddOn, ObjectSet, LocationSet, FixedText, Orientation
Fixed text	FixedText		

In figure “UML diagram of the boilerplate approach” the UML diagram of the data structure of the requirement boilerplates for the AML implementation is depicted. A description of each UML object can be found in the appendix 10.1.6.

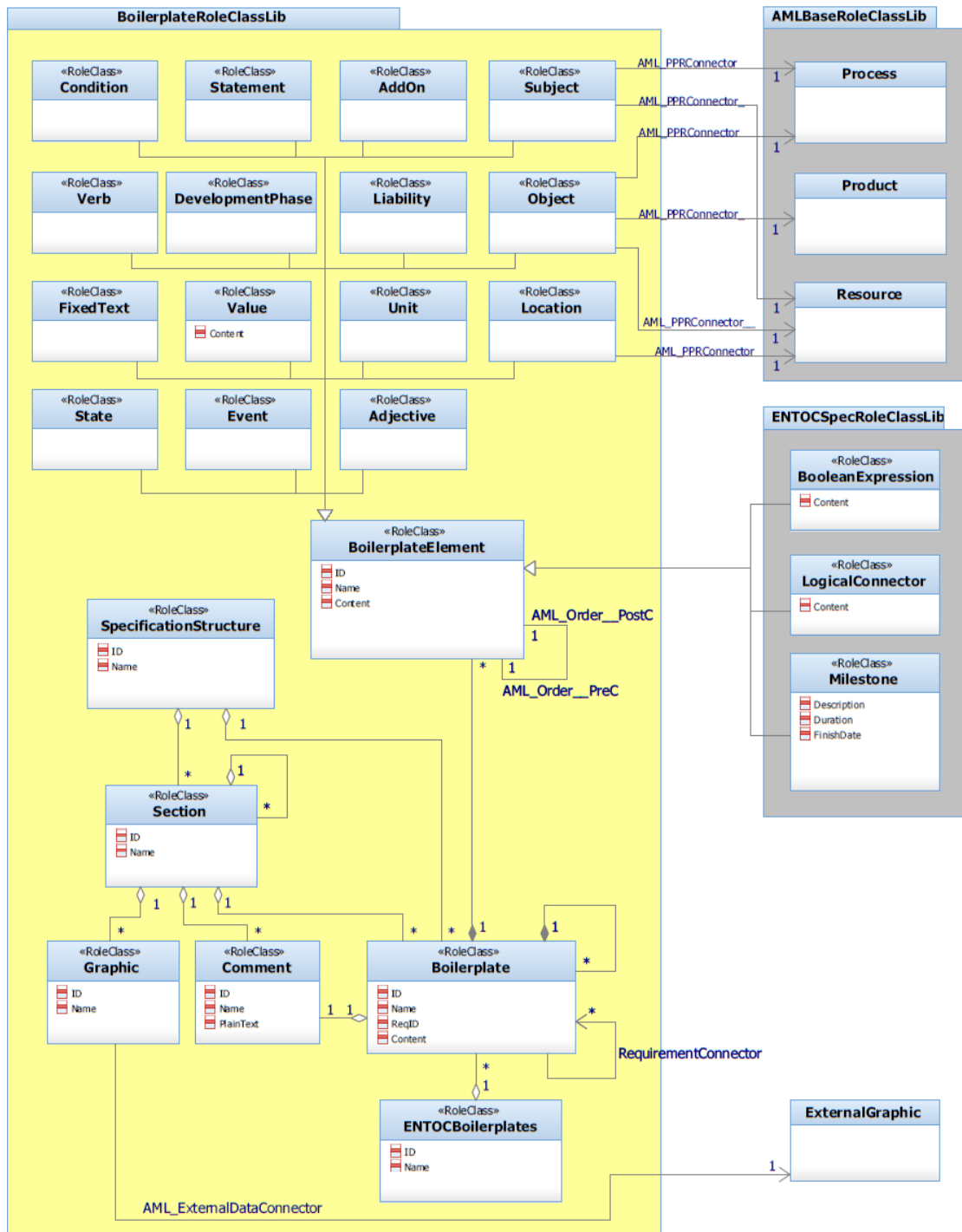


Figure 36: UML diagram of the boilerplate approach

## 7.4 Example of General Functionality

The previously described boilerplate approach saved in the AML format makes use of the role class boilerplate elements as building blocks for boilerplates. A specific boilerplate is defined as a system unit class. Finally, specific requirements are defined in the instance hierarchy as instances of the system unit class boilerplates. To show the general functionality and to present how to make use of the concept the example previously introduced in 7.2 is mapped into the data model developed in 7.3. The corresponding AML system unit class library as well as the AML instance hierarchy are presented next.

The requirement *'Robot 1 transports the left side door to the welding table 3'* (presented in 7.2) has the form subject-verb-object. The corresponding subject-verb-object-boilerplate is shown in the following section of the system unit class library and called Boilerplate\_Ba1. The connectors "PreC" and "PostC" are of the Class "Order" from the "AutomationMLInterfaceClassLib" and used to define the order between the boilerplate elements.

### SystemUnitClassLib

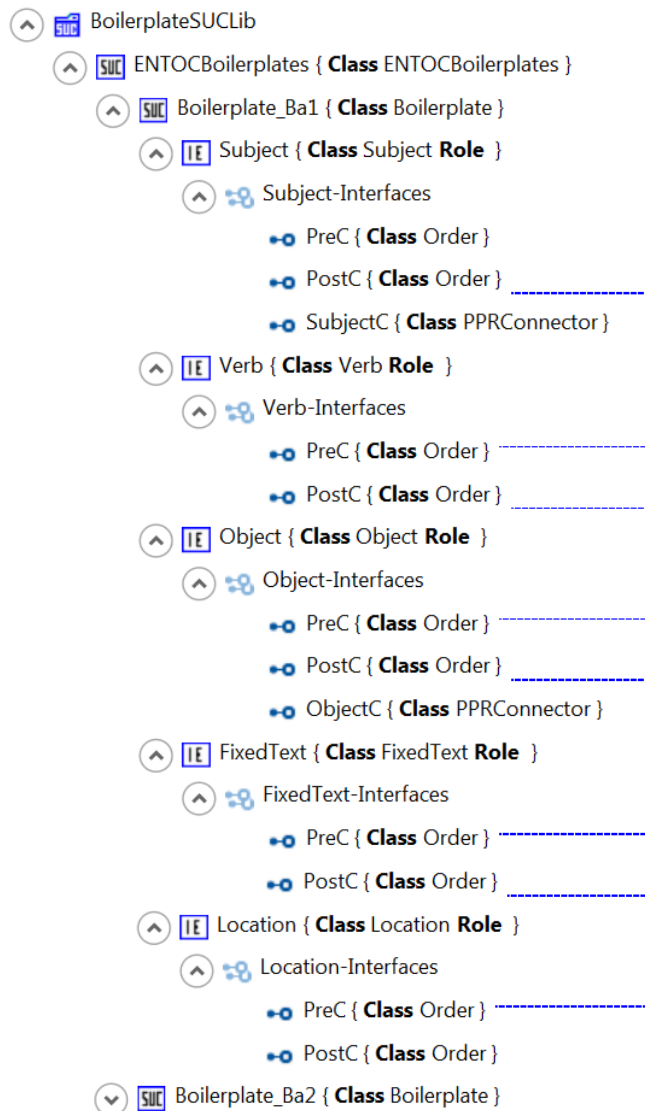


Figure 37: Example usage of boilerplate as AutomationML-SystemUnitClass

Making use of the boilerplate Ba1 an instance is created in the instance hierarchy and parameterized with the specific content or values of the specific requirement. The resulting instance hierarchy is shown as next.

### InstanceHierarchy

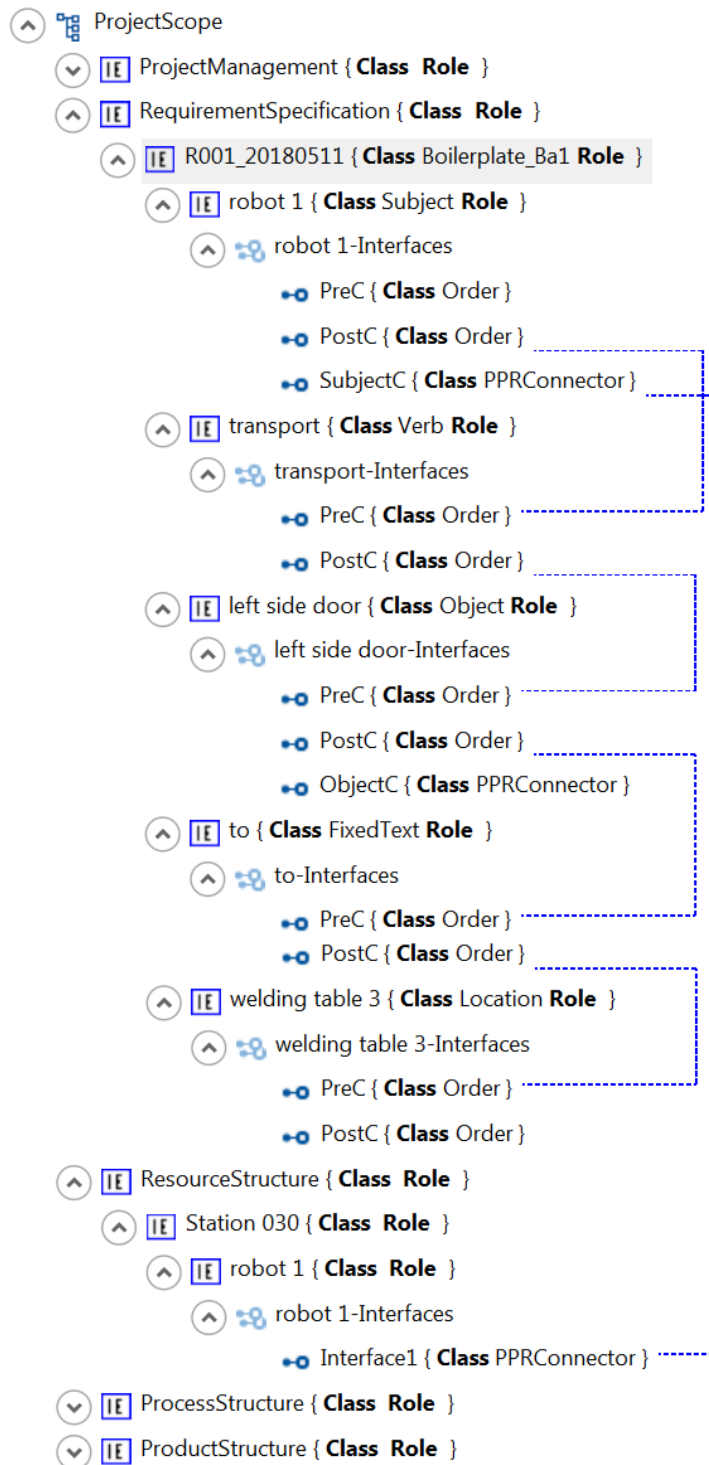




Figure 38: Example usage of boilerplates within AutomationML-InstanceHierarchy

	<p style="text-align: center;"> <b>ENTOC</b>        Engineering tool chain for efficient and iterative        development of smart factories        ITEA 3, 15015        Project Coordinator: Thomas Bär, Daimler AG     </p>	
--	---	---

## 8 Conclusion and Outlook



Verbal statements with references to images and tables are currently used to describe requirements for production equipment in industrial contexts. Those specifications are sometimes ambiguous and are not suited to be evaluated by computer programs.

An approach of formalization of requirements has been presented within this document. The term "formalization" indicates a certain degree of unambiguity regarding a set of statements. Formal descriptions of requirements can be used in different ways:

- for clear and unambiguous understanding between customer and supplier of production equipment,
- for automatic evaluation (e.g. in terms of comparisons between request for offer and offers, search of components in catalogs, assistency in selection of related components, etc.) of requirement statements by computer programs like construction, ordering, process control or simulation systems.

Several concepts and language constructs have been developed (chapter **Fehler! Verweisquelle konnte nicht gefunden werden.** to **Fehler! Verweisquelle konnte nicht gefunden werden.**), which can be used independently from each other, but are developed with the intention to use them complementarily. All language constructs have been described by use of lower level syntax elements provided by the AutomationML standard [IEC62714:2018], which makes it possible to use one generic basic parsing algorithm for usage of all the concepts. The developed language constructs make it possible to describe important aspects of products, production processes, production resources and projects for creation of production systems. A customized version of the Boilerplate methodology (chapter **Fehler! Verweisquelle konnte nicht gefunden werden.**) would make it possible to formalize nearly 100% of text parts of today's requirement specifications. However, the effort for creation of tools supporting automatic evaluation of formalized requirement specifications is different with regard to the approaches presented in this document. The expected tendency is, that more universal approaches (Boilerplates) are more difficult to be exploited by tools than the more simple approaches presented here.

Simple hand-written usage examples have been presented within the various chapters. They prove the theoretical usability of the concepts. Tools for editing more complex examples are necessary in order to create more complicated requirement specifications. Those tools will be developed within task 2.3 of the ENTOC project. A broader coverage of requirement specification scenarios may be reached by use of these tools. Those scenarios will be evaluated within working package 6 of the ENTOC project.

	<p style="text-align: center;"> <b>ENTOC</b>        Engineering tool chain for efficient and iterative        development of smart factories        ITEA 3, 15015        Project Coordinator: Thomas Bär, Daimler AG     </p>	
--	---	---

## 9 References

- [Sch2017] Schlag, A., Vielhaber, M.: Auto-Generated Specification of Assembly Units by Formalized Requirements for a higher Maturity in the Engineering Process, IFAC 2017. Available only (last visited 2018-07-8): <https://www.sciencedirect.com/science/article/pii/S240589631731474X>
- [Gil2017] Gilz, T.: Modellbasierte virtuelle Produktentwicklung – Requirements Engineering und Requirements Management. Springer-Vieweg, pages 53-75, 2014.
- [Man1995] Mannion, M., Keepence, B.: SMART Requirements. ACM SIGSOFT – Software Engineering Notes Vol 20 No 2, 1995.
- [Rup2014] Rupp, C., The SOPHISTS: Requirements Engineering. Carl Hanser Verlag, pages 215-246, 2014.
- [Sch2008] Schwärzler, L.: Modellbasierte Anforderungsanalyse – Was haben Weihnachtsgebäck und Anforderungen gemeinsam? Online-Ausgabe. Requirements Engineering, 2008.
- [VDI2015-1] VDI/VDE 3682:Part1: Formalized process description – Informationmodel, 2015.
- [VDI2105-2] VDI/VDE 3682:Part2: Formalized process description – Concept and graphical representation, 2015.
- [Aml2016a] AutomationML e.V.: Whitepaper AutomationML, Part 1 – Architectural and general requirements, Version 2.0.0, April 2016.
- [HJD2011] Hull, E., Jackson, K., Dick, J.: Requirements Engineering, Springer-Verlag London, 2011.
- [IEC62714:2018] IEC: Engineering data exchange format for use in industrial automation systems engineering - Automation Markup Language - Part 1: Architecture and general requirements
- [IEC6242:2008] IEC: Representation of process control engineering – Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools. IEC 6242/Ed.1.
- [ISO14977:1996] ISO/IEC: Information technology - Syntactic metalanguage - Extended BNF. ISO/IEC 14977/ Ed. 1, 1996. Available online (last visited 2018-06-13): [http://standards.iso.org/ittf/PubliclyAvailableStandards/s026153\\_ISO\\_IEC\\_14977\\_1996\(E\).zip](http://standards.iso.org/ittf/PubliclyAvailableStandards/s026153_ISO_IEC_14977_1996(E).zip)
- [Poh2010] Pohl, K.: Requirements Engineering - Fundamentals, Principles, and Techniques, Springer-Verlag Berlin Heidelberg, 2010.



ENTOC  
Engineering tool chain for efficient and iterative  
development of smart factories  
ITEA 3, 15015

Project Coordinator: Thomas Bär, Daimler AG





## 10 Appendix

### 10.1 Tables of UML to AutomationML Mapping

#### 10.1.1 Chapter 2 – Project Planning

##### RoleClasses



```

▲ [RC] ProjectManagementRoleClassLib
  [RC] Project { Class Process }
  [RC] MileStonePlan { Class AutomationMLBaseRole }
  [RC] MileStone { Class AutomationMLBaseRole }
  [RC] ResultType { Class AutomationMLBaseRole }
  [RC] ResultDefinition { Class AutomationMLBaseRole }

```



RoleClass name	<b>Project</b>
Description	This RoleClass "Project" shall be used to identify an InternalElement, which describes the process of engineering and commissioning of technical equipment. It is no direct part of the triple process-product-resource, but it is linked to the resource part and describes a first part of the life-cycle of the production resource.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process
Attributes	<b>Name</b> A string identifying an instance of the RoleClass "Project". Only characters are allowed according to following regular expression [A-Za-z0-9_-], while the first character is one of [A-Za-z].
	<b>Description</b> This is the verbal description of the "Project" RoleClass instance.

RoleClass name	<b>MileStonePlan</b>
Description	This RoleClass "MileStonePlan" shall be used to identify the required and possibly agreed plan (depending on the finalisation status within the negotiation between customer and supplier of the production resource) for engineering and commissioning of a production resource.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Attributes	<b>Name</b> A string identifying an instance of the RoleClass "MileStonePlan". Only characters are allowed according to following regular expression [A-Za-z0-9_-],

	<p style="text-align: center;"> <b>ENTOC</b>  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015    Project Coordinator: Thomas Bär, Daimler AG </p>	
--	---	---



	while the first character is one of [A-Za-z].
	<p><b>Description</b></p> This is the verbal description of the "MileStonePlan" RoleClass instance.
	<p><b>AbsoluteStartDate</b></p> A dateTime value according to the W3C XML Schema specification, which indicates the start date of the project.

RoleClass name	<b>MileStone</b>
Description	This RoleClass "MileStone" shall be used to identify identify an InternalElement, which represents a single mile-stone within a MileStonePlan. Thus those MileStone InternalElements must be direct sub-elements of a MileStonePlan InternalElement.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Attributes	<p><b>Name</b></p> A string identifying an instance of the RoleClass "MileStone". Only characters are allowed according to following regular expression [A-Za-z0-9_-], while the first character is one of [A-Za-z].
	<p><b>Description</b></p> This is the verbal description of the "MileStone" RoleClass instance.
	<p><b>TimeSpan</b></p> A floating point value (double precision) of seconds, which are defined (expected) for the time distance of this MileStone related to the last ending PredecessorMileStone. Remarks: <ul style="list-style-type: none"> <li>(1) Relative due dates may be computed based on the absolute start time of the MileStonePlan and the duration of MileStones.</li> <li>(2) Using the unit 'second' may be unhandy for human readers, but may be transformed by tools, but it enables this approach to be used for description of fast sub-processes.</li> </ul>
	<p><b>FinishedDate</b></p> A dateTime value according to the W3C XML Schema specification, which indicates the date when this mile-stone has been reached. Based on this value tools may compute expedited finishing of following mile-stones.

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

RoleClass name	<b>ResultType</b>
Description	This RoleClass "ResultType" describes a result of an engineering or commissioning step. This may be defined by the Description attribute e.g. as "Construction files delivered", "Control program tested", "Acceptance test passed", ...
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Attributes	<p><b>Name</b></p> <p>A string identifying an instance of the RoleClass "ResultType". Only characters are allowed according to following regular expression [A-Za-z0-9_-], while the first character is one of [A-Za-z].</p>
	<p><b>Description</b></p> <p>This is the verbal description of the "ResultType" RoleClass instance.</p>

RoleClass name	<b>ResultDefinition</b>
Description	This RoleClass "ResultType" associates PlantResources to ResultTypes. They are sub-elements of milestones, which add planned time information to the result specifications.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Attributes	<p><b>DeliveryDate</b></p> <p>A dateTime value according to the W3C XML Schema specification, which indicates the date when the result has been delivered by the supplier.</p>
	<p><b>DeliveredBy</b></p> <p>A character string identifying the name of suppliers employee, who has initiated the delivery of the result.</p>
	<p><b>AcceptanceDate</b></p> <p>A dateTime value according to the W3C XML Schema specification, which indicates the date when the result has been accepted by the customer.</p>
	<p><b>AcceptedBy</b></p> <p>A character string identifying the name of customer's employee, who has initiated the acceptance of the result.</p>

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---



## Interface Classes

- ProjectManagementInterfaceClassLib
  - SuccessorMileStoneConnector { **Class** AutomationMLBaselInterface }
  - PredecessorMileStoneConnector { **Class** AutomationMLBaselInterface }
  - ResultDefinitionConnector { **Class** AutomationMLBaselInterface }
  - ResultTypeConnector { **Class** AutomationMLBaselInterface }
  - RessourceConnector { **Class** AutomationMLBaselInterface }

InterfaceClass name	<b>SuccessorMileStoneConnector</b>
Description	This InterfaceClass "SuccessorMileStoneConnector" describes the connection point of a mile-stone, which is dependent on other predecessor mile-stones, which have to be finished before this successor mile-stone starts. It is used within InternalLinks indicating the relationship between successor- and predecessor-mile-stones. "SuccessorMileStoneConnector" therein is used on RefPartnerSideA only.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaselInterface
Attributes	-

InterfaceClass name	<b>PredecessorMileStoneConnector</b>
Description	This InterfaceClass "PredecessorMileStoneConnector" describes the connection point of a mile-stone, which has to be finished before the successor mile-stone starts. It is used within InternalLinks indicating the relationship between successor- and predecessor-mile-stones. "PredecessorMileStoneConnector" therein is used on RefPartnerSideB only.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaselInterface
Attributes	-

InterfaceClass name	<b>ResultDefinitionConnector</b>
Description	This InterfaceClass "ResultDefinitionConnector" describes the connection point of a ResultDefinition. It may be used for relations to ResultTypes and to PlantResources. Those relations will be defined by using InternalLinks. Within those links "ResultDefinitionConnector" shall only appear on RefPartnerSideA.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaselInterface

	<p style="text-align: center;"> <b>ENTOC</b>        Engineering tool chain for efficient and iterative        development of smart factories        ITEA 3, 15015        Project Coordinator: Thomas Bär, Daimler AG     </p>	
--	---	---

Attributes	-
------------	---

InterfaceClass name	<b>ResultTypeConnector</b>
Description	This InterfaceClass "ResultTypeConnector" describes the connection point of a ResultType. It may be used for relations to ResultDefinitions. Those relations will be defined by using InternalLinks. Within those links "ResultTypeConnector" shall only appear on RefPartnerSideB.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
Attributes	-

InterfaceClass name	<b>ResourceConnector</b>
Description	This InterfaceClass "ResourceConnector" describes the connection point of a PlantResource. It may be used for relations to ResultDefinitions. Those relations will be defined by using InternalLinks. Within those links "ResourceConnector" shall only appear on RefPartnerSideB.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
Attributes	-

The RoleClasses and InterfaceClasses are formally described by the following AutomationML file content:

```

<?xml version="1.0" encoding="utf-8"?>
<CAEXFile FileName="MileStoneRSL.aml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <AdditionalInformation>
    <WriterHeader>
      <WriterName>ENTOC consortium</WriterName>
      <WriterID>ENTOC consortium</WriterID>
      <WriterVendor>ENTOC consortium</WriterVendor>
      <WriterVendorURL>https://entoc.eu/</WriterVendorURL>
      <WriterVersion>1.0.0</WriterVersion>
      <WriterRelease>v1.0</WriterRelease>
      <LastWritingDateTime>2017-01-
12T13:05:07.3289785+01:00</LastWritingDateTime>
      <WriterProjectTitle>Mile Stone Requirement Specification Language
(MileStoneRSL)</WriterProjectTitle>
      <WriterProjectID>Mile Stone Requirement Specification Language
(MileStoneRSL)</WriterProjectID>
    </WriterHeader>
  </AdditionalInformation>
  <AdditionalInformation AutomationMLVersion="2.0" />
  <AdditionalInformation UserDefined="true">Author: ifak e.V.
Magdeburg</AdditionalInformation>

```



ENTOC  
Engineering tool chain for efficient and iterative  
development of smart factories  
ITEA 3, 15015



Project Coordinator: Thomas Bär, Daimler AG

```
<ExternalReference Path="AutomationMLInterfaceClassLib.aml"
Alias="AutomationMLInterfaceClassLib" />
<ExternalReference Path="AutomationMLBaseRoleClassLib.aml"
Alias="AutomationMLBaseRoleClassLib" />
<InterfaceClassLib Name="ProjectManagementInterfaceClassLib">
  <Version>1.0.0</Version>
  <InterfaceClass Name="SuccessorMilestoneConnector"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface">
    <Description>This InterfaceClass "SuccessorMilestoneConnector"
describes the connection point of a mile-stone, which is dependent on other predecessor mile-
stones, which have to be finished before this successor mile-stone starts. It is used within
InternalLinks indicating the relationship between successor- and predecessor-mile-stones.
"SuccessorMilestoneConnector" therein is used on RefPartnerSideA only.</Description>
    <Version>1.0.0</Version>
  </InterfaceClass>
  <InterfaceClass Name="PredecessorMilestoneConnector"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface">
    <Description>This InterfaceClass "PredecessorMilestoneConnector"
describes the connection point of a mile-stone, which has to be finished before the successor
mile-stone starts. It is used within InternalLinks indicating the relationship between
successor- and predecessor-mile-stones. "PredecessorMilestoneConnector" therein is used on
RefPartnerSideB only.</Description>
    <Version>1.0.0</Version>
  </InterfaceClass>
  <InterfaceClass Name="ResultDefinitionConnector"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface">
    <Description>This InterfaceClass "ResultDefinitionConnector" describes
the connection point of a ResultDefinition. It may be used for relations to ResultTypes and to
PlantRessources. Those relations will be defined by using InternalLinks. Within those links
"ResultDefinitionConnector" shall only appear on RefPartnerSideA.</Description>
    <Version>1.0.0</Version>
  </InterfaceClass>
  <InterfaceClass Name="ResultTypeConnector"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface">
    <Description>This InterfaceClass "ResultTypeConnector" describes the
connection point of a ResultType. It may be used for relations to ResultDefinitions. Those
relations will be defined by using InternalLinks. Within those links "ResultTypeConnector"
shall only appear on RefPartnerSideB.</Description>
    </InterfaceClass>
  <InterfaceClass Name="RessourceConnector"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface">
    <Description>This InterfaceClass "RessourceConnector" describes the
connection point of a PlantRessource. It may be used for relations to ResultDefinitions. Those
relations will be defined by using InternalLinks. Within those links "RessourceConnector"
shall only appear on RefPartnerSideB.</Description>
    </InterfaceClass>
  </InterfaceClassLib>
  <RoleClassLib Name="ProjectManagementRoleClassLib">
    <Version>1.0.0</Version>
    <RoleClass Name="Project"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process">
      <Description>This RoleClass "Project" shall be used to identify an
InternalElement, which describes the process of engineering and commissioning of technical
equipment. It is no direct part of the triple process-product-resource, but it is linked to
the resource part and describes a first part of the life-cycle of the production
resource.</Description>
      <Version>1.0.0</Version>
      <Attribute Name="Name" AttributeDataType="xs:string">
        <Description>A string identifying an instance of the RoleClass
"Project". Only characters are allowed according to following regular expression [A-Za-z0-9_-
], while the first character is one of [A-Za-z].</Description>
      </Attribute>
      <Attribute Name="Description" AttributeDataType="xs:string">
        <Description>This is the verbal description of the "Project"
RoleClass instance.</Description>
      </Attribute>
    </RoleClass>
    <RoleClass Name="MilestonePlan"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
      <Description>This RoleClass "MilestonePlan" shall be used to identify
the required and possibly agreed plan (depending on the finalisation status within the
```





ENTOC  
Engineering tool chain for efficient and iterative  
development of smart factories  
ITEA 3, 15015



Project Coordinator: Thomas Bär, Daimler AG

```
negotiation between customer and supplier of the production resource) for engineering and
commissioning of a production resource.</Description>
  <Version>1.0.0</Version>
  <Attribute Name="Name" AttributeDataType="xs:string">
    <Description>A string identifying an instance of the RoleClass
"MileStonePlan". Only characters are allowed according to following regular expression [A-Za-
z0-9_-], while the first character is one of [A-Za-z].</Description>
  </Attribute>
  <Attribute Name="Description" AttributeDataType="xs:string">
    <Description>This RoleClass "MileStonePlan" shall be used to
identify the required and possibly agreed plan (depending on the finalisation status within
the negotiation between customer and supplier of the production resource) for engineering and
commissioning of a production resource.</Description>
  </Attribute>
  <Attribute Name="AbsoluteStartDate" AttributeDataType="xs:dateTime">
    <Description>A dateTime value according to the W3C XML Schema
specification, which indicates the start date of the project.</Description>
  </Attribute>
</RoleClass>
<RoleClass Name="MileStone"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
  <Description>This RoleClass "MileStone" shall be used to identify
identify an InternalElement, which represents a single mile-stone within a MileStonePlan. Thus
those MileStone InternalElements must be direct sub-elements of a MileStonePlan
InternalElement.</Description>
  <Version>1.0.0</Version>
  <Attribute Name="Name" AttributeDataType="xs:string">
    <Description>A string identifying an instance of the RoleClass
"MileStone". Only characters are allowed according to following regular expression [A-Za-z0-
9_-], while the first character is one of [A-Za-z].</Description>
  </Attribute>
  <Attribute Name="Description" AttributeDataType="xs:string">
    <Description>This is the verbal description of the "MileStone"
RoleClass instance.</Description>
  </Attribute>
  <Attribute Name="TimeSpan" AttributeDataType="xs:double" Unit="s">
    <Description>A floating point value (double precision) of
seconds, which are defined (expected) for the time distance of this MileStone related to the
last ending PredecessorMileStone.
Remarks:
(1) Relative due dates may be computed based on the absolute start time of the MileStonePlan
and the duration of MileStones.
(2) Using the unit 'second' may be unhandy for human readers, but may be transformed by tools,
but it enables this approach to be used for description of fast sub-processes.</Description>
  </Attribute>
  <Attribute Name="FinishedDate" AttributeDataType="xs:dateTime">
    <Description>A dateTime value according to the W3C XML Schema
specification, which indicates the date when this mile-stone has been reached. Based on this
value tools may compute expedited finishing of following mile-stones.</Description>
  </Attribute>
</RoleClass>
<RoleClass Name="ResultType"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
  <Description>This RoleClass "ResultType" describes a result of an
engineering or commissioning step. This may be defined by the Description attribute e.g. as
"Construction files delivered", "Control program tested", "Acceptance test passed",
...</Description>
  <Version>1.0.0</Version>
  <Attribute Name="Name" AttributeDataType="xs:string">
    <Description>A string identifying an instance of the RoleClass
"ResultType". Only characters are allowed according to following regular expression [A-Za-z0-
9_-], while the first character is one of [A-Za-z].</Description>
  </Attribute>
  <Attribute Name="Description" AttributeDataType="xs:string">
    <Description>This is the verbal description of the "ResultType"
RoleClass instance.</Description>
  </Attribute>
</RoleClass>
<RoleClass Name="ResultDefinition"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
```



	<p style="text-align: center;"> <b>ENTOC</b>        Engineering tool chain for efficient and iterative        development of smart factories        ITEA 3, 15015          Project Coordinator: Thomas Bär, Daimler AG     </p>	
--	---	---

```

    <Description>This RoleClass "ResultType" associates PlantRessources to
ResultTypes. They are sub-elements of milestones, which add planned time information to the
result specifications. </Description>
    <Version>1.0.0</Version>
    <Attribute Name="DeliveryDate" AttributeDataType="xs:dateTime">
      <Description>A dateTime value according to the W3C XML Schema
specification, which indicates the date when the result has been delivered by the
supplier.</Description>
    </Attribute>
    <Attribute Name="DeliveredBy" AttributeDataType="xs:string">
      <Description>A character string identifying the name of suppliers
employee, who has initiated the delivery of the result. </Description>
    </Attribute>
    <Attribute Name="AcceptanceDate" AttributeDataType="xs:dateTime">
      <Description>A dateTime value according to the W3C XML Schema
specification, which indicates the date when the result has been accepted by the
customer.</Description>
    </Attribute>
    <Attribute Name="AcceptedBy">
      <Description>A character string identifying the name of
customer's employee, who has initiated the acceptance of the result.</Description>
    </Attribute>
  </RoleClass>
</RoleClassLib>
</CAEXFile>

```



	<p style="text-align: center;"> <b>ENTOC</b>          Engineering tool chain for efficient and iterative          development of smart factories          ITEA 3, 15015          Project Coordinator: Thomas Bär, Daimler AG       </p>	
--	---	---

### 10.1.2 Chapter 3 – Product Variability Handling



#### VariabilityRoleClassLib

RoleClass name	<b>Variant</b>
Description	<p>This class shall be used to list all the major component of the product, which are of interest to the customer and whose selection may influence the subsequently assembly.</p> <p>Variants are listed into groups whose cardinality points out how many of them must be chosen from each group.</p>
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Product
Attributes	<p><b>Name</b></p> <p>A string identifying an instance of the RoleClass "Variant".</p>

RoleClass name	<b>Item</b>
Description	<p>This class shall be used to define the minor elements of the product, whose presence within the final assembly is still crucial but whose selection is of no interest in terms of product design or performance.</p> <p>All instances of class Item are listed in a dedicated Group from which some of them are chosen according to the picked variants.</p>
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Product
Attributes	<p><b>Name</b></p> <p>A string identifying an instance of the RoleClass "Item".</p>

RoleClass name	<b>VariantConstraint</b>
Description	<p>This class shall be used to point out the logical relations among variants and items. The instances of this class are stored in a Group whose cardinality is "all" meaning that all constraints are always true and cannot be neglected.</p>
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/LogicObject
Attributes	<p><b>Name</b></p> <p>A string identifying an instance of the RoleClass "VariantConstraint".</p>

RoleClass name	<b>ResourceConstraint</b>
----------------	---------------------------

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---



Description	Instances of this class are to be used within the operation they refer to, stating what requirements a resource must fulfill to be eligible to execute the operation.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Resource
Attribuites	<p><b>Name</b></p> <p>A string identifying an instance of the RoleClass "ResourceConstraint".</p>

RoleClass name	<b>VariabilityGroup</b>
Description	Generalization of the class Group belonging to the class library AMLRoleClassLib. This class is employed top define groups of both variants and items.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Group
Attribuites	<p><b>Name</b></p> <p>A string identifying an instance of the RoleClass "ResourceConstraint".</p>

RoleClass name	<b>VariabilityProduct</b>
Description	Generalization of the class Product belonging to the class library AMLRoleClassLib. This class is employed top define groups of variant constraints.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Product
Attribuites	<p><b>Name</b></p> <p>A string identifying an instance of the RoleClass "ResourceConstraint".</p>

RoleClass name	<b>VariabilityProcess</b>
Description	Generalization of the class Process belonging to the class library AMLRoleClassLib. This class is employed top define groups of resource consraints.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process
Attribuites	<p><b>Name</b></p> <p>A string identifying an instance of the RoleClass "ResourceConstraint".</p>

### SchedulingRoleClassLib



	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

RoleClass name	<b>SchedulingConnector</b>
Description	Instances of this class are used to specify scheduling constraints among operations and thus they are employed within the <i>Preconditions</i> sub-group. They help to state the execution order of the operations.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/LogicObject
Attributes	<b>Name</b> A string identifying an instance of the RoleClass "SchedulingConnector".

RoleClass name	<b>StartAfterEnd</b>
Description	Instances of this class are used to point out that the it refers to operation must start after another operation (referenced within the instance) is completed.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/SchedulingConnector
Attributes	<b>Name</b> A string identifying an instance of the RoleClass "StartAfterEnd".

RoleClass name	<b>StartAfterStart</b>
Description	Instances of this class are used to point out that the it refers to operation must start after another operation (referenced within the instance) has started.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/SchedulingConnector
Attributes	<b>Name</b> A string identifying an instance of the RoleClass "StartAfterStart".

RoleClass name	<b>StartBeforeStart</b>
Description	Instances of this class are used to point out that the it refers to operation must start before another operation (referenced within the instance) has started.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/SchedulingConnector
Attributes	<b>Name</b> A string identifying an instance of the RoleClass "StartBeforeStart".

	<b>ENTOC</b> Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015  Project Coordinator: Thomas Bär, Daimler AG	
--	--	---

RoleClass name	<b>StartBeforeSEnd</b>
Description	Instances of this class are used to point out that the it refers to operation must start before another operation (referenced within the instance) is completed.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/SchedulingConnector
Attributes	<b>Name</b> A string identifying an instance of the RoleClass "StartBeforeSEnd".

The RoleClasses and InterfaceClasses are formally described by the following AutomationML file content:

```

<?xml version="1.0" encoding="utf-8"?>
<CAEXFile FileName="VariabilitySchedulingRCLib.aml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <AdditionalInformation>
    <WriterHeader>
      <WriterName>ENTOC consortium</WriterName>
      <WriterID>ENTOC consortium</WriterID>
      <WriterVendor>ENTOC consortium</WriterVendor>
      <WriterVendorURL>https://entoc.eu/</WriterVendorURL>
      <WriterVersion>1.0.0</WriterVersion>
      <WriterRelease>v1.0</WriterRelease>
      <LastWritingDateTime>2017-01-
12T13:05:07.3289785+01:00</LastWritingDateTime>
      <WriterProjectTitle>Variability Scheduling Requirement Specification
Language</WriterProjectTitle>
      <WriterProjectID>Variability Scheduling Requirement Specification
Language</WriterProjectID>
    </WriterHeader>
  </AdditionalInformation>
  <AdditionalInformation AutomationMLVersion="2.0" />
  <AdditionalInformation UserDefined="true">Author: Chalmers
University</AdditionalInformation>
  <ExternalReference Path="AutomationMLInterfaceClassLib.aml"
Alias="AutomationMLInterfaceClassLib" />
  <ExternalReference Path="CommunicationInterfaceClassLib.aml"
Alias="CommunicationInterfaceClassLib" />
  <ExternalReference Path="AutomationMLBaseRoleClassLib.aml"
Alias="AutomationMLBaseRoleClassLib" />
  <ExternalReference Path="CommunicationRoleClassLib.aml"
Alias="CommunicationRoleClassLib" />
  <ExternalReference Path="AR APC InterfaceClassLib.aml"
Alias="AutomationProjectConfigurationInterfaceClassLib" />
  <ExternalReference Path="AutomationMLDMIRoleClassLib.aml"
Alias="AutomationMLDMIRoleClassLib" />
  <ExternalReference Path="AutomationMLExtendedRoleClassLib.aml"
Alias="AutomationMLExtendedRoleClassLib" />
  <ExternalReference Path="ExternalDataReference_BPR-Libraries.aml"
Alias="ExternalDataReference" />
  <ExternalReference Path="AutomationMLCSRoleClassLib.aml"
Alias="AutomationMLCSRoleClassLib" />
  <RoleClassLib Name="VariabilityRoleClassLib">
    <Version>1.0.0</Version>
    <RoleClass Name="Variant"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Product" />
    <RoleClass Name="Item"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Product" />
    <RoleClass Name="SchedulingConstraint"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Frame" />
    <RoleClass Name="ResourceConstraint"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Frame" />
    <RoleClass Name="VariantConstarint"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Frame" />
  </RoleClassLib>
</CAEXFile>

```



ENTOC  
Engineering tool chain for efficient and iterative  
development of smart factories  
ITEA 3, 15015








Project Coordinator: Thomas Bär, Daimler AG

```
</RoleClassLib>
<RoleClassLib Name="SchedulingRoleClass">
  <Version>1.0.0</Version>
  <RoleClass Name="SchedulingConnector"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole/LogicObject">
    <RoleClass Name="StartAfterEnd"
RefBaseClassPath="SchedulingRoleClass/SchedulingConnector" />
    <RoleClass Name="EndAfterStart"
RefBaseClassPath="SchedulingRoleClass/SchedulingConnector" />
    <RoleClass Name="EndBeforeStart"
RefBaseClassPath="SchedulingRoleClass/SchedulingConnector" />
    <RoleClass Name="EndBeforeEnd"
RefBaseClassPath="SchedulingRoleClass/SchedulingConnector" />
  </RoleClass>
</RoleClassLib>
<RoleClassLib Name="AutomationMLBaseRoleClassLib">
  <Description>Automation Markup Language Base Role Class Library - Part 1
Content extended with Part 3 and Part 4 Content</Description>
  <Version>2.2.2</Version>
  <RoleClass Name="AutomationMLBaseRole">
    <RoleClass Name="Group" RefBaseClassPath="AutomationMLBaseRole">
      <Attribute Name="AssociatedFacet" AttributeDataType="xs:string"
/>
      <RoleClass Name="InterlockingSourceGroup"
RefBaseClassPath="Group" />
      <RoleClass Name="InterlockingTargetGroup"
RefBaseClassPath="Group" />
    </RoleClass>
    <RoleClass Name="Facet" RefBaseClassPath="AutomationMLBaseRole" />
    <RoleClass Name="Port" RefBaseClassPath="AutomationMLBaseRole">
      <Attribute Name="Direction" AttributeDataType="xs:string" />
      <Attribute Name="Cardinality">
        <Attribute Name="MinOccur"
AttributeDataType="xs:unsignedInt" />
        <Attribute Name="MaxOccur"
AttributeDataType="xs:unsignedInt" />
      </Attribute>
      <Attribute Name="Category" AttributeDataType="xs:string" />
      <ExternalInterface Name="ConnectionPoint" ID="9942bd9c-c19d-
44e4-a197-11b9edf264e7"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PortConnector" />
    </RoleClass>
    <RoleClass Name="Resource" RefBaseClassPath="AutomationMLBaseRole" />
    <RoleClass Name="Product" RefBaseClassPath="AutomationMLBaseRole" />
    <RoleClass Name="Process" RefBaseClassPath="AutomationMLBaseRole" />
    <RoleClass Name="Structure" RefBaseClassPath="AutomationMLBaseRole">
      <RoleClass Name="ProductStructure" RefBaseClassPath="Structure"
/>
      <RoleClass Name="ProcessStructure" RefBaseClassPath="Structure"
/>
      <RoleClass Name="ResourceStructure" RefBaseClassPath="Structure"
/>
    </RoleClass>
    <RoleClass Name="PropertySet" RefBaseClassPath="AutomationMLBaseRole"
/>
    <RoleClass Name="Frame" RefBaseClassPath="AutomationMLBaseRole" />
    <RoleClass Name="LogicObject" RefBaseClassPath="AutomationMLBaseRole"
/>
  </RoleClass>
</RoleClassLib>
</CAEXFile>
```

### 10.1.3 Chapter 4 – Process-Driven Production Planning



#### Role Classes

- 
 CoreProcessPlanningRoleClassLib
  -  ManufacturingProcess{Class: Process }
  -  HandlingProcess{Class: Process }
  -  Fastener{Class: Product }

RoleClass name	<b>ManufacturingProcess</b>
Description	The role class “ManufacturingProcess” defines a process, which has a resulting product based on input products and the characteristic type of the manufacturing process (e.g. spot welding, screwing, etc.).
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process
Path for element reference	ProcessPlanningRoleClassLib/ManufacturingProcess
Attribuites	<b>Charatertistic</b> Defines the concrete characteristic of a manufacturing process like e.g. spot welding

RoleClass name	<b>HandlingProcess</b>
Description	The role class “Handlingprocess” defines a process, which handles a product by changing location or verifying required and defined properties of a product.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process
Path for element reference	ProcessPlanningRoleClassLib/HandlingProcess
Attribuites	<b>Characteristic</b> Defines the concrete characteristic of a handling process like e.g. rotating, turinn, etc.

RoleClass name	<b>Fastener</b>
Description	The role class “Fastener” defines a connection element of a product.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Product
Path for element reference	ProcessPlanningRoleClassLib/Fastener

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---



Attributes	None
------------	------

### Interface Classes

- ▶ 📁 CoreProcessPlanningInterfaceClassLib
  - ProcessFollowsProcess {Class: Order }
  - ProductHasFastener {Class: Order }
  - FastenerCreatesProduct {Class: Order }
  - ProductUsedByProcess {Class: PPRConnector }
  - ProductProducedByProcess {Class: PPRConnector }
  - FastenerUsedByProcess {Class: PPRConnector }
  - ResourceHandlesProduct {Class: PPRConnector }
  - ResourceHandlesFastener {Class: PPRConnector }
  - ResourceSupportsProcess {Class: PPRConnector }
  - ProcessRunsOnResource {Class: PPRConnector }
  - ResourceFollowsResource {Class: Order }
  - ProcessHandlesProduct {Class: PPRConnector }

InterfaceClass name	<b>ProcessFollowsProcess</b>
Description	The interface class “ProcessFollowsProcess” defines the order and/or sequence of a process chain.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order
Path for element reference	ProcessPlanningInterfaceClassLib/ProcessFollowsProcess
Attributes	None

InterfaceClass name	<b>ProductHasFastener</b>
Description	The interface class “ProductHasFastener” defines that the product has connection elements.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order
Path for element reference	ProcessPlanningInterfaceClassLib/ProductHasFastener
Attributes	None

	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---



InterfaceClass name	<b>FastenerCreatesProduct</b>
Description	The interface class “FastenerCreatesProduct” defines that the connection element is part of the structure of the product.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order
Path for element reference	ProcessPlanningInterfaceClassLib/FastenerCreatesProduct
Attributes	None

InterfaceClass name	<b>ProductUsedByProcess</b>
Description	The interface class “ProductUsedByProcess” defines the usage of a product by a process.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector
Path for element reference	ProcessPlanningInterfaceClassLib/ProductUsedByProcess
Attributes	None

InterfaceClass name	<b>ProductProducedByProcess</b>
Description	The interface class “ProductProducedByProcess” defines that a product is produced by the process.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector
Path for element reference	ProcessPlanningInterfaceClassLib/ProductProducedByProcess
Attributes	None

InterfaceClass name	<b>FastenerUsedByProcess</b>
---------------------	------------------------------





	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

Description	The interface class “FastenerUsedByProcess” defines that a connection element is processed by a process.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector
Path for element reference	ProcessPlanningInterfaceClassLib/FastenerUsedByProcess
Attributes	None

InterfaceClass name	<b>ResourceHandlesProduct</b>
Description	The interface class “ResourceHandlesProduct” defines, that a resource handles a product.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector
Path for element reference	ProcessPlanningInterfaceClassLib/ResourceHandlesProduct
Attributes	None

InterfaceClass name	<b>ResourceHandlesFastener</b>
Description	The interface class “ResourceHandlesFastener” defines that a connectin element is done by the related resource.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector
Path for element reference	ProcessPlanningInterfaceClassLib/ResourceHandlesFastener
Attributes	None

InterfaceClass name	<b>ResourceSupportsProcess</b>
Description	The interface class “ResourceSupportsProcess” defines that a resource supports a process to process.



	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector
Path for element reference	ProcessPlanningInterfaceClassLib/ResourceSupportsProcess
Attributes	None

InterfaceClass name	<b>ProcessRunsOnResource</b>
Description	The interface class “ProcessRunsOnResource” defines that a process is processed on the related resource.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector
Path for element reference	ProcessPlanningInterfaceClassLib/ProcessRunsOnResource
Attributes	None

InterfaceClass name	<b>ResourceFollowsResource</b>
Description	The interface class “ResourceFollowsResource” defines the order of resources describing the material flow.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order
Path for element reference	ProcessPlanningInterfaceClassLib/ResourceFollowsResource
Attributes	None

InterfaceClass name	<b>ProcessHandlesProduct</b>
Description	The interface class “ProcessHandlesProduct” defines that a process handles a product to process.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector
Path for element	ProcessPlanningInterfaceClassLib/ProcessHandlesProduct

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

reference	
Attributes	None

The RoleClasses and InterfaceClasses are formally described by the following AutomationML file content:

```

<?xml version="1.0" encoding="utf-8"?>
<CAEXFile FileName="CoreProcessPlanningLibs.aml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <AdditionalInformation>
    <WriterHeader>
      <WriterName>ENTOC consortium</WriterName>
      <WriterID>ENTOC consortium</WriterID>
      <WriterVendor>ENTOC consortium</WriterVendor>
      <WriterVendorURL>www.AutomationML.org</WriterVendorURL>
      <WriterVersion>1.0.0</WriterVersion>
      <WriterRelease>v1.0</WriterRelease>
      <LastWritingDateTime>2018-03-
26T15:51:30.2465102+02:00</LastWritingDateTime>
      <WriterProjectTitle>Production Process Planning for Requirement
Specification Language</WriterProjectTitle>
      <WriterProjectID>Production Process Planning for Requirement
Specification Language</WriterProjectID>
    </WriterHeader>
  </AdditionalInformation>
  <AdditionalInformation AutomationMLVersion="2.0" />
  <AdditionalInformation UserDefined="true">Author: Daimler AG
Sindelfingen</AdditionalInformation>
  <InterfaceClassLib Name="CoreProcessPlanningInterfaceClassLib">
    <Version>0</Version>
    <InterfaceClass Name="ProcessFollowsProcess"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order"
ChangeMode="state">
      <Description>The interface class "ProcessFollowsProcess" defines the
order and/or sequence of a process chain.</Description>
      <Version>1.0.0</Version>
      <Copyright>Daimler AG</Copyright>
    </InterfaceClass>
    <InterfaceClass Name="ProductHasFastener"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order"
ChangeMode="state">
      <Description>The interface class "ProductHasFastener" defines that the
product has connection elements.</Description>
      <Version>1.0.0</Version>
      <Copyright>Daimler AG</Copyright>
    </InterfaceClass>
    <InterfaceClass Name="FastenerCreatesProduct"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order"
ChangeMode="state">
      <Description>The interface class "FastenerCreatesProduct" defines that
the connection element is part of the structure of the product.</Description>
      <Version>1.0.0</Version>
      <Copyright>Daimler AG</Copyright>
    </InterfaceClass>
    <InterfaceClass Name="ProductUsedByProcess"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector"
ChangeMode="state">
      <Description>The interface class "ProductUsedByProcess" defines the
usage of a product by a process.</Description>
      <Version>1.0.0</Version>
      <Copyright>Daimler AG</Copyright>
    </InterfaceClass>
    <InterfaceClass Name="ProductProducedByProcess"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector"
ChangeMode="state">
      <Description>The interface class "ProductProducedByProcess" defines
that a product is produced by the process.</Description>

```



ENTOC  
Engineering tool chain for efficient and iterative  
development of smart factories  
ITEA 3, 15015



Project Coordinator: Thomas Bär, Daimler AG

```
<Version>1.0.0</Version>
<Copyright>Daimler AG</Copyright>
</InterfaceClass>
<InterfaceClass Name="FastenerUsedByProcess"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector"
ChangeMode="state">
  <Description>The interface class "FastenerUsedByProcess" defines that a
connection element is processed by a process.</Description>
  <Version>1.0.0</Version>
  <Copyright>Daimler AG</Copyright>
</InterfaceClass>
<InterfaceClass Name="ResourceHandlesProduct"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector"
ChangeMode="state">
  <Description>The interface class "ResourceHandlesProduct" defines, that
a resource handles a product.</Description>
  <Version>1.0.0</Version>
  <Copyright>Daimler AG</Copyright>
</InterfaceClass>
<InterfaceClass Name="ResourceHandlesFastener"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector"
ChangeMode="state">
  <Description>The interface class "ResourceHandlesFastener" defines that
a connection element is done by the related resource.</Description>
  <Version>1.0.0</Version>
  <Copyright>Daimler AG</Copyright>
</InterfaceClass>
<InterfaceClass Name="ResourceSupportsProcess"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector"
ChangeMode="state">
  <Description>The interface class "ResourceSupportsProcess" defines that
a resource supports a process to process.</Description>
  <Version>1.0.0</Version>
  <Copyright>Daimler AG</Copyright>
</InterfaceClass>
<InterfaceClass Name="ProcessRunsOnResource"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector"
ChangeMode="state">
  <Description>The interface class "ProcessRunsOnResource" defines that a
process is processed on the related resource.</Description>
  <Version>1.0.0</Version>
  <Copyright>Daimler AG</Copyright>
</InterfaceClass>
<InterfaceClass Name="ResourceFollowsResource"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order"
ChangeMode="state">
  <Description>The interface class "ResourceFollowsResource" defines the
order of resources describing the material flow.</Description>
  <Version>1.0.0</Version>
  <Copyright>Daimler AG</Copyright>
</InterfaceClass>
<InterfaceClass Name="ProcessHandlesProduct"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector"
ChangeMode="state">
  <Description>The interface class "ProcessHandlesProduct" defines that a
process handles a product to process.</Description>
  <Version>1.0.0</Version>
  <Copyright>Daimler AG</Copyright>
</InterfaceClass>
</InterfaceClassLib>
<RoleClassLib Name="CoreProcessPlanningRoleClassLib">
  <Version>0</Version>
  <RoleClass Name="ManufacturingProcess"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process"
ChangeMode="state">
    <Description>The role class "ManufacturingProcess" defines a process,
which has a resulting product based on input products and the characteristic type of the
manufacturing process (e.g. spot welding, screwing, etc.).</Description>
    <Version>1.0.0</Version>
    <Copyright>Daimler AG</Copyright>
    <Attribute Name="Characteristic" AttributeDataType="xs:string" />
  </RoleClass>
</RoleClassLib>
</Lib>
```



ENTOC  
Engineering tool chain for efficient and iterative  
development of smart factories  
ITEA 3, 15015





Project Coordinator: Thomas Bär, Daimler AG

```
</RoleClass>
<RoleClass Name="HandlingProcess"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process"
ChangeMode="state">
  <Description>The role class "Handlingprocess" defines a process, which
handles a product by changing location or verifying required and defined properties of a
product. </Description>
  <Version>1.0.0</Version>
  <Copyright>Daimler AG</Copyright>
  <Attribute Name="Chracteristic" AttributeDataType="xs:string" />
</RoleClass>
<RoleClass Name="Fastener"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Product"
ChangeMode="state">
  <Description>The role class "Fastener" defines a connection element of
a product.</Description>
  <Version>1.0.0</Version>
  <Copyright>Daimler AG</Copyright>
</RoleClass>
</RoleClassLib>
</CAEXFile>
```

## 10.1.4 Chapter 5 – Production Resource Planning

### Role Classes



- MaterialsHandlingClassLib
  - Turner { **Class** Transport }
    - VerticalTurner { **Class** Turner }
    - LateralTurner { **Class** Turner }
    - LongitudinalTurner { **Class** Turner }
  - MaterialHandlingConveyor { **Class** Conveyor }
    - StraightConveyor { **Class** MaterialHandlingConveyor }
    - CurvedConveyor { **Class** MaterialHandlingConveyor }
    - VerticalConveyor { **Class** MaterialHandlingConveyor }
  - Positioner { **Class** Transport }
    - LinearPositioner { **Class** Positioner }
    - ArealPositioner { **Class** Positioner }
    - FreePositioner { **Class** Positioner }
  - ConveyorTechnology { **Class** Resource }
    - BeltCarrier { **Class** ConveyorTechnology }
    - ChainCarrier { **Class** ConveyorTechnology }
    - SkidCarrier { **Class** ConveyorTechnology }
    - RollCarrier { **Class** ConveyorTechnology }
    - LinkBeltCarrier { **Class** ConveyorTechnology }
    - BallCarrier { **Class** ConveyorTechnology }
    - WheelCarrier { **Class** ConveyorTechnology }
  - MaterialHandlingStorage { **Class** Storage }
    - SortedStorage { **Class** MaterialHandlingStorage }
    - PartlySortedStorage { **Class** MaterialHandlingStorage }
    - UnsortedStorage { **Class** MaterialHandlingStorage }
  - MaterialHandlingConnectionPoint { **Class** ResourceStructure }
  - ConveyorAttachments { **Class** Resource }
    - Stopper { **Class** ConveyorAttachments }
    - Pusher { **Class** ConveyorAttachments }
  - EndEffector { **Class** Fixture }
    - Gripper { **Class** EndEffector }
      - HydraulicGripper { **Class** EndEffector }
      - VacuumGripper { **Class** EndEffector }
      - ServoElectricGripper { **Class** EndEffector }
      - MagneticGripper { **Class** EndEffector }
      - PneumaticGripper { **Class** EndEffector }
  - DriveTechnology { **Class** Actuator }
  - LoadingEquipment { **Class** Product }
    - Container { **Class** LoadingEquipment }
    - Pallet { **Class** LoadingEquipment }
    - Box { **Class** LoadingEquipment }
  - ComboElement { **Class** ResourceStructure }

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

RoleClass name	<b>Turner</b>
Description	A general functional role for the elements, which are turning materials without any position change.
Parent class	<b>Undefined with the required libraries</b>
Path for element reference	MaterialsHandlingClassLib/Turner
Attributes	<b>MaximumLoad</b> (DataType="xs:double", Unit="kg") Maximum sustainable load.
	<b>MaximumAngularAcceleration</b> (DataType="xs:double", Unit="rad/s^2") Angular acceleration of the rotation.
	<b>MaximumAngularVelocity</b> (DataType="xs:double", Unit="rad/s") Angular velocity of the rotation.
	<b>Range</b> (DataType="xs:double", Unit="deg") Range of the turner.

RoleClass name	<b>VerticalTurner</b>
Description	An element, which is turning materials on vertical axis.
Parent class	MaterialsHandlingClassLib/Turner
Path for element reference	MaterialsHandlingClassLib/Turner/VerticalTurner
Attributes	None

RoleClass name	<b>LateralTurner</b>
Description	An element, which is turning materials on lateral axis.
Parent class	MaterialsHandlingClassLib/Turner
Path for element reference	MaterialsHandlingClassLib/Turner/LateralTurner
Attributes	None

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---



RoleClass name	<b>LongitudinalTurner</b>
Description	An element, which is turning materials on longitudinal axis.
Parent class	MaterialsHandlingClassLib/Turner
Path for element reference	MaterialsHandlingClassLib/Turner/LongitudinalTurner
Attributes	None

RoleClass name	<b>MaterialHandlingConveyor</b>
Description	A functional role for the elements, which move materials from one location to another but does not change its own place.
Parent class	<b>Undefined with the required libraries</b>
Path for element reference	MaterialsHandlingClassLib/MaterialHandlingConveyor
Attributes	<b>Width</b> (DataType="xs:double", Unit="m") Usable width of the conveyor.
	<b>Velocity</b> (DataType="xs:double", Unit="m/s") Velocity of the conveyor.

RoleClass name	<b>StraightConveyor</b>
Description	Moves materials from one location to another on a straight line.
Parent class	MaterialsHandlingClassLib/MaterialHandlingConveyor
Path for element reference	MaterialsHandlingClassLib/MaterialHandlingConveyor/StraightConveyor
Attributes	<b>Tilt</b> (DataType="xs:double", Unit="deg") Rotational angle over the y-axis.
	<b>Length</b> (DataType="xs:double", Unit="m") Usable length of the conveyor.

RoleClass name	<b>Bevel</b>
----------------	--------------





	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

Description	Moves materials from one location to another and changes the direction of output with an angle.
Parent class	MaterialsHandlingClassLib/MaterialHandlingConveyor/StraightConveyor
Path for element reference	MaterialsHandlingClassLib/MaterialHandlingConveyor/StraightConveyor/Bevel
Attributes	<p><b>Angle</b> (DataType="xs:double", Unit="deg") Angle of bevel.</p>

RoleClass name	<b>CurvedConveyor</b>
Description	Moves materials from one location to another on a curve.
Parent class	MaterialsHandlingClassLib/MaterialHandlingConveyor
Path for element reference	MaterialsHandlingClassLib/MaterialHandlingConveyor/CurvedConveyor
Attributes	<p><b>Angle</b> (DataType="xs:double", Unit="deg") Angle of the curve.</p>
	<p><b>Radius</b> (DataType="xs:double", Unit="m") Center radius.</p>

RoleClass name	<b>SpiralConveyor</b>
Description	Moves materials from one location to another on a spiral.
Parent class	MaterialsHandlingClassLib/MaterialHandlingConveyor
Path for element reference	MaterialsHandlingClassLib/MaterialHandlingConveyor/SpiralConveyor
Attributes	<p><b>Height</b> (DataType="xs:double", Unit="m") Height of the spiral.</p>



RoleClass name	<b>VerticalConveyor</b>
Description	Moves materials from one location to another on a line flapped around the connection point.

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

Parent class	MaterialsHandlingClassLib/MaterialHandlingConveyor
Path for element reference	MaterialsHandlingClassLib/MaterialHandlingConveyor/VerticalConveyor
Attributes	<p><b>SwingVelocity</b> (DataType="xs:double", Unit="m/s") Velocity of the swinging from one tilt angle to another.</p>
	<p><b>Length</b> (DataType="xs:double", Unit="m") Usable length of the conveyor.</p>
	<p><b>TiltList</b> (Unit="deg", RefSemantic="OrderedListType") List of tilt angles.</p>

RoleClass name	<b>Positioner</b>
Description	A functional role for the elements, which carry materials from one location to another (Positioner has also location change).
Parent class	<b>Undefined with the required libraries</b>
Path for element reference	MaterialsHandlingClassLib/Positioner
Attributes	<p><b>MaximumAcceleration</b> (DataType="xs:double", Unit="m/s<sup>2</sup>") Maximum acceleration if the linear movement.</p>
	<p><b>MaximumVelocity</b> (DataType="xs:double", Unit="m/s") Maximum velocity of the linear movement.</p>
	<p><b>MaximumLoad</b> (DataType="xs:double" Unit="kg") Maximum sustainable load.</p>

RoleClass name	<b>LinearPositioner</b>
Description	A positioner, which carries materials on a linear axis.
Parent class	MaterialsHandlingClassLib/Positioner
Path for element reference	MaterialsHandlingClassLib/Positioner/LinearPositioner
Attributes	<p><b>Range</b> (DataType="xs:double", Unit="m") Usable range of the positioner.</p>



	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

RoleClass name	<b>VerticalPositioner</b>
Description	A positioner, which carries materials in linear vertical axis.
Parent class	MaterialsHandlingClassLib/Positioner/LinearPositioner
Path for element reference	MaterialsHandlingClassLib/Positioner/LinearPositioner/VerticalPositioner
Attributes	None

RoleClass name	<b>LateralPositioner</b>
Description	A positioner, which carries materials in linear lateral axis.
Parent class	MaterialsHandlingClassLib/Positioner/LinearPositioner
Path for element reference	MaterialsHandlingClassLib/Positioner/LinearPositioner/LateralPositioner
Attributes	None

RoleClass name	<b>LongitudinalPositioner</b>
Description	A positioner, which carries materials on a linear longitudinal axis.
Parent class	MaterialsHandlingClassLib/Positioner/LinearPositioner
Path for element reference	MaterialsHandlingClassLib/Positioner/LinearPositioner/LongitudinalPositioner
Attributes	None



RoleClass name	<b>ArealPositioner</b>
Description	A positioner, which carries materials on a defined area.
Parent class	MaterialsHandlingClassLib/Positioner
Path for element reference	MaterialsHandlingClassLib/Positioner/ArealPositioner
Attributes	<b>Range</b> (DataType="xs:double", Unit="m")

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

	Usable range of the positioner.
	<p><b>Direction</b> (DataType="", Unit="m[]", RefSemantic="OrderedListType") Direction of the movement as vector.</p>

RoleClass name	<b>FreePositioner</b>
Description	A positioner, which can carry materials in three dimensional space.
Parent class	MaterialsHandlingClassLib/Positioner
Path for element reference	MaterialsHandlingClassLib/Positioner/FreePositioner
Attributes	<p><b>TaskSpace</b> (DataType="", Unit="") The task space region of the positioner</p>
	<p><b>TaskSpace_x_min</b> (DataType="xs:double", Unit="m") Minimum x space.</p>
	<p><b>TaskSpace_x_max</b> (DataType="xs:double", Unit="m") Maximum x space.</p>
	<p><b>TaskSpace_y_min</b> (DataType="xs:double", Unit="m") Minimum y space.</p>
	<p><b>TaskSpace_y_max</b> (DataType="xs:double", Unit="m") Maximum y space.</p>
	<p><b>TaskSpace_z_min</b> (DataType="xs:double", Unit="m") Minimum z space.</p>
	<p><b>TaskSpace_z_max</b> (DataType="xs:double", Unit="m") Maximum z space.</p>

RoleClass name	<b>ConveyorTechnology</b>
Description	Defines the carrier element of a conveyor like rolls, belts, chains, etc.
Parent class	AutomationMLBaseRole/Resource
Path for element reference	MaterialsHandlingClassLib/ConveyorTechnology

	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---



Attributes	None
------------	------

RoleClass name	<b>BeltCarrier</b>
Description	Belt type carrier element.
Parent class	MaterialsHandlingClassLib/ConveyorTechnology
Path for element reference	MaterialsHandlingClassLib/ConveyorTechnology/BeltCarrier
Attributes	None

RoleClass name	<b>ChainCarrier</b>
Description	Chain type carrier element.
Parent class	MaterialsHandlingClassLib/ConveyorTechnology
Path for element reference	MaterialsHandlingClassLib/ConveyorTechnology/ChainCarrier
Attributes	None

RoleClass name	<b>SkidCarrier</b>
Description	Skid type carrier element.
Parent class	MaterialsHandlingClassLib/ConveyorTechnology
Path for element reference	MaterialsHandlingClassLib/ConveyorTechnology/SkidCarrier
Attributes	None

RoleClass name	<b>RollCarrier</b>
Description	Roll type carrier element.
Parent class	MaterialsHandlingClassLib/ConveyorTechnology
Path for element reference	MaterialsHandlingClassLib/ConveyorTechnology/RollCarrier

	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---



Attributes	None
------------	------

RoleClass name	<b>LinkBeltCarrier</b>
Description	Link belt type carrier element.
Parent class	MaterialsHandlingClassLib/ConveyorTechnology
Path for element reference	MaterialsHandlingClassLib/ConveyorTechnology/LinkBeltCarrier
Attributes	None

RoleClass name	<b>BallCarrier</b>
Description	Ball type carrier element.
Parent class	MaterialsHandlingClassLib/ConveyorTechnology
Path for element reference	MaterialsHandlingClassLib/ConveyorTechnology/BallCarrier
Attributes	None

RoleClass name	<b>WheelCarrier</b>
Description	Wheel type carrier element.
Parent class	MaterialsHandlingClassLib/ConveyorTechnology
Path for element reference	MaterialsHandlingClassLib/ConveyorTechnology/WheelCarrier
Attributes	None

RoleClass name	<b>MaterialHandlingStorage</b>
Description	Represents storage for materials.
Parent class	<b>Undefined with the required libraries</b>
Path for element reference	MaterialsHandlingClassLib/MaterialHandlingStorage

	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---



Attributes	None
------------	------

RoleClass name	<b>SortedStorage</b>
Description	Represents a storage, in which all translational and rotational degrees of freedom for materials are defined.
Parent class	MaterialsHandlingClassLib/MaterialHandlingStorage
Path for element reference	MaterialsHandlingClassLib/MaterialHandlingStorage/SortedStorage
Attributes	None

RoleClass name	<b>PartlySortedStorage</b>
Description	Represents a storage, in which some translational and rotational degrees of freedom for materials are defined.
Parent class	MaterialsHandlingClassLib/MaterialHandlingStorage
Path for element reference	MaterialsHandlingClassLib/MaterialHandlingStorage/PartlySortedStorage
Attributes	None

RoleClass name	<b>UnsortedStorage</b>
Description	Represents a storage, in which translational and rotational degrees of freedom for materials are arbitrary.
Parent class	MaterialsHandlingClassLib/MaterialHandlingStorage
Path for element reference	MaterialsHandlingClassLib/MaterialHandlingStorage/UnsortedStorage
Attributes	None

RoleClass name	<b>ConveyorAttachments</b>
Description	A general role for conveyor attachments.
Parent class	AutomationMLBaseRole/Resource

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

Path for element reference	MaterialsHandlingClassLib/ConveyorAttachments
Attributes	None



RoleClass name	<b>Stopper</b>
Description	Decelerates or stops materials safely.
Parent class	MaterialsHandlingClassLib/ConveyorAttachments
Path for element reference	MaterialsHandlingClassLib/ConveyorAttachments/Stopper
Attributes	None

RoleClass name	<b>Pusher</b>
Description	Diverts materials from one conveyor line to another.
Parent class	MaterialsHandlingClassLib/ConveyorAttachments
Path for element reference	MaterialsHandlingClassLib/ConveyorAttachments/Pusher
Attributes	None

RoleClass name	<b>EndEffector</b>
Description	Represents a device at the end of a manipulator, designed to interact with the environment.
Parent class	<b>Undefined with the required libraries</b>
Path for element reference	MaterialsHandlingClassLib/EndEffector
Attributes	None

RoleClass name	<b>Gripper</b>
Description	Represents a device which enables the holding of an object to be manipulated.





	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

Parent class	MaterialsHandlingClassLib/EndEffector
Path for element reference	MaterialsHandlingClassLib/EndEffector/Gripper
Attributes	<p><b>Weight</b> (DataType="xs:double", Unit="kg") Weight of the gripper.</p>
	<p><b>GrippingForce</b> (DataType="xs:double", Unit="N") Gripping force of the handling element.</p>

RoleClass name	<b>HydraulicGripper</b>
Description	Represent s a hydraulic driven gripper.
Parent class	MaterialsHandlingClassLib/EndEffector/Gripper
Path for element reference	MaterialsHandlingClassLib/EndEffector/Gripper/HydraulicGripper
Attributes	None

RoleClass name	<b>VacuumGripper</b>
Description	Represent s a vacuum based gripping mechanism.
Parent class	MaterialsHandlingClassLib/EndEffector/Gripper
Path for element reference	MaterialsHandlingClassLib/EndEffector/Gripper/VacuumGripper
Attributes	None

RoleClass name	<b>ServoElectricGripper</b>
Description	Represent s a servo electric driven gripper.
Parent class	MaterialsHandlingClassLib/EndEffector/Gripper
Path for element reference	MaterialsHandlingClassLib/EndEffector/Gripper/ServoElectricGripper
Attributes	None



	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---

RoleClass name	<b>MagneticGripper</b>
Description	Represent s a magnetism based gripping mechanism.
Parent class	MaterialsHandlingClassLib/EndEffector/Gripper
Path for element reference	MaterialsHandlingClassLib/EndEffector/Gripper/MagneticGripper
Attributes	None

RoleClass name	<b>PneumaticGripper</b>
Description	Represent s a pneumatic driven gripper.
Parent class	MaterialsHandlingClassLib/EndEffector/Gripper
Path for element reference	MaterialsHandlingClassLib/EndEffector/Gripper/PneumaticGripper
Attributes	None

RoleClass name	<b>DriveTechnology</b>
Description	A general role for the driving technology in material flow. Could be electrical, pneumatical or hydraulic.
Parent class	<b>Undefined with the required libraries</b>
Path for element reference	MaterialsHandlingClassLib/DriveTechnology
Attributes	<b>MomentOfInertia</b> (DataType="xs:double", Unit="kgm^2") Moment of inertia for the load.

RoleClass name	<b>LoadingEquipment</b>
Description	A general role for loading equipments like containers, pallets, boxes. They move together with the product on the production line. It is also possible that they are delivered together with the product. That is an element which has the characteristics of both the product and the resource.
Parent class	AutomationMLBaseRole/Product
Path for element	MaterialsHandlingClassLib/LoadingEquipment



	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

reference	
Attributes	<p><b>Width</b> (DataType="xs:double", Unit="m") Width of the loading equipment.</p>
	<p><b>Height</b> (DataType="xs:double", Unit="m") Height of the loading equipment.</p>
	<p><b>Capacity</b> (DataType="xs:double", Unit="kg") Capacity of the loading equipment.</p>
	<p><b>Length</b> (DataType="xs:double", Unit="m") Length of the loading equipment.</p>

RoleClass name	<b>Container</b>
Description	Represents any container for materials.
Parent class	MaterialsHandlingClassLib/LoadingEquipment
Path for element reference	MaterialsHandlingClassLib/LoadingEquipment/Container
Attributes	None

RoleClass name	<b>Pallet</b>
Description	Represents any pallet for materials.
Parent class	MaterialsHandlingClassLib/LoadingEquipment
Path for element reference	MaterialsHandlingClassLib/LoadingEquipment/Pallet
Attributes	None

RoleClass name	<b>Box</b>
Description	Represents any pallet for materials.
Parent class	MaterialsHandlingClassLib/LoadingEquipment
Path for element reference	MaterialsHandlingClassLib/LoadingEquipment/Box


	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

Attributes	None
------------	------


RoleClass name	<b>ComboElement</b>
Description	A marker role to show that the component consists of subcomponents.
Parent class	AutomationMLBaseRole/Structure/ResourceStructure
Path for element reference	MaterialsHandlingClassLib/ComboElement
Attributes	None


### Interface Classes

#### MaterialsHandlingInterfaceClassLib

-  MaterialHandlingProductionProcessConnector { **Class** AutomationMLBaseInterface }

#### MaterialHandlingInterface { **Class** Order }



-  MaterialHandlingConnector { **Class** MaterialHandlingInterface }

-  MaterialHandlingConnectionPointConnector { **Class** MaterialHandlingInterface }

The *MaterialHandlingInterfaceClassLib* contains four interface class types. The *MaterialHandlingInterface* is a base interface for the linking of components in the material flow. It is derived from AutomationML standard interface *Order* and inherits its *Direction* attribute. The *Direction* attribute defines if the connection point is used for input, output or both. There are two interfaces derived from this interface: *MaterialHandlingConnector* and *MaterialHandlingConnectionPointConnector*. *MaterialHandlingConnector* is designed to connect the components for the module level. *MaterialHandlingConnectionPointConnector* connects the components for the port level. The role class *MaterialHandlingConnectionPoint* represents a container for the *MaterialHandlingConnectionPointConnector* interface.

The last interface in the library *MaterialHandlingProductionProcessConnector* is for linking material flow requirement modules and manufacturing technology. So that, it is possible to combine different aspects in the plant design.



InterfaceClass name	<b>MaterialHandlingInterface</b>
Description	General interface to define the connection of material flow elements.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order
Path for element reference	MaterialsHandlingInterfaceClassLib/MaterialHandlingInterface
Attributes	<p><b>Direction</b> (DataType="xs:string", inherited)</p> <p>Allowed values are "In", "Out" and "InOut" describing the direction of the material flow.</p>

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

InterfaceClass name	<b>MaterialHandlingConnector</b>
Description	Connects the material handling components directly with each other (1. Level).
Parent class	MaterialsHandlingInterfaceClassLib/MaterialHandlingInterface
Path for element reference	MaterialsHandlingInterfaceClassLib/MaterialHandlingInterface/MaterialHandlingConnector
Attributes	<p><b>Direction</b> (DataType="xs:string", inherited)</p> <p>Allowed values are "In", "Out" and "InOut" describing the direction of the material flow.</p>

InterfaceClass name	<b>MaterialHandlingConnectionPointConnector</b>
Description	Connects the material handling components through connection points (2. Level).
Parent class	MaterialsHandlingInterfaceClassLib/MaterialHandlingInterface
Path for element reference	MaterialsHandlingInterfaceClassLib/MaterialHandlingInterface/MaterialHandlingConnectionPointConnector
Attributes	<p><b>Direction</b> (DataType="xs:string", inherited)</p> <p>Allowed values are "In", "Out" and "InOut" describing the direction of the material flow.</p>

InterfaceClass name	<b>MaterialHandlingProductionProcessConnector</b>
Description	Defines the connection between material handling level and production process level.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
Path for element reference	MaterialsHandlingInterfaceClassLib/MaterialHandlingProductionProcessConnector
Attributes	None

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

InterfaceClass name	<b>MaterialHandlingConnectionPoint</b>
Description	Represents the point where the material transfer occurs.
Parent class	ResourceStructure
Path for element reference	MaterialsHandlingInterfaceClassLib/MaterialHandlingConnectionPoint
Attributes	None



The RoleClasses and InterfaceClasses are formally described by the following AutomationML file content:

**Insertion of AML File of MatieralHandling components → Input from tarakos**

### Role Classes

Not only the resources but also processes are essential in the material handling technology. According to the VDI 2411 guideline (Verein Deutscher Ingenieure (VDI), 1970), following function types are existing in a material flow system: machining, handling, conveying, storing and placing. The guideline VDI 2860 (Verein deutscher Ingenieure (VDI), 1990) defines only conveying, storage and handling as the functional elements in the material flow (see **Fehler! Verweisquelle konnte nicht gefunden werden.**). It also defines and systematizes the handling to elementary and composite functions: saving, changing quantities, moving, securing and verification. The role classes of the material handling processes are created generally after the standard guideline VDI 2860 to serve as requirements when during planning information about the material handling process need to be defined. Nevertheless, extra roles are also needed to describe requirements for some existing activities in the workflow, e.g. roles for congestion behavior or roles such as source, sink, loading process, discharging process, etc.

- MaterialsHandlingProcessClassLib
  - Behavior { **Class** Process }
    - Congestion { **Class** Behavior }
    - Priority { **Class** Behavior }
    - Consolidation { **Class** Behavior }
    - Sorting { **Class** Behavior }
    - Dividing { **Class** Behavior }
      - Partition { **Class** Dividing }
      - Distribution { **Class** Dividing }
      - Branching { **Class** Dividing }
  - Moving { **Class** Process }
    - Turning { **Class** Moving }
    - Translating { **Class** Moving }
    - Swinging { **Class** Moving }
    - Orientating { **Class** Moving }
    - Positioning { **Class** Moving }
    - Arranging { **Class** Moving }
    - Directing { **Class** Moving }
    - HandingOver { **Class** Moving }
    - Conveying { **Class** Moving }
  - Securing { **Class** Process }
    - Holding { **Class** Securing }
    - Loosening { **Class** Securing }
    - Tightening { **Class** Securing }
    - Relieving { **Class** Securing }
  - Verification { **Class** Process }
    - Inspection { **Class** Verification }
      - PresenceCheck { **Class** Inspection }
      - IdentityCheck { **Class** Inspection }
      - ShapeCheck { **Class** Inspection }
      - SizeCheck { **Class** Inspection }
      - ColorCheck { **Class** Inspection }
      - WeightCheck { **Class** Inspection }
      - PositionCheck { **Class** Inspection }
      - OrientationCheck { **Class** Inspection }
    - Measuring { **Class** Verification }
      - Counting { **Class** Measuring }
      - OrientationMeasuring { **Class** Measuring }
      - PositionMeasuring { **Class** Measuring }
  - Source&Sink { **Class** Process }
    - Source { **Class** Source&Sink }
    - Sink { **Class** Source&Sink }
  - GoodProcess { **Class** Process }
    - Loading { **Class** GoodProcess }
    - Discharging { **Class** GoodProcess }

	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---



RoleClass name	<b>Behavior</b>
Description	Defines handling processes for distribution, routing, etc.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process
Path for element reference	MaterialsHandlingProcessesLib/Behaviour
Attributes	None

RoleClass name	<b>Congestion</b>
Description	Defines accumulation behavior.
Parent class	MaterialsHandlingProcessesLib/Behaviour
Path for element reference	MaterialsHandlingProcessesLib/Behaviour/Congestion
Attributes	None

RoleClass name	<b>Priority</b>
Description	Defines the priority in the material flow.
Parent class	MaterialsHandlingProcessesLib/Behaviour
Path for element reference	MaterialsHandlingProcessesLib/Behaviour/Priority
Attributes	None

RoleClass name	<b>Consolidation</b>
Description	Defines consolidation of multiple material handling objects.
Parent class	MaterialsHandlingProcessesLib/Behaviour
Path for element reference	MaterialsHandlingProcessesLib/Behaviour/Consolidation
Attributes	None





	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

RoleClass name	<b>Sorting</b>
Description	Defines the sorting process in material handling.
Parent class	MaterialsHandlingProcessesLib/Behaviour
Path for element reference	MaterialsHandlingProcessesLib/Behaviour/Sorting
Attributes	None

RoleClass name	<b>Dividing</b>
Description	A general process role to define how to create subsets from an amount.
Parent class	MaterialsHandlingProcessesLib/Behaviour
Path for element reference	MaterialsHandlingProcessesLib/Behaviour/Dividing
Attributes	None

RoleClass name	<b>Partition</b>
Description	Defines a process to form subsets with defined size.
Parent class	MaterialsHandlingProcessesLib/Behaviour/Dividing
Path for element reference	MaterialsHandlingProcessesLib/Behaviour/Dividing/Partition
Attributes	None

RoleClass name	<b>Distribution</b>
Description	Defines a process to form subsets with defined size and transfer them to a defined place.
Parent class	MaterialsHandlingProcessesLib/Behaviour/Dividing
Path for element reference	MaterialsHandlingProcessesLib/Behaviour/Dividing/Distribution
Attributes	None



	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

RoleClass name	<b>Branching</b>
Description	Defines a process to split a material flow to smaller flows.
Parent class	MaterialsHandlingProcessesLib/Behaviour/Dividing
Path for element reference	MaterialsHandlingProcessesLib/Behaviour/Dividing/Branching
Attributes	None

RoleClass name	<b>Moving</b>
Description	Defines a changement in the spatial arrangement of a body.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process
Path for element reference	MaterialsHandlingProcessesLib/Moving
Attributes	None

RoleClass name	<b>Turning</b>
Description	Changing the orientation of the body without changing the position.
Parent class	MaterialsHandlingProcessesLib/Moving
Path for element reference	MaterialsHandlingProcessesLib/Moving/Turning
Attributes	None

RoleClass name	<b>Translating</b>
Description	Moving the body in a linear direction without changing the orientation.
Parent class	MaterialsHandlingProcessesLib/Moving
Path for element reference	MaterialsHandlingProcessesLib/Moving/Translating
Attributes	None



	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---

RoleClass name	<b>Swinging</b>
Description	Rotation of the body over a center of an axis, which is not within the body, resulting an orientation and position change.
Parent class	MaterialsHandlingProcessesLib/Moving
Path for element reference	MaterialsHandlingProcessesLib/Moving/Swinging
Attributes	None

RoleClass name	<b>Orientating</b>
Description	Change of the orientation from an undefined to a defined one. Position change of the body is disregarded.
Parent class	MaterialsHandlingProcessesLib/Moving
Path for element reference	MaterialsHandlingProcessesLib/Moving/Orientating
Attributes	None

RoleClass name	<b>Positioning</b>
Description	Moving a body to a defined position. Orientation of the body is disregarded.
Parent class	MaterialsHandlingProcessesLib/Moving
Path for element reference	MaterialsHandlingProcessesLib/Moving/Positioning
Attributes	None

RoleClass name	<b>Arranging</b>
Description	Moving a body from an undefined place to a defined orientation and position providing arrangement.
Parent class	MaterialsHandlingProcessesLib/Moving
Path for element reference	MaterialsHandlingProcessesLib/Moving/Arranging

	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---



Attributes	None
------------	------

RoleClass name	<b>Directing</b>
Description	Moving a body on a defined path from a defined place to another defined place.
Parent class	MaterialsHandlingProcessesLib/Moving
Path for element reference	MaterialsHandlingProcessesLib/Moving/Directing
Attributes	None

RoleClass name	<b>HandingOver</b>
Description	Moving a body on an undefined path from a defined place to another defined place.
Parent class	MaterialsHandlingProcessesLib/Moving
Path for element reference	MaterialsHandlingProcessesLib/Moving/HandingOver
Attributes	None

RoleClass name	<b>Conveying</b>
Description	Moving a body from any place to another place. Orientation and position of the body during the movement is not necessarily defined.
Parent class	MaterialsHandlingProcessesLib/Moving
Path for element reference	MaterialsHandlingProcessesLib/Moving/Conveying
Attributes	None

RoleClass name	<b>Securing</b>
Description	A general role to define securing a defined status.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process

	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---



Path for element reference	MaterialsHandlingProcessesLib/Securing
Attributes	None

RoleClass name	<b>Holding</b>
Description	Securing the orientation and position of a body.
Parent class	MaterialsHandlingProcessesLib/Securing
Path for element reference	MaterialsHandlingProcessesLib/Securing/Holding
Attributes	None

RoleClass name	<b>Loosening</b>
Description	Loosening the orientation and position of a body, reversal of holding.
Parent class	MaterialsHandlingProcessesLib/Securing
Path for element reference	MaterialsHandlingProcessesLib/Securing/Loosening
Attributes	None

RoleClass name	<b>Tightening</b>
Description	Securing the orientation and position of a body with applying force.
Parent class	MaterialsHandlingProcessesLib/Securing
Path for element reference	MaterialsHandlingProcessesLib/Securing/Tightening
Attributes	None

RoleClass name	<b>Relieving</b>
Description	Relieving the force which is blocking the orientational and positional change, reversal of tightening.

	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---



Parent class	MaterialsHandlingProcessesLib/Securing
Path for element reference	MaterialsHandlingProcessesLib/Securing/Relieving
Attributes	None

RoleClass name	<b>Verification</b>
Description	Defines a general role for inspection and measuring of properties and status.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process
Path for element reference	MaterialsHandlingProcessesLib/Verification
Attributes	None

RoleClass name	<b>Inspection</b>
Description	Inspection of properties and status. Admission of information, comparison with target properties, status or decisions.
Parent class	MaterialsHandlingProcessesLib/Verification
Path for element reference	MaterialsHandlingProcessesLib/Verification/Inspection
Attributes	None

RoleClass name	<b>PresenceCheck</b>
Description	Determining if a body is present in a defined place.
Parent class	MaterialsHandlingProcessesLib/Verification/Inspection
Path for element reference	MaterialsHandlingProcessesLib/Verification/Inspection/PresenceCheck
Attributes	None

RoleClass name	<b>IdentityCheck</b>
----------------	----------------------

	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---



Description	Determining if a body satisfies the defined properties.
Parent class	MaterialsHandlingProcessesLib/Verification/Inspection
Path for element reference	MaterialsHandlingProcessesLib/Verification/Inspection/IdentityCheck
Attributes	None

RoleClass name	<b>ShapeCheck</b>
Description	Determining if a body has the defined shape.
Parent class	MaterialsHandlingProcessesLib/Verification/Inspection
Path for element reference	MaterialsHandlingProcessesLib/Verification/Inspection/ShapeCheck
Attributes	None

RoleClass name	<b>SizeCheck</b>
Description	Determining if a body has the defined sizes/measurements.
Parent class	MaterialsHandlingProcessesLib/Verification/Inspection
Path for element reference	MaterialsHandlingProcessesLib/Verification/Inspection/SizeCheck
Attributes	None

RoleClass name	<b>ColorCheck</b>
Description	Determining if a body or body areas have the defined colors.
Parent class	MaterialsHandlingProcessesLib/Verification/Inspection
Path for element reference	MaterialsHandlingProcessesLib/Verification/Inspection/ColorCheck
Attributes	None

RoleClass name	<b>WeightCheck</b>
----------------	--------------------

	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---

Description	Determining if a body has the defined weight.
Parent class	MaterialsHandlingProcessesLib/Verification/Inspection
Path for element reference	MaterialsHandlingProcessesLib/Verification/Inspection/WeightCheck
Attributes	None



RoleClass name	<b>PositionCheck</b>
Description	Determining if a body has the defined position.
Parent class	MaterialsHandlingProcessesLib/Verification/Inspection
Path for element reference	MaterialsHandlingProcessesLib/Verification/Inspection/PositionCheck
Attributes	None

RoleClass name	<b>OrientationCheck</b>
Description	Determining if a body has the defined orientation.
Parent class	MaterialsHandlingProcessesLib/Verification/Inspection
Path for element reference	MaterialsHandlingProcessesLib/Verification/Inspection/OrientationCheck
Attributes	None

RoleClass name	<b>Measuring</b>
Description	Measuring the values in accordance to reference values.
Parent class	MaterialsHandlingProcessesLib/Verification
Path for element reference	MaterialsHandlingProcessesLib/Verification/Measuring
Attributes	None

RoleClass name	<b>Counting</b>
----------------	-----------------





	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

Description	Determines the amount of bodies.
Parent class	MaterialsHandlingProcessesLib/Verification/Measuring
Path for element reference	MaterialsHandlingProcessesLib/Verification/Measuring/Counting
Attributes	None

RoleClass name	<b>OrientationMeasuring</b>
Description	Determines the orientation of a body in accordance to a reference coordinate system.
Parent class	MaterialsHandlingProcessesLib/Verification/Measuring
Path for element reference	MaterialsHandlingProcessesLib/Verification/Measuring/OrientationMeasuring
Attributes	None

RoleClass name	<b>PositionMeasuring</b>
Description	Determines the position of a body in accordance to a reference coordinate system.
Parent class	MaterialsHandlingProcessesLib/Verification/Measuring
Path for element reference	MaterialsHandlingProcessesLib/Verification/Measuring/PositionMeasuring
Attributes	None



RoleClass name	<b>SourceAndSink</b>
Description	General role to define source and sink processes.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process
Path for element reference	MaterialsHandlingProcessesLib/SourceAndSink
Attributes	None

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

RoleClass name	<b>Source</b>
Description	Defines the source of a material flow.
Parent class	MaterialsHandlingProcessesLib/SourceAndSink
Path for element reference	MaterialsHandlingProcessesLib/SourceAndSink/Source
Attributes	<b>DelayMode</b> (DataType="xs:string", Unit="") Specified or determined randomly within a selectable time range.
	<b>CycleTime</b> (DataType="xs:integer", Unit="s") Cycle time.
	<b>DelayTime</b> (DataType="xs:integer", Unit="s") Delay time.
	<b>OutputMode</b> (DataType="xs:string", Unit="") Creating goods randomly or in a synchronized manner.

RoleClass name	<b>Sink</b>
Description	Defines the sink of a material flow.
Parent class	MaterialsHandlingProcessesLib/SourceAndSink
Path for element reference	MaterialsHandlingProcessesLib/SourceAndSink/Sink
Attributes	None

RoleClass name	<b>GoodProcess</b>
Description	Defines the sink of a material flow.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process
Path for element reference	MaterialsHandlingProcessesLib/GoodProcess
Attributes	<b>ProcessTime</b> (DataType="xs:nonNegativeInteger", Unit="s") Process time

	<p style="text-align: center;"> <b>ENTOC</b>          Engineering tool chain for efficient and iterative          development of smart factories          ITEA 3, 15015          Project Coordinator: Thomas Bär, Daimler AG       </p>	
--	---	---

RoleClass name	<b>Loading</b>
Description	Defines a good loading process.
Parent class	MaterialsHandlingProcessesLib/GoodProcess
Path for element reference	MaterialsHandlingProcessesLib/GoodProcess/Loading
Attributes	<b>ProcessTime</b> (DataType="xs:nonNegativeInteger", Unit="s") Process time








RoleClass name	<b>Discharging</b>
Description	Defines a good discharging process.
Parent class	MaterialsHandlingProcessesLib/GoodProcess
Path for element reference	MaterialsHandlingProcessesLib/GoodProcess/Discharging
Attributes	<b>ProcessTime</b> (DataType="xs:nonNegativeInteger", Unit="s") Process time

The RoleClasses and InterfaceClasses are formally described by the following AutomationML file content:

**Insertion of AML File of MatieralHandlingProcesses → Input from tarakos**



### 10.1.5 Chapter 6 – Production Facility Requirement Specifications as a Result of PPR-Model-Based Production Planning

#### Role Classes

- ⤴  GenericAMLRequirementModelRoleClassLib
  - ⤴  RequirementStructure { **Class** Structure }
  - ⤴  Requirement { **Class** AutomationMLBaseRole }
    -  RequirementSubject { **Class** AutomationMLBaseRole }
    -  RequirementObject { **Class** AutomationMLBaseRole }
    -  RequirementCondition { **Class** AutomationMLBaseRole }
    -  RequirementAction { **Class** AutomationMLBaseRole }

RoleClass name	<b>RequirementStructure</b>
Description	The RoleClass "RequirementStructure" represents an element to structure the requirement specifications.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Structure
Attributes	<p><b>Type</b></p> <p>The attribute "Type" defines the type of the structure. Recommended types are document, specification, chapter, subchapter, subsubchapter, section.</p>



RoleClass name	<b>Requirement</b>
Description	The RoleClass "Requirement" represents a specific requirement dependent on the characteristic, perspective, liability and system activity.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Attributes	<p><b>Characteristic</b></p> <p>The attribute "Characteristic" describes the characteristic of a "Requirement". This attribute should be used to define specific boilerplates.</p> <p><b>Perspective</b></p> <p>The attribute "Perspective" defines the view of the requirement and defines the characteristic in a more detailed way. Recommended perspectives are product, process, resource, time plan.</p> <p><b>Liability</b></p> <p>The attribute "Liability" defines the liability of the specific requirement. In requirements engineering the types legally binding, strongly recommended and used in future are defined. For a strong requirement specification the use of the</p>

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

	type used in future is not recommended.
	<p><b>SystemActivity</b></p> <p>The attribute "SystemActivity" describes the action of the requirement. Three types are defined in requirements engineering. Process, provide the ability to process and be able to process.</p>
	<p><b>DisplayedName</b></p> <p>The Attribute "DisplayedName" represents the human readyble requirement as a sentence.</p>
	<p><b>ReqID</b></p> <p>The Attribute "ReqID" gives the possibility th identify the Requirement in a more human-readyble way than a UUID does.</p>
	<p><b>StateOfAcceptance</b></p> <p>The attribute "StateOfAcceptance" defines the place where the supplier has the ability to "accept", "decline" or "accept with further changes" the requirement.</p>
	<p><b>Comment</b></p> <p>The attribute "Comment" defines the place where the supplier has the ability to annotate his concerns regarding to the "StateOfAcceptance".</p>

RoleClass name	<b>RequirementSubject</b>
Description	The RoleClass "RequirementSubject" defines the subjects of a specific requirement.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Attributes	<p><b>Article</b></p> <p>The attribute "Article" defines the article of the related subject.</p>

RoleClass name	<b>RequirementObject</b>
Description	The RoleClass "RequirementObject" defines the objects of a specific requirement.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Attributes	<p><b>Article</b></p>



	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

	<p>The attribute "Article" defines the article of the related object.</p>
	<p><b>PrePosition</b></p> <p>The attribute "PrePosition" defines the preposition of the object dependent on the order and purpose of the object.</p>

RoleClass name	<b>RequirementCondition</b>
Description	The RoleClass "RequirementCondition" defines the condition of a specific requirement. This element is optional. It is recommended to implement this class as an abstract class.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Attributes	<p><b>State</b></p> <p>The attribute "State" defines the required state of the condition.</p>
	<p><b>Type</b></p> <p>The attribute "Type" defines the type of the "RequirementCondition". Two types are recommended, "BooelanExpression" and "LogicalConnection".</p>
	<p><b>Characteristic</b></p> <p>The attribute "Characteristic" defines the required characteristic of the typcasted "RequirementCondition".</p>
	<p><b>PrePosition</b></p> <p>The attribute "PrePosition" defines the preposition of the "RequirementCondition" dependent on the order and purpose of it.</p>

RoleClass name	<b>RequirementAction</b>
Description	The RoleClass "RequirementAction" defines the action of a specific requirement.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Attributes	<p><b>Support</b></p> <p>The attribute "Support" describes the wording of the related action.</p>

### Interface Classes

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

- GenericAMLRequirementModelInterfaceClassLib
  - RequirementConnector { Class AutomationMLBaseInterface }
  - OperatorConnector { Class AutomationMLBaseInterface }

InterfaceClass name	<b>RequirementConnector</b>
Description	The InterfaceClass "RequirementConnector" describes the relation between different specific requirements. With the usage of InternalLinks the direction is defined with RefPartnerSideA and RefPartnerSideB.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
Attributes	None

InterfaceClass name	<b>OperatorConnector</b>
Description	The InterfaceClass "OperatorConnector" defines the reference between RequirementCondition and PropertySet. The direction has to be used with the view from the RequirementCondition (RefPartnerSideA) to PropertySet (RefPartnerSideB).
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
Attributes	None

The RoleClasses and InterfaceClasses are formally described by the following AutomationML file content:

```
<?xml version="1.0" encoding="utf-8"?>
<CAEXFile FileName="201805120807_DaimlerRequirementsRoleClassLib.aml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <AdditionalInformation>
    <WriterHeader>
      <WriterName>AutomationML Editor</WriterName>
      <WriterID>916578CA-FE0D-474E-A4FC-9E1719892369</WriterID>
      <WriterVendor>AutomationML e.V.</WriterVendor>
      <WriterVendorURL>www.AutomationML.org</WriterVendorURL>
      <WriterVersion>4.9.2.0</WriterVersion>
      <WriterRelease>4.9.2.0</WriterRelease>
      <LastWritingDateTime>2018-05-
12T12:37:51.3118442+02:00</LastWritingDateTime>
      <WriterProjectTitle>unspecified</WriterProjectTitle>
      <WriterProjectID>unspecified</WriterProjectID>
    </WriterHeader>
  </AdditionalInformation>
  <AdditionalInformation AutomationMLVersion="2.0" />
  <AdditionalInformation UserDefined="true">Author: Daimler AG
Sindelfingen</AdditionalInformation>
  <InterfaceClassLib Name="DaimlerRequirementsInterfaceClassLib" ChangeMode="create">
    <Description>RoleClassLib for Requirement Specification of Prudction
Facilities</Description>
```



ENTOC  
Engineering tool chain for efficient and iterative  
development of smart factories  
ITEA 3, 15015



Project Coordinator: Thomas Bär, Daimler AG

```
<Version>1.0.0</Version>
<InterfaceClass Name="RequirementConnector" ChangeMode="create"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface">
  <Description>The InterfaceClass "RequirementConnector" describes the
realtion between different specific requirements. With the usage of internal links the
direction is defined with RefPartnerSideA and RefPartnerSideB.</Description>
  <Version>1.0.0</Version>
  <Copyright>Daimler AG</Copyright>
</InterfaceClass>
<InterfaceClass Name="OperatorConnector" ChangeMode="create"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface">
  <Description>The InterfaceClass "OperatorConnector" defines the
reference between RequirementCondition and PropertySet. The direction has to be used with the
view from the RequirementCondition (RefPartnerSideA) to PropertySet
(RefPartnerSideB).</Description>
  <Version>1.0.0</Version>
  <Copyright>Daimler AG</Copyright>
</InterfaceClass>
</InterfaceClassLib>
<RoleClassLib Name="DaimlerRequirementsRoleClassLib" ChangeMode="create">
  <Description>RoleClassLib for Requirement Specification of Prudction
Facilities</Description>
  <Version>1.0.0</Version>
  <Copyright>Daimler AG</Copyright>
  <RoleClass Name="RequirementStructure"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Structure"
ChangeMode="create">
  <Description>The RoleClass "RequirementStructure" represents an element
to structure the requirement specifications.</Description>
  <Version>1.0.0</Version>
  <Copyright>Daimler AG</Copyright>
  <Attribute Name="Type" AttributeDataType="xs:string">
    <Description>The attribute "Type" defines the type of the
structure. Recommended types are document, specification, chapter, subchapter, subsubchapter,
section.</Description>
  </Attribute>
  <RoleClass Name="Requirement" ChangeMode="create"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
    <Description>The RoleClass "Requirement" represents a specific
requirement dependent on the characteristic, perspective, liability and system
activity.</Description>
    <Version>1.0.0</Version>
    <Copyright>Daimler AG</Copyright>
    <Attribute Name="Characteristic" AttributeDataType="xs:string">
      <Description>The attribute "Characteristic" describes the
characteristic of a "Requirement". This attribute should be used to define specific
boilerplates.</Description>
    </Attribute>
    <Attribute Name="Perspective" AttributeDataType="xs:string">
      <Description>The attribute "Perspective" defines the view
of the requirement and defines the characteristic in a more detailed way. Recommended
perspectives are product, process, resource, timeplan.</Description>
    </Attribute>
    <Attribute Name="Liability" AttributeDataType="xs:string">
      <Description>The attribute "Liability" defines the
liability of the specific requirement. In requirements engineering the types legally binding,
strongly recommended and used in future are defined. For a strong requirement specification
the use of the type used in future is not recommended.</Description>
    </Attribute>
    <Attribute Name="SystemActivity" AttributeDataType="xs:string">
      <Description>The attribute "SystemActivity" describes the
action of the requirement. Three types are defined in requirements engineering. Process,
provide the ability to process and be able to process.</Description>
    </Attribute>
    <Attribute Name="DisplayedName" AttributeDataType="xs:string">
      <Description>The Attribute "DisplayedName" represents the
human readyble requirement as a sentence.</Description>
    </Attribute>
    <Attribute Name="ReqID" AttributeDataType="xs:string">
      <Description>The Attribute "ReqID" gives the possibility
th identifiy the Requirement in a more human-readyble way than a UUID does.</Description>
```





```
</Attribute>
<Attribute Name="StateOfAcceptance"
AttributeDataType="xs:string">
    <Description>The attribute "StateOfAcceptance" defines
the place where the supplier has the ability to "accept", "decline" or "accept with further
changes" the requirement.</Description>
</Attribute>
<Attribute Name="Comment" AttributeDataType="xs:string">
    <Description>The attribute "Comment" defines the place
where the supplier has the ability to annotate his concerns regarding to the
"StateOfAcceptance"</Description>
</Attribute>
<RoleClass Name="RequirementSubject" ChangeMode="create"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
    <Description>The RoleClass "RequirementSubject" defines
the subjects of a specific requirement.</Description>
<Version>1.0.0</Version>
<Copyright>Daimler AG</Copyright>
<Attribute Name="Article" AttributeDataType="xs:string">
    <Description>The attribute "Article" defines the
article of the related subject.</Description>
</Attribute>
</RoleClass>
<RoleClass Name="RequirementObject" ChangeMode="create"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
    <Description>The RoleClass "RequirementObject" defines
the objects of a specific requirement.</Description>
<Version>1.0.0</Version>
<Copyright>Daimler AG</Copyright>
<Attribute Name="Article" AttributeDataType="xs:string">
    <Description>The attribute "Article" defines the
article of the related object.</Description>
</Attribute>
<Attribute Name="PrePosition">
    <Description>The attribute "PrePosition" defines
the preposition of the objects dependent on the order and purpose of the
objects.</Description>
</Attribute>
</RoleClass>
<RoleClass Name="RequirementCondition" ChangeMode="create"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
    <Description>The RoleClass "RequirementCondition" defines
the condition of a specific requirement. This element is optional.</Description>
<Version>1.0.0</Version>
<Copyright>Daimler AG</Copyright>
<Attribute Name="State" AttributeDataType="xs:boolean">
    <Description>The attribute "State" defines the
required state of the "RequirementCondition".</Description>
</Attribute>
<Attribute Name="Type" AttributeDataType="xs:string">
    <Description>The attribute "Type" defines the
type of the "RequirementCondition". Two types are recommended, "BooelanExpression" and
"LogicalConnection"</Description>
</Attribute>
<Attribute Name="Characteristic"
AttributeDataType="xs:string">
    <Description>The attribute "Characteristic"
defines the required characteristic of the typcasted "RequirementCondition".</Description>
</Attribute>
<Attribute Name="PrePostion"
AttributeDataType="xs:string">
    <Description>The attribute "PrePosition" defines
the preposition of the "RequirementCondition" dependent on the order and purpose of the
"RequirementCondition".</Description>
</Attribute>
</RoleClass>
<RoleClass Name="RequirementAction" ChangeMode="create"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
    <Description>The RoleClass "RequirementAction" defines
the action of a specific requirement.</Description>
<Version>1.0.0</Version>
```



ENTOC  
Engineering tool chain for efficient and iterative  
development of smart factories  
ITEA 3, 15015



Project Coordinator: Thomas Bär, Daimler AG



```
<Copyright>Daimler AG</Copyright>  
<Attribute Name="Support" AttributeDataType="xs:string">  
  <Description>The attribute "Support" describes  
the wording of the action.</Description>  
  </Attribute>  
</RoleClass>  
</RoleClass>  
</RoleClassLib>  
</CAEXFile>
```

## 10.1.6 Requirement Specification based on Natural Language Boilerplates

### Role Classes

- ⤴ RC BoilerplateRoleClassLib
  - ⤴ RC SpecificationStructure { **Class** AutomationMLBaseRole }
    - ⤴ RC Section { **Class** AutomationMLBaseRole }
      - ⤴ RC Boilerplate { **Class** AutomationMLBaseRole }
        - ⤴ RC BoilerplateElement { **Class** AutomationMLBaseRole }
          - ⤵ RC Subject { **Class** BoilerplateElement }
          - ⤵ RC Verb { **Class** BoilerplateElement }
          - ⤵ RC Object { **Class** BoilerplateElement }
          - ⤵ RC FixedText { **Class** BoilerplateElement }
          - ⤵ RC DevelopmentPhase { **Class** BoilerplateElement }
          - ⤵ RC Liability { **Class** BoilerplateElement }
          - ⤵ RC Value { **Class** BoilerplateElement }
          - ⤵ RC Unit { **Class** BoilerplateElement }
          - ⤵ RC Adjective { **Class** BoilerplateElement }
          - ⤵ RC State { **Class** BoilerplateElement }
          - ⤵ RC Location { **Class** BoilerplateElement }
          - ⤵ RC Event { **Class** BoilerplateElement }
          - ⤵ RC AddOn { **Class** BoilerplateElement }
          - ⤵ RC Condition { **Class** BoilerplateElement }
          - ⤵ RC Statement { **Class** BoilerplateElement }
        - RC Comment { **Class** AutomationMLBaseRole }
        - ⤵ RC Graphic { **Class** AutomationMLBaseRole }
        - RC ENTOCBoilerplates { **Class** AutomationMLBaseRole }

RoleClass name	<b>BoilerplateElement</b>
Description	This RoleClass “BoilerplateElement” shall be used to identify an InternalElement, which describes a part of a requirement sentence. It is a place holder for a subject, a verb, an object, an AddOn, or any other part of a requirement.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Attributes	<b>Name</b> A string identifying an instance of the RoleClass "BoilerplateElement". Only characters are allowed according to following regular expression [A-Za-z0-9_-], while the first character is one of [A-Za-z].

	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---



	<p><b>Content</b></p> <p>A string specifying the content of an InternalElement of the RoleClass "BoilerplateElement".</p>
--	---

RoleClass name	<b>Subject</b>
Description	This RoleClass "Subject" shall be used to identify an InternalElement, which describes the subject of a requirement sentence.
Parent class	BoilerplateRoleClassLib/BoilerplateElement
Attributes	None

RoleClass name	<b>Verb</b>
Description	This RoleClass "Verb" shall be used to identify an InternalElement, which describes the verb or activity of a requirement sentence. An activity can for example be performed by a resource.
Parent class	BoilerplateRoleClassLib/BoilerplateElement
Attributes	None

RoleClass name	<b>Object</b>
Description	This RoleClass "Object" shall be used to identify an InternalElement, which describes the object of a requirements sentence.
Parent class	BoilerplateRoleClassLib/BoilerplateElement
Attributes	None

RoleClass name	<b>DevelopmentPhase</b>
Description	This RoleClass "DevelopmentPhase" shall be used to identify an InternalElement, which describes to which development phase the requirement sentence is assigned. The instance can have the value "shall", "will", or none.
Parent class	BoilerplateRoleClassLib/BoilerplateElement

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---



Attributes	None
------------	------

RoleClass name	<b>Liability</b>
Description	This RoleClass "Liability" can be used to identify an InternalElement, which describes the importance of a requirement sentence. The instance can have the values "shall", "should", or "will".
Parent class	BoilerplateRoleClassLib/BoilerplateElement
Attributes	None

RoleClass name	<b>Condition</b>
Description	This RoleClass "Condition" shall be used to identify an InternalElement, which describes the condition of a requirements sentence. An InternalElement with the role "Condition" is usually decomposed into subparts. An example for a requirement sentence with a condition: If Condition, then Statement.
Parent class	BoilerplateRoleClassLib/BoilerplateElement
Attributes	None

RoleClass name	<b>Statement</b>
Description	This RoleClass "Statement" shall be used to identify an InternalElement, which describes the statement of a requirements sentence. An InternalElement with the role "Statement" is usually decomposed into subparts. An example for a requirement sentence with a statement: If Condition, then Statement.
Parent class	BoilerplateRoleClassLib/BoilerplateElement
Attributes	None

RoleClass name	<b>AddOn</b>
Description	This RoleClass "AddOn" shall be used to identify an InternalElement, which describes an optional part of a requirements sentence. An AddOn consist of other boilerplate elements and expresses different arrangements. An example for an AddOn: From the carrier stack.

	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---



Parent class	BoilerplateRoleClassLib/BoilerplateElement
Attributes	None

RoleClass name	<b>Value</b>
Description	This RoleClass "Value" shall be used to identify an InternalElement, which describes a value of a requirements sentence.
Parent class	BoilerplateRoleClassLib/BoilerplateElement
Attributes	<b>Content</b> The attribute content of an InternalElement specifies its current value. It is of type "float" to cover a wide range of possible numbers.

RoleClass name	<b>Unit</b>
Description	This RoleClass "Unit" shall be used to identify an InternalElement, which describes the unit of a value in a requirements sentence.
Parent class	BoilerplateRoleClassLib/BoilerplateElement
Attributes	None

RoleClass name	<b>Adjective</b>
Description	This RoleClass "Adjective" shall be used to identify an InternalElement, which describes an adjective in a requirements sentence.
Parent class	BoilerplateRoleClassLib/BoilerplateElement
Attributes	None

RoleClass name	<b>State</b>
Description	This RoleClass "State" shall be used to identify an InternalElement, which describes the state of a resource in a requirements sentence or condition.
Parent class	BoilerplateRoleClassLib/BoilerplateElement

	<p>ENTOC  Engineering tool chain for efficient and iterative  development of smart factories  ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	--	---



Attributes	None
------------	------

RoleClass name	<b>Location</b>
Description	This RoleClass "Location" shall be used to identify an InternalElement, which describes a location in a requirements sentence. A location is part of the resources.
Parent class	BoilerplateRoleClassLib/BoilerplateElement
Attributes	None

RoleClass name	<b>Event</b>
Description	This RoleClass "Event" shall be used to identify an InternalElement, which describes an event within a requirement sentence or condition.
Parent class	BoilerplateRoleClassLib/BoilerplateElement
Attributes	None

RoleClass name	<b>FixedText</b>
Description	This RoleClass "FixedText" shall be used to identify an InternalElement, which describes the text parts of a requirements sentence which are used to describe the semantic of this sentence. For example: "If", "then", "in", "with", "from", "to", ",", etc.
Parent class	BoilerplateRoleClassLib/BoilerplateElement
Attributes	None

RoleClass name	<b>SpecificationStructure</b>
Description	This RoleClass "SpecificationStructure" shall be used to identify an InternalElement, which builds a structure of a specification. It collects sections, boilerplate instances (requirements), graphics and comments in a structured hierarchy.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole

	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---



Attributes	<p><b>Name</b></p> <p>A string identifying an instance of the RoleClass "SpecificationStructure". Only characters are allowed according to following regular expression [A-Za-z0-9_-], while the first character is one of [A-Za-z].</p>
------------	--

RoleClass name	<b>Section</b>
Description	This RoleClass "Section" shall be used to identify an InternalElement, which describes a section or subsection and collects requirement instances and comments in a structured hierarchy.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Attributes	<p><b>Name</b></p> <p>A string identifying an instance of the RoleClass "Section". Only characters are allowed according to following regular expression [A-Za-z0-9_-], while the first character is one of [A-Za-z].</p>

RoleClass name	<b>Graphic</b>
Description	This RoleClass "Graphic" shall be used to identify an InternalElement, which represents graphical information like illustrations or tables. An InternalElement with the role "Graphic" has an external interface with a link or an URI.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Attributes	<p><b>Name</b></p> <p>A string identifying an instance of the RoleClass "Graphic". Only characters are allowed according to following regular expression [A-Za-z0-9_-], while the first character is one of [A-Za-z].</p>

RoleClass name	<b>Comment</b>
Description	This RoleClass "Comment" shall be used to identify an InternalElement, which as plain text represents additional information in the requirements specification. It also represents requirements as plain text, if these can't be further formalized.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Attributes	<p><b>Name</b></p> <p>A string identifying an instance of the RoleClass "Comment". Only characters are allowed according to following regular expression [A-Za-z0-9_-], while the first</p>






	<p>ENTOC Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015</p> <p>Project Coordinator: Thomas Bär, Daimler AG</p>	
--	---	---

	character is one of [A-Za-z].
	<p><b>PlainText</b> A string with the content of the comment.</p>



RoleClass name	<b>Boilerplate</b>
Description	This RoleClass "Boilerplate" shall be used to identify an InternalElement, which describes a requirement sentence.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Attributes	<p><b>Name</b> A string identifying an instance of the RoleClass "Boilerplate". Only characters are allowed according to following regular expression [A-Za-z0-9_-], while the first character is one of [A-Za-z].</p>
	<p><b>ReqID</b> A string uniquely identifying the instance of a boilerplate, hence a requirement statement. The string consists of a numeration, a classification, as well as a date, to be able to uniquely define and trace the requirement statement within all development phases of the specification. The ReqID is stated together with the requirement statement in the specification document.</p>

RoleClass name	<b>ENTOCBoilerplate</b>
Description	This RoleClass "ENTOCBoilerplates" shall be used to identify an InternalElement, which compiles boilerplates in the SUC.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Attributes	<p><b>Name</b> A string identifying an instance of the RoleClass "Boilerplate". Only characters are allowed according to following regular expression [A-Za-z0-9_-], while the first character is one of [A-Za-z].</p>

### Interface Classes


 BoilerplateInterfaceClassLib  
 RequirementConnector { **Class** AutomationMLBaseInterface }

InterfaceClass name	<b>RequirementConnector</b>
---------------------	-----------------------------

	<b>ENTOC</b> Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015  Project Coordinator: Thomas Bär, Daimler AG	
--	--	---

Description	This InterfaceClass "RequirementConnector" shall be used to describe a reference from a requirement to another.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
Attributes	None

The RoleClasses and InterfaceClasses are formally described by the following AutomationML file content:

```

<?xml version="1.0" encoding="utf-8"?>
<CAEXFile FileName="BRTBoilerplatesRCLib.aml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <AdditionalInformation>
    <WriterHeader>
      <WriterName>[Name des Exporters]</WriterName>
      <WriterID>[Exporter internal Name]</WriterID>
      <WriterVendor>[SW-Hersteller Exporter]</WriterVendor>
      <WriterVendorURL>[SW-Hersteller URL]</WriterVendorURL>
      <WriterVersion>[x.x.x]</WriterVersion>
      <WriterRelease>[xySW Vx]</WriterRelease>
      <LastWritingDateTime>2017-01-
12T13:05:07.3289785+01:00</LastWritingDateTime>
      <WriterProjectTitle>[xy Export Project]</WriterProjectTitle>
      <WriterProjectID>[xy Export Project]</WriterProjectID>
    </WriterHeader>
  </AdditionalInformation>
  <AdditionalInformation AutomationMLVersion="2.0" />
  <ExternalReference Path="AutomationMLInterfaceClassLib.aml"
Alias="AutomationMLInterfaceClassLib" />
  <ExternalReference Path="CommunicationInterfaceClassLib.aml"
Alias="CommunicationInterfaceClassLib" />
  <ExternalReference Path="AutomationMLBaseRoleClassLib.aml"
Alias="AutomationMLBaseRoleClassLib" />
  <InterfaceClassLib Name="BoilerplateInterfaceClassLib">
    <Version>1.0.0</Version>
    <InterfaceClass Name="RequirementConnector"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface">
      <Description>This InterfaceClass "RequirementConnector" shall be used
to describe a reference from a requirement to another.</Description>
    </InterfaceClass>
  </InterfaceClassLib>
  <RoleClassLib Name="BoilerplateRoleClassLib">
    <Description>Role class library for boilerplate specification
components</Description>
    <Version>1.0.0</Version>
    <RoleClass Name="SpecificationStructure"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
      <Description>This RoleClass "SpecificationStructure" shall be used to
identify an InternalElement, which builds a structure of a specification. It collects
sections, boilerplate instances (requirements), graphics and comments in a structured
hierarchy.</Description>
    </RoleClass Name="Section"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
      <Description>This RoleClass "Section" shall be used to identify
an InternalElement, which describes a section or subsection and collects requirement instances
and comments in a structured hierarchy.</Description>
    </RoleClass Name="Boilerplate"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
      <Description>This RoleClass "Boilerplate" shall be used
to identify an InternalElement, which describes a requirement sentence. </Description>
    </RoleClass Name="BoilerplateElement"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
      <Description>This RoleClass "BoilerplateElement"
shall be used to identify an InternalElement, which describes a part of a requirement

```



sentence. It is a place holder for a subject, a verb, an object, an Addn, or any other part of a requirement.</Description>

```
<RoleClass Name="Subject"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
```

<Description>This RoleClass "Subject" shall be used to identify an InternalElement, which describes the subject of a requirement sentence.</Description>

```
<Attribute Name="Attribute1"
AttributeDataType="xs:string">
<Description>Content</Description>
</Attribute>
<ExternalInterface Name="Interface1"
```

```
ID="3056d349-269d-47b4-b1e2-2ecf1b1cacc" />
```

```
</RoleClass>
<RoleClass Name="Verb"
```

```
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
```

<Description>This RoleClass "Verb" shall be used to identify an InternalElement, which describes the verb or activity of a requirement sentence. An activity can for example be performed by a resource.</Description>

```
<ExternalInterface Name="Interface1"
ID="7c652149-f5a1-4e9e-bda1-e5c00f8e50c6" />
```

```
</RoleClass>
<RoleClass Name="Object"
```

```
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
```

<Description>This RoleClass "Object" shall be used to identify an InternalElement, which describes the object of a requirements sentence.</Description>

```
<ExternalInterface Name="Interface1"
ID="a5ed476b-e959-4a5c-ba76-e97f8fb9ee72" />
```

```
</RoleClass>
<RoleClass Name="FixedText"
```

```
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
```

<Description>This RoleClass "FixedText" shall be used to identify an InternalElement, which describes the text parts of a requirements sentence which are used to describe the semantic of this sentence. For example: "If", "then", "in", "with", "from", "to", ",", "...</Description>

```
<ExternalInterface Name="Interface1"
ID="8f1278bc-3e04-443a-a36f-62e75625b6b3" />
```

```
</RoleClass>
<RoleClass Name="DevelopmentPhase"
```

```
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
```

<Description>This RoleClass "DevelopmentPhase" shall be used to identify an InternalElement, which describes to which development phase the requirement sentence is assigned. The instance can have the value "shall", "will", or none.</Description>

```
<ExternalInterface Name="Interface1"
ID="c6c9eb5f-ef3b-4e84-8683-20a814fe0021" />
```

```
</RoleClass>
<RoleClass Name="Liability"
```

```
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
```

<Description>This RoleClass "Liability" can be used to identify an InternalElement, which describes the importance of a requirement sentence. The instance can have the values "shall", "should", or "will".</Description>

```
<ExternalInterface Name="Interface1"
ID="0c576230-7f68-49c5-a050-cb665e7a739e" />
```

```
</RoleClass>
<RoleClass Name="Value"
```

```
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
```



<Description>This RoleClass "Value" shall be used to identify an InternalElement, which describes a value of a requirements sentence.</Description>

```
<ExternalInterface Name="Interface1"
ID="842e9dd4-b774-4ef9-a482-661bdd9fb2f5" />
```

```
</RoleClass>
```



```
<RoleClass Name="Unit"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
    <Description>This RoleClass "Unit" shall
be used to identify an InternalElement, which describes the unit of a value in a requirements
sentence.</Description>
    <ExternalInterface Name="Interfacel"
ID="5e3e75ca-626c-43c3-8a80-ae9eb94e582b" />
    </RoleClass>
    <RoleClass Name="Adjective"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
    <Description>This RoleClass "Adjective"
shall be used to identify an InternalElement, which describes an adjective in a requirements
sentence.</Description>
    <ExternalInterface Name="Interfacel"
ID="715a6786-22dc-438f-a5a2-8cd5d0b35d86" />
    </RoleClass>
    <RoleClass Name="State"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
    <Description>This RoleClass "State" shall
be used to identify an InternalElement, which describes the state of a resource in a
requirements sentence or condition.</Description>
    <ExternalInterface Name="Interfacel"
ID="db2be535-6acd-42ea-8a60-5ad2354b8391" />
    </RoleClass>
    <RoleClass Name="Location"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
    <Description>This RoleClass "Location"
shall be used to identify an InternalElement, which describes a location in a requirements
sentence. A location is part of the resources.</Description>
    <ExternalInterface Name="Interfacel"
ID="63422b74-ba6d-4442-9b7a-d131d7c568c4" />
    </RoleClass>
    <RoleClass Name="Event"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
    <Description>This RoleClass "Event" shall
be used to identify an InternalElement, which describes an event within a requirement sentence
or condition.</Description>
    <ExternalInterface Name="Interfacel"
ID="a95b9bd4-ef5d-4ebc-aad0-3c5a129aec1f" />
    </RoleClass>
    <RoleClass Name="AddOn"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
    <Description>This RoleClass "AddOn" shall
be used to identify an InternalElement, which describes an optional part of a requirements
sentence. An AddOn consist of other boilerplate elements and expresses different arrangements.
An example for an AddOn: From the carrier stack.</Description>
    <ExternalInterface Name="Interfacel"
ID="1989fdca-2d2f-4d13-bed7-e5edcb579350" />
    </RoleClass>
    <RoleClass Name="Condition"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
    <Description>This RoleClass "Condition"
shall be used to identify an InternalElement, which describes the condition of a requirements
sentence. An InternalElement with the role "Condition" is usually decomposed into subparts. An
example for a requirement sentence with a condition: If Condition, then
Statement.</Description>
    <ExternalInterface Name="Interfacel"
ID="0637c3bb-a236-457d-b424-c87cc78fbf95" />
    </RoleClass>
    <RoleClass Name="Statement"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
    <Description>This RoleClass "Statement"
shall be used to identify an InternalElement, which describes the statement of a requirements
```

	<b>ENTOC</b> Engineering tool chain for efficient and iterative development of smart factories ITEA 3, 15015  Project Coordinator: Thomas Bär, Daimler AG	
--	--	---

sentence. An InternalElement with the role "Statement" is usually decomposed into subparts. An example for a requirement sentence with a statement: If Condition, then Statement.</Description>

```

                                <ExternalInterface Name="Interfacel"
ID="0c576230-7f68-49c5-a050-cb665e7a739e" />
                                </RoleClass>
                                </RoleClass>
                                </RoleClass>
                                <RoleClass Name="Comment"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
                                <Description>This RoleClass "Comment" shall be used to
identify an InternalElement, which as plain text represents additional information in the
requirements specification. It also represents requirements as plain text, if these can't be
further formalized. </Description>
                                </RoleClass>
                                <RoleClass Name="Graphic"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
                                <Description>This RoleClass "Graphic" shall be used to
identify an InternalElement, which represents graphical information like illustrations or
tables. An InternalElement with the role "Graphic" has an external interface with a link or an
URI.</Description>
                                <ExternalInterface Name="Interfacel" ID="e0c8f655-5cf0-
45f7-9667-816c8aeef905" />
                                </RoleClass>
                                </RoleClass>
                                <RoleClass Name="ENTOCBoilerplates"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
                                <Description>This RoleClass "ENTOCBoilerplates" shall be used to
identify an InternalElement, which compiles boilerplates in the SUC. </Description>
                                </RoleClass>
                                </RoleClassLib>
</CAEXFile>

```

The SystemUnitClasses are formally described by the following AutomationML file content, which additionally contains all necessary libraries:

```

<?xml version="1.0" encoding="utf-8"?>
<CAEXFile FileName="BRTBoilerplatesSUCLib_new.aml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <AdditionalInformation>
    <WriterHeader>
      <WriterName>[Name des Exporters]</WriterName>
      <WriterID>[Exporter internal Name]</WriterID>
      <WriterVendor>[SW-Hersteller Exporter]</WriterVendor>
      <WriterVendorURL>[SW-Hersteller URL]</WriterVendorURL>
      <WriterVersion>[x.x.x]</WriterVersion>
      <WriterRelease>[xySW Vx]</WriterRelease>
      <LastWritingDateTime>2017-01-
12T13:05:07.3289785+01:00</LastWritingDateTime>
      <WriterProjectTitle>[xy Export Project]</WriterProjectTitle>
      <WriterProjectID>[xy Export Project]</WriterProjectID>
    </WriterHeader>
  </AdditionalInformation>
  <AdditionalInformation AutomationMLVersion="2.0" />
  <ExternalReference Path="AutomationMLInterfaceClassLib.aml"
Alias="AutomationMLInterfaceClassLib" />
  <ExternalReference Path="CommunicationInterfaceClassLib.aml"
Alias="CommunicationInterfaceClassLib" />
  <ExternalReference Path="AutomationMLBaseRoleClassLib.aml"
Alias="AutomationMLBaseRoleClassLib" />
  <InstanceHierarchy Name="ProjectScope">
    <Version>1.0.0</Version>
    <InternalElement Name="ProjectManagement" ID="cf592285-e6af-4799-85f8-
c3c0d04d87d1">
      <InternalElement Name="InternalElement1" ID="6c19862c-74ce-45b7-8598-
22ec3b4376ae" />
    </InternalElement>
    <InternalElement Name="RequirementSpecification" ID="8c16a940-c338-4420-92ed-
28fb9d6d900b">

```



ENTOC  
Engineering tool chain for efficient and iterative  
development of smart factories  
ITEA 3, 15015



Project Coordinator: Thomas Bär, Daimler AG

```
<InternalElement Name="R001_20180511"
RefBaseSystemUnitPath="BoilerplateSUCLib/ENTOCBoilerplates/Boilerplate_Ba1" ID="d8e8831a-7825-
4eb0-981f-b8b3e40abd88">
    <InternalElement Name="robot 1" ID="640f6db1-e00a-4d0b-8dcf-
91769e81c600"
RefBaseSystemUnitPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/Boil
erplateElement/Subject">
    <ExternalInterface Name="PreC" ID="97ba8e47-572d-4a6b-
b5d3-f794a9773a4d"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    <ExternalInterface Name="PostC" ID="b77ca686-8ba2-4715-
842f-f83acd685d63"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    <ExternalInterface Name="SubjectC" ID="f5c9cb36-9e07-
4e4d-826c-bfdc433e69d3"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector" />
    <InternalLink Name="InternalLink1" />
    <InternalLink Name="Link1" RefPartnerSideA="640f6db1-
e00a-4d0b-8dcf-91769e81c600:SubjectC" RefPartnerSideB="6ee5e028-93d2-41aa-add2-
45e037999960:Interface1" />
    </InternalElement>
    <InternalElement Name="transport" ID="4e60de76-63b4-4120-8153-
13dbd943ffab"
RefBaseSystemUnitPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/Boil
erplateElement/Verb">
    <ExternalInterface Name="PreC" ID="1036fbbd-6270-453a-
8dfb-75931be11329"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    <ExternalInterface Name="PostC" ID="11892686-13e5-4f37-
97fb-22783972a586"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    </InternalElement>
    <InternalElement Name="left side door" ID="bf32305a-72db-4217-
8757-ad91cd48a8fb"
RefBaseSystemUnitPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/Boil
erplateElement/Object">
    <ExternalInterface Name="PreC" ID="8f395ff0-995a-465d-
b328-a3eef893eadc"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    <ExternalInterface Name="PostC" ID="ded65a78-a683-4762-
8b3d-d635a88e9ccd"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    <ExternalInterface Name="ObjectC" ID="3b6757c5-617b-4250-
a059-51a0464400a9"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector" />
    </InternalElement>
    <InternalElement Name="to" ID="dfc8fd3d-0d46-4f35-b9b2-
44996ff44849"
RefBaseSystemUnitPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/Boil
erplateElement/FixedText">
    <ExternalInterface Name="PreC" ID="ef316f01-0209-4b7f-
b8ef-a142cff5d5ddf"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    <ExternalInterface Name="PostC" ID="74cee679-caee-4f70-
8eb9-e23c4547f4a8"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    </InternalElement>
    <InternalElement Name="welding table 3" ID="d02e8463-bcb5-4c03-
b482-64456a8c35c3"
RefBaseSystemUnitPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/Boil
erplateElement/Location">
    <ExternalInterface Name="PreC" ID="972a7b50-4fac-4f9c-
b50c-e51a3c83ac4f"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    <ExternalInterface Name="PostC" ID="dff421d6-3b71-40c3-
b4e2-11e2f2be3e52"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    </InternalElement>
    <InternalLink Name="Link3" RefPartnerSideA="4e60de76-63b4-4120-
8153-13dbd943ffab:PostC" RefPartnerSideB="bf32305a-72db-4217-8757-ad91cd48a8fb:PreC" />
</InternalElement>
```



ENTOC  
Engineering tool chain for efficient and iterative  
development of smart factories  
ITEA 3, 15015



Project Coordinator: Thomas Bär, Daimler AG

```
<InternalLink Name="Link4" RefPartnerSideA="bf32305a-72db-4217-
8757-ad91cd48a8fb:PostC" RefPartnerSideB="dfc8fd3d-0d46-4f35-b9b2-44996ff44849:PreC" />
<InternalLink Name="Link5" RefPartnerSideA="640f6db1-e00a-4d0b-
8dcf-91769e81c600:PostC" RefPartnerSideB="4e60de76-63b4-4120-8153-13dbd943ffab:PreC" />
<InternalLink Name="Link6" RefPartnerSideA="dfc8fd3d-0d46-4f35-
b9b2-44996ff44849:PostC" RefPartnerSideB="d02e8463-bcb5-4c03-b482-64456a8c35c3:PreC" />
</InternalElement>
</InternalElement>
<InternalElement Name="ResourceStructure" ID="83ffc853-d1de-4ae4-8723-
ce50d31fed97">
  <InternalElement Name="Station 030" ID="2e1942c4-16f2-4e54-90c7-
6d02446df5e7">
    <InternalElement Name="robot 1" ID="6ee5e028-93d2-41aa-add2-
45e037999960">
      <ExternalInterface Name="Interface1" ID="101242b0-4055-
44f1-a3cd-9886efc05c5e"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector" />
      </InternalElement>
    </InternalElement>
  </InternalElement>
  <InternalElement Name="ProcessStructure" ID="e4679b12-a969-449f-84c1-
ebc475cd3f72">
    <InternalElement Name="InternalElement1" ID="607100ab-bb0b-45d6-9adf-
f7902a75621c" />
    </InternalElement>
    <InternalElement Name="ProductStructure" ID="f558cee6-c9ef-4ca9-bc89-
93f9582387a4">
      <InternalElement Name="InternalElement1" ID="2983a0fe-ed00-4052-ae82-
e37c17630996" />
      </InternalElement>
    </InternalElement>
  </InstanceHierarchy>
  <InterfaceClassLib Name="AutomationMLInterfaceClassLib">
    <Description>Standard Automation Markup Language Interface Class Library - Part
1 Content extended with Part 3 and Part 4 Content</Description>
    <Version>2.2.2</Version>
    <InterfaceClass Name="AutomationMLBaseInterface">
      <InterfaceClass Name="Order"
RefBaseClassPath="AutomationMLBaseInterface">
        <Attribute Name="Direction" AttributeDataType="xs:string" />
      </InterfaceClass>
      <InterfaceClass Name="PortConnector"
RefBaseClassPath="AutomationMLBaseInterface" />
      <InterfaceClass Name="InterlockingConnector"
RefBaseClassPath="AutomationMLBaseInterface" />
      <InterfaceClass Name="PPRConnector"
RefBaseClassPath="AutomationMLBaseInterface" />
      <InterfaceClass Name="ExternalDataConnector"
RefBaseClassPath="AutomationMLBaseInterface">
        <Attribute Name="refURI" AttributeDataType="xs:anyURI" />
      <InterfaceClass Name="COLLADAInterface"
RefBaseClassPath="ExternalDataConnector">
        <Attribute Name="refType" AttributeDataType="xs:string"
/>
      </InterfaceClass>
      <InterfaceClass Name="PLCopenXMLInterface"
RefBaseClassPath="ExternalDataConnector">
        <InterfaceClass Name="LogicInterface"
RefBaseClassPath="PLCopenXMLInterface">
          <InterfaceClass Name="SequencingLogicInterface"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnecto
r/PLCopenXMLInterface/LogicInterface" />
          <InterfaceClass Name="BehaviourLogicInterface"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnecto
r/PLCopenXMLInterface/LogicInterface" />
          <InterfaceClass
Name="SequencingBehaviourLogicInterface"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnecto
r/PLCopenXMLInterface/LogicInterface" />
```



ENTOC  
 Engineering tool chain for efficient and iterative  
 development of smart factories  
 ITEA 3, 15015



Project Coordinator: Thomas Bär, Daimler AG

```

      <InterfaceClass Name="InterlockingLogicInterface"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnecto
r/PLCopenXMLInterface/LogicInterface" />
      </InterfaceClass>
      <InterfaceClass Name="LogicElementInterface"
RefBaseClassPath="PLCopenXMLInterface" />
      <InterfaceClass Name="VariableInterface"
RefBaseClassPath="PLCopenXMLInterface">
      <InterfaceClass
Name="InterlockingVariableInterface"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnecto
r/PLCopenXMLInterface/VariableInterface">
      <Attribute Name="SafeConditionEquals"
AttributeDataType="xs:boolean">
      <DefaultValue>true</DefaultValue>
      </Attribute>
      </InterfaceClass>
      </InterfaceClass>
      </InterfaceClass>
      </InterfaceClass>
      <InterfaceClass Name="Communication"
RefBaseClassPath="AutomationMLBaseInterface">
      <InterfaceClass Name="SignalInterface"
RefBaseClassPath="Communication" />
      </InterfaceClass>
      <InterfaceClass Name="AttachmentInterface"
RefBaseClassPath="AutomationMLBaseInterface" />
      </InterfaceClass>
      </InterfaceClassLib>
      <InterfaceClassLib Name="BoilerplateInterfaceClassLib">
      <Version>1.0.0</Version>
      <InterfaceClass Name="RequirementConnector"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface">
      <Description>This InterfaceClass "RequirementConnector" shall be used
to describe a reference from a requirement to another.</Description>
      </InterfaceClass>
      </InterfaceClassLib>
      <RoleClassLib Name="AutomationMLBaseRoleClassLib">
      <Description>Automation Markup Language Base Role Class Library - Part 1
Content extended with Part 3 and Part 4 Content</Description>
      <Version>2.2.2</Version>
      <RoleClass Name="AutomationMLBaseRole">
      <RoleClass Name="Group" RefBaseClassPath="AutomationMLBaseRole">
      <Attribute Name="AssociatedFacet" AttributeDataType="xs:string" />
      />
      <RoleClass Name="InterlockingSourceGroup"
RefBaseClassPath="Group" />
      <RoleClass Name="InterlockingTargetGroup"
RefBaseClassPath="Group" />
      </RoleClass>
      <RoleClass Name="Facet" RefBaseClassPath="AutomationMLBaseRole" />
      <RoleClass Name="Port" RefBaseClassPath="AutomationMLBaseRole">
      <Attribute Name="Direction" AttributeDataType="xs:string" />
      <Attribute Name="Cardinality">
      <Attribute Name="MinOccur"
AttributeDataType="xs:unsignedInt" />
      <Attribute Name="MaxOccur"
AttributeDataType="xs:unsignedInt" />
      </Attribute>
      <Attribute Name="Category" AttributeDataType="xs:string" />
      <ExternalInterface Name="ConnectionPoint" ID="9942bd9c-c19d-
44e4-a197-11b9edf264e7"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PortConnector" />
      </RoleClass>
      <RoleClass Name="Resource" RefBaseClassPath="AutomationMLBaseRole" />
      <RoleClass Name="Product" RefBaseClassPath="AutomationMLBaseRole" />
      <RoleClass Name="Process" RefBaseClassPath="AutomationMLBaseRole" />
      <RoleClass Name="Structure" RefBaseClassPath="AutomationMLBaseRole">
      <RoleClass Name="ProductStructure" RefBaseClassPath="Structure"
  
```





```

        <RoleClass Name="ProcessStructure" RefBaseClassPath="Structure"
/>
        <RoleClass Name="ResourceStructure" RefBaseClassPath="Structure"
/>
    </RoleClass>
    <RoleClass Name="PropertySet" RefBaseClassPath="AutomationMLBaseRole"
/>
    <RoleClass Name="Frame" RefBaseClassPath="AutomationMLBaseRole" />
    <RoleClass Name="LogicObject" RefBaseClassPath="AutomationMLBaseRole"
/>
    </RoleClass>
</RoleClassLib>
<RoleClassLib Name="EntocSpecRCL">
    <Version>1.0.0</Version>
    <RoleClass Name="Boilerplate">
        <RoleClass Name="Characteristic"
RefBaseClassPath="EntocSpecRCL/Boilerplate" />
    </RoleClass>
    <RoleClass Name="Requirement">
        <RoleClass Name="RequirementPart"
RefBaseClassPath="EntocSpecRCL/Requirement">
            <RoleClass Name="Object"
RefBaseClassPath="EntocSpecRCL/Requirement/RequirementPart" />
            <RoleClass Name="SystemActivity"
RefBaseClassPath="EntocSpecRCL/Requirement/RequirementPart" />
            <RoleClass Name="Liability"
RefBaseClassPath="EntocSpecRCL/Requirement/RequirementPart" />
            <RoleClass Name="Subject"
RefBaseClassPath="EntocSpecRCL/Requirement/RequirementPart" />
            <RoleClass Name="Condition"
RefBaseClassPath="EntocSpecRCL/Requirement/RequirementPart">
                <RoleClass Name="BooleanCondition"
RefBaseClassPath="EntocSpecRCL/Requirement/RequirementPart/Condition" />
            </RoleClass>
            <RoleClass Name="Action"
RefBaseClassPath="EntocSpecRCL/Requirement/RequirementPart" />
        </RoleClass>
    </RoleClass>
    <RoleClass Name="LogicalConnector">
        <RoleClass Name="Conjunction"
RefBaseClassPath="EntocSpecRCL/LogicalConnector">
            <RoleClass Name="AND"
RefBaseClassPath="EntocSpecRCL/LogicalConnector/Conjunction" />
        </RoleClass>
        <RoleClass Name="Disjunction"
RefBaseClassPath="EntocSpecRCL/LogicalConnector">
            <RoleClass Name="OR"
RefBaseClassPath="EntocSpecRCL/LogicalConnector/Disjunction" />
        </RoleClass>
        <RoleClass Name="Implication"
RefBaseClassPath="EntocSpecRCL/LogicalConnector">
            <RoleClass Name="IMPLIES"
RefBaseClassPath="EntocSpecRCL/LogicalConnector/Implication" />
            <RoleClass Name="IF_THEN"
RefBaseClassPath="EntocSpecRCL/LogicalConnector/Implication" />
            <RoleClass Name="IF_AND_ONLY_IF"
RefBaseClassPath="EntocSpecRCL/LogicalConnector/Implication" />
            <RoleClass Name="ONLY_IF"
RefBaseClassPath="EntocSpecRCL/LogicalConnector/Implication" />
        </RoleClass>
        <RoleClass Name="Biconditional"
RefBaseClassPath="EntocSpecRCL/LogicalConnector">
            <RoleClass Name="XNOR"
RefBaseClassPath="EntocSpecRCL/LogicalConnector/Biconditional" />
        </RoleClass>
        <RoleClass Name="AlternativeDenial"
RefBaseClassPath="EntocSpecRCL/LogicalConnector">
            <RoleClass Name="NAND"
RefBaseClassPath="EntocSpecRCL/LogicalConnector/AlternativeDenial" />
        </RoleClass>
    </RoleClass>
</RoleClassLib>
```



```
<RoleClass Name="JointDenial"
RefBaseClassPath="EntocSpecRCL/LogicalConnector">
  <RoleClass Name="NOR"
RefBaseClassPath="EntocSpecRCL/LogicalConnector/JointDenial" />
  </RoleClass>
  <RoleClass Name="Negation"
RefBaseClassPath="EntocSpecRCL/LogicalConnector">
  <RoleClass Name="NOT"
RefBaseClassPath="EntocSpecRCL/LogicalConnector/Negation" />
  </RoleClass>
  <RoleClass Name="ExclusiveDisjunction"
RefBaseClassPath="EntocSpecRCL/LogicalConnector">
  <RoleClass Name="XOR"
RefBaseClassPath="EntocSpecRCL/LogicalConnector/ExclusiveDisjunction" />
  </RoleClass>
</RoleClass>
<RoleClass Name="BooleanExpression">
  <RoleClass Name="LESS_THAN"
RefBaseClassPath="EntocSpecRCL/BooleanExpression" />
  <RoleClass Name="GREATER_THAN"
RefBaseClassPath="EntocSpecRCL/BooleanExpression" />
  <RoleClass Name="EQUAL_TO"
RefBaseClassPath="EntocSpecRCL/BooleanExpression" />
  <RoleClass Name="LESS_THAN_OR_EQUAL_TO"
RefBaseClassPath="EntocSpecRCL/BooleanExpression" />
  <RoleClass Name="GREATER_THAN_OR_EQUAL_TO"
RefBaseClassPath="EntocSpecRCL/BooleanExpression" />
  <RoleClass Name="NOT_EQUAL_TO"
RefBaseClassPath="EntocSpecRCL/BooleanExpression" />
</RoleClass>
<RoleClass Name="SchedulingConnector">
  <RoleClass Name="StartAfterEnd" />
  <RoleClass Name="EndAfterStart" />
  <RoleClass Name="EndAfterStart" />
  <RoleClass Name="EndAfterEnd" />
</RoleClass>
</RoleClassLib>
<RoleClassLib Name="BoilerplateRoleClassLib">
  <Description>Role class library for boilerplate specification
components</Description>
  <Version>1.0.0</Version>
  <RoleClass Name="SpecificationStructure"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
    <Description>This RoleClass "SpecificationStructure" shall be used to
identify an InternalElement, which builds a structure of a specification. It collects
sections, boilerplate instances (requirements), graphics and comments in a structured
hierarchy.</Description>
    <RoleClass Name="Section"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
      <Description>This RoleClass "Section" shall be used to identify
an InternalElement, which describes a section or subsection and collects requirement instances
and comments in a structured hierarchy.</Description>
      <RoleClass Name="Boilerplate"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
        <Description>This RoleClass "Boilerplate" shall be used
to identify an InternalElement, which describes a requirement sentence. </Description>
        <RoleClass Name="BoilerplateElement"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
          <Description>This RoleClass "BoilerplateElement"
shall be used to identify an InternalElement, which describes a part of a requirement
sentence. It is a place holder for a subject, a verb, an object, an Addn, or any other part of
a requirement.</Description>
          <RoleClass Name="Subject"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
            <Description>This RoleClass "Subject"
shall be used to identify an InternalElement, which describes the subject of a requirement
sentence.</Description>
            <Attribute Name="Attribute1"
AttributeDataType="xs:string">
              <Description>Content</Description>
```



ENTOC  
Engineering tool chain for efficient and iterative  
development of smart factories  
ITEA 3, 15015



Project Coordinator: Thomas Bär, Daimler AG

```
</Attribute>
<ExternalInterface Name="Interfacel"
ID="3056d349-269d-47b4-b1e2-2ecf1b1cacc" />
</RoleClass>
<RoleClass Name="Verb"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
<Description>This RoleClass "Verb" shall
be used to identify an InternalElement, which describes the verb or activity of a requirement
sentence. An activity can for example be performed by a resource.</Description>
<ExternalInterface Name="Interfacel"
ID="7c652149-f5a1-4e9e-bda1-e5c00f8e50c6" />
</RoleClass>
<RoleClass Name="Object"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
<Description>This RoleClass "Object" shall
be used to identify an InternalElement, which describes the object of a requirements
sentence.</Description>
<ExternalInterface Name="Interfacel"
ID="a5ed476b-e959-4a5c-ba76-e97f8fb9ee72" />
</RoleClass>
<RoleClass Name="FixedText"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
<Description>This RoleClass "FixedText"
shall be used to identify an InternalElement, which describes the text parts of a requirements
sentence which are used to describe the semantic of this sentence. For example: "If", "then",
"in", "with", "from", "to", "\", "...</Description>
<ExternalInterface Name="Interfacel"
ID="8f1278bc-3e04-443a-a36f-62e75625b6b3" />
</RoleClass>
<RoleClass Name="DevelopmentPhase"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
<Description>This RoleClass
"DevelopmentPhase" shall be used to identify an InternalElement, which describes to which
development phase the requirement sentence is assigned. The instance can have the value
"shall", "will", or none.</Description>
<ExternalInterface Name="Interfacel"
ID="c6c9eb5f-ef3b-4e84-8683-20a814fe0021" />
</RoleClass>
<RoleClass Name="Liability"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
<Description>This RoleClass "Liability"
can be used to identify an InternalElement, which describes the importance of a requirement
sentence. The instance can have the values "shall", "should", or "will".</Description>
<ExternalInterface Name="Interfacel"
ID="0c576230-7f68-49c5-a050-cb665e7a739e" />
</RoleClass>
<RoleClass Name="Value"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
<Description>This RoleClass "Value" shall
be used to identify an InternalElement, which describes a value of a requirements
sentence.</Description>
<ExternalInterface Name="Interfacel"
ID="842e9dd4-b774-4ef9-a482-661bdd9fb2f5" />
</RoleClass>
<RoleClass Name="Unit"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
<Description>This RoleClass "Unit" shall
be used to identify an InternalElement, which describes the unit of a value in a requirements
sentence.</Description>
<ExternalInterface Name="Interfacel"
ID="5e3e75ca-626c-43c3-8a80-ae9eb94e582b" />
</RoleClass>
```



```
<RoleClass Name="Adjective"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
  <Description>This RoleClass "Adjective"
shall be used to identify an InternalElement, which describes an adjective in a requirements
sentence.</Description>
  <ExternalInterface Name="Interfacel"
ID="715a6786-22dc-438f-a5a2-8cd5d0b35d86" />
</RoleClass>
<RoleClass Name="State"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
  <Description>This RoleClass "State" shall
be used to identify an InternalElement, which describes the state of a resource in a
requirements sentence or condition.</Description>
  <ExternalInterface Name="Interfacel"
ID="db2be535-6acd-42ea-8a60-5ad2354b8391" />
</RoleClass>
<RoleClass Name="Location"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
  <Description>This RoleClass "Location"
shall be used to identify an InternalElement, which describes a location in a requirements
sentence. A location is part of the resources.</Description>
  <ExternalInterface Name="Interfacel"
ID="63422b74-ba6d-4442-9b7a-d131d7c568c4" />
</RoleClass>
<RoleClass Name="Event"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
  <Description>This RoleClass "Event" shall
be used to identify an InternalElement, which describes an event within a requirement sentence
or condition.</Description>
  <ExternalInterface Name="Interfacel"
ID="a95b9bd4-ef5d-4ebc-aad0-3c5a129aec1f" />
</RoleClass>
<RoleClass Name="AddOn"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
  <Description>This RoleClass "AddOn" shall
be used to identify an InternalElement, which describes an optional part of a requirements
sentence. An AddOn consist of other boilerplate elements and expresses different arrangements.
An example for an AddOn: From the carrier stack.</Description>
  <ExternalInterface Name="Interfacel"
ID="1989fdca-2d2f-4d13-bed7-e5edcb579350" />
</RoleClass>
<RoleClass Name="Condition"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
  <Description>This RoleClass "Condition"
shall be used to identify an InternalElement, which describes the condition of a requirements
sentence. An InternalElement with the role "Condition" is usually decomposed into subparts. An
example for a requirement sentence with a condition: If Condition, then
Statement.</Description>
  <ExternalInterface Name="Interfacel"
ID="0637c3bb-a236-457d-b424-c87cc78fbf95" />
</RoleClass>
<RoleClass Name="Statement"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement">
  <Description>This RoleClass "Statement"
shall be used to identify an InternalElement, which describes the statement of a requirements
sentence. An InternalElement with the role "Statement" is usually decomposed into subparts. An
example for a requirement sentence with a statement: If Condition, then
Statement.</Description>
  <ExternalInterface Name="Interfacel"
ID="0c576230-7f68-49c5-a050-cb665e7a739e" />
</RoleClass>
</RoleClass>
</RoleClass>
```



ENTOC  
Engineering tool chain for efficient and iterative  
development of smart factories  
ITEA 3, 15015



Project Coordinator: Thomas Bär, Daimler AG

```
<RoleClass Name="Comment"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
  <Description>This RoleClass "Comment" shall be used to
  identify an InternalElement, which as plain text represents additional information in the
  requirements specification. It also represents requirements as plain text, if these can't be
  further formalized. </Description>
</RoleClass>
<RoleClass Name="Graphic"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
  <Description>This RoleClass "Graphic" shall be used to
  identify an InternalElement, which represents graphical information like illustrations or
  tables. An InternalElement with the role "Graphic" has an external interface with a link or an
  URI.</Description>
  <ExternalInterface Name="Interfacel" ID="e0c8f655-5cf0-
  45f7-9667-816c8aeef905" />
</RoleClass>
</RoleClass>
<RoleClass Name="ENTOCBoilerplates"
RefBaseClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole">
  <Description>This RoleClass "ENTOCBoilerplates" shall be used to
  identify an InternalElement, which compiles boilerplates in the SUC. </Description>
</RoleClass>
</RoleClassLib>
<SystemUnitClassLib Name="BoilerplateSUCLib">
  <Version>1.0.0</Version>
  <SystemUnitClass Name="ENTOCBoilerplates"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/ENTOCBoilerplates">
    <SystemUnitClass Name="Boilerplate_Bal"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate">
      <InternalElement Name="Subject" ID="c125774b-8448-4da4-861d-
  19d7f8889f70"
RefBaseSystemUnitPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/Boil
  erplateElement/Subject">
        <ExternalInterface Name="PreC" ID="ce2d12f6-9c0a-4fee-
  b0f1-f1071120b746"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
          <ExternalInterface Name="PostC" ID="0d000ca0-5a40-4927-
  9536-fb424a097156"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
            <ExternalInterface Name="SubjectC" ID="08ae37dd-e969-
  47b8-8083-d905cf80b3d1"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector" />
              <InternalLink Name="InternalLink1" />
            </InternalElement>
          <InternalElement Name="Verb" ID="f0414b86-3318-4019-97bb-
  52fd128c1792"
RefBaseSystemUnitPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/Boil
  erplateElement/Verb">
                <ExternalInterface Name="PreC" ID="dcce2516-ab5f-4181-
  adca-5fab5e0c8e9c"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
                  <ExternalInterface Name="PostC" ID="1167bdb9-7f64-43fe-
  b682-b09cf52393ef"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
                    </InternalElement>
                  <InternalElement Name="Object" ID="5e7ea656-1c53-452a-b616-
  8b6ada2960f3"
RefBaseSystemUnitPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/Boil
  erplateElement/Object">
                        <ExternalInterface Name="PreC" ID="f92abf40-3df1-4b29-
  bffa-4a3857df99f9"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
                          <ExternalInterface Name="PostC" ID="c84772e6-fa18-4001-
  99d6-ab2beeddc9"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
                            <ExternalInterface Name="ObjectC" ID="713f275f-822b-43cf-
  a817-7702814f3e0c"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector" />
                              </InternalElement>
```



ENTOC  
Engineering tool chain for efficient and iterative  
development of smart factories  
ITEA 3, 15015



Project Coordinator: Thomas Bär, Daimler AG

```
<InternalElement Name="FixedText" ID="517ec510-8b15-46b0-bb2c-3fd7d07cdc8e"
RefBaseSystemUnitPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement/FixedText">
    <ExternalInterface Name="PreC" ID="04712390-c1ca-4859-b7ad-ad4c1ab9a3a8"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    <ExternalInterface Name="PostC" ID="ae7e86c8-2e92-4ca9-ab37-7d990dfb1387"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
</InternalElement>
<InternalElement Name="Location" ID="e311165d-8de1-4554-823a-2f5786a511d2"
RefBaseSystemUnitPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement/Location">
    <ExternalInterface Name="PreC" ID="7c189a3e-6f04-4d9e-b018-a0bfa5b168b3"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    <ExternalInterface Name="PostC" ID="5cc8e71a-9a50-4387-80b8-cff86039af85"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
</InternalElement>
<InternalLink Name="Link3" RefPartnerSideA="f0414b86-3318-4019-97bb-52fd128c1792:PostC" RefPartnerSideB="5e7ea656-1c53-452a-b616-8b6ada2960f3:PreC" />
<InternalLink Name="Link4" RefPartnerSideA="5e7ea656-1c53-452a-b616-8b6ada2960f3:PostC" RefPartnerSideB="517ec510-8b15-46b0-bb2c-3fd7d07cdc8e:PreC" />
<InternalLink Name="Link5" RefPartnerSideA="c125774b-8448-4da4-861d-19d7f8889f70:PostC" RefPartnerSideB="f0414b86-3318-4019-97bb-52fd128c1792:PreC" />
<InternalLink Name="Link6" RefPartnerSideA="517ec510-8b15-46b0-bb2c-3fd7d07cdc8e:PostC" RefPartnerSideB="e311165d-8de1-4554-823a-2f5786a511d2:PreC" />
</SystemUnitClass>
<SystemUnitClass Name="Boilerplate_Ba2"
RefBaseClassPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate">
    <InternalElement Name="Subject" ID="f45b0bec-f785-45ed-a6ff-86191e1caa45"
RefBaseSystemUnitPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement/Subject">
        <ExternalInterface Name="PreC" ID="36e3bc67-a0a8-448d-81dc-73d4f43130f8"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
        <ExternalInterface Name="PostC" ID="96703a20-f107-4537-8dac-07a64400972c"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
        <ExternalInterface Name="SubjectC" ID="5820b963-2f0c-49f3-8c64-ca772f167387"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PPRConnector" />
        <InternalLink Name="InternalLink1" />
    </InternalElement>
    <InternalElement Name="FixedText" ID="70e981ab-2bfc-48ae-8e58-29fdc8cbafe9"
RefBaseSystemUnitPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement/FixedText">
        <ExternalInterface Name="PreC" ID="a4d65f3d-156a-4395-af8c-877c752857ae"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
        <ExternalInterface Name="PostC" ID="26c2ba7e-13fc-4fae-b596-3fef73b9c8e9"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    </InternalElement>
    <InternalElement Name="BooleanExpression" ID="3c97880c-2a51-4697-be90-125bbb2bf21c" RefBaseSystemUnitPath="EntocSpecRCL/BooleanExpression">
        <ExternalInterface Name="PreC" ID="96081e8c-16ea-4b82-8c06-42fa60407beb"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
        <ExternalInterface Name="PostC" ID="7fff0985-f3a7-4c8d-9188-2c129e97b6dd"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    </InternalElement>
    <InternalElement Name="Value" ID="4164ddae-b2bc-434c-9cf1-07270f5cd0fb">
```



ENTOC  
Engineering tool chain for efficient and iterative  
development of smart factories  
ITEA 3, 15015



Project Coordinator: Thomas Bär, Daimler AG

```
RefBaseSystemUnitPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement/Value">
    <ExternalInterface Name="PreC" ID="6610efa3-dc66-4e3c-8a07-fa28cdae45f0"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    <ExternalInterface Name="PostC" ID="3e1dba8f-02b1-4766-86df-833857bd9aa8"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    </InternalElement>
    <InternalElement Name="Unit" ID="8b1e1d43-5fdd-4fe1-8865-e2557a760178"
RefBaseSystemUnitPath="BoilerplateRoleClassLib/SpecificationStructure/Section/Boilerplate/BoilerplateElement/Unit">
    <ExternalInterface Name="PreC" ID="bc588c8e-4707-4b1b-8bef-85e3bb0b5cfb"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    <ExternalInterface Name="PostC" ID="21aa74e7-6673-4dfd-bf1e-0a7d420c2d19"
RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order" />
    </InternalElement>
    <InternalLink Name="Link1" RefPartnerSideA="f45b0bec-f785-45ed-a6ff-86191e1caa45:PostC" RefPartnerSideB="70e981ab-2bfc-48ae-8e58-29fdc8cbafe9:PreC" />
    <InternalLink Name="Link2" RefPartnerSideA="70e981ab-2bfc-48ae-8e58-29fdc8cbafe9:PostC" RefPartnerSideB="3c97880c-2a51-4697-be90-125bbb2bf21c:PreC" />
    <InternalLink Name="Link3" RefPartnerSideA="3c97880c-2a51-4697-be90-125bbb2bf21c:PostC" RefPartnerSideB="4164ddae-b2bc-434c-9cf1-07270f5cd0fb:PreC" />
    <InternalLink Name="Link4" RefPartnerSideA="4164ddae-b2bc-434c-9cf1-07270f5cd0fb:PostC" RefPartnerSideB="8b1e1d43-5fdd-4fe1-8865-e2557a760178:PreC" />
    </SystemUnitClass>
  </SystemUnitClass>
</SystemUnitClassLib>
</CAEXFile>
```