



<AutomationML/>

**The Glue for Seamless
Automation Engineering**

**Application Recommendation
Modelling of Material Handling in
AutomationML**

Version 1.0.0

State: March 2020

Table of contents

Table of contents	2
List of figures	3
Preface	4
1 Introduction.....	5
1.1 Tool chains.....	5
2 Motivation for modelling material handling related information with AutomationML.....	7
2.1 Basics.....	7
2.2 Scope	8
2.3 References.....	8
3 Basic concept.....	9
3.1 Module level (Level 1).....	9
3.2 Port level (Level 2)	11
3.3 Combo Element	13
4 Material handling class libraries	14
4.1 Material handling interface class library.....	14
4.2 Material handling resource role classes	18
4.2.2 Conveyors	20
4.2.3 Positioners.....	23
4.2.4 Carrier mediums.....	27
4.2.5 Connection point	29
4.2.6 Conveyor attachments	29
4.2.7 Load carrying equipments.....	30
4.2.8 Drive technology.....	39
4.2.9 Combo element	39
4.2.10 Identification devices.....	40
4.3 Material handling process role classes.....	41
4.3.1 Handling	44
4.3.2 Moving.....	46
4.3.3 Securing	50
4.3.4 Inspection	52
4.3.5 Source and sink.....	56
4.3.6 Transport unit processes.....	59
4.3.7 Storing	61
4.4 Material handling product role classes	62
5 Examples.....	67
5.1 Level 1 (module level).....	67
5.2 Level 2 (port level)	70

List of figures

<i>Figure 1 Tool chain for material handling system design (Variation 1).....</i>	<i>6</i>
<i>Figure 2 Tool chain for material handling system design (Variation 2).....</i>	<i>6</i>
<i>Figure 3 Example Scene with three conveyors (RC) and one turntable (TT).....</i>	<i>9</i>
<i>Figure 4 Modelling of module level (Level 1). O for output and I for input.....</i>	<i>10</i>
<i>Figure 5 The relationship between two material handling elements in module level</i>	<i>11</i>
<i>Figure 6 Modelling of port level (Level 2).....</i>	<i>12</i>
<i>Figure 7 The relationship between two material handling elements in port level</i>	<i>13</i>
<i>Figure 8 MaterialHandlingInterfaceClassLib.....</i>	<i>14</i>
<i>Figure 9 MaterialHandlingResourceClassLib</i>	<i>18</i>
<i>Figure 10 Sub-functions of the material handling processes and their formation based on [4]</i>	<i>42</i>
<i>Figure 11 MaterialHandlingProcessClassLib.....</i>	<i>43</i>
<i>Figure 12 MaterialHandlingProductClassLib</i>	<i>62</i>
<i>Figure 13 The connection between turntable and conveyors for module level (1. level)</i>	<i>67</i>
<i>Figure 14 Usage of the PPR interface and the Source role to define a source for transport units.....</i>	<i>68</i>
<i>Figure 15 Usage of the PPR interface and the Sink role to define sinks for transport units.....</i>	<i>69</i>
<i>Figure 16 The connection between turntable and conveyors for port level (2. level)</i>	<i>70</i>
<i>Figure 17 Usage of the PPR interface and Loading role to define loading process.....</i>	<i>71</i>
<i>Figure 18 Usage of the TransportUnitCapacity interface and Carrying role to define the load carrier capacity.....</i>	<i>72</i>

Preface

AutomationML provides the basis for an efficient data exchange within the engineering process of production systems. The AutomationML standard series IEC 62714 "Engineering data exchange format for use in industrial automation systems engineering" already contains many use cases and guidelines of how system engineering information is modelled.

In order to specify these definitions with examples, to apply them to specific use cases, and to facilitate the first steps with AutomationML, specific issues for the modelling of data in AutomationML are illustrated in Best Practice Recommendations (BPR) and Application Recommendations (AR).

In addition, the AR shall provide a consistent realisation for specific use cases and shall thus, complement the AutomationML standard documents.

1 Introduction

Material handling is a general term which contains the movement (transportation), protection, storage, control and coding of materials and products through a manufacturing system. The movement of the materials, which is an essential part of material handling, can be defined as the *material flow*. Material flow includes a set of actions in an assembly line in which a product is taken from a loading station, passes through intermediate steps like assembling or packaging, and arrives at the shipment. These operations are made by transportation systems such as carriers, conveyors, vehicles or sometimes robots [1].

This document introduces the usage of AutomationML standard format to define material handling related resources and processes. The eCI@ss standard [2] and the guideline VDI 2411 [3] have influenced the concept modeling of the components, but they are not adopted strictly as the priority lies on the usability of the models for data exchange purpose already in the earlier phases of the engineering planning.

The main approach of this document is to define the material handling resources according to their functionality. Thereby the functional necessity can be determined in the early phases of the layout planning and the corresponding element can be enriched with information through the further phases of production planning. For example, a conveyor is needed to transport goods (mentioned as *transport units* in this document) from one place to another. It can be modelled with a functional semantics in the material handling system without stating the type of the carrying medium. After selecting the carrying medium, it can be specified as a second semantics (in this case “*ConveyorTechnology*” role). The instance of the conveyor object gets the necessary attributes or a child object that adopts this semantics as the carrying medium can be created.

Types of processes in material handling systems are also a vital part of this document. A role class library for processes is created according to the standard guideline VDI 2860 [4], which contains following main process types: moving, securing and verification. Furthermore, the practical modeling experience has suggested the need of some extra semantics: e.g. semantics for accumulation behavior or roles such as source, sink, loading process, unloading process, etc.

The standard IEC 62264 [5] provides a terminology and a set of concepts and models for the integration of control systems with enterprise systems. There are defined classes dealing with the machinery available in production environments such as robots, conveyor belts, etc., as well as structures depicting processes, production steps and operational data. There is an ongoing work to integrate IEC 62264 in AutomationML standard [6], which can be used for a control systems engineering based view of manufacturing systems.

1.1 Tool chains

It is meaningful to define the target tool landscape, in which the material handling related information are involved. The planning of the material handling has been broken down into a series of individual, specialized phases, which involves powerful but separate tools. Software tools, which are widely used in the field, have only limited possibilities to exchange planning data in this heterogeneous system landscape.

The tool chain of material handling design varies in the beginning phase. The first variation starts with the layout sketching and selecting the necessary resources for the process (see *Figure 1*). The information sources are mostly existing projects, PLM/SAP systems or proprietary data bases. Parameters, which can be defined in this step, are name, identification, description, position and geometry of the components, layers and dimensions of the production system. CAD tools (i.e. NX-MCD, AutoCAD, Catia, Solidworks, Pro-Engineer, etc.) are used for 2D or 3D layout sketching.

The next step in the design would be the process planning and detailed layout planning. Successor-predecessor relationship, process flow, operations and actions are defined in detail. Ergonomics analysis and offline robot programming are also some important activities, which have influence on

material handling design. Import tools for this phase of planning are Process Designer, Tecnomatix, tarakos tool suite and Delmia.

The layout and process planning are validated through the material flow simulation. Process Simulate, Delmia and NX-MCD are some important simulation tools.

The whole project can be tested with virtual commissioning at the end. Tools like WinMOD, RF-Suite and SIMIT can be named for this purpose.

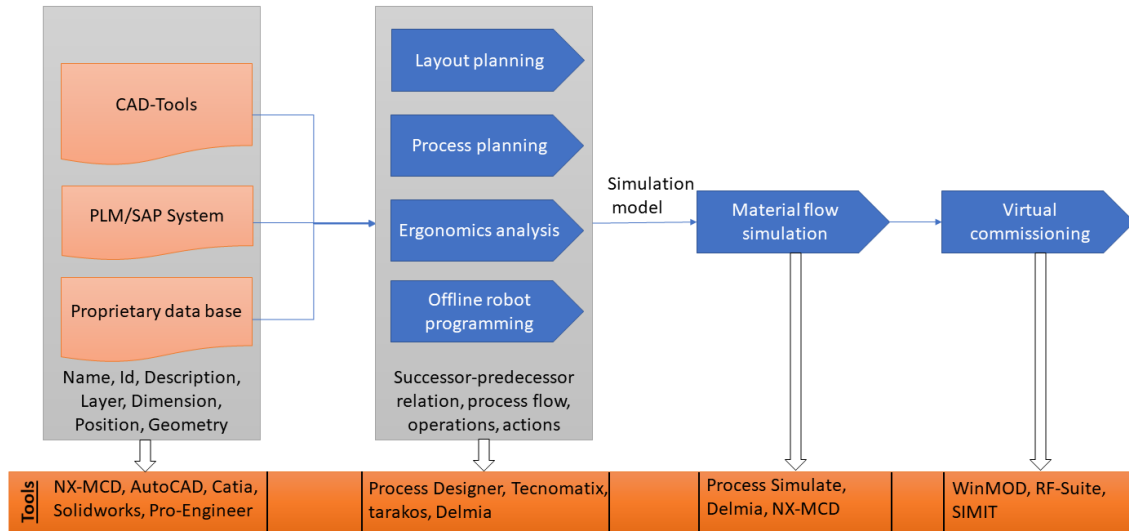


Figure 1 Tool chain for material handling system design (Variation 1)

The second variation of the material handling tool chain starts with the process planning (see Figure 2). The necessary process steps for manufacturing the product are defined without defining the resources concretely. Different business methods like Value Stream Planning can be used for production planning and control or the management of services [7]. The selection of resources to execute the processes takes place after process planning phase. The rest of the tool chain is like the variation 1 as explained above.

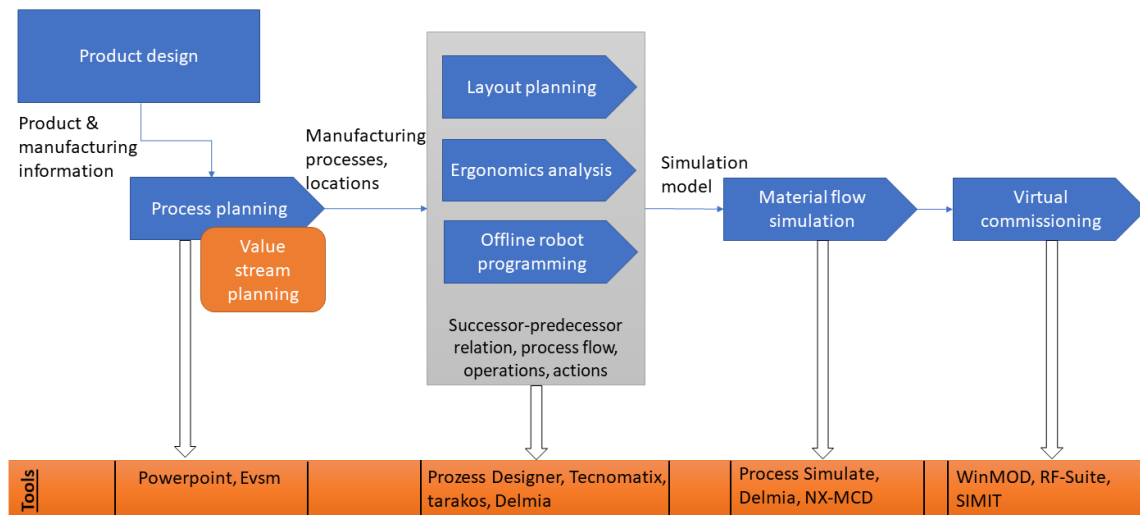


Figure 2 Tool chain for material handling system design (Variation 2)

2 Motivation for modelling material handling related information with AutomationML

As the product life cycles getting shorter every day, factory planning places increasing emphasis on the versatility of a production plant. The competitiveness of a company is also based on the ability to make the necessary adjustments economically and responsively.

The digital factory tools make it possible to avoid expensive planning errors. However, there are still many interruptions during the data exchange between the software tools in practice. These have been mostly bridged by renewed manual input of already known data. This represents a potential source of error. Another important point is that, an (at least partially) automated transfer of existing data could shorten the reaction times to changes in the planning.

The lack of integration of all necessary planning data represents a vast time and cost factor. The motivation for modelling material handling related information in AutomationML is to cover these needs of data-exchange from layout planning, process description, simulation and visualization of material handling systems.

Planners of material handling systems should consider aspects of layout design, material flow characteristics, construction characteristics and process description. As mentioned before, depending on the current level of detail and planning phase different data are necessary and separate tools are used. To give an example, for a first draft the information about a “conveyor” including attributes like length and width are enough. Later, behaviour description, like speed and accumulating places, should be added. The design engineer needs information about the carrier medium (rollers, belt conveyor, chain conveyor). For simulation the type of the carrier medium is not important, but the relation predecessor/successor is required.

During the requirements specification phase, different levels of details are available and must be documented accordingly. 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 description concerning the transportation means. Requirements specification may also include demands on characteristics of the product being handled. This ranges from general requirements defining i.e. the mass to be transported and the transportation time or speed, up to detailed requirements specifying carrying mediums to be used or specific conveying technologies like automated guided vehicles (AGV). Other requirements are necessary to adequately describe, and later simulate, a production system’s material handling resources and processes, which 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 on AutomationML RoleClassLibraries and InterfaceClassLibraries has been developed.

2.1 Basics

AutomationML, which is standardised as the IEC 62714, is a neutral, free, and XML-based data exchange format. It has been developed in order to support the data exchange between engineering tools in a heterogeneous engineering tool landscape.

Due to the different aspects of AutomationML, the IEC 62714 consists of different parts. The relevant parts for this document are shown in the following table:

Table 1 – Overview of AutomationML parts

Part	Title	Description
Part 1	Architecture and general requirements	This part specifies the general AutomationML architecture, the modelling of the engineering data, classes, instances, relations, references, hierarchies, basic AutomationML libraries and extended AutomationML concepts.
Part 2	Role class libraries	This part specifies additional AutomationML libraries.
Part 3	Geometry and kinematics	This part specifies the modelling of geometry and kinematics information.

Further parts may be added in the future in order to e.g. interconnect further data standards to AutomationML.

2.2 Scope

This application recommendation proposes a modelling method of material handling systems, processes and transport units by means of the engineering data format AutomationML. It will describe the recommended use of role and interface classes as well as the recommended structures to be considered within the instance hierarchy of an AutomationML project.

2.3 References

The following documents are referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

*Extensible Markup Language (XML) 1.0:2008, W3C Recommendation (available at <
https://www.w3.org/TR/2008/REC-xml-20081126/>)*

Whitepaper AutomationML Part 1 – AutomationML Architecture and general requirements, November 2018

Whitepaper AutomationML Part 2 –AutomationML Role class libraries, January 2017

Whitepaper AutomationML Part 3 –AutomationML Geometry and kinematics, March 2017

- [1] Gino Dini and Dieter Spath, "Material Flow," in *CIRP Encyclopedia of Production Engineering*.: Springer Berlin Heidelberg, 2014, p. 844.
- [2] eCI@ss e.V. , "ISO/TS29002-5. Industrial automation systems and integration - Exchange of characteristic data - Part 5: Identification scheme," Version 4.0, 2019.
- [3] Verein Deutscher Ingenieure (VDI), "VDI 2411: Begriffe und Erläuterungen im Förderwesen.," Berlin, 1970.
- [4] Verein deutscher Ingenieure (VDI), "VDI 2860: Handhabungsfunktionen, Handhabungseinrichtungen: Begriffe, Definitionen, Symbole," Düsseldorf, 1990.
- [5] International Electrotechnical Commission (IEC), "Enterprise-control system integration—Part 2: Objects and attributes for enterprise-control system integration," International Standard IEC 62264-2:2016 Edition 2.0., 2016.
- [6] Bernhard Wally, Christian Huemer, Alexandra Mazak, and Manuel Wimmer, "IEC 62264-2 for AutomationML," in *Proc. 5th AutomationML User Conference* , Gothenburg, 2018.
- [7] sixsigmablackbelt.de. (2019, Feb.) Wertstromanalyse – Wertstromdesign – value stream mapping. [Online]. <https://www.sixsigmablackbelt.de/wertstromanalyse-value-stream-mapping/>

3 Basic concept

The primary approach is to define reusable elements by categorizing and dissecting 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 production phase. An abstract function defining role can be assigned to a 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 carrying medium is implemented as a second role *ConveyorTechnology*, which ascertain if it is a belt, chain or band conveyor. An element, which performs the conveyor role, can contain a child element to define the conveyor technology. More examples of the concept are given in section 5. 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 modules. An AutomationML document can include only one or both modelling levels or a mixture of them, which can occur during the translation of different planning phases.

3.1 Module level (Level 1)

Figure 3 depicts a small example scene. There are three conveyors, one is the product source and the other two are sinks. In between is a turntable, which divides the product flow.

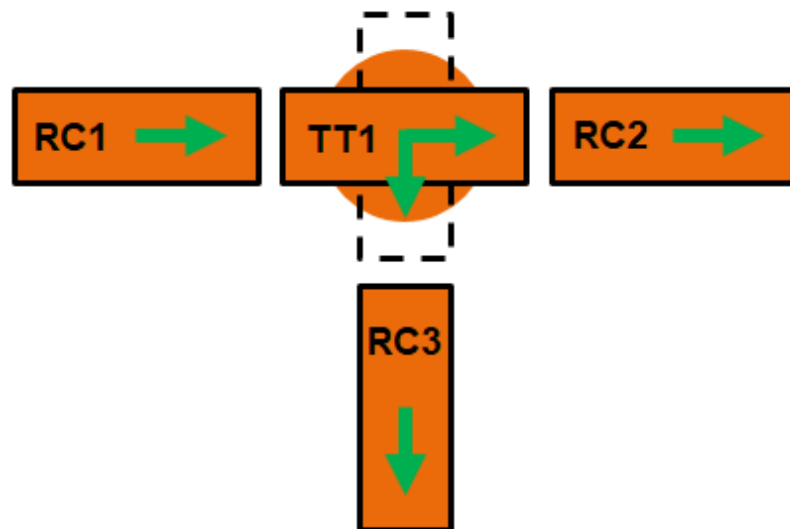


Figure 3 Example Scene with three conveyors (RC) and one turntable (TT)

Figure 4 depicts the module level. All conveyors are modelled as *InternalElements* and it is recommended but not necessary, that these resource descriptions have a direct or indirect reference to any role of the *MaterialHandlingResourceClassLib*. The possible transport directions are modelled with a simple 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.

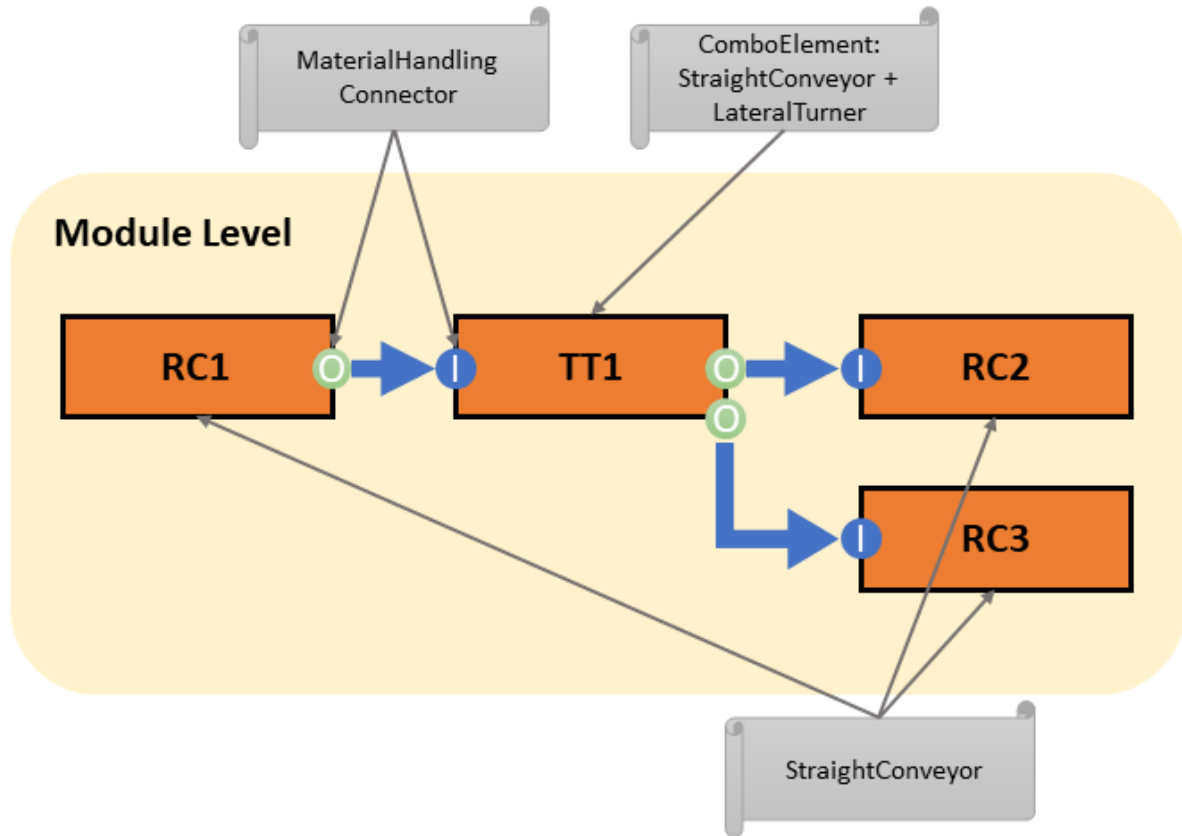


Figure 4 Modelling of module level (Level 1). O for output and I for input.

The relationship between two material handling elements in module level can be seen as an UML diagram in Figure 5. RC1 element implements the role of a *StraightConveyor*, which has a parent role class *MaterialHandlingConveyor*. TT1 element contains the conveying and turning functionalities, so it contains one element which fulfils the role of a *StraightConveyor* and another one for the role of a *LateralTurner*. This situation in TT1 is marked with the role *ComboElement* (see Combo Element). Both RC1 and TT1 contains an *ExternalInterface*, which implements the *MaterialHandlingConnector* interface class. The course of the material flow is defined through the *Direction* attributes in the external interfaces, which are connected with an *InternalLink*. The value of the *Direction* can be Input(I), Output(O) or both (InOut).

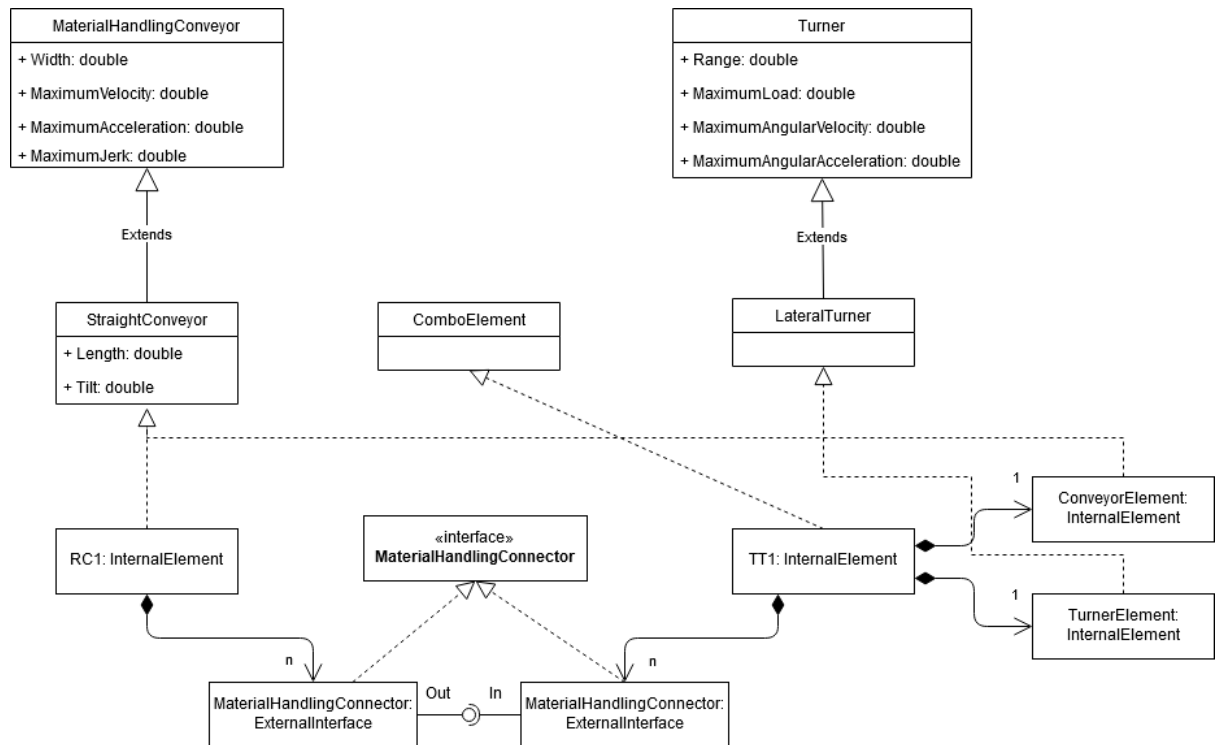


Figure 5 The relationship between two material handling elements in module level

3.2 Port level (Level 2)

The port level, depicted in *Figure 6*, offers more modelling possibilities. It adds connection point (port) descriptions to each material handling resource and offers a more detailed specification of the connections between the modules. In the example, the same course of the material flow as in module level example is defined, but in an elaborated way. The conveyor objects get an internal element child of the type *MaterialHandlingConnectionPoint*. The course of the material flow is established through the external interfaces in the connection points, not through the external interfaces directly inside the material handling component.

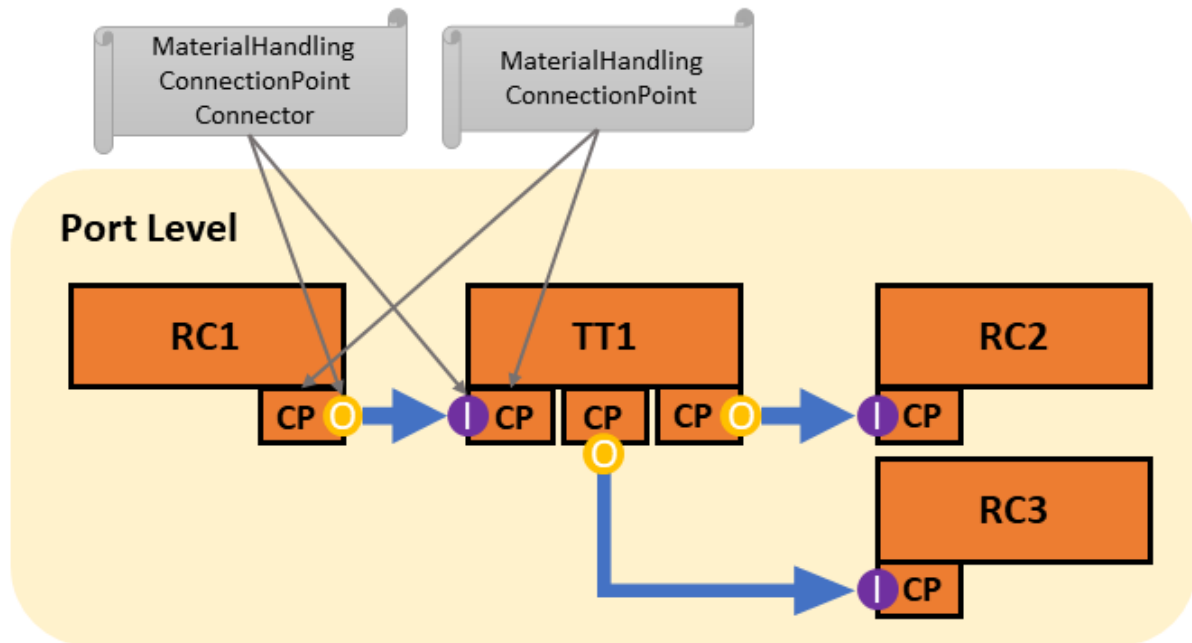


Figure 6 Modelling of port level (Level 2)

As seen in the UML diagram in Figure 7, the relationship between two material handling elements differs from the module level with *ConnectionPoint* elements. The connection point is a child element of the material handling resource component and fulfils the role of a *MaterialHandlingConnectionPoint*. It contains an *ExternalInterface* instance, which implements the *MaterialHandlingConnectionPointConnector* interface. The course of the material flow is again defined through the external interfaces, which are connected with an *InternalLink*, with the help of the *Direction* attribute; similar to the module level.

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 i.e. for handover areas, but this is not defined in this document. Other possible extensions may be links to logic information, communication, wiring, etc.

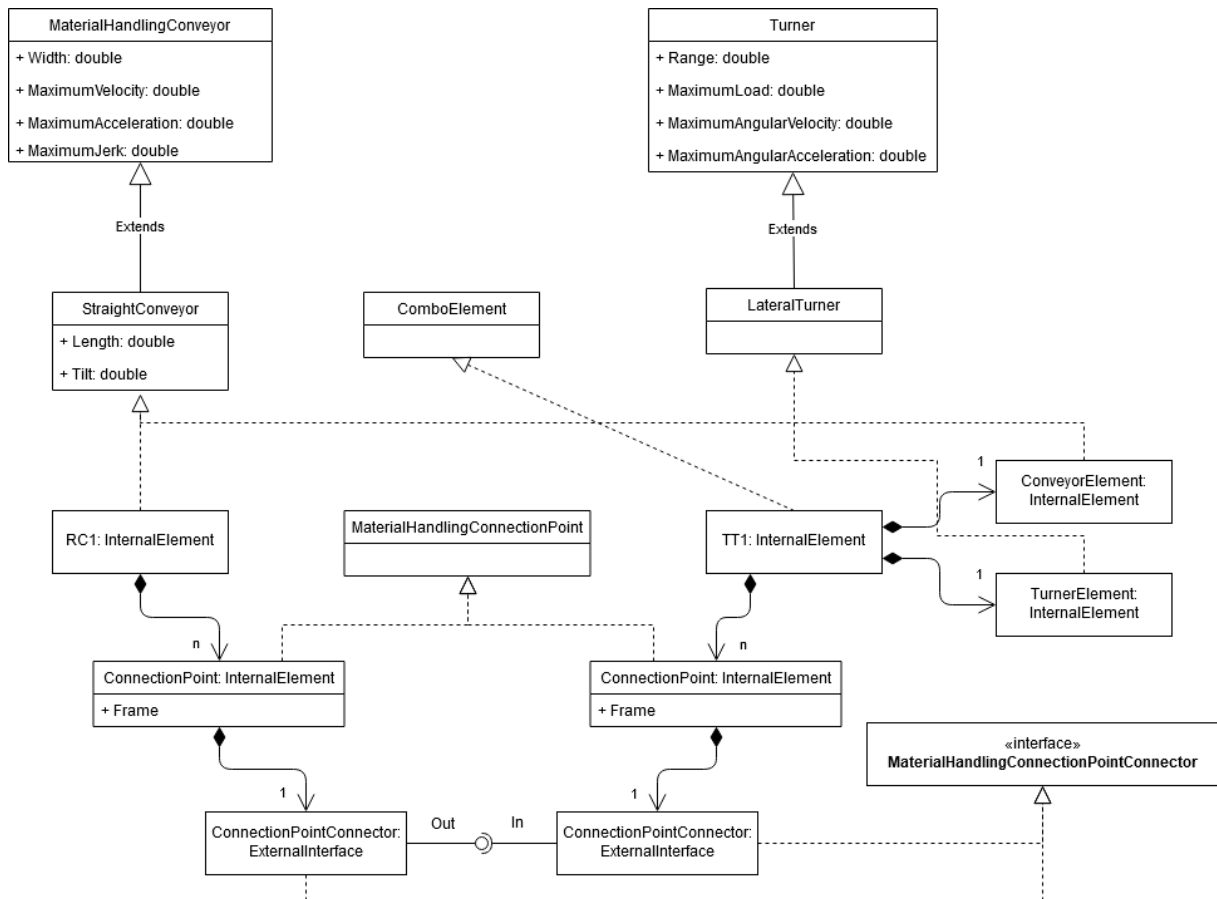


Figure 7 The relationship between two material handling elements in port level

3.3 Combo Element

If a component consists of two or more main child components, the role class *ComboElement* is used to flag this element as a combination of more than one elements. For example, a turntable which contains a rotational part with the supported role *Turner* and a transporting part with the supported role *MaterialHandlingConveyor*. Another example would be a linear portal with a lifting element with the supported role *VerticalPositioner* and a rail element with the supported role *HorizontalPositioner*. Thereby it is possible to define elements with complex functionalities by splitting them up to small functional blocks, using superposition principle. A turner example is shown in section 5.2.

4 Material handling class libraries

The current versions of the material handling libraries are:

- MaterialHandlingInterfaceClassLib V 1.5.0
- MaterialHandlingResourceClassLib V 1.6.0
- MaterialHandlingProcessClassLib V 1.4.0
- MaterialHandlingProductClassLib V 1.3.0

4.1 Material handling interface class library

This section contains detailed descriptions of interface classes.

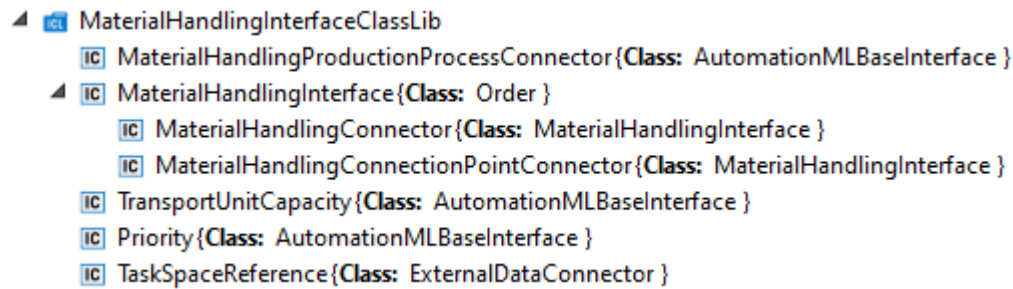


Figure 8 MaterialHandlingInterfaceClassLib

The *MaterialHandlingInterfaceClassLib* contains seven new interface class types.

The first interface in the library *MaterialHandlingProductionProcessConnector* is for the linking of processes in material handling and manufacturing technology. It is possible to define the transfer of material between material flow process and production process in the plant design with the help of this interface class.

The *MaterialHandlingInterface* is a base interface for the linking of components in the material handling. 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.

TransportUnitCapacity interface can be used to define the capacity of a load carrier for a specific type of transport unit.

Priority interface defines the priority of a transport unit during the accumulation. If the accumulation process does not have a pre-defined priority rule, it can be defined via this interface with the attribute *PriorityDegree*.

TaskSpaceReference interface defines a reference to an external MathML equation, which is used to specify a complex task space for a free positioner (see Positioners). The MathML equation can be defined in a discrete file or embedded in a COLLADA file with a corresponding node and unique id. This element should be used only if the task space cannot be defined with the *TaskSpace* attribute of the *FreePositioner* element, which means it is not a cube or a rectangular prism.

Class name	MaterialHandlingProductionProcessConnector
Description	Defines the connection between material handling level and production process level
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
Path for element reference	MaterialHandlingInterfaceClassLib/MaterialHandlingProductionProcessConnector

Class name	MaterialHandlingInterface
Description	General interface to define the connection of material flow elements
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Order
Path for element reference	MaterialHandlingInterfaceClassLib/MaterialHandlingInterface

Class name	MaterialHandlingConnector
Description	Connects the material handling components directly with each other (Module level)
Parent class	MaterialHandlingInterfaceClassLib/MaterialHandlingInterface
Path for element reference	MaterialHandlingInterfaceClassLib/MaterialHandlingInterface/ MaterialHandlingConnector

Class name	MaterialHandlingConnectionPointConnector
Description	Connects the material handling components through connection points (Port level)
Parent class	MaterialHandlingInterfaceClassLib/MaterialHandlingInterface
Path for element reference	MaterialHandlingInterfaceClassLib/MaterialHandlingInterface/ MaterialHandlingConnectionPointConnector

Class name	TransportUnitCapacity
Description	Defines the capacity of a load carrier for a corresponding transport unit.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
Path for element reference	MaterialHandlingInterfaceClassLib/TransportUnitCapacity

Class name	Priority
Description	Defines the priority of a transport unit during accumulation.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface
Path for element reference	MaterialHandlingInterfaceClassLib/Priority
Attributes	<div> <div>"PriorityDegree" (AttributeDataType="xs:PositiveInteger")</div> <div>Defines the degree of priority compared to linked elements</div> </div>

Class name	TaskSpaceReference
Description	Defines a reference to an external MathML equation, which is used to specify a complex task space for a free positioner.
Parent class	AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ ExternalDataConnector
Path for element reference	MaterialHandlingInterfaceClassLib/TaskSpaceReference

4.2 Material handling resource role classes

This section contains detailed descriptions of resource role classes, which are used to define material handling elements.

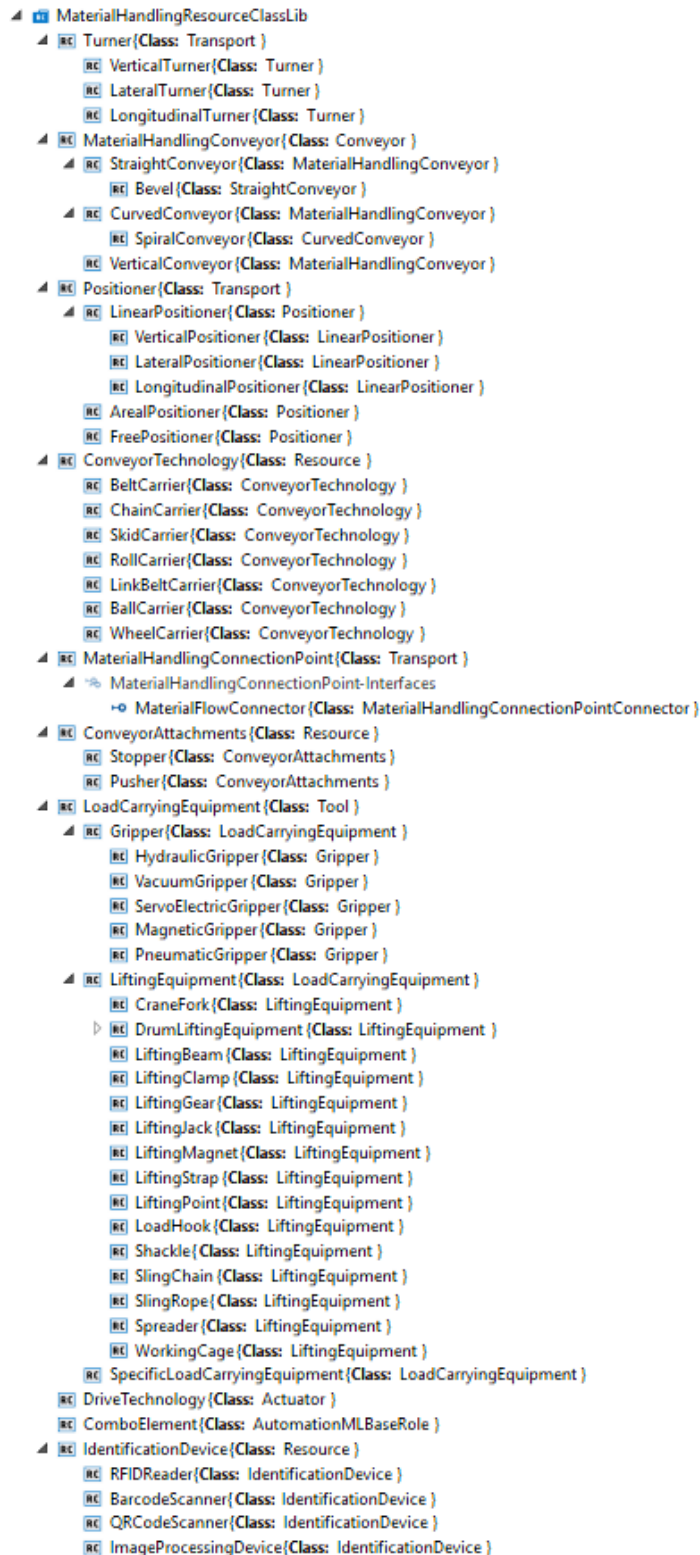


Figure 9 MaterialHandlingResourceClassLib

4.2.1.1 Turners

Class name	Turner	
Description	A general functional role for the elements, which are turning materials without any position change	
Parent class	AutomationMLDMIRoleClassLib/DiscManufacturingEquipment/Transport	
Path for element reference	MaterialHandlingResourceClassLib/Turner	
Attributes	MaximumLoad (DataType="xs:double", Unit="kg")	Maximum sustainable load
	MaximumAngularAcceleration (DataType="xs:double", Unit="rad/s^2")	Angular acceleration of the rotation
	MaximumAngularVelocity (DataType="xs:double", Unit="rad/s")	Angular velocity of the rotation
	Range (DataType="xs:double", Unit="deg")	Range of the turner

Class name	VerticalTurner
Description	An element, which is turning materials on vertical axis
Parent class	MaterialHandlingResourceClassLib/Turner
Path for element reference	MaterialHandlingResourceClassLib/Turner/VerticalTurner

Class name	LateralTurner
Description	An element, which is turning materials on lateral axis
Parent class	MaterialHandlingResourceClassLib/Turner
Path for element reference	MaterialHandlingResourceClassLib/Turner/LateralTurner

Class name	LongitudinalTurner
Description	An element, which is turning materials on longitudinal axis
Parent class	MaterialHandlingResourceClassLib/Turner
Path for element reference	MaterialHandlingResourceClassLib/Turner/LongitudinalTurner

4.2.2 Conveyors

Class name	MaterialHandlingConveyor	
Description	A functional role for the elements, which move materials from one location to another but does not change its own place	
Parent class	AutomationMLExtendedRoleClassLib/Conveyor	
Path for element reference	MaterialHandlingResourceClassLib/MaterialHandlingConveyor	
Attributes	Width (DataType="xs:double", Unit="m")	Usable width of the conveyor
	MaximumVelocity (DataType="xs:double", Unit="m/s")	Maximum velocity of the conveyor
	MaximumAcceleration (DataType="xs:double", Unit="m/s^2")	Maximum acceleration of the conveyor
	MaximumJerk (DataType="xs:double", Unit="m/s^3")	Maximum jerk of the conveyor (Optional)

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

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

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

Class name	SpiralConveyor	
Description	Moves materials from one location to another on a spiral	
Parent class	MaterialHandlingResourceClassLib/MaterialHandlingConveyor/CurvedConveyor	
Path for element reference	MaterialHandlingResourceClassLib/MaterialHandlingConveyor/CurvedConveyor/SpiralConveyor	
Attributes	Height (DataType="xs:double", Unit="m")	Height of the spiral

Class name	VerticalConveyor	
Description	Moves materials from one location to another on a line flapped around the connection point	
Parent class	MaterialHandlingResourceClassLib/MaterialHandlingConveyor	
Path for element reference	MaterialHandlingResourceClassLib/MaterialHandlingConveyor/VerticalConveyor	
Attributes	SwingVelocity (DataType="xs:double", Unit="m/s")	Velocity of the swinging from one tilt angle to another
	Length (DataType="xs:double", Unit="m")	Length of the conveyor
	TiltList (Unit="deg", RefSemantic="OrderedListType")	List of tilt angles

4.2.3 Positioners

Class name	Positioner	
Description	A functional role for the elements, which carry materials from one location to another. (Positioner has also location change.)	
Parent class	AutomationMLDMIRoleClassLib/DiscManufacturingEquipment/Transport	
Path for element reference	MaterialHandlingResourceClassLib/Positioner	
Attributes	MaximumAcceleration (DataType="xs:double", Unit="m/s^2")	Maximum acceleration of the linear movement
	MaximumVelocity (DataType="xs:double", Unit="m/s")	Maximum velocity of the linear movement
	MaximumLoad (DataType="xs:double", Unit="kg")	Maximum sustainable load

Class name	LinearPositioner	
Description	A positioner, which carries materials on a linear axis	
Parent class	MaterialHandlingResourceClassLib/Positioner	
Path for element reference	MaterialHandlingResourceClassLib/Positioner/LinearPositioner	
Attributes	Range Unit="m") (DataType="xs:double",	Range of the positioner

Class name	VerticalPositioner	
Description	A positioner, which carries materials on linear vertical axis	
Parent class	MaterialHandlingResourceClassLib/Positioner/LinearPositioner	
Path for element reference	MaterialHandlingResourceClassLib/Positioner/LinearPositioner/VerticalPositioner	

Class name	LateralPositioner	
Description	A positioner, which carries materials on linear lateral axis	
Parent class	MaterialHandlingResourceClassLib/Positioner/LinearPositioner	
Path for element reference	MaterialHandlingResourceClassLib/Positioner/LinearPositioner/LateralPositioner	

Class name	LongitudinalPositioner
Description	A positioner, which carries materials on linear longitudinal axis
Parent class	MaterialHandlingResourceClassLib/Positioner/LinearPositioner
Path for element reference	MaterialHandlingResourceClassLib/Positioner/LinearPositioner/ LongitudinalPositioner

Class name	ArealPositioner	
Description	A positioner, which carries materials in a defined area	
Parent class	MaterialHandlingResourceClassLib/Positioner	
Path for element reference	MaterialHandlingResourceClassLib/Positioner/ArealPositioner	
Attributes	Range (DataType="xs:double", Unit="m")	Range of the positioner
	Direction (DataType = "", Unit="m[]", RefSemantic="OrderedListType")	Direction of the movement as vector

Class name	FreePositioner	
Description	A positioner, which can carry materials in 3D space	
Parent class	MaterialHandlingResourceClassLib/Positioner	
Path for element reference	MaterialHandlingResourceClassLib/Positioner/FreePositioner	
Attributes	TaskRoute	An ordered list of points which define the route of the positioner (Optional)
	TaskRoute.Point	A single point on the route of the positioner
	TaskRoute.Point.x (DataType="xs:double", Unit="m")	X coordinate of the point
	TaskRoute.Point.y (DataType="xs:double", Unit="m")	Y coordinate of the point
	TaskRoute.Point.z (DataType="xs:double", Unit="m")	Z coordinate of the point
	TaskSpace	The task space region of the positioner (Optional)
	TaskSpace.x_min (DataType="xs:double", Unit="m")	Minimum x value
	TaskSpace.x_max (DataType="xs:double", Unit="m")	Maximum x value
	TaskSpace.y_min (DataType="xs:double", Unit="m")	Minimum y value
	TaskSpace.y_max (DataType="xs:double", Unit="m")	Maximum y value
	TaskSpace.z_min (DataType="xs:double", Unit="m")	Minimum z value
	TaskSpace.z_max (DataType="xs:double", Unit="m")	Maximum z value

4.2.4 Carrier mediums

Class name	ConveyorTechnology
Description	Defines the carrier element of a conveyor like rolls, belts, chains, etc.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Resource
Path for element reference	MaterialHandlingResourceClassLib/ConveyorTechnology

Class name	BeltCarrier
Description	Belt type carrier element
Parent class	MaterialHandlingResourceClassLib/ConveyorTechnology
Path for element reference	MaterialHandlingResourceClassLib/ConveyorTechnology/BeltCarrier

Class name	ChainCarrier
Description	Chain type carrier element
Parent class	MaterialHandlingResourceClassLib/ConveyorTechnology
Path for element reference	MaterialHandlingResourceClassLib/ConveyorTechnology/ChainCarrier

Class name	SkidCarrier
Description	Skid type carrier element
Parent class	MaterialHandlingResourceClassLib/ConveyorTechnology
Path for element reference	MaterialHandlingResourceClassLib/ConveyorTechnology/SkidCarrier

Class name	RollCarrier
Description	Roll type carrier element
Parent class	MaterialHandlingResourceClassLib/ConveyorTechnology
Path for element reference	MaterialHandlingResourceClassLib/ConveyorTechnology/RollCarrier

Class name	LinkBeltCarrier
Description	Link belt type carrier element
Parent class	MaterialHandlingResourceClassLib/ConveyorTechnology
Path for element reference	MaterialHandlingResourceClassLib/ConveyorTechnology/LinkBeltCarrier

Class name	BallCarrier
Description	Ball type carrier element
Parent class	MaterialHandlingResourceClassLib/ConveyorTechnology
Path for element reference	MaterialHandlingResourceClassLib/ConveyorTechnology/BallCarrier

Class name	WheelCarrier
Description	Wheel type carrier element
Parent class	MaterialHandlingResourceClassLib/ConveyorTechnology
Path for element reference	MaterialHandlingResourceClassLib/ConveyorTechnology/WheelCarrier

4.2.5 Connection point

Class name	MaterialHandlingConnectionPoint	
Description	Represent the point where the material transfer occurs	
Parent class	AutomationMLDMIRoleClassLib/DiscManufacturingEquipment/Transport	
Path for element reference	MaterialHandlingResourceClassLib/MaterialHandlingConnectionPoint	
Interfaces	"MaterialFlowConnector" (ClassReference="MaterialHandlingInterfaceClassLib/ MaterialHandlingInterface/ MaterialHandlingConnectionPointConnector)	Used to link connection points with each other.
Attributes	Frame	Standard attribute defined in AutomationML Whitepaper 3 (Optional)

4.2.6 Conveyor attachments

Class name	ConveyorAttachments	
Description	A general role for conveyor attachments	
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Resource	
Path for element reference	MaterialHandlingResourceClassLib/ConveyorAttachments	

Class name	Stopper	
Description	Decelerates or stops materials safely	
Parent class	MaterialHandlingResourceClassLib/ConveyorAttachments	
Path for element reference	MaterialHandlingResourceClassLib/ConveyorAttachments/Stopper	

Class name	Pusher
Description	Diverts materials from one conveyor line to another
Parent class	MaterialHandlingResourceClassLib/ConveyorAttachments
Path for element reference	MaterialHandlingResourceClassLib/ConveyorAttachments/Pusher

4.2.7 Load carrying equipments

Class name	LoadCarryingEquipment	
Description	Represents an equipment that can be used to carry or lift loads.	
Parent class	AutomationMLDMIRoleClassLib/DiscManufacturingEquipment/Tool	
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment	
Attributes	MaximumLoad (DataType="xs:double", Unit="kg")	Maximum load that the equipment can carry.

Class name	Gripper	
Description	Represents the end effector of a manipulator which is used to hold an object.	
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment	
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/Gripper	
Attributes	GrippingForce (DataType="xs:double", Unit="N")	Gripping force of the handling element

Class name	HydraulicGripper
Description	Represents a hydraulic driven gripper
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/Gripper
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ Gripper/HydraulicGripper

Class name	VacuumGripper
Description	Represents a vacuum based gripping mechanism
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/Gripper
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ Gripper/VacuumGripper

Class name	ServoElectricGripper
Description	Represents a servo electric driven gripper
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/Gripper
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ Gripper/ServoElectricGripper

Class name	MagneticGripper
Description	Represents a magnetism based gripping mechanism
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/Gripper
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ Gripper/MagneticGripper

Class name	PneumaticGripper
Description	Represents a pneumatic driven gripper
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/Gripper
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ Gripper/PneumaticGripper

Class name	LiftingEquipment
Description	Any equipment that can be used to lift loads or help load suspension
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment

Class name	CraneFork
Description	Represents a crane fork
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/LiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/CraneFork

Class name	LiftingBeam
Description	Represents a lifting beam
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/LiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/LiftingBeam

Class name	LiftingClamp
Description	Represents a lifting clamp
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/LiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/LiftingClamp

Class name	LiftingGear
Description	Represents a lifting gear
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/LiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/LiftingGear

Class name	LiftingJack
Description	Represents a lifting jack
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/LiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/LiftingJack

Class name	LiftingMagnet
Description	Represents a lifting magnet
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/LiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/LiftingMagnet

Class name	LiftingStrap
Description	Represents a lifting strap
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/LiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/LiftingStrap

Class name	LiftingPoint
Description	Represents a lifting point
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/LiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/LiftingPoint

Class name	LoadHook
Description	Represents a load hook
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/LiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/LoadHook

Class name	Shackle
Description	Represents a shackle
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/LiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/Shackle

Class name	SlingChain
Description	Represents a sling chain
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/LiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/SlingChain

Class name	SlingRope
Description	Represents a sling rope
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/LiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/SlingRope

Class name	Spreader
Description	Represents a spreader
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/LiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/Spreader

Class name	WorkingCage
Description	Represents a working cage
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/LiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/WorkingCage

Class name	DrumLiftingEquipment
Description	Represents an equipment to lift or carry drums
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/LiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/DrumLiftingEquipment

Class name	DrumPicker
Description	Represents a drum picker
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/DrumLiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/DrumLiftingEquipment/DrumPicker

Class name	DrumDumper
Description	Represents a drum dumper
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/DrumLiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/DrumLiftingEquipment/DrumDumper

Class name	DrumTongs
Description	Represents drum tongs
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/DrumLiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/DrumLiftingEquipment/DrumTongs

Class name	DrumLifter
Description	Represents a drum lifter
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/DrumLiftingEquipment
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ LiftingEquipment/DrumLiftingEquipment/DrumLifter

Class name	SpecificLoadCarryingEquipment
Description	Represents a specific load carrying equipment
Parent class	MaterialHandlingResourceClassLib/LoadCarryingEquipment/
Path for element reference	MaterialHandlingResourceClassLib/LoadCarryingEquipment/ SpecificLoadCarryingEquipment

4.2.8 Drive technology

Class name	DriveTechnology
Description	A general role for the driving technology in material handling. Can be used to define a requirement for the further aspects of engineering planning.
Parent class	AutomationMLCSRoleClassLib/ControlEquipment/Actuator
Path for element reference	MaterialHandlingResourceClassLib/DriveTechnology
Attributes	MomentOfInertia (DataType="xs:double", Unit="kgm^2") Moment of inertia for the load

4.2.9 Combo element

Class name	ComboElement
Description	A marker role to show that the component consists of subcomponents
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole
Path for element reference	MaterialHandlingResourceClassLib/ComboElement

4.2.10 Identification devices

Class name	IdentificationDevice
Description	Devices that identify and track objects or tags attached to objects.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Resource
Path for element reference	MaterialHandlingResourceClassLib/IdentificationDevice

Class name	RFIDReader
Description	A device used to gather information from an RFID tag.
Parent class	MaterialHandlingResourceClassLib/IdentificationDevice
Path for element reference	MaterialHandlingResourceClassLib/IdentificationDevice/RFIDReader

Class name	BarcodeScanner
Description	A device that decodes and physically captures information contained in barcodes.
Parent class	MaterialHandlingResourceClassLib/IdentificationDevice
Path for element reference	MaterialHandlingResourceClassLib/IdentificationDevice/BarcodeScanner

Class name	QRCodeScanner
Description	A device that decodes and physically captures information contained in QR-codes.
Parent class	MaterialHandlingResourceClassLib/IdentificationDevice
Path for element reference	MaterialHandlingResourceClassLib/IdentificationDevice/QRCodeScanner

Class name	ImageProcessingDevice
Description	Devices that use image processing algorithms to identify and tracks objects.
Parent class	MaterialHandlingResourceClassLib/IdentificationDevice
Path for element reference	MaterialHandlingResourceClassLib/IdentificationDevice/ImageProcessingDevice

4.3 Material handling process role classes

Not only the resources but also processes are essential in the material handling technology. According to the VDI 2411 [3] guideline, following function types are existing in a material handling system: machining, handling, conveying, storing and placing. The guideline VDI 2860 [4] defines only conveying, storage and handling as the functional elements in the material handling (see *Figure 10*). 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. Nevertheless, extra roles are also needed to describe some existing activities in the workflow, e.g. roles for accumulation behavior or roles such as source, sink, loading process, unloading process, etc.

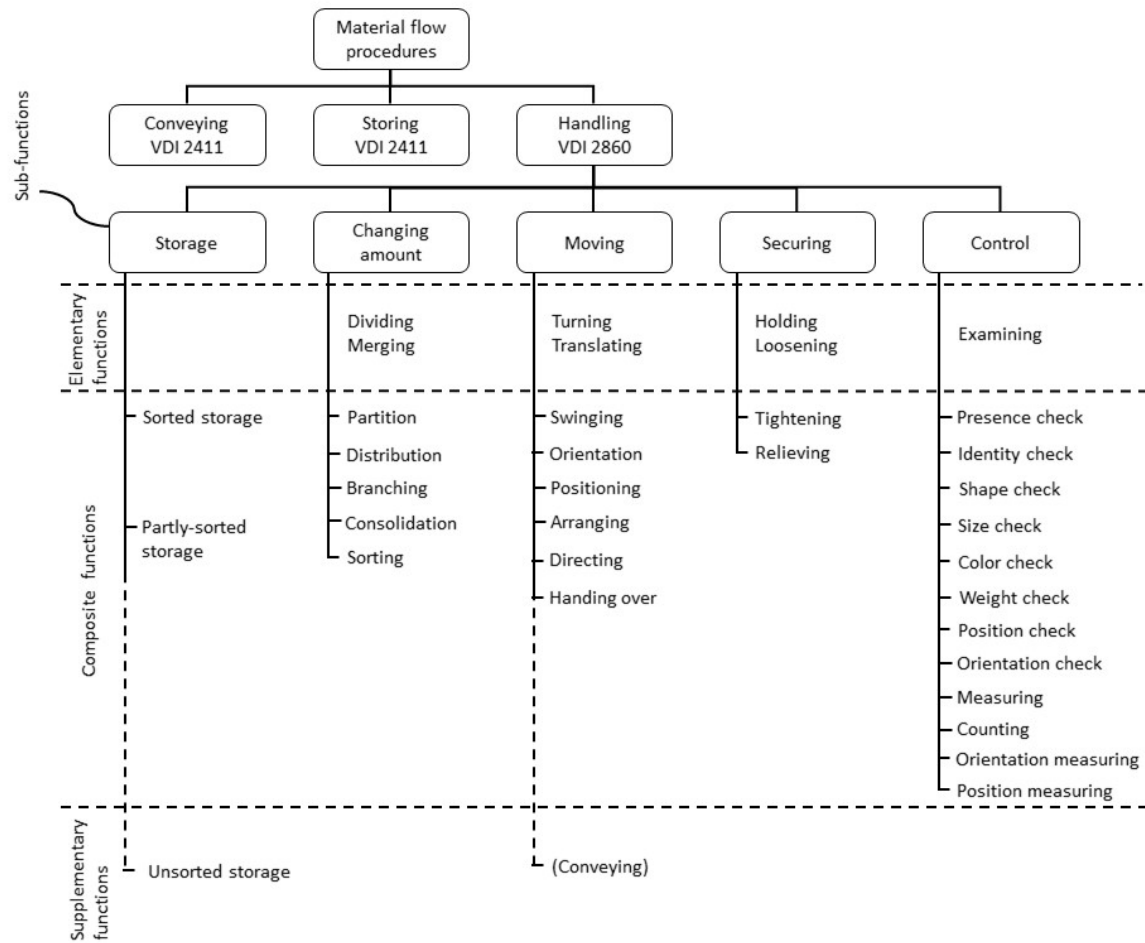


Figure 10 Sub-functions of the material handling processes and their formation based on [4]

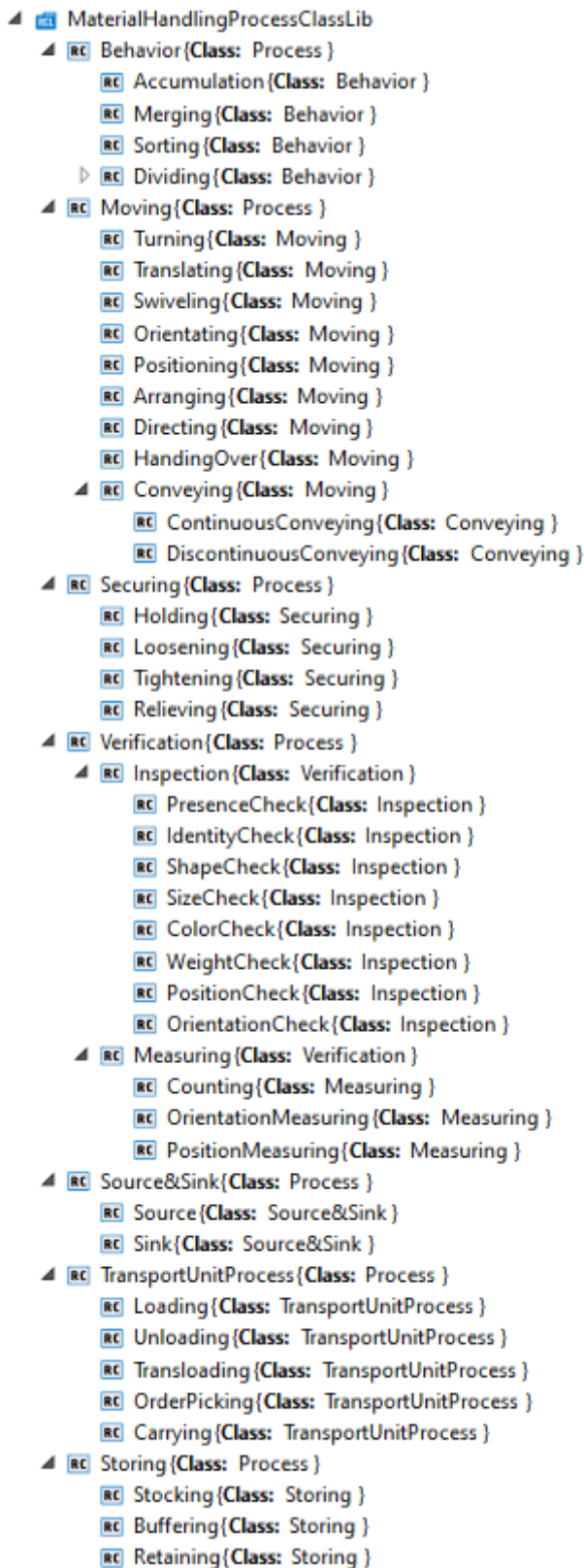


Figure 11 MaterialHandlingProcessClassLib

4.3.1 Handling

Class name	Behavior
Description	Defines handling processes for the accumulation, merging, sorting, dividing, etc.
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process
Path for element reference	MaterialHandlingProcessClassLib/Behavior

Class name	Accumulation
Description	Defines accumulation behavior
Parent class	MaterialHandlingProcessClassLib/Behavior
Path for element reference	MaterialHandlingProcessClassLib/Behavior/Accumulation

Class name	Merging	
Description	Defines merging of multiple material handling objects	
Parent class	MaterialHandlingProcessClassLib/Behavior	
Path for element reference	MaterialHandlingProcessClassLib/Behavior/Merging	
Attributes	PriorityRule (DataType="xs:string")	Defines the priority rule as FIFO, LIFO, Sequence or PriorityByProduct

Class name	Sorting
Description	Defines the sorting process in material handling
Parent class	MaterialHandlingProcessClassLib/Behavior
Path for element reference	MaterialHandlingProcessClassLib/Behavior/Sorting

Class name	Dividing
Description	A general process role to define how to create subsets from an amount
Parent class	MaterialHandlingProcessClassLib/Behavior
Path for element reference	MaterialHandlingProcessClassLib/Behavior/Dividing

Class name	Partition
Description	Defines a process to form subsets with defined size
Parent class	MaterialHandlingProcessClassLib/Behavior/Dividing
Path for element reference	MaterialHandlingProcessClassLib/Behavior/Dividing/Partition

Class name	Distribution
Description	Defines a process to form subsets with defined size and transfer them to a defined place
Parent class	MaterialHandlingProcessClassLib/Behavior/Dividing
Path for element reference	MaterialHandlingProcessClassLib/Behavior/Dividing/Distribution

Class name	Branching
Description	Defines a process to split a material flow to smaller flows
Parent class	MaterialHandlingProcessClassLib/Behavior/Dividing
Path for element reference	MaterialHandlingProcessClassLib/Behavior/Dividing/Branching

4.3.2 Moving

Class name	Moving	
Description	Defines a change in the spatial arrangement of a body	
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process	
Path for element reference	MaterialHandlingProcessClassLib/Moving	
Attributes	RotationOfTransportUnit (DataType="xs:string")	Defines the rotation of a transport unit in the process as Lengthwise, Crosswise or Random
	LateralPosition (DataType="xs:string")	Defines the lateral position of a transport unit relative to the transport direction as LeftHand, RightHand, Centered or Random

Class name	Turning
Description	Changing the orientation of the body without changing the position
Parent class	MaterialHandlingProcessClassLib/Moving
Path for element reference	MaterialHandlingProcessClassLib/Moving/Turning

Class name	Translating
Description	Moving a body in a linear direction without changing the orientation
Parent class	MaterialHandlingProcessClassLib/Moving
Path for element reference	MaterialHandlingProcessClassLib/Moving/Translating

Class name	Swiveling
Description	Rotation of a body over a centre of an axis, which is not within the body, resulting an orientation and position change
Parent class	MaterialHandlingProcessClassLib/Moving
Path for element reference	MaterialHandlingProcessClassLib/Moving/Swiveling

Class name	Orientating
Description	Change of the orientation from an undefined to a defined one. Position change of the body is disregarded.
Parent class	MaterialHandlingProcessClassLib/Moving
Path for element reference	MaterialHandlingProcessClassLib/Moving/Orientating

Class name	Positioning
Description	Moving a body to a defined position. Orientation of the body is disregarded.
Parent class	MaterialHandlingProcessClassLib/Moving
Path for element reference	MaterialHandlingProcessClassLib/Moving/Positioning

Class name	Arranging
Description	Moving a body from an undefined place to a defined orientation and position providing arrangement.
Parent class	MaterialHandlingProcessClassLib/Moving
Path for element reference	MaterialHandlingProcessClassLib/Moving/Arranging

Class name	Directing
Description	Moving a body on a defined path from a defined place to another defined place
Parent class	MaterialHandlingProcessClassLib/Moving
Path for element reference	MaterialHandlingProcessClassLib/Moving/Directing

Class name	HandingOver
Description	Moving a body on an undefined path from a defined place to another defined place
Parent class	MaterialHandlingProcessClassLib/Moving
Path for element reference	MaterialHandlingProcessClassLib/Moving/HandingOver

Class name	Conveying
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	MaterialHandlingProcessClassLib/Moving
Path for element reference	MaterialHandlingProcessClassLib/Moving/Conveying

Class name	ContinuousConveying
Description	Conveying process that create a continuous conveyance flow.
Parent class	MaterialHandlingProcessClassLib/Moving/Conveying
Path for element reference	MaterialHandlingProcessClassLib/Moving/Conveying/ContinuousConveying

Class name	DiscontinuousConveying
Description	Conveying process to move a transport unit in individual working cycles between take-up and delivery points.
Parent class	MaterialHandlingProcessClassLib/Moving/Conveying
Path for element reference	MaterialHandlingProcessClassLib/Moving/Conveying/DiscontinuousConveying

4.3.3 Securing

Class name	Securing
Description	A general role to define securing a defined status
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process
Path for element reference	MaterialHandlingProcessClassLib/Securing

Class name	Holding
Description	Securing the orientation and position of a body
Parent class	MaterialHandlingProcessClassLib/Securing
Path for element reference	MaterialHandlingProcessClassLib/Securing/Holding

Class name	Loosening
Description	Loosening the orientation and position of a body, reversal of holding.
Parent class	MaterialHandlingProcessClassLib/Securing
Path for element reference	MaterialHandlingProcessClassLib/Securing/Loosening

Class name	Tightening
Description	Securing the orientation and position of a body with applying force
Parent class	MaterialHandlingProcessClassLib/Securing
Path for element reference	MaterialHandlingProcessClassLib/Securing/Tightening

Class name	Relieving
Description	Relieving the force, which is blocking the orientational and positional change, reversal of tightening.
Parent class	MaterialHandlingProcessClassLib/Securing
Path for element reference	MaterialHandlingProcessClassLib/Securing/Relieving

Class name	Verification
Description	Defines a general role for inspection and measuring of properties and status
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process
Path for element reference	MaterialHandlingProcessClassLib/Verification

4.3.4 Inspection

Class name	Inspection
Description	Inspection of properties and status. Admission of information, comparison with target properties, status or decisions.
Parent class	MaterialHandlingProcessClassLib/Verification
Path for element reference	MaterialHandlingProcessClassLib/Verification/Inspection

Class name	PresenceCheck
Description	Determining if a body is present in a defined place
Parent class	MaterialHandlingProcessClassLib/Verification/Inspection
Path for element reference	MaterialHandlingProcessClassLib/Verification/Inspection/PresenceCheck

Class name	IdentityCheck
Description	Determining if a body satisfies the defined properties
Parent class	MaterialHandlingProcessClassLib/Verification/Inspection
Path for element reference	MaterialHandlingProcessClassLib/Verification/Inspection/IdentityCheck

Class name	ShapeCheck
Description	Determining if a body has the defined shape.
Parent class	MaterialHandlingProcessClassLib/Verification/Inspection
Path for element reference	MaterialHandlingProcessClassLib/Verification/Inspection/ShapeCheck

Class name	SizeCheck
Description	Determining if a body has the defined sizes/measurements.
Parent class	MaterialHandlingProcessClassLib/Verification/Inspection
Path for element reference	MaterialHandlingProcessClassLib/Verification/Inspection/SizeCheck

Class name	ColorCheck
Description	Determining if a body or body areas have the defined colors
Parent class	MaterialHandlingProcessClassLib/Verification/Inspection
Path for element reference	MaterialHandlingProcessClassLib/Verification/Inspection/ColorCheck

Class name	WeightCheck
Description	Determining if a body has the defined weight
Parent class	MaterialHandlingProcessClassLib/Verification/Inspection
Path for element reference	MaterialHandlingProcessClassLib/Verification/Inspection/WeightCheck

Class name	PositionCheck
Description	Determining if a body has the defined position
Parent class	MaterialHandlingProcessClassLib/Verification/Inspection
Path for element reference	MaterialHandlingProcessClassLib/Verification/Inspection/PositionCheck

Class name	OrientationCheck
Description	Determining if a body has the defined orientation
Parent class	MaterialHandlingProcessClassLib/Verification/Inspection
Path for element reference	MaterialHandlingProcessClassLib/Verification/Inspection/OrientationCheck

Class name	Measuring
Description	Measuring the values in accordance to reference values
Parent class	MaterialHandlingProcessClassLib/Verification
Path for element reference	MaterialHandlingProcessClassLib/Verification/Measuring

Class name	Counting
Description	Determining the amount of bodies
Parent class	MaterialHandlingProcessClassLib/Verification/Measuring
Path for element reference	MaterialHandlingProcessClassLib/Verification/Measuring/Counting

Class name	OrientationMeasuring
Description	Determining the orientation of a body in accordance to a reference coordinate system
Parent class	MaterialHandlingProcessClassLib/Verification/Measuring
Path for element reference	MaterialHandlingProcessClassLib/Verification/Measuring/OrientationMeasuring

Class name	PositionMeasuring
Description	Determining the position of a body in accordance to a reference coordinate system
Parent class	MaterialHandlingProcessClassLib/Verification/Measuring
Path for element reference	MaterialHandlingProcessClassLib/Verification/Measuring/PositionMeasuring

4.3.5 Source and sink

Class name	Source&Sink
Description	General role to define source and sink processes
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process
Path for element reference	MaterialHandlingProcessClassLib/Source&Sink

Class name	Source	
Description	Defines the source of a material flow	
Parent class	MaterialHandlingProcessClassLib/Source&Sink	
Path for element reference	MaterialHandlingProcessClassLib/Source&Sink/Source	
Attributes	DelayMode (DataType="xs:string")	Specified or determined randomly within a selectable time range
	CycleTime (DataType="xs:PositiveInteger", Unit="s")	Cycle time
	DelayTime (DataType="xs:PositiveInteger", Unit="s")	Delay time
	OutputMode (DataType="xs:string")	Creating transport units randomly or in a synchronized manner
	RotationOfTransportUnit (DataType="xs:string")	Defines the rotation of a transport unit in the process as Lengthwise, Crosswise or Random
	LateralPosition (DataType="xs:string")	Defines the lateral position of a transport unit relative to the transport direction as LeftHand, RightHand, Centered or Random

Class name	Sink	
Description	Defines the sink of a material flow	
Parent class	MaterialHandlingProcessClassLib/Source&Sink	
Path for element reference	MaterialHandlingProcessClassLib/Source&Sink/Sink	
Attributes	InputMode (DataType="xs:string", Unit="")	Receiving transport units randomly or in a synchronized manner
	DelayMode (DataType="xs:string", Unit="")	Specified or determined randomly within a selectable time range
	CycleTime (DataType="xs:PositiveInteger", Unit="s")	Cycle time
	DelayTime (DataType="xs:PositiveInteger", Unit="s")	Delay time
	RotationOfTransportUnit (DataType="xs:string")	Defines the rotation of a transport unit in the process as Lengthwise, Crosswise or Random
	LateralPosition (DataType="xs:string")	Defines the lateral position of a transport unit relative to the transport direction as LeftHand, RightHand, Centered or Random

4.3.6 Transport unit processes

Class name	TransportUnitProcess	
Description	A general class to define transport unit processes	
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process	
Path for element reference	MaterialHandlingProcessClassLib/TransportUnitProcess	
Attributes	ProcessTime (DataType="xs:PositiveInteger", Unit="s")	The period during which the transport unit process executed.

Class name	Loading	
Description	Defines a transport unit loading process	
Parent class	MaterialHandlingProcessClassLib/TransportUnitProcess	
Path for element reference	MaterialHandlingProcessClassLib/TransportUnitProcess/Loading	

Class name	Unloading	
Description	Defines a transport unit unloading process	
Parent class	MaterialHandlingProcessClassLib/TransportUnitProcess	
Path for element reference	MaterialHandlingProcessClassLib/TransportUnitProcess/Unloading	

Class name	Transloading
Description	Transloading is the process of transferring a shipment from one mode of transportation to another.
Parent class	MaterialHandlingProcessClassLib/TransportUnitProcess
Path for element reference	MaterialHandlingProcessClassLib/TransportUnitProcess/Transloading

Class name	OrderPicking
Description	Order picking stands for the compilation of transport units according to given orders from an available complete assortment.
Parent class	MaterialHandlingProcessClassLib/TransportUnitProcess
Path for element reference	MaterialHandlingProcessClassLib/TransportUnitProcess/OrderPicking

Class name	Carrying	
Description	Defines the process of carrying transport units with a component or a load carrier.	
Parent class	MaterialHandlingProcessClassLib/TransportUnitProcess	
Path for element reference	MaterialHandlingProcessClassLib/TransportUnitProcess/Carrying	
Attributes	“LoadCapacity” (AttributeDataType=“xs:double”, Unit=“kg”)	Defines transport unit capacity in kilograms.
	“AmountCapacity” (AttributeDataType=“xs:PositiveInteger”, Unit=“pce”)	Defines transport unit capacity in pieces.

4.3.7 Storing

Class name	Storing	
Description	Storing is according to VDI 2411 every planned interruption in the material flow.	
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Process	
Path for element reference	MaterialHandlingProcessClassLib/Storing	
Attributes	CapacityAmount (DataType="xs:PositiveInteger", Unit="pce")	Capacity defined in amount of transport units.
	CapacityLength (DataType="xs:double", Unit="m")	Defines the length which allows the storage of transport units with different sizes.
	PriorityRule (DataType="xs:string")	Defines the priority rule as FIFO, LIFO, Sequence or PriorityByProduct

Class name	Stocking	
Description	Storing transport units for longer periods.	
Parent class	MaterialHandlingProcessClassLib/Storing	
Path for element reference	MaterialHandlingProcessClassLib/Storing/Stocking	
Attributes	MediumSingleCycleTime (DataType="xs:PositiveInteger", Unit="s")	Medium time that is spent to load a transport unit to storage or unload a transport unit from storage
	MediumDoubleCycleTime (DataType="xs:PositiveInteger", Unit="s")	Medium time that is spent to load a transport unit to storage, pick up another transport unit from storage and unload

Class name	Buffering
Description	Buffering is a similar process to storing where transport units are kept in a buffer to ensure trouble-free operation.
Parent class	MaterialHandlingProcessClassLib/Storing
Path for element reference	MaterialHandlingProcessClassLib/Storing/Buffering

Class name	Retaining
Description	Retaining is a similar process to storage, where transport units are deliberately and intentionally kept in a store, but only in the short term.
Parent class	MaterialHandlingProcessClassLib/Storing
Path for element reference	MaterialHandlingProcessClassLib/Storing/ Retaining

4.4 Material handling product role classes

The role classes in the *MaterialHandlingProductClassLib* are used to define the product related aspects regarding material handling view. There are two main components in this library, transport unit and load carrier. Every object, which are moved by resources, has been considered as products. If there is a need to define a specific product semantics, it can be derived from the *TransportUnit* role. *LoadCarrier* is used to define elements like containers, pallets, boxes, etc. They move together with transport units on the production line. It is also possible that they are delivered together with transport units in the final product. That is an element which has the characteristics of both the product and the resource. The *SpecificLoadCarrier* role, which is a child of the *LoadCarrier* role, allows to specify the type of a load carrier with the *Type* attribute.

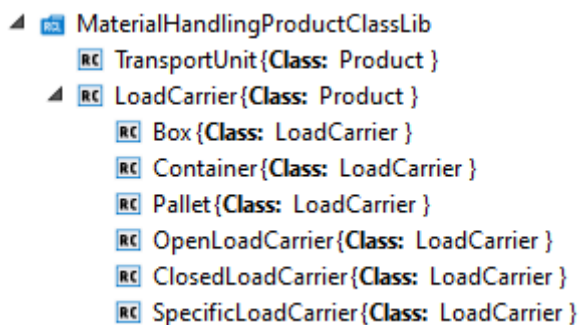


Figure 12 *MaterialHandlingProductClassLib*

Class name	TransportUnit	
Description	A moving object, which are handled by resources to create a final product, or the product itself	
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Product	
Path for element reference	MaterialHandlingProductClassLib/TransportUnit	
Attributes	BasicForm (DataType="xs:string")	Basic form of the product if the measurements are not defined accurately. Can be linear, laminar, cubic, cylindrical or free shape (undefined).
	Height (DataType="xs:double", Unit="m")	Height of the transport unit
	Height.Min (DataType="xs:double", Unit="m")	Minimum height of the transport unit
	Height.Max (DataType="xs:double", Unit="m")	Maximum height of the transport unit
	Width (DataType="xs:double", Unit="m")	Width of the transport unit
	Width.Min(DataType="xs:double", Unit="m")	Minimum width of the transport unit
	Width.Max(DataType="xs:double",Unit="m")	Maximum width of the transport unit
	Length (DataType="xs:double", Unit="m")	Length of the transport unit
	Length.Min(DataType="xs:double",Unit="m")	Minimum length of the transport unit
	Length.Max(DataType="xs:double",Unit="m")	Maximum length of the transport unit
	Weight (DataType="xs:double", Unit="kg")	Weight of the transport unit
	Weight.Min(DataType="xs:double",Unit="kg")	Minimum weight of the transport unit
	Weight.Max(DataType="xs:double",Unit="kg")	Maximum weight of the transport unit
	SpecialProperties (DataType="xs:string")	Other properties which can be defined freely i.e. Shock sensitivity

Class name	LoadCarrier	
Description	A general role for load carriers like containers, pallets, boxes. They move together with transport units on the production line. It is also possible that they are delivered together with transport units in the final product. That is an element which has the characteristics of both the product and the resource.	
Parent class	AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Product	
Path for element reference	MaterialHandlingProductClassLib/LoadCarrier	
Attributes	Width (DataType="xs:double", Unit="m")	Width of the load carrier
	Height (DataType="xs:double", Unit="m")	Height of the load carrier
	Length (DataType="xs:double", Unit="m")	Length of the load carrier

Class name	Box
Description	Represents box for materials
Parent class	MaterialHandlingProductClassLib/LoadCarrier
Path for element reference	MaterialHandlingProductClassLib/LoadCarrier/Box

Class name	Container
Description	Represents any container for materials
Parent class	MaterialHandlingProductClassLib/LoadCarrier
Path for element reference	MaterialHandlingProductClassLib/LoadCarrier/Container

Class name	Pallet
Description	Represents any pallet for materials
Parent class	MaterialHandlingProductClassLib/LoadCarrier
Path for element reference	MaterialHandlingProductClassLib/LoadCarrier/Pallet

Class name	OpenLoadCarrier
Description	Represents a general open load carrier for materials like shelf boards, pallets, etc.
Parent class	MaterialHandlingProductClassLib/LoadCarrier
Path for element reference	MaterialHandlingProductClassLib/LoadCarrier/OpenLoadCarrier

Class name	ClosedLoadCarrier
Description	Represents a general closed load carrier for materials like containers, pallet cages, etc.
Parent class	MaterialHandlingProductClassLib/LoadCarrier
Path for element reference	MaterialHandlingProductClassLib/LoadCarrier/ClosedLoadCarrier

Class name	SpecificLoadCarrier	
Description	Represents a specific load carrier for materials like skids, racks, etc.	
Parent class	MaterialHandlingProductClassLib/LoadCarrier	
Path for element reference	MaterialHandlingProductClassLib/LoadCarrier/SpecificLoadCarrier	
Attributes	Type (DataType="xs:string",)	Type of the specific load carrier

5 Examples

This chapter illustrates the practical usage of material handling libraries. Examples are provided for both module and port modelling levels.

5.1 Level 1 (module level)

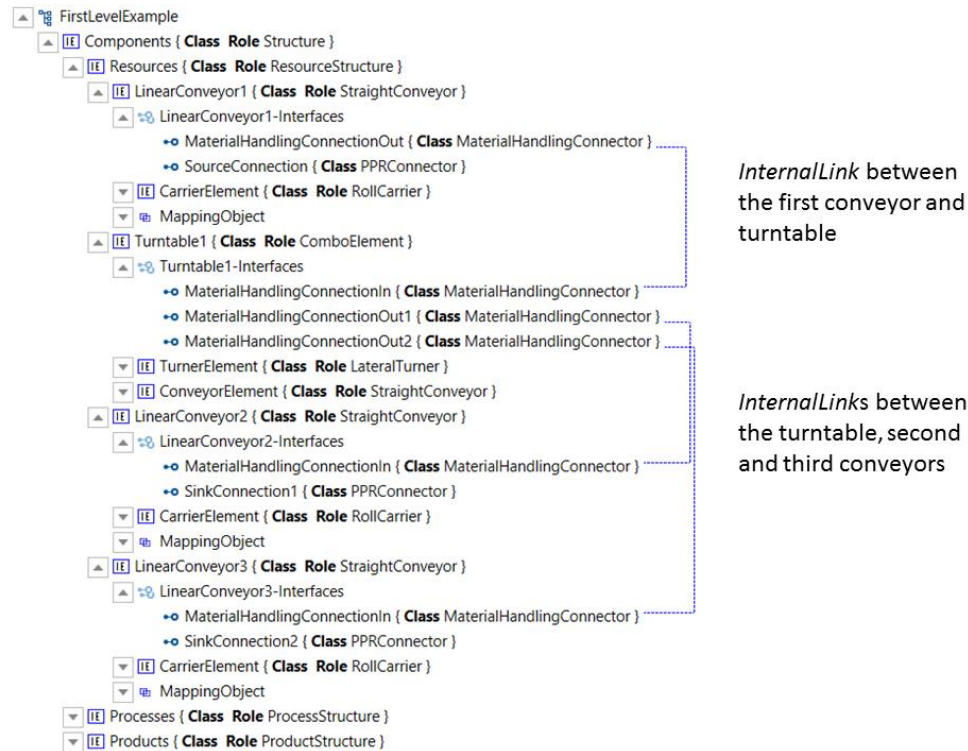
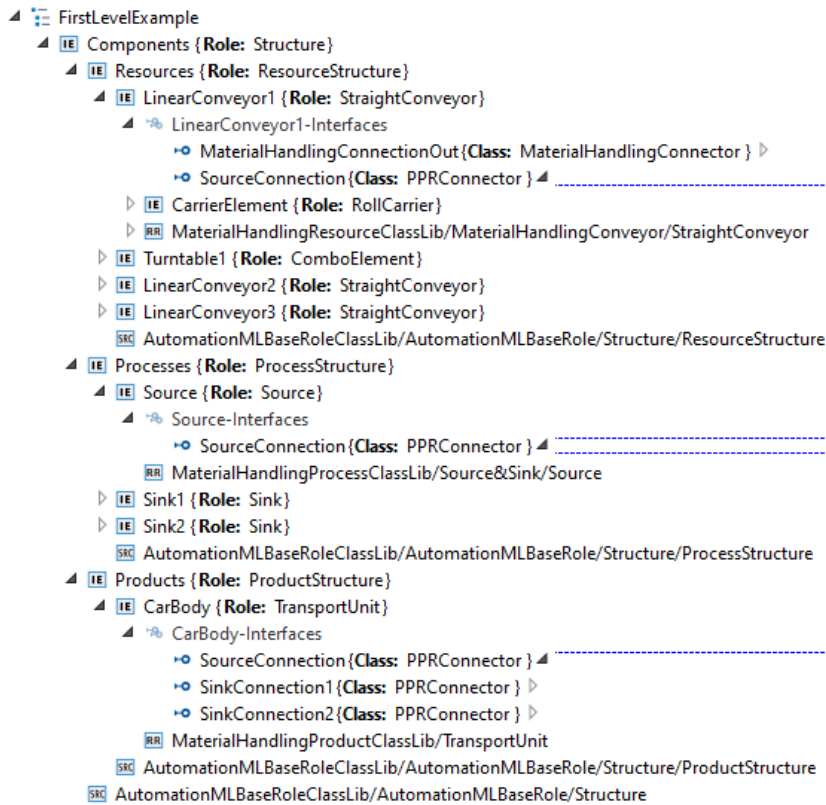


Figure 13 The connection between turntable and conveyors for module level (1. level)

The first example contains three conveyors and one turntable, parallel to the first example scene for the module level (see Figure 3). The first conveyor *LinearConveyor1* is connected to the *Turntable1*, which distributes the product to the *LinearConveyor2* and *LinearConveyor3*. Linear conveyors are assigned to the material handling role class *StraightConveyor*. Turntable is a combination of a turner and conveyor element, which are assigned to the *LateralTurner* and *StraightConveyor* roles in the order. The role requirement *ComboElement* indicates that the turntable is a combination of multiple elements.

Conveyor elements contain a child element with the name *CarrierElement*, which is assigned to the material handling role *RollCarrier*. This role shows what kind of conveyor technology is used in the component.

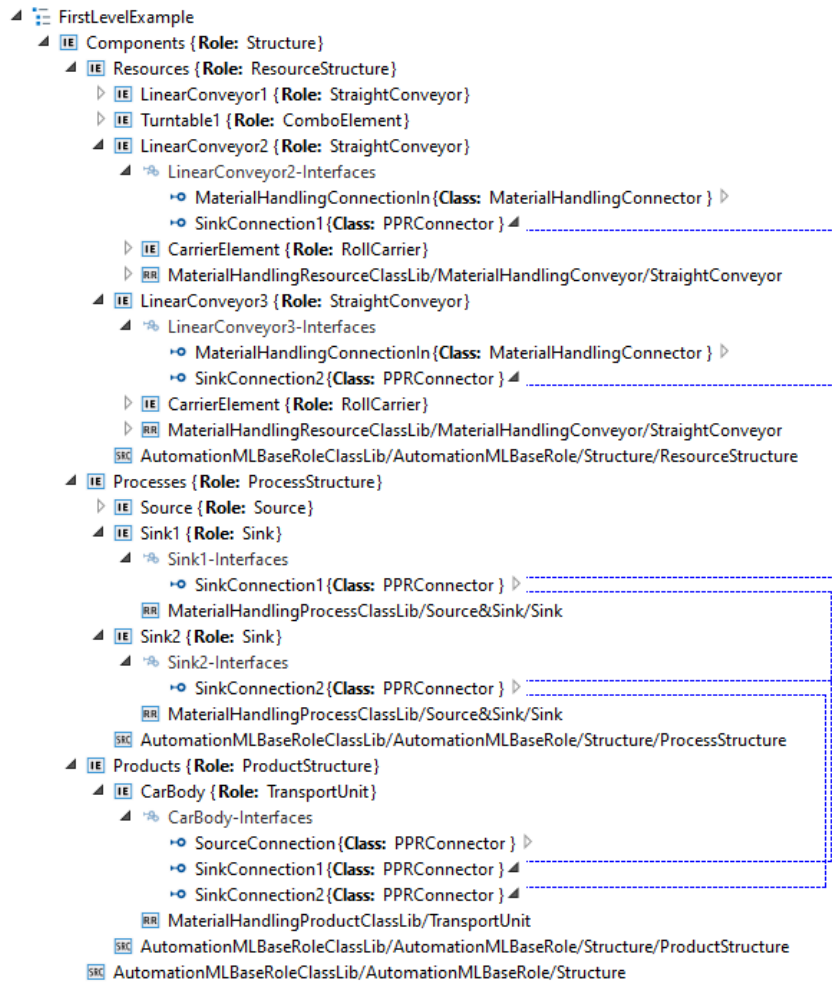
The connections between the turntable and conveyors are realized by the *MaterialHandlingConnector* interface class. The direction of the material flow can be identified through the *Direction* attribute of the *MaterialHandlingConnector*, which is inherited from base interface class *Order* (*MaterialHandlingConnector* is derived from *MaterialHandlingInterface*, which is derived from *Order*). In addition, the name of the interface class has been chosen correspondingly to show the material flow direction in this example (In, Out).



InternalLinks to define the source for a transport unit in the module level

Figure 14 Usage of the PPR interface and the Source role to define a source for transport units

Figure 14 show how the PPR interface from base AutomationML interfaces can be utilized to define the source for transport units (car body in this example) in the module level. In this case, transport units (car body) enter the system through the *LinearConveyor1* element. A source element, which is assigned to the role *Source* from material handling process class library, is used to define the process.



InternalLinks to define sinks for a transport unit in the module level

Figure 15 Usage of the PPR interface and the Sink role to define sinks for transport units

Similar to the source concept, sinks are assigned to the *Sink* role from the material handling process class library and connected with the product and resource by the PPR interface. *LinearConveyor2* is connected with *Sink1* and *LinearConveyor3* is with *Sink2* in this case.

5.2 Level 2 (port level)

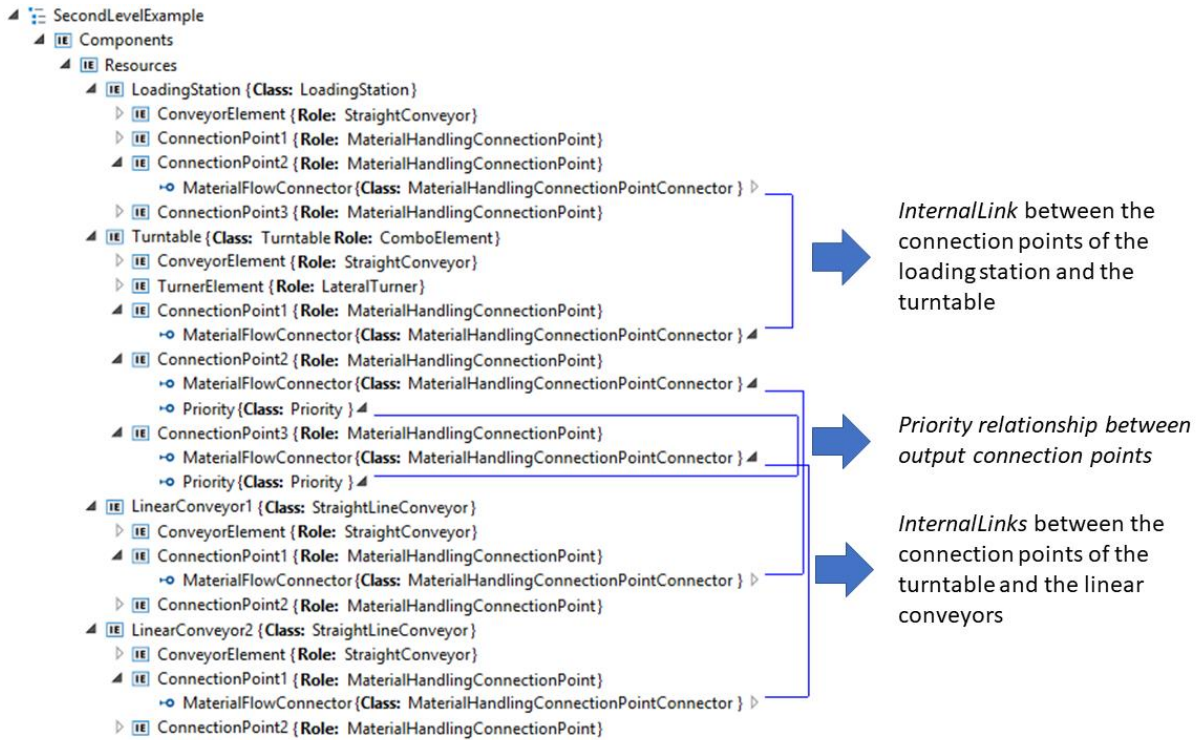


Figure 16 The connection between turntable and conveyors for port level (2. level)

The main difference between module level (1. Level) concept and port level (2. Level) concept can be seen through the example in Figure 16. The resource components are connected with each other via connection points, which are assigned to the *MaterialHandlingConnectionPoint* role class. This role class has an interface *MaterialHandlingConnectionPointConnector*, which is derived from *MaterialHandlingInterface* (similar to *MaterialHandlingConnector* interface). This interface also inherits *Direction* attribute from *Order*. Connection point elements contain *Frame* attribute, so it is possible to say where the transport of the product from one component to another occurs.

This example scene is very similar to the first example, but there is a loading station as the first element in the material flow instead of the linear conveyor. The other components are the same. The loading station is an element with three connection points. One of the connection points defines the input port, which is in this case *ConnectionPoint1*. *ConnectionPoint2* and *ConnectionPoint3* are output ports, which are also connected with the *Priority* interface. This interface defines which one of the output ports should be preferred first with the help of the *PriorityDegree* attribute.

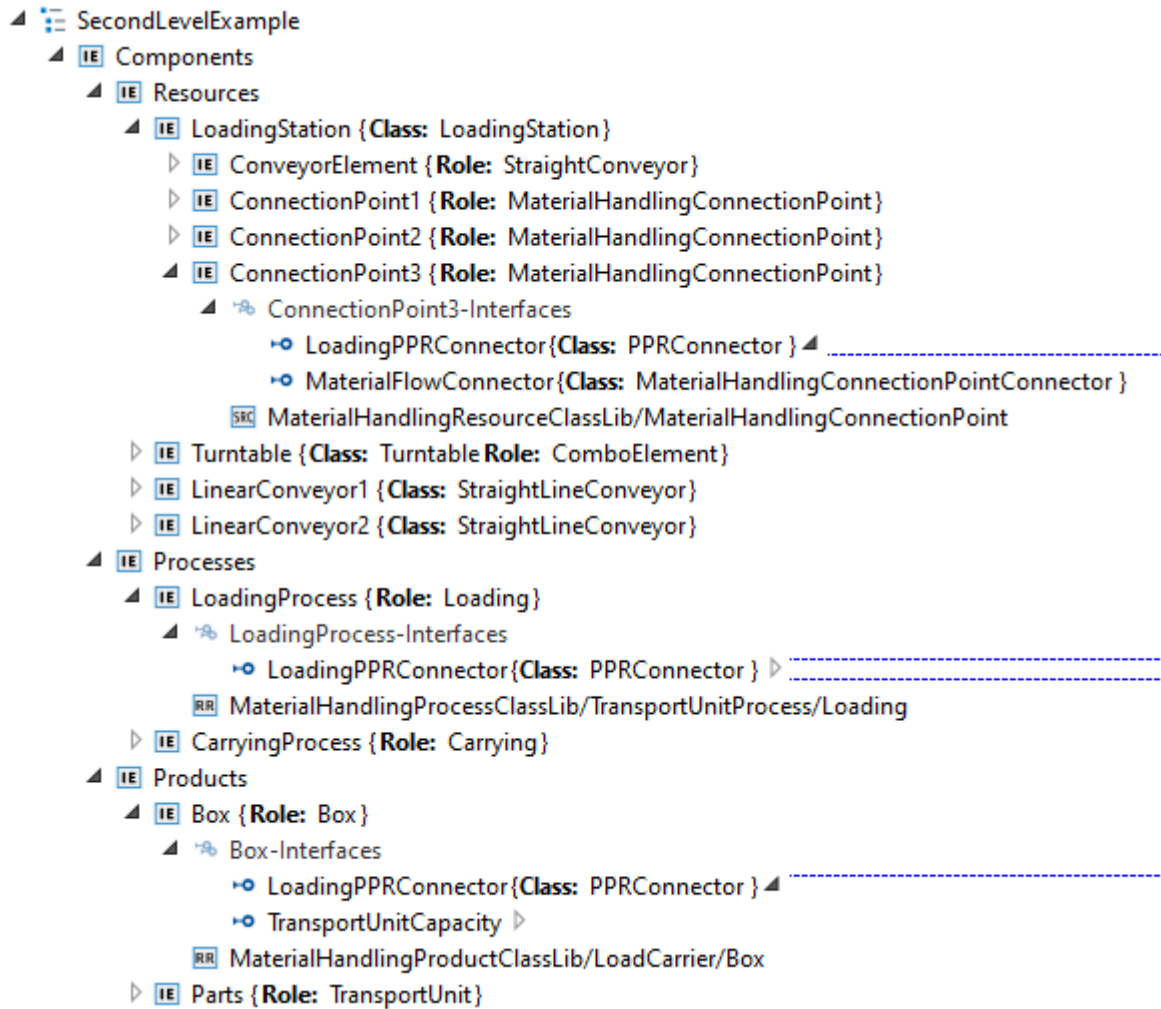


Figure 17 Usage of the PPR interface and Loading role to define loading process

Figure 17 shows how the material handling process role *Loading* can be utilized to define the loading point of a load carrier (a box containing parts in this case) through the *PPRConnector* interface. An internal element with the name *LoadingProcess* implements the process role class *Loading*. This internal element is connected with the *ConnectionPoint3* from the resource *LoadingStation* and the load carrier element *Box* through PPR-Interface. Thereby it is possible to define which product is loaded in which connection point through the loading process.

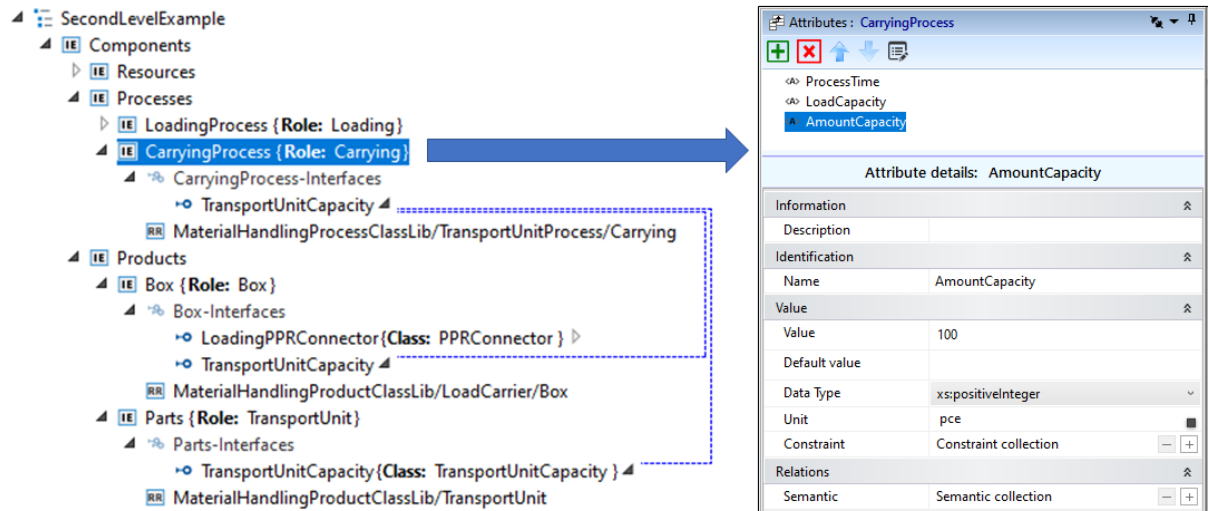


Figure 18 Usage of the *TransportUnitCapacity* interface and *Carrying* role to define the load carrier capacity

Figure 16 shows how the material handling process role *Carrying* is used to define the loading capacity of a load carrier for a specific type of transport unit with the help of *TransportUnitCapacity* interface. The internal element *CarryingProcess* with the role *Carrying* is connected with the load carrier *Box* and the transport unit element *Parts*. With the help of *AmountCapacity* attribute, which is defined in the role *Carrying*, the amount of parts, which can be carried with the load carrier, is defined.