



# **Creation of a role class library for material handling**

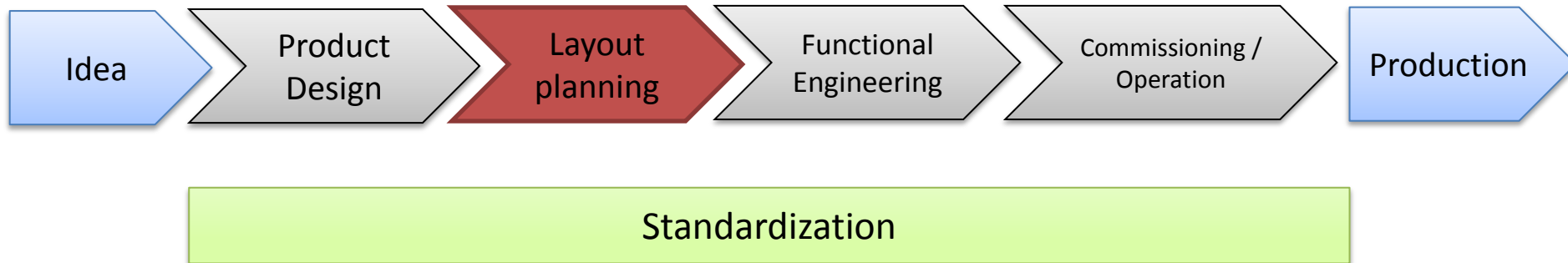
**15.09.2015**

**Ender Yemenicioglu**

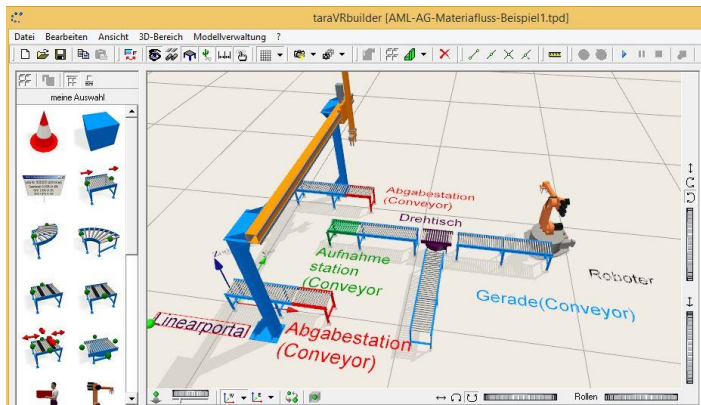


## Motivation:

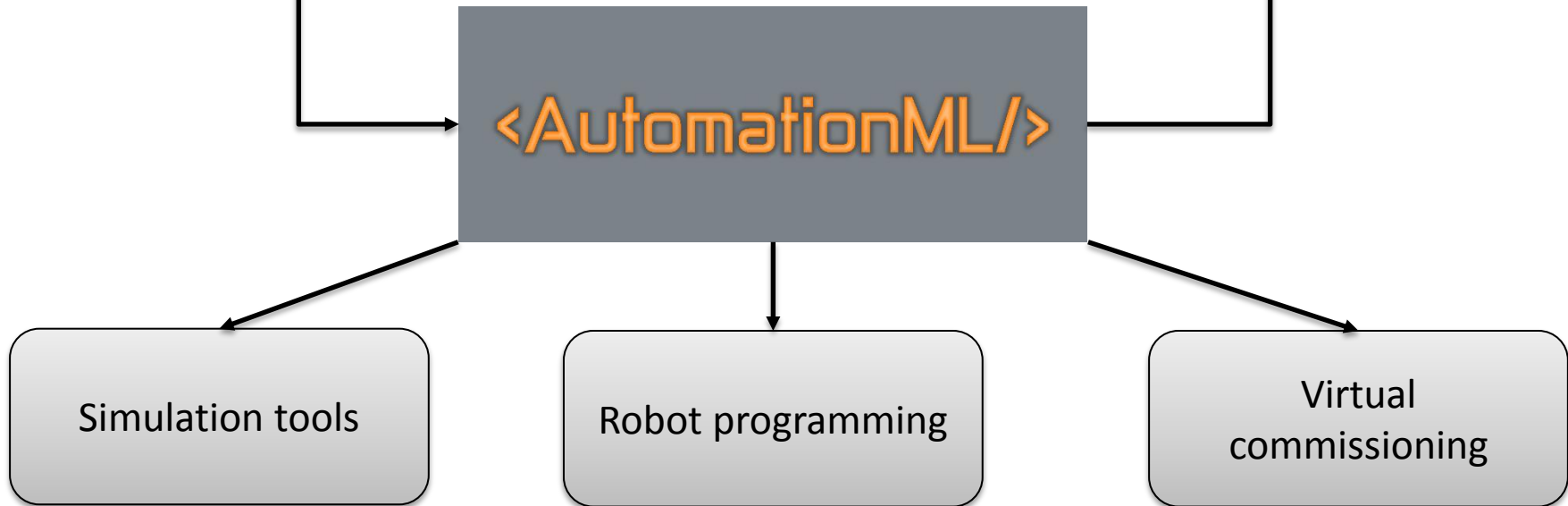
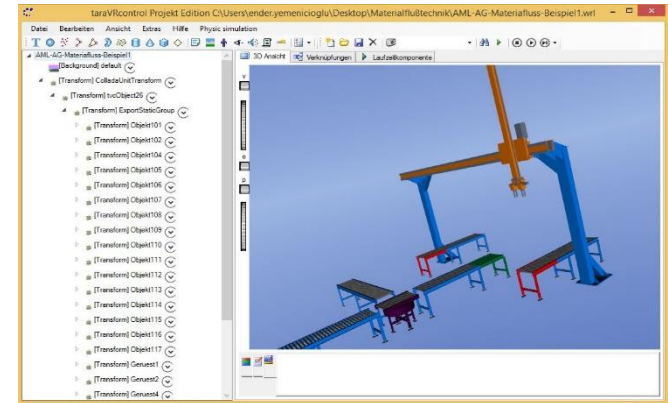
In the mechatronic engineering process the product design phase is followed by the layout planning step. Already in this early step of facility design there are a lot of information in use. Existing standard facilities or sub-components are commonly used during the planning of a new facility. The layout planning can be a basis template for further steps like process simulation, material handling simulation, robot programming or virtual commissioning. Transferring the existing information to the engineering tools for the further steps is an important task in the work flow. For this reason standardization is necessary.



## Layout planning tool



## Visualisation tool





*What is already defined in AML-Whitepaper?*

- 1) Meta information like name, version, creation date, etc.
- 2) Topology (InstanceHierarchy objects)
- 3) Position (Frame attribute)
- 4) Geometry (Extern referencing of Collada files)
- 5) Connection points from components (InternalLink object)
- 6) AutomationMLExtendedRoleClassLib:
  - Turntable, Conveyor, AGV, etc.
  - No attributes
  - There is no "transport" role which is in „Disc... Equipment“ library.

A new approach is necessary!

## Existing role class library:

RoleClassLib

- AutomationMLExtendedRoleClassLib
  - ProductionLine { **Class:** ResourceStructure }
  - WorkCell { **Class:** ResourceStructure }
  - ProcessCell { **Class:** ResourceStructure }
  - Unit { **Class:** ResourceStructure }
  - ProductionUnit { **Class:** ResourceStructure }
  - StorageZone { **Class:** ResourceStructure }
  - StorageUnit { **Class:** ResourceStructure }
  - Turntable { **Class:** Transport }
  - Conveyor { **Class:** Transport }
    - BeltConveyor { **Class:** Conveyor }
    - RollConveyor { **Class:** Conveyor }
    - ChainConveyor { **Class:** Conveyor }
    - PalletConveyor { **Class:** Conveyor }
    - OverheadConveyor { **Class:** Conveyor }
  - LiftingTable { **Class:** Transport }
  - AGV { **Class:** Transport }
  - Transposer { **Class:** Transport }
  - CarrierHandlingSystem { **Class:** Transport }

- AutomationMLDMIRoleClassLib
  - DiscManufacturingEquipment { **Class:** Resource }
    - Transport { **Class:** DiscManufacturingEquipment }
    - Storage { **Class:** DiscManufacturingEquipment }
    - Fixture { **Class:** DiscManufacturingEquipment }

No attributes

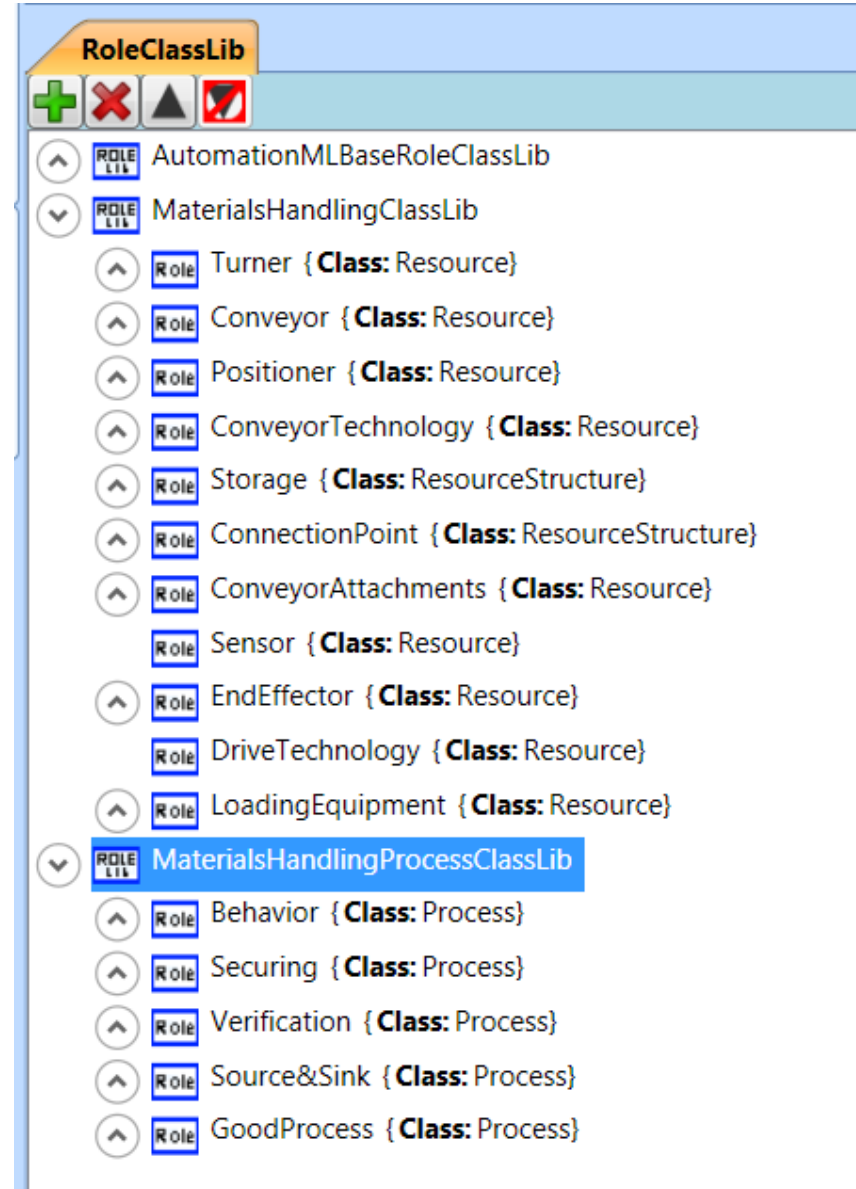
```
<RoleClass Name="Turntable" RefBaseClassPath=
"AutomationMLDMIRoleClassLib@AutomationMLDMIRoleClassLib/DiscManufacturingEquipment/Transport" />
<RoleClass Name="Conveyor" RefBaseClassPath=
"AutomationMLDMIRoleClassLib@AutomationMLDMIRoleClassLib/DiscManufacturingEquipment/Transport">
  <RoleClass Name="BeltConveyor" RefBaseClassPath="Conveyor" />
  <RoleClass Name="RollConveyor" RefBaseClassPath="Conveyor" />
  <RoleClass Name="ChainConveyor" RefBaseClassPath="Conveyor" />
  <RoleClass Name="PalletConveyor" RefBaseClassPath="Conveyor" />
  <RoleClass Name="OverheadConveyor" RefBaseClassPath="Conveyor" />
</RoleClass>
```

Categorized by loading equipment, which is a subcomponent of the conveyor



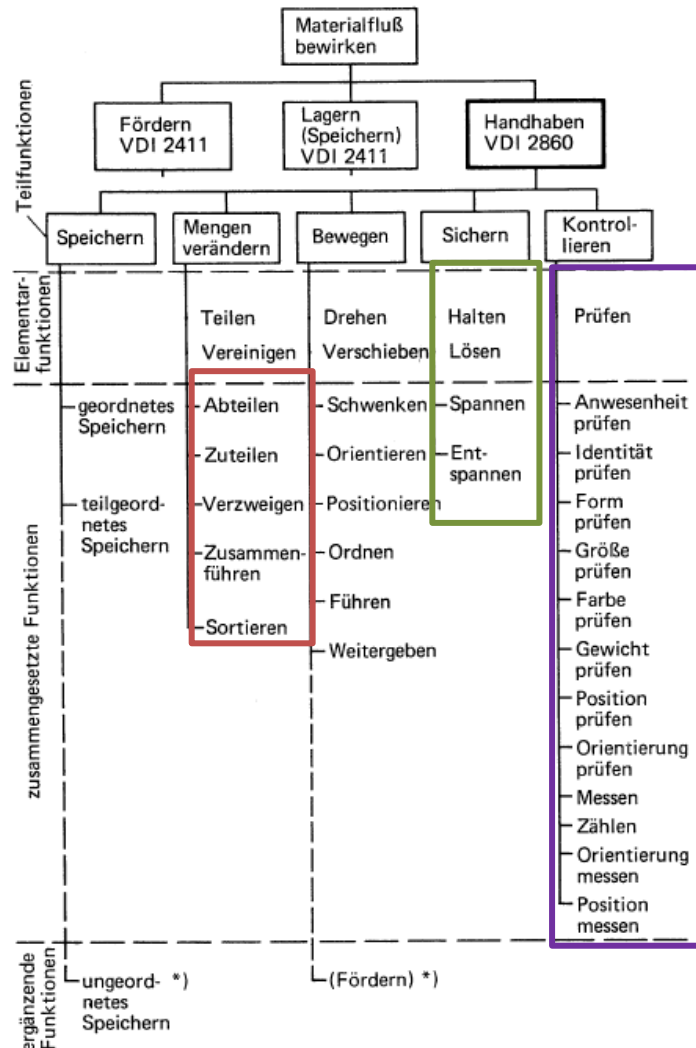
## Creation of a standard role class library:

- Small function-based roles
- Separation of concerns: Product, resource, process
- Derived from standard base roles like „Resource“ or „ResourceStructure“
- Comparison with standards like VDI 2860 and VDI 2411
- Difference of conveyor and positioner: Positioner moves together with the last, conveyor stays steady and conveys the last
- Loading equipment as a role, not the categorization





## Process roles in accordance with VDI 2860



- RL Sorting { **Class:** Behavior }
- ▲ RL Dividing { **Class:** Behavior }
  - RL Partition { **Class:** Dividing }
  - RL Distribution { **Class:** Dividing }
  - RL Branching { **Class:** Dividing }

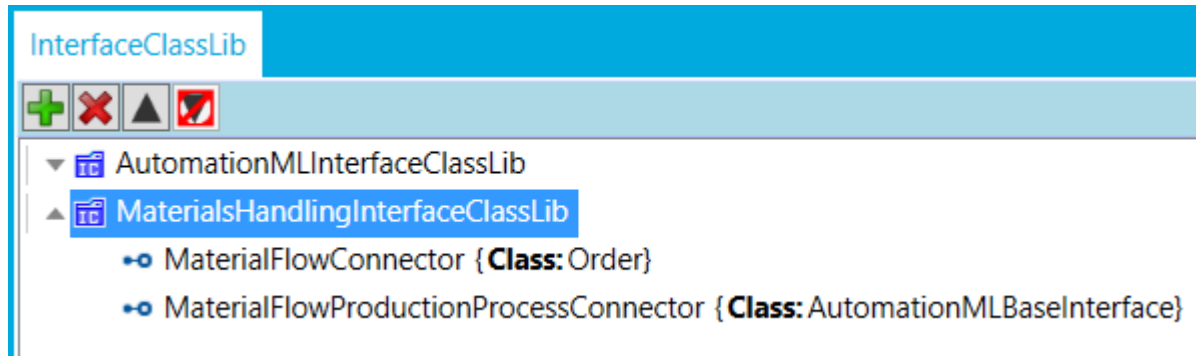
- ▲ RL Securing { **Class:** Process }
  - RL Holding { **Class:** Securing }
  - RL Loosening { **Class:** Securing }
  - RL Tightening { **Class:** Securing }
  - RL Relieving { **Class:** Securing }

- ▲ RL Verification { **Class:** Process }
  - ▲ RL Inspection { **Class:** Verification }
    - RL PresenceCheck { **Class:** Inspection }
    - RL ShapeCheck { **Class:** Inspection }
    - RL SizeCheck { **Class:** Inspection }
    - RL ColorCheck { **Class:** Inspection }
    - RL WeightCheck { **Class:** Inspection }
    - RL PositionCheck { **Class:** Inspection }
    - RL OrientationCheck { **Class:** Inspection }
  - ▲ RL Measuring { **Class:** Verification }
    - RL Counting { **Class:** Measuring }
    - RL PositionMeasuring { **Class:** Measuring }
    - RL OrientationMeasuring { **Class:** Measuring }





## Material handling interfaces



### 1) MaterialFlowConnector:

- For the definition of connection points between material flow components
- Derived from „Order“ interface, which has the „Direction“ attribute
- Values for „Direction“: In, Out, In/Out

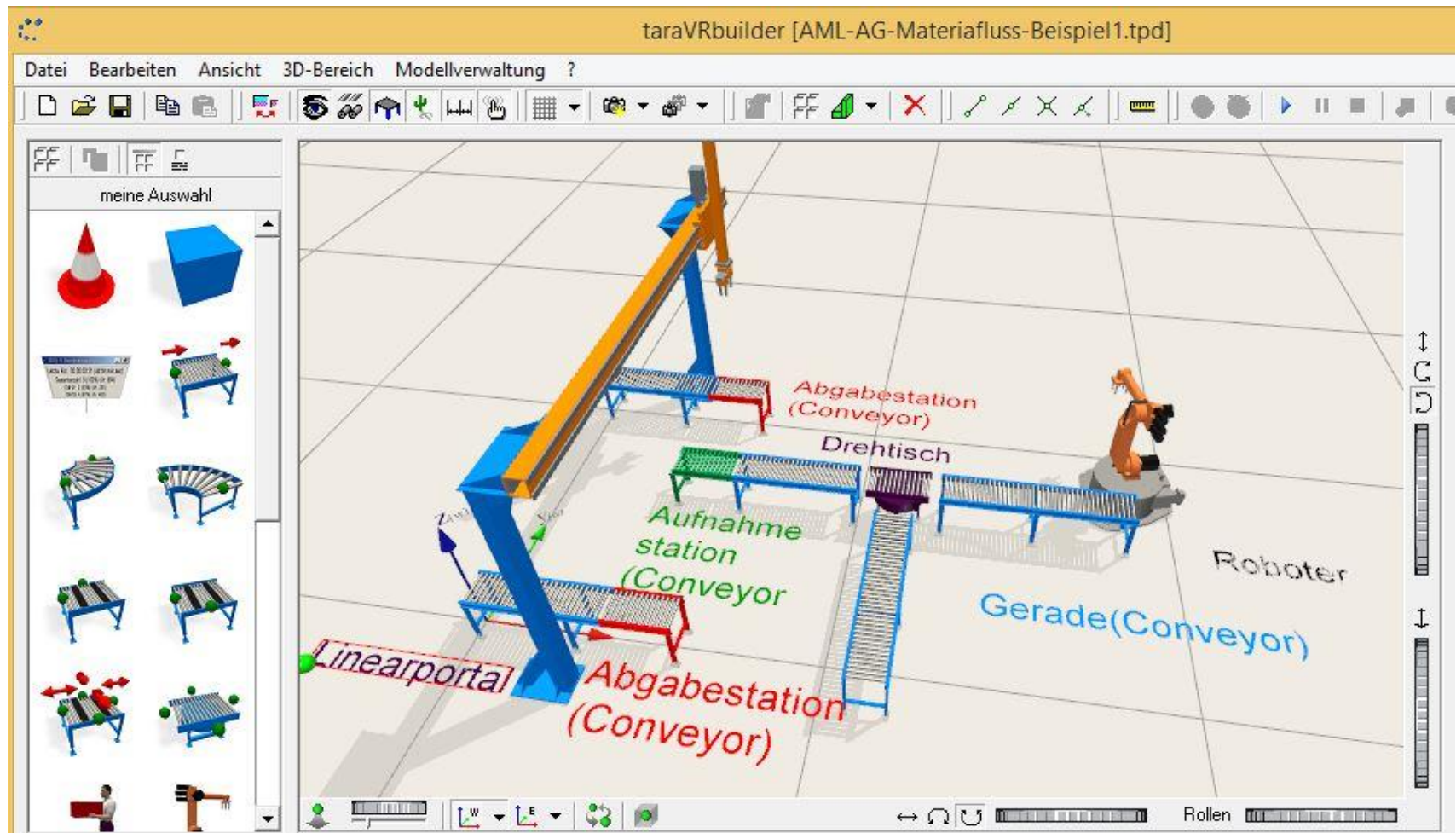
### 2) MaterialFlowProductionProcessConnector:

- AutomationML allows the defining of processes in multiple layers in the same file
- A connection between the material flow process and the production process





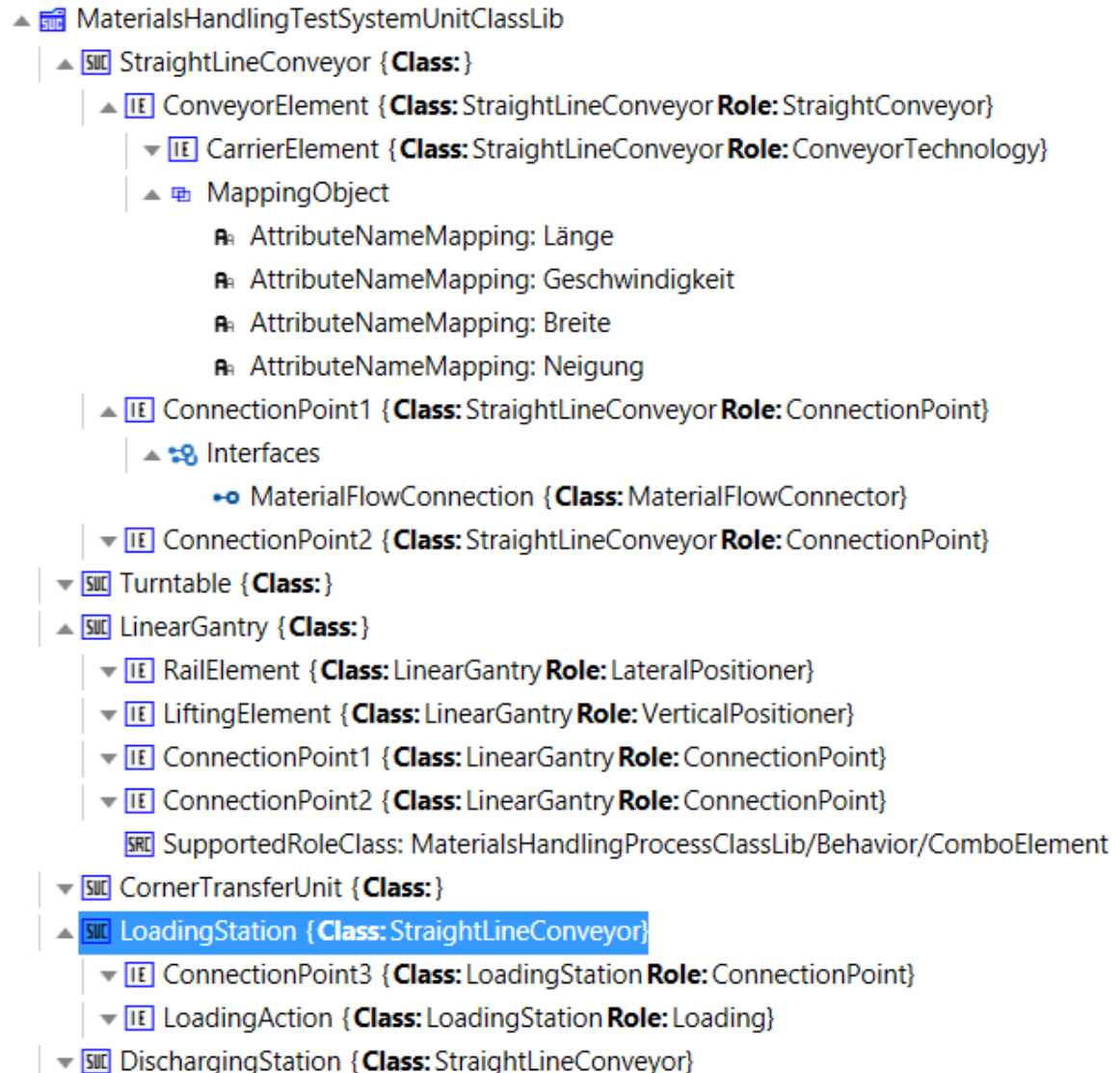
## Example scene for material handling





## SystemUnit class library:

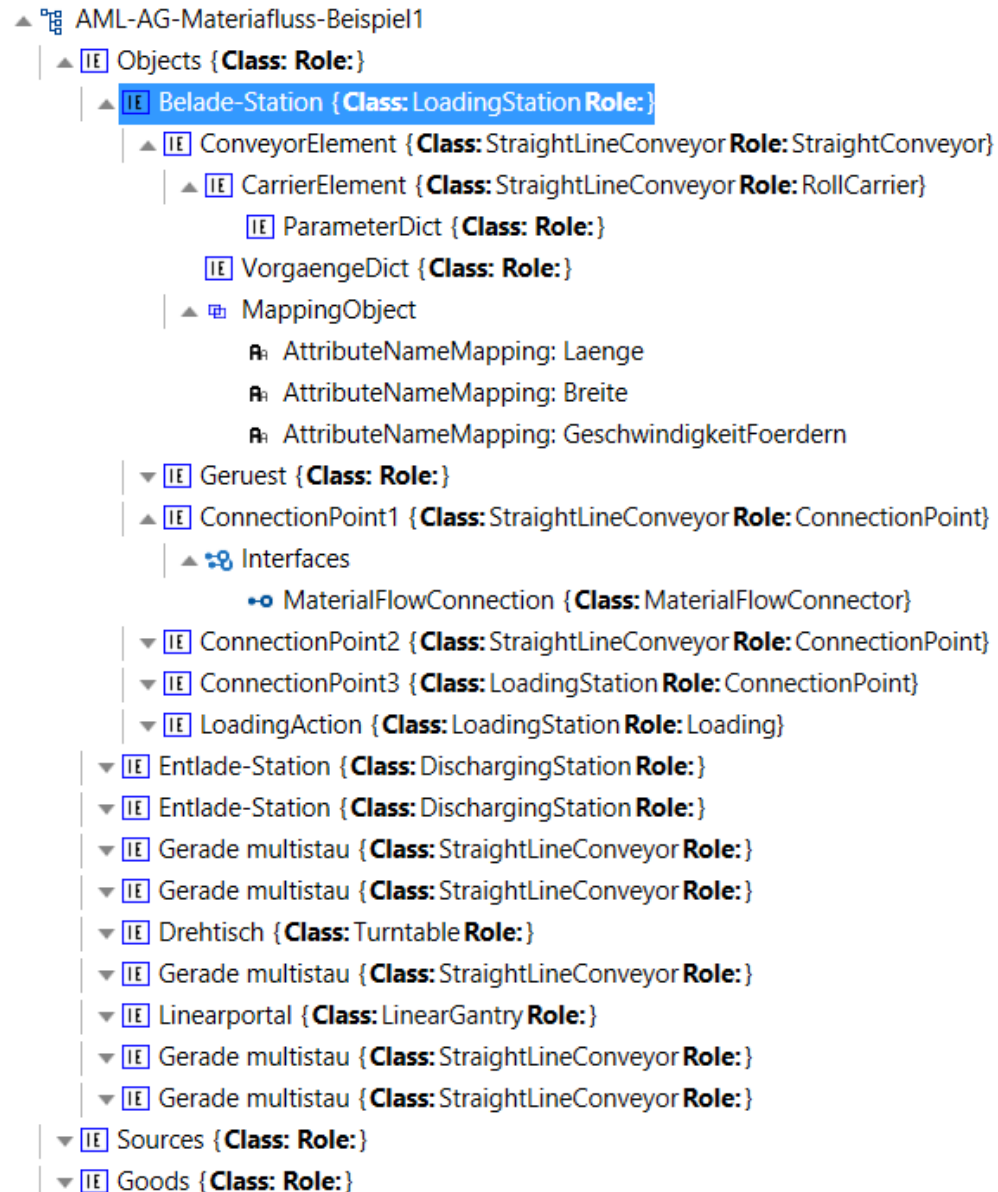
- 6 example implementations: Straight line conveyor, Turntable, Linear gantry, Corner transfer unit, Loading station, Discharging station
- Internal elements, which are referencing the roles
- Connection points which contain the interface „Material Flow Connector“
- Attribute Mapping
- A conveyor has a carrier element, is not categorized with it
- Complicated classes with more than one internal element i.e. Linear gantry or corner transfer unit
- Marker Role: ComboElement
- Usage of inheritance: i.e. Loading station inherits straight line conveyor and defines 2 new sub elements





## Instance hierarchy:

- First approach of an instance hierarchy with the new material handling role class library
- Example scene as the starting point
- An exported AML-data from taraVRBuilder has been edited manually
- A mixture of standard and non-standard elements. i.e. Geruest has no standard role.
- Already implemented:
  - Process roles
  - Linking of the components
  - Attribute mapping





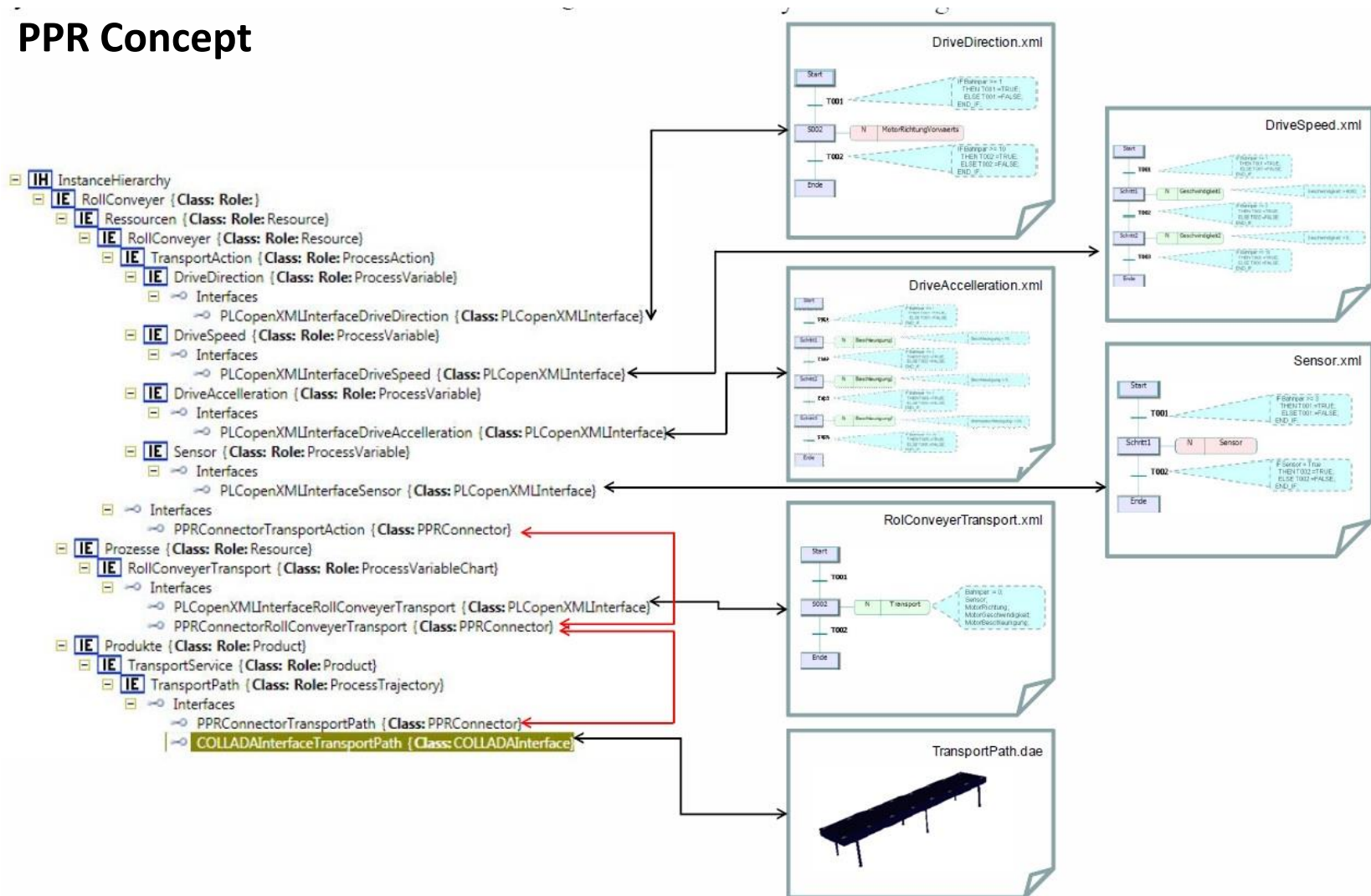
## Next steps:

- Enhancement of the role class library with new aspects of material handling
- Implementing more components
- Product-Process-Resource relationship
- Implementing control aspects of material handling
- Implementing transporting strategies
- Support of the material handling role class library in tarakos software





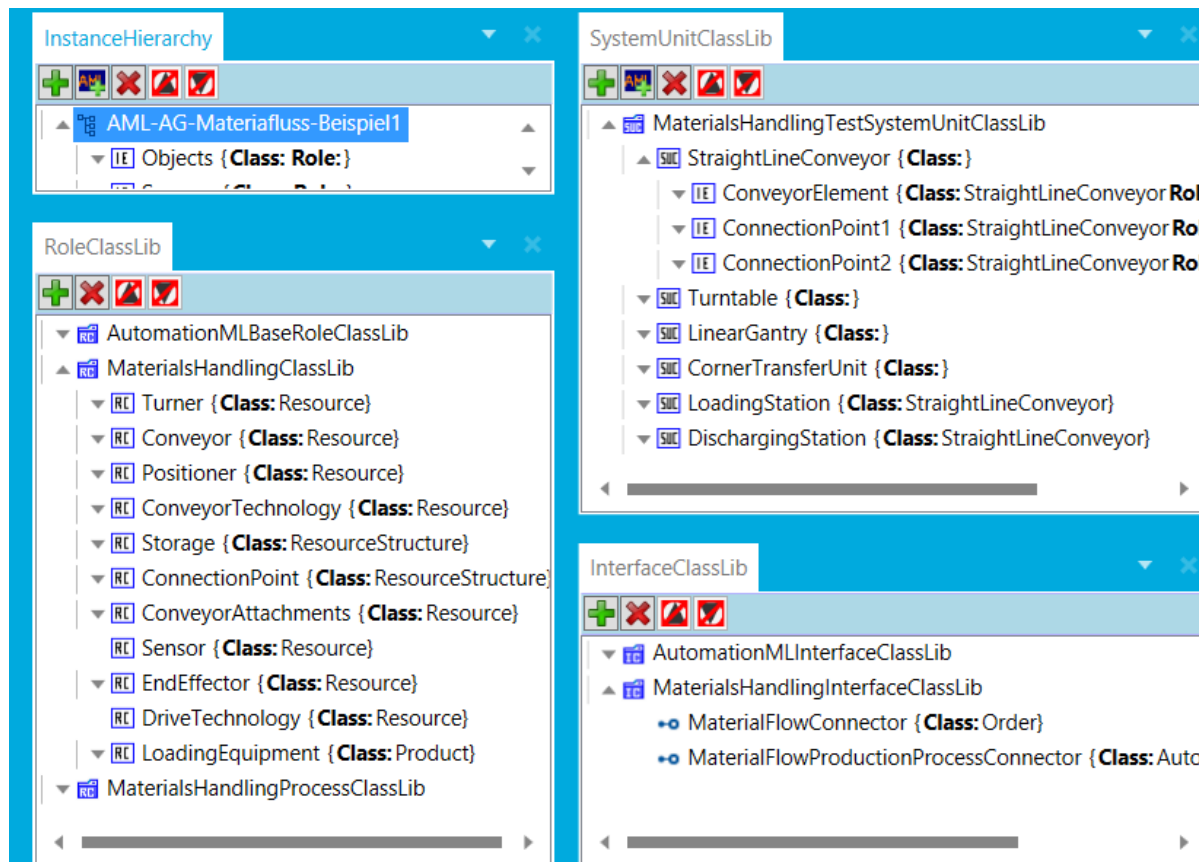
## PPR Concept



\* Lüder, Arndt, Lorenz Hundt, and Andreas Keibel. "Description of manufacturing processes using AutomationML." *Emerging Technologies and Factory Automation (ETFA), 2010 IEEE Conference on*. IEEE, 2010.



## Live demonstration of the further implementation details in the example file





**Questions?**  
**Thank you for your attention.**