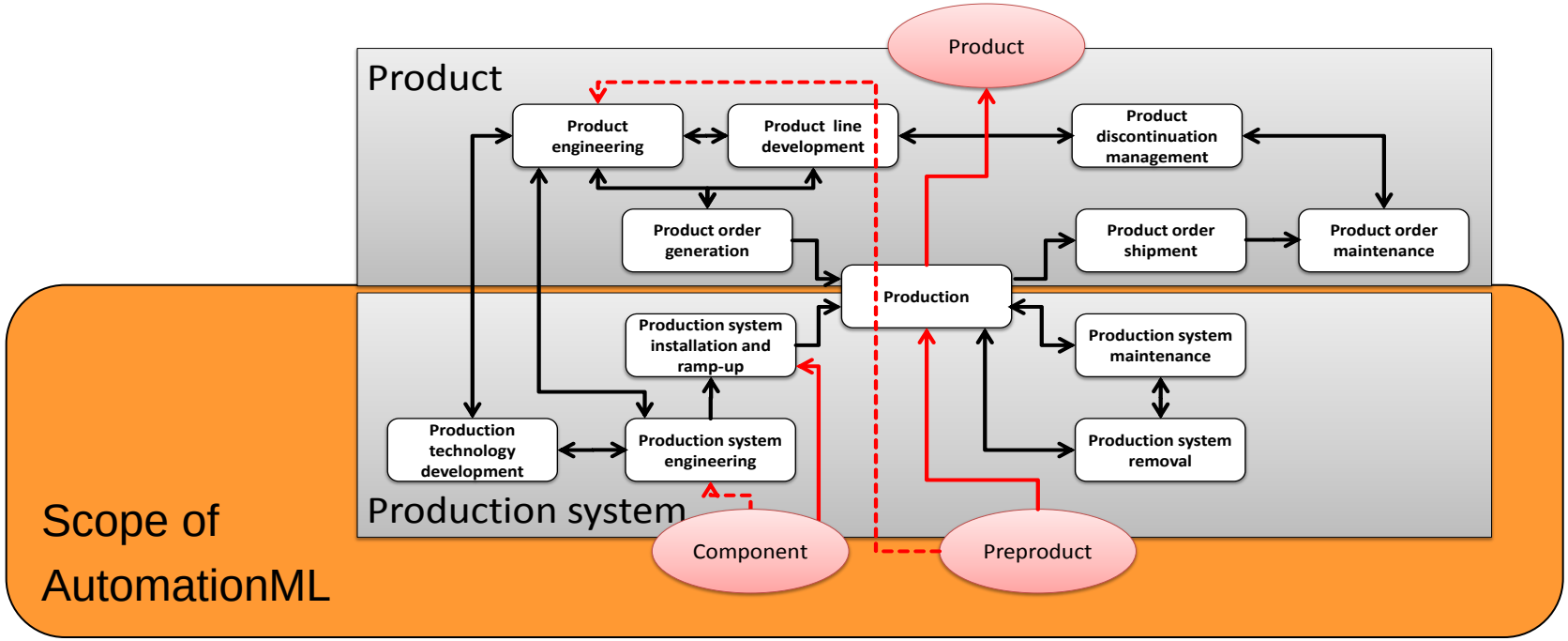
Modeling semantics with AutomationML using the PPR-concept – a beginners workshop

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

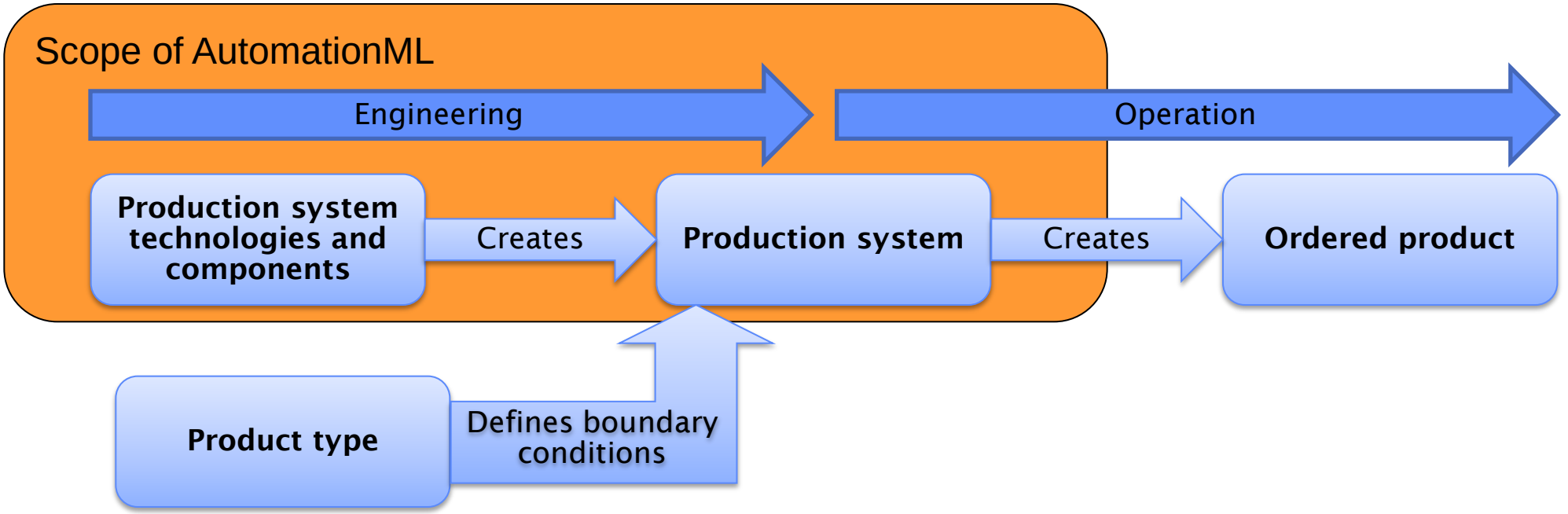
apl. Prof. Dr.- Ing. habil. A. Lüder

- **Engineering and use of production systems is a complex and interdisciplinary process**
  - **Consideration of both products to be produced and production systems used therefore**



OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB
apl. Prof. Dr.- Ing. habil. A. Lüder

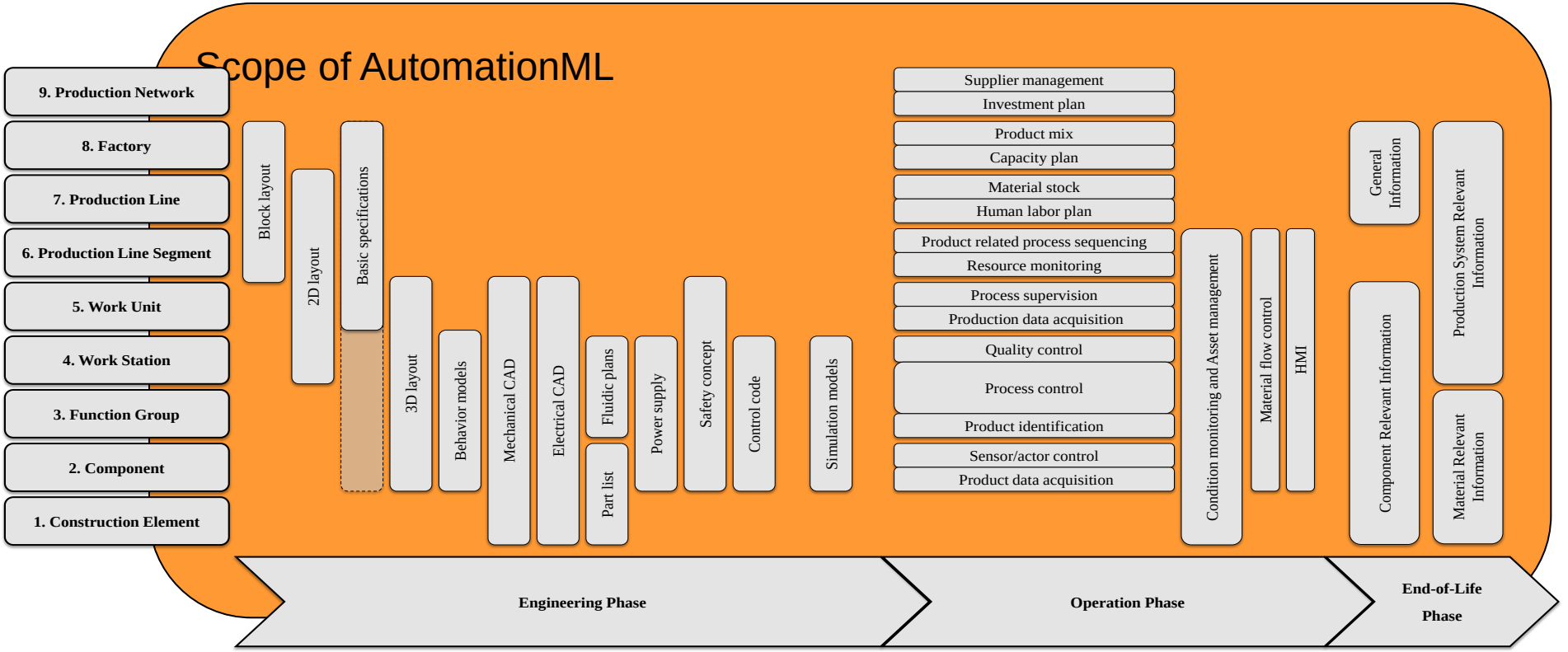
Detailed dependencies between product and production systems



apl. Prof. Dr.- Ing. habil. A. Lüder

Production systems are complex

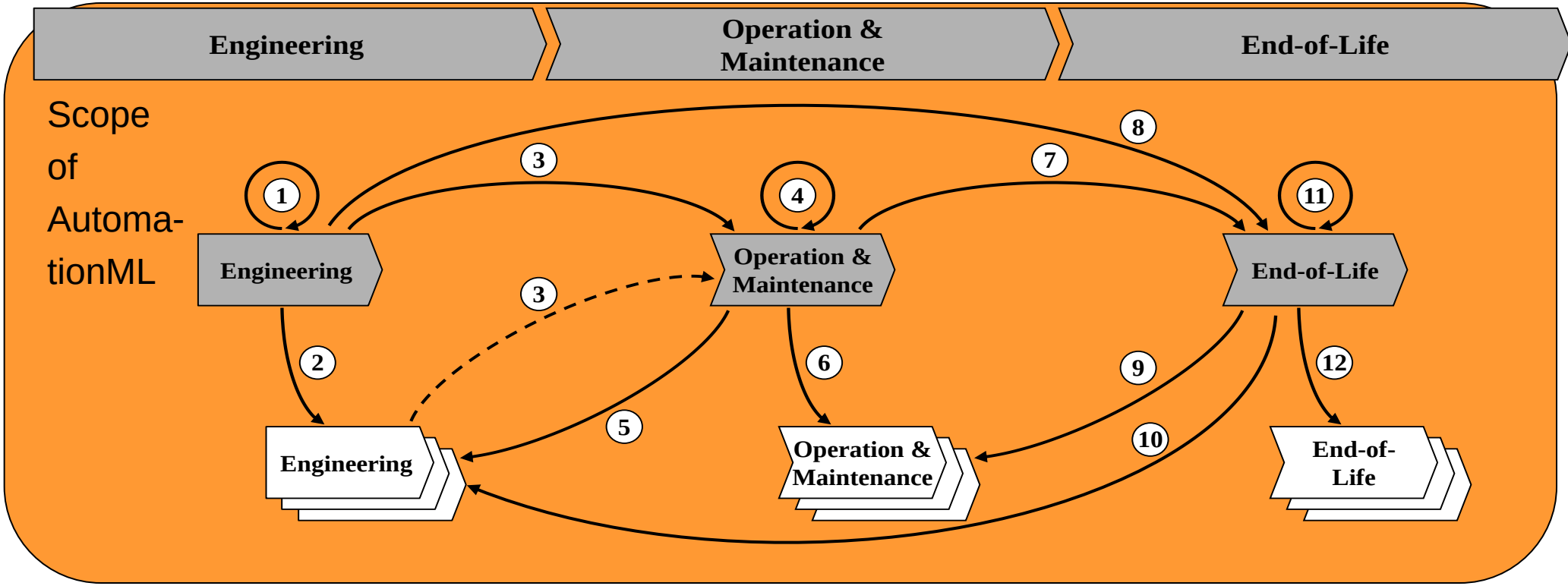
- During the life cycle phases and the different layers of the system hierarchy different information are relevant



OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- **The life cycle phases of different systems are interlinked**
  - **Reuse of gained information / knowledge**



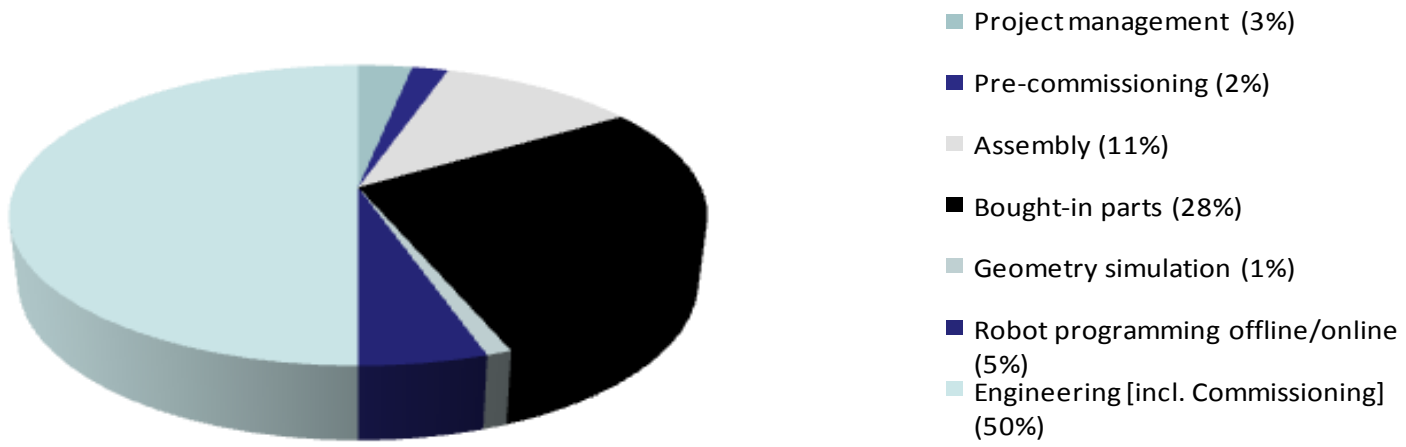
- The engineering of production systems is a complex process:
  - Involving several engineers of
  - Several engineering disciplines,
  - Executing several engineering activities, and
  - Using / creating several engineering artefacts
- Huge amount of human labour involved
- Repetitions of several engineering activities within different engineering tools
  - No appropriate means for data exchange between these tools

➔ Means for lossless data exchange along the entire engineering tool chain necessary

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

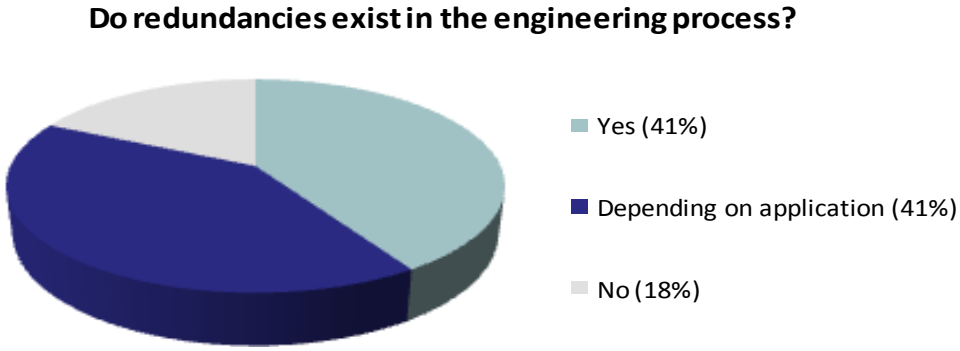
apl. Prof. Dr.- Ing. habil. A. Lüder

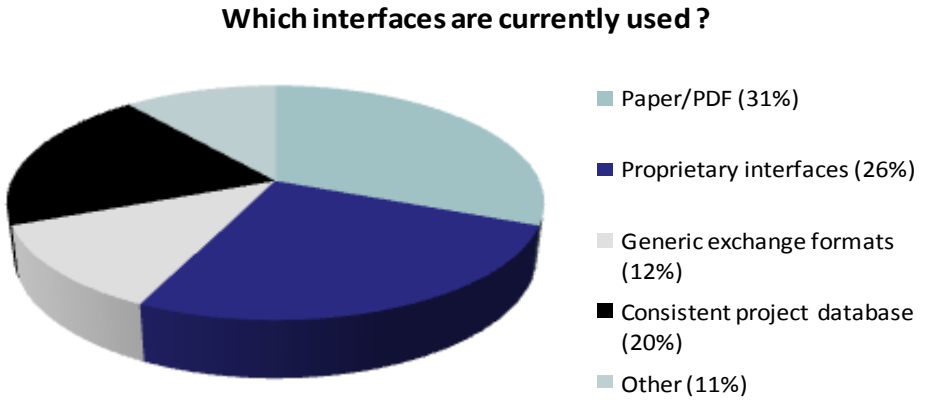
- A significant cost factor of industrial production systems is the engineering process.
- To reduce this factor new methods and description means are necessary!



Source: AIDA 2005

- **What are the avoidable costs?**
- **A survey revealed:**
  - That 82% of the interviewed experts say that redundancy on planning steps exits.
  - That the pdf/paper interface is the most widespread interface with 31%.
  - That only 12% of the interviewed experts use standardized interfaces.





Source: Common project with Austria, the TU Vienna and the OvGU Magdeburg, 2013

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- Many software solutions developed by many engineering organizations and companies

➔Three main philosophies to ensure lossless data exchange along the engineering activity and tool chain:

  - One Tool For All
  - Best of Breed
  - Integration Framework

- Requirements:
  - Different data models,
  - Different data exchange methodologies and technologies, and
  - Different software systems
- All of them with special advantages and disadvantages

System layout developed by plant planner with “Microstation”

Geometry of product developed by product designer with “NX”

Robot program developed by robot programmer with “RobCAD”

Mechanical construction by mechanical engineering with “AutoCAD”

Mechanical construction by electrical engineering with “EPLAN”

PLC code by software engineer with “TIA Portal”

Virtual commissioning with “RobSim”

Tool chains

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

Integration framework

Covers all engineering phases and engineering disciplines

Data are stored consistently in a central data base

Advantages: high degree of guarantee of consistency, sound basement for project management

Disadvantages: high tool complexity, high learning curve, difficult to adapt to changing engineering process needs

Tool chain

Tools are connected by common data exchange formats

Enables “best-of-bread” tool application strategy depending on engineering process needs

Advantages: easy to adapt to changing engineering process needs, optimal tool selection possible

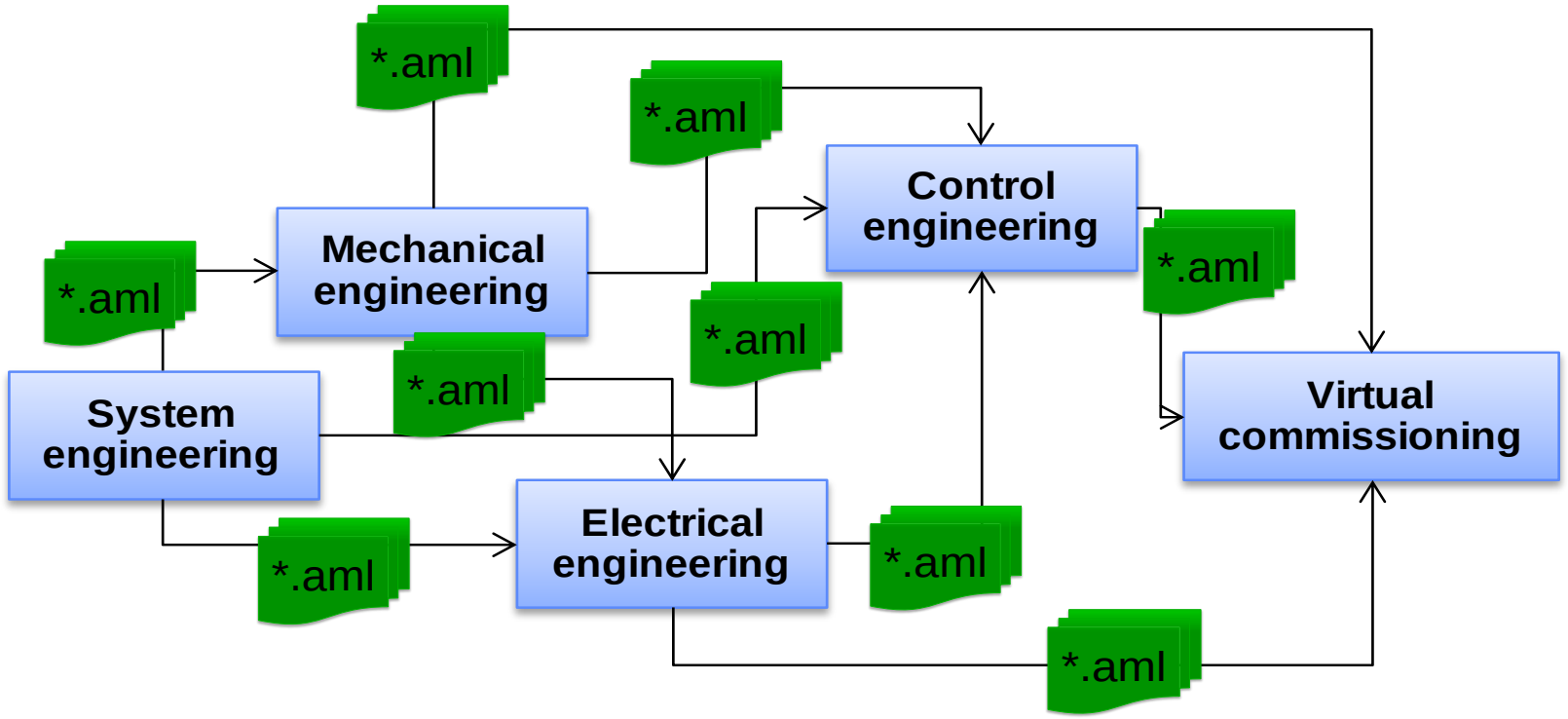
Disadvantages: Adaptation may lead to high adaptation costs, possible data lost between tools

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

Tool chains

- Usually applied within engineering projects of SMEs and/or projects with more than one company involved
- Existing engineering tools are combined via bilateral data exchange or via a centralized data broker

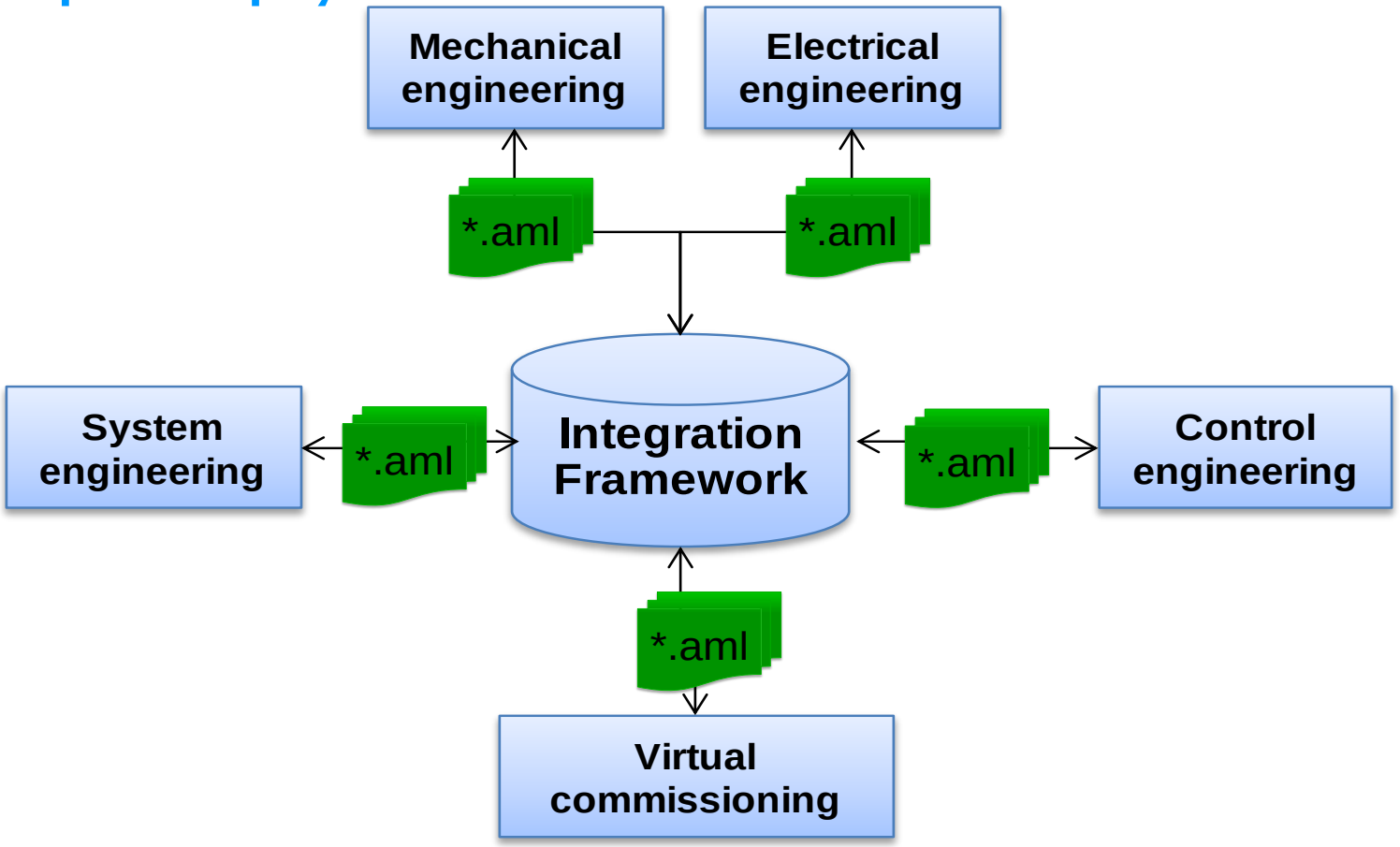


"Best of Breed" Based Engineering Network

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- Combination of existing engineering tools like for “Best of Breed” philosophy



“Integration Framework” Based Engineering Network

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- To enable the necessary data exchange between the possibly changing engineering tools standardized data exchange formats like AutomationML and STEP might be preferable.
- Have to
  - Cover possibly all but at least most of the information required and/or
  - Be created within the engineering process of production systems
- Set of (sometimes contradicting) requirements to be fulfilled for such data exchange formats:
  - Data format adaptable to different application cases and flexible with respect to extensions and changes
  - Efficient data representation
  - Human readable data representation
  - Data representation based on international standards

➔ Lead to an XML based data format

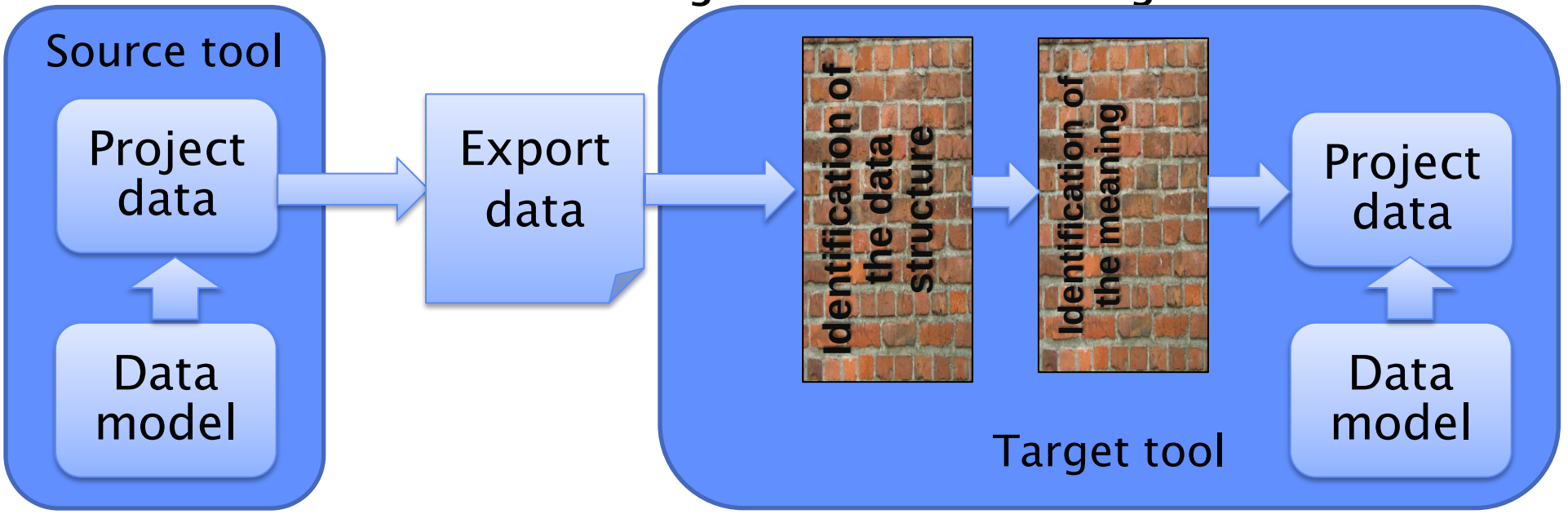
OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

Data formats

- **Requirements for data exchange between engineering tools:**
  - Two sets of levels of standardisation
    - » Syntax levels
    - » Semantic levels
- **Syntax levels**
  - Correct technical representation of the data objects within the data exchange format
  - Providing the vocabulary of the data exchange
- **Semantic levels**
  - Interpretation of data objects
    - » i.e. their meaning within the conceptualisation of objects within the engineering tool chain

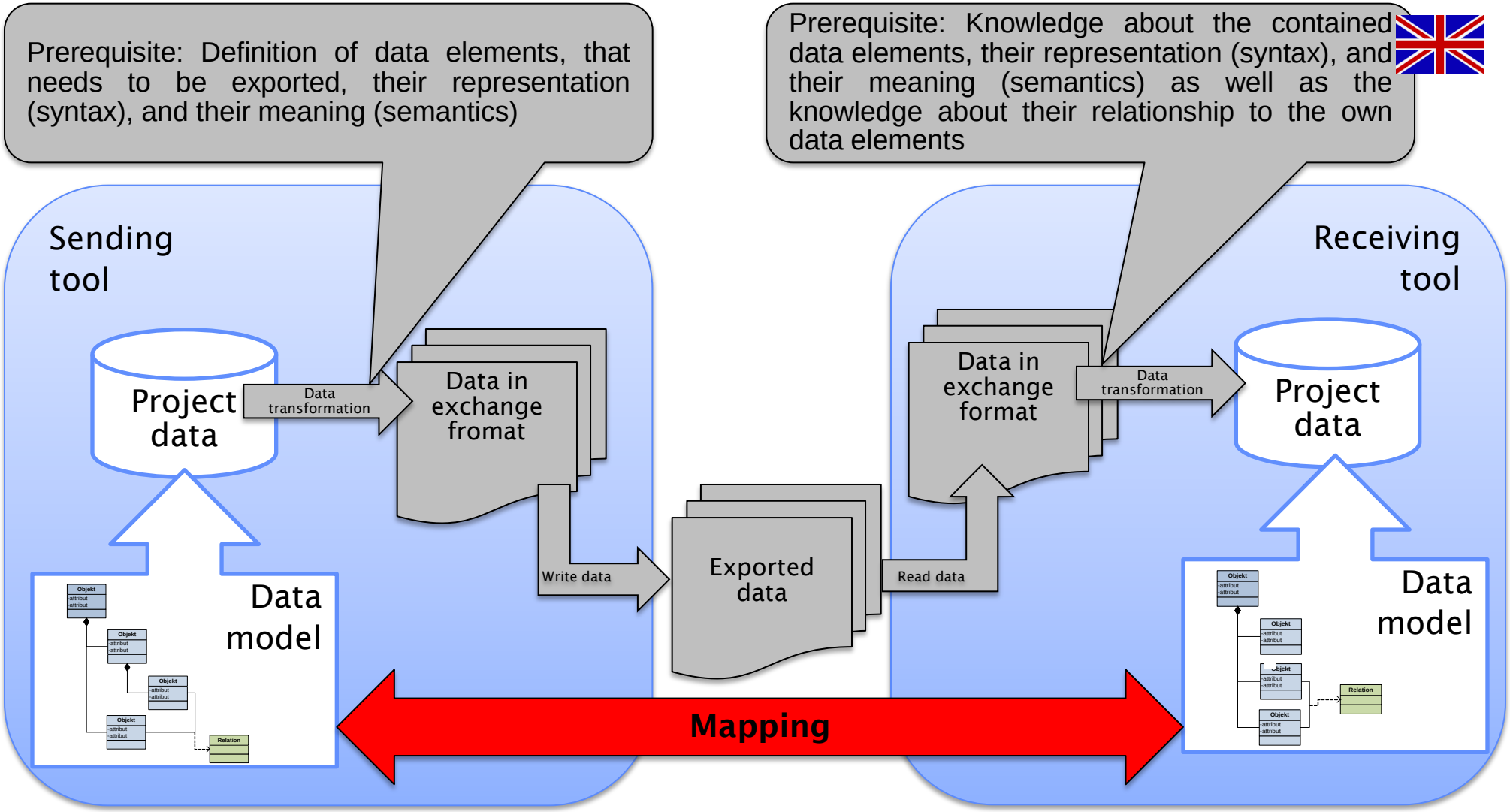
- **Data exchange process between two different tools requires two logical steps:**
  - **Identification of data structure in accordance with the syntax of the data**
  - **Identification of the meaning of the data according to the semantics**



  - **Both steps must be supported by the exporter as well as by the importer**

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB
apl. Prof. Dr.- Ing. habil. A. Lüder

Data exchange process

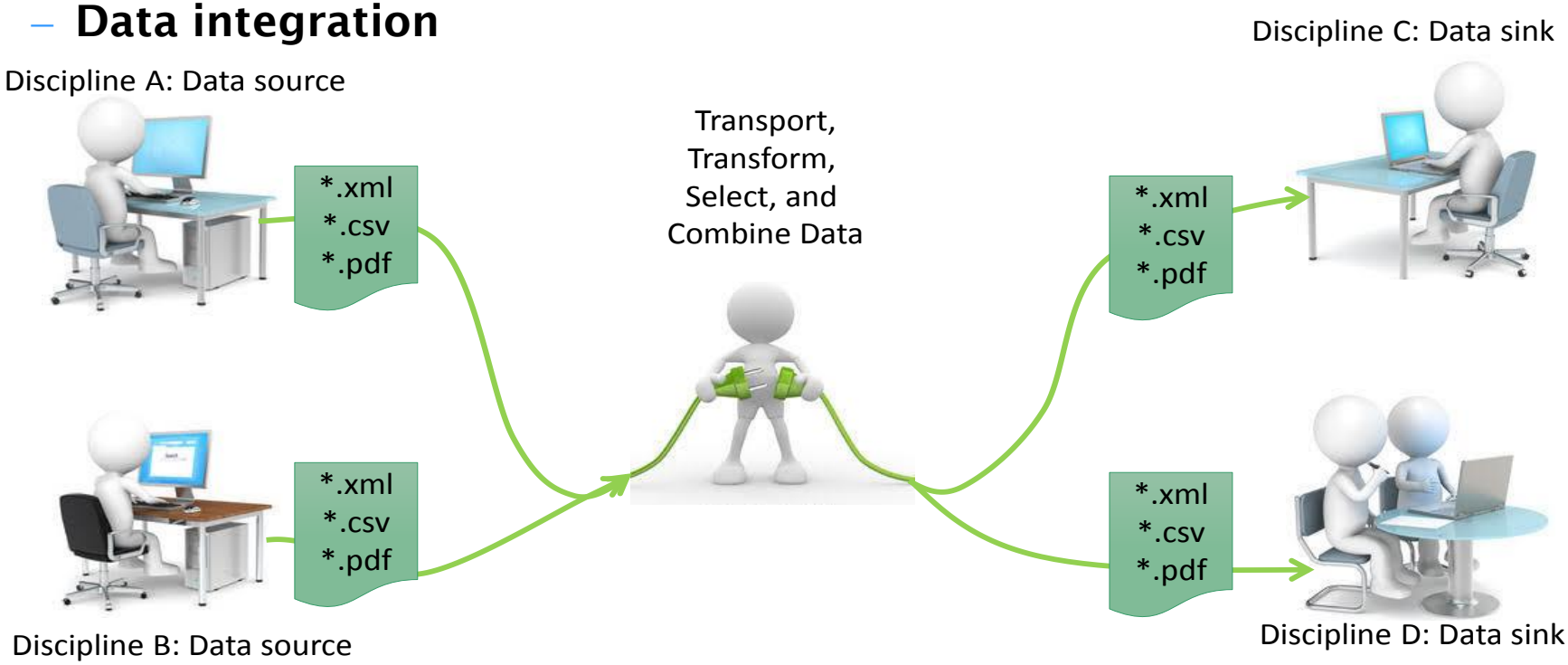


Data exchange process

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- **Data exchange process between more than two different tools requires beyond syntax and semantics:**
  - **Identification of identical objects**
  - **Data integration**

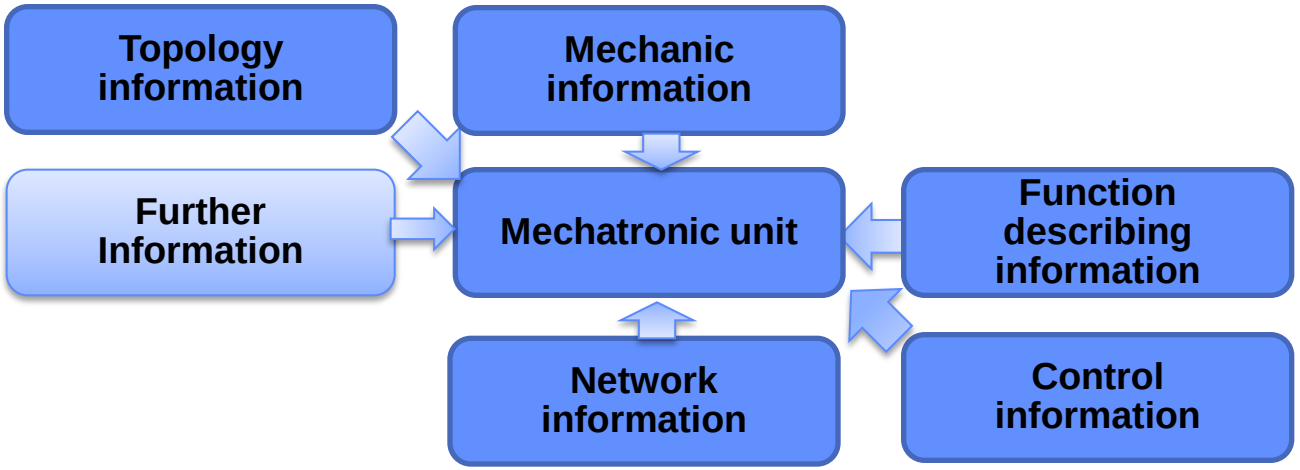


  - **Both must be supported by the data logistic system**

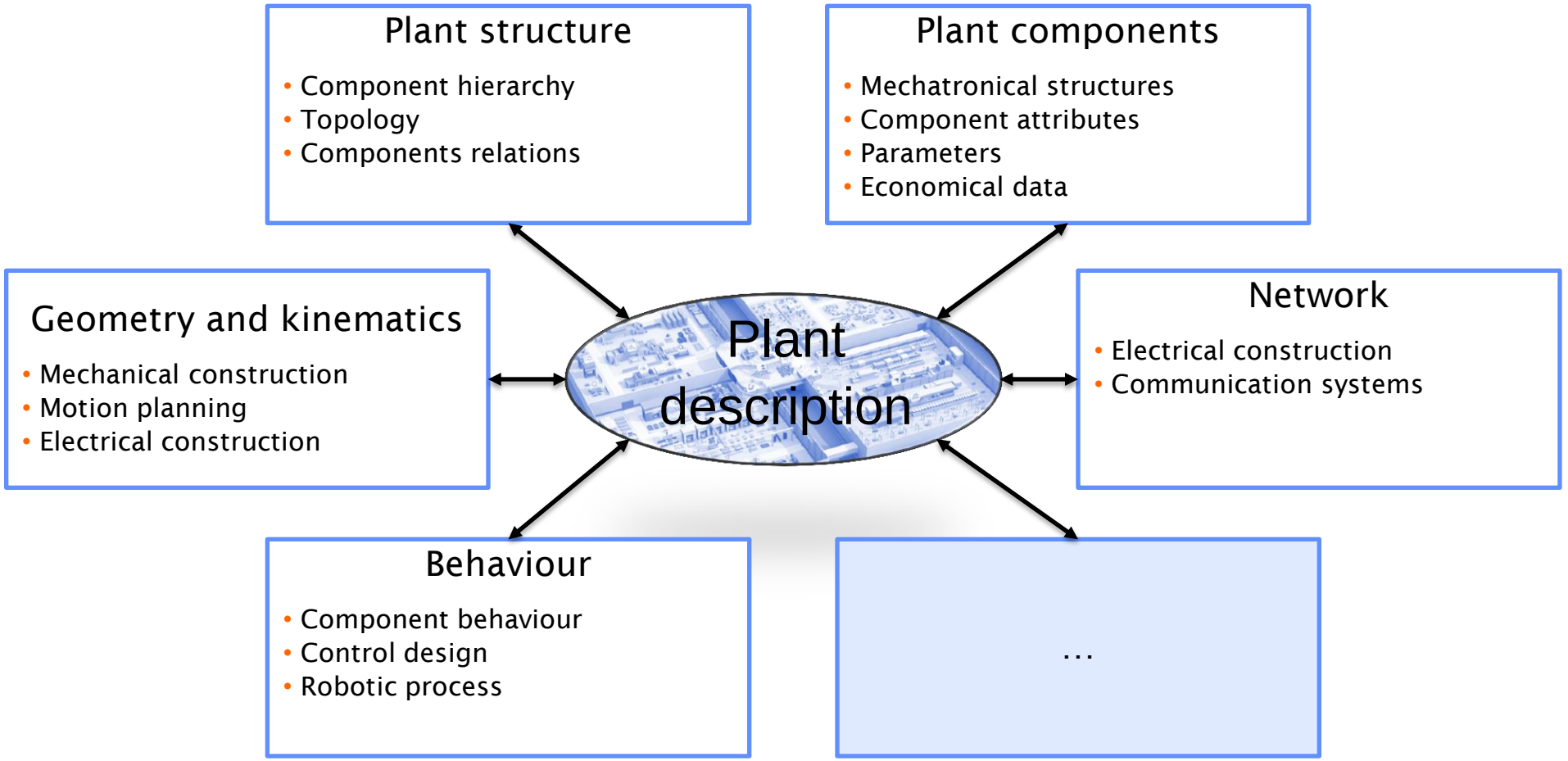
Requirements to the data exchange process

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB
apl. Prof. Dr.- Ing. habil. A. Lüder

- **AutomationML**
  - Has been developed in response to the data exchange problem within a development effort of different research entities and companies
  - Is a combination of different XML based data formats following the principles of human readability and the overall is more than the sum of its parts
  - Is developed as transient format, i.e. data are delivered and deleted
  - Enables the modelling of information related to:



Which data contents are covered by AutomationML?

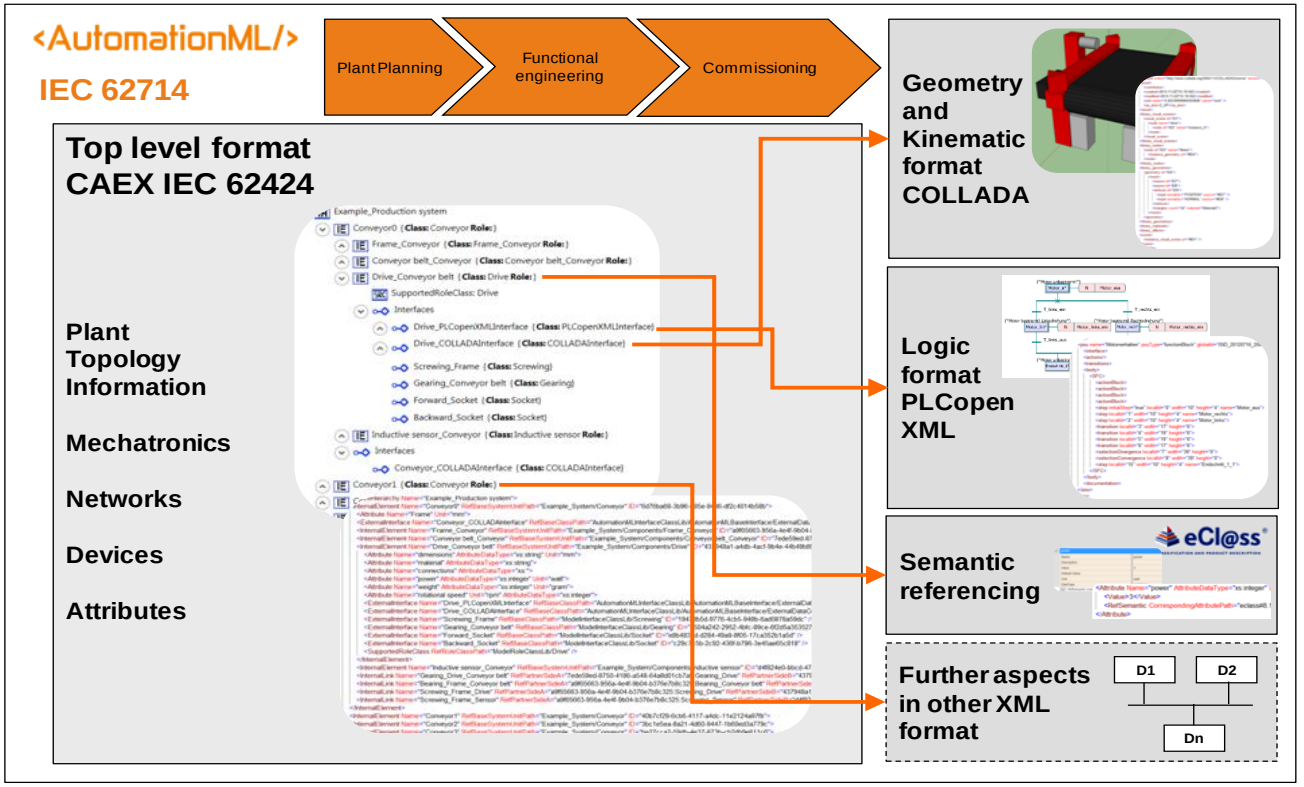


OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

Objectives of AutomationML

- Standardized in IEC 62714
- Object oriented structuring of engineering data
- Utilizes and integrates existing data exchange formats



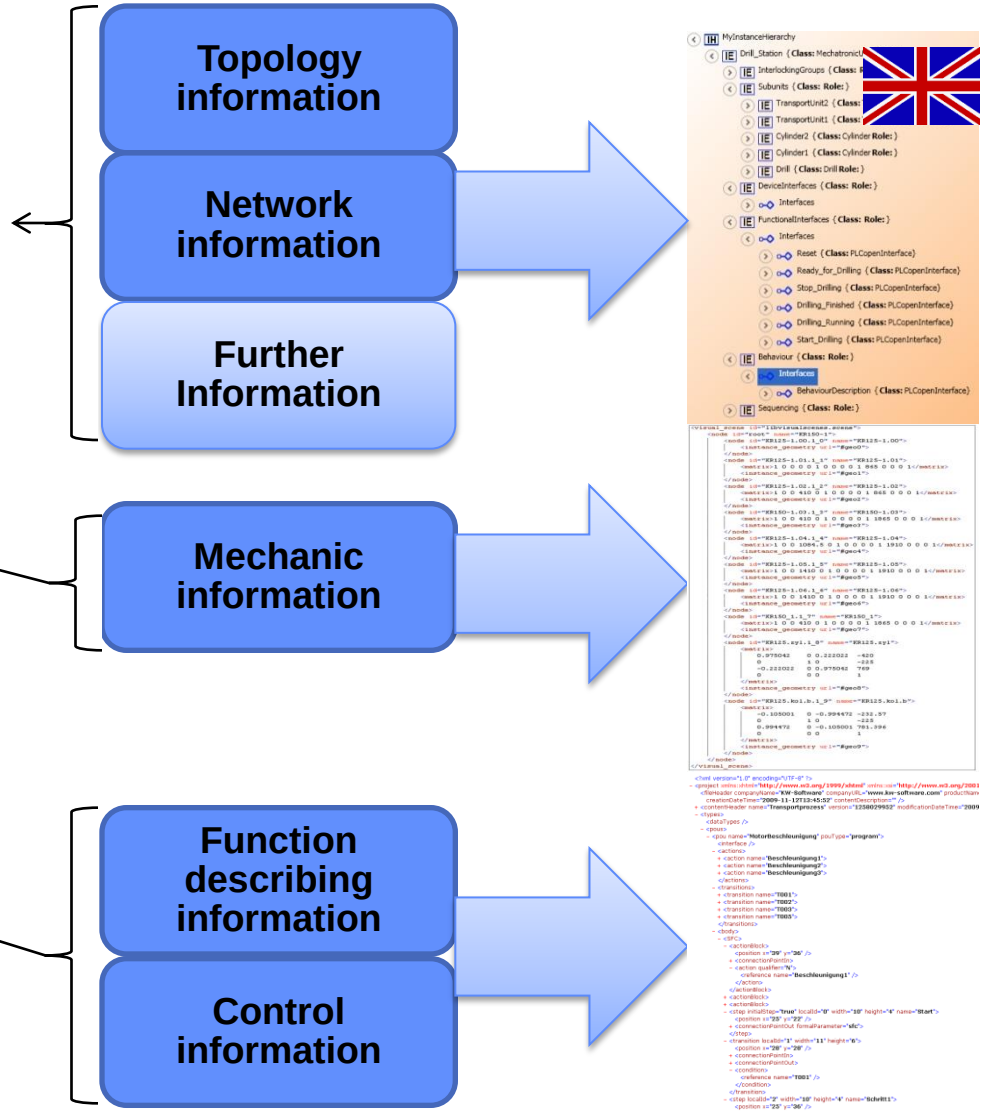
OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

AutomationML Basics

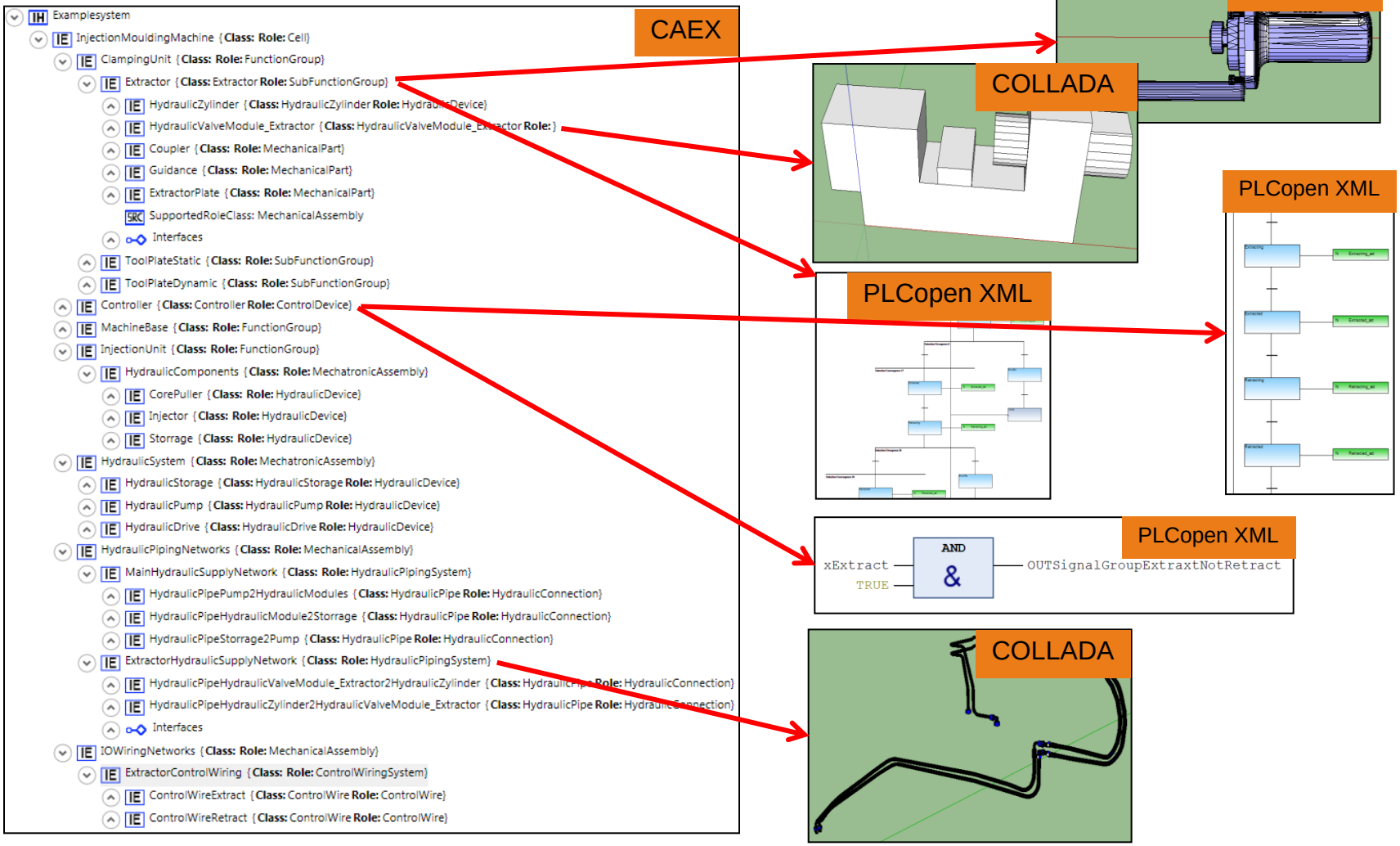
AutomationML combines the data formats

- CAEX (IEC 62424) to describe system hierarchies as well as attributes for system elements and devices
- COLLADA (Standard of KHRONOS Group) to describe geometry und kinematic information
- PLCopen XML (Standard of PLCopen for modelling of IEC 61131 projects) for behavior information modelling



OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder



OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

AutomationML Basics

- **AutomationML is**
  - A data format, that allows a vendor-independent data exchange of engineering data of production systems.
  - A storage format for information.
  - Applicable within the entire engineering process by connecting different discipline specific engineering tools.
  - Object orientated and allows the modelling of production system components as data objects aggregating different aspects.
  - A combination and adaptation of already existing industry formats that were developed for exchange and storage of different engineering aspects.
  - A consistent and distributed document architecture, that enables the handling of large data sets and the outsourcing of libraries to external documents.

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

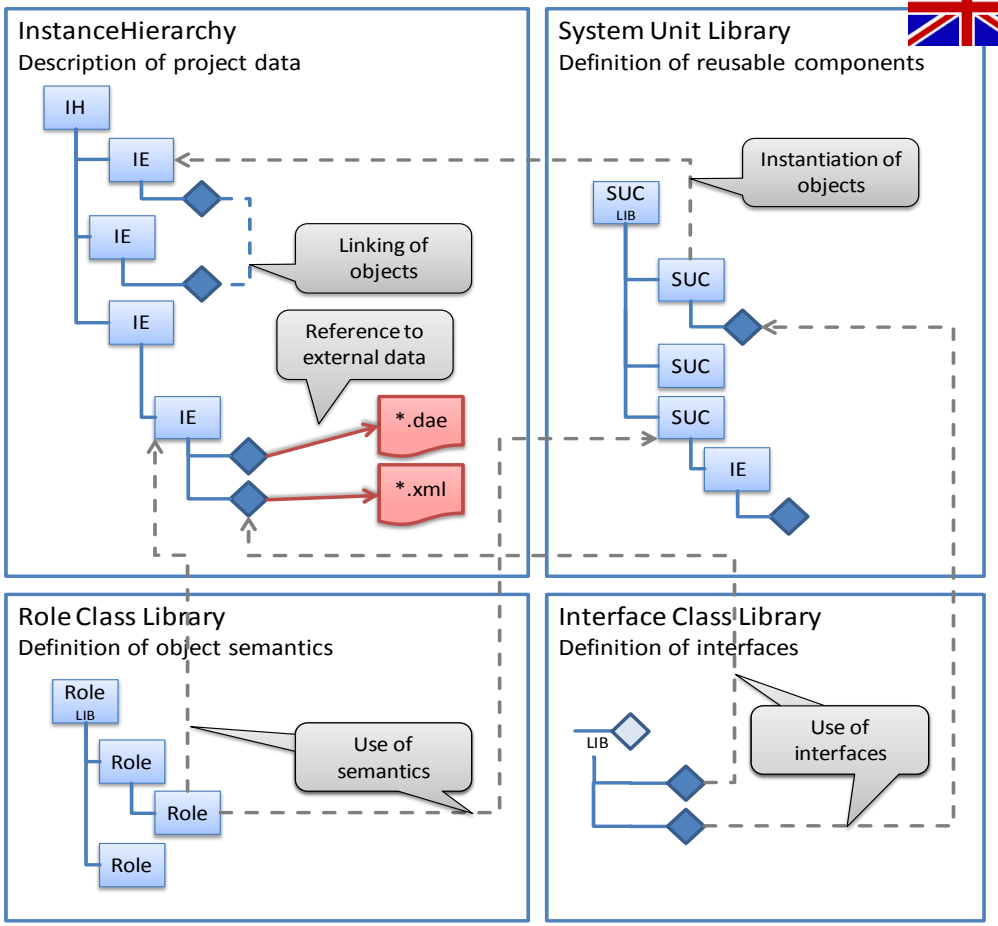
apl. Prof. Dr.- Ing. habil. A. Lüder

- **AutomationML is not**
  - Tool functionality.
  - Capable to review conditions, attribute values, relations, references, or semantic correctness of data objects.
  - Capable to check consistence or to review and match version of data objects.
  - Capable of automatic standardization of user specific information.
  - Capable of automatic creation of libraries.
  - Capable of automatic management of versions and variants.
  - A project management tool.
  - A project management database.

➔ **But it allows the storage of all data required for that.**

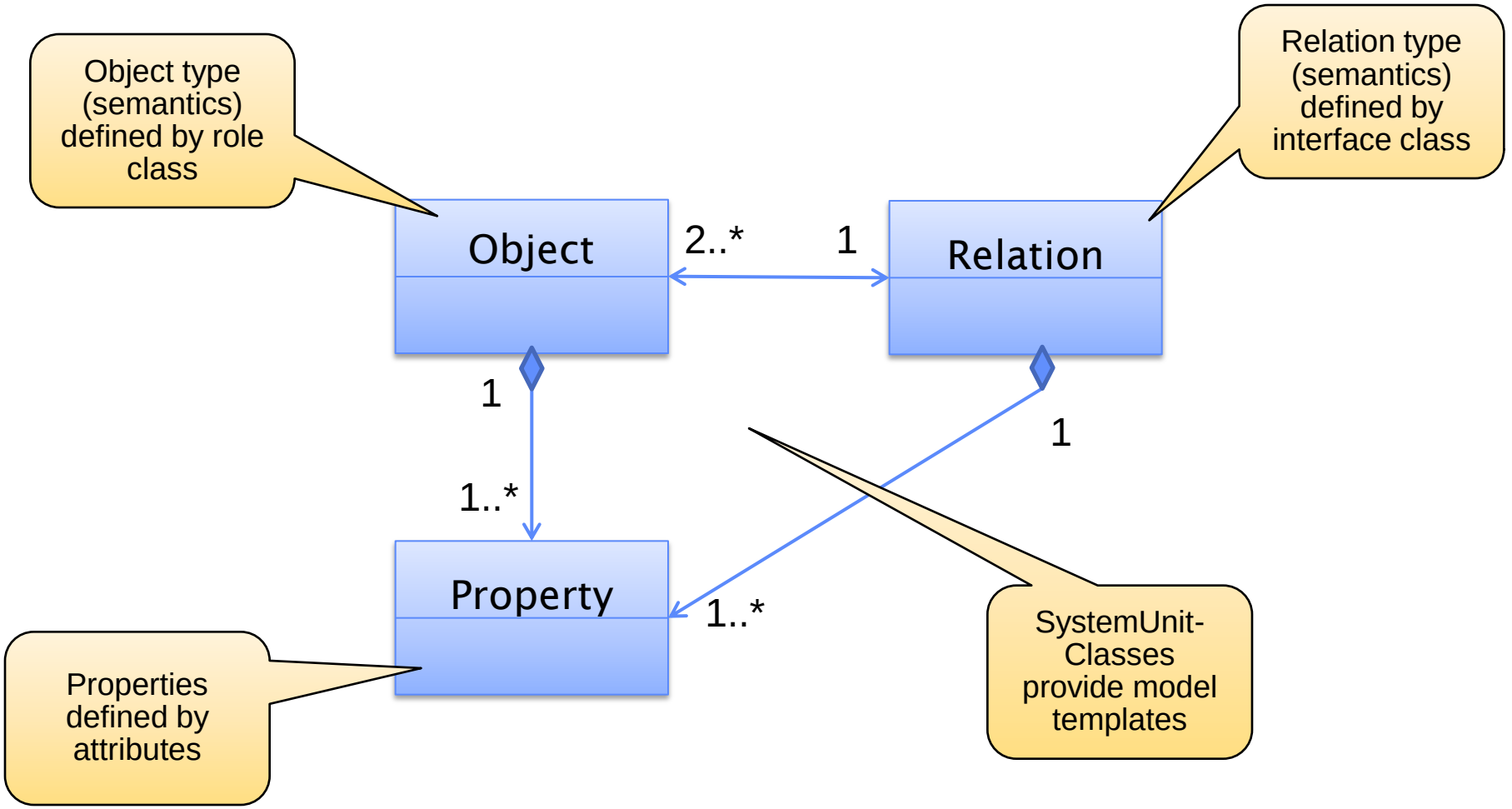
apl. Prof. Dr.- Ing. habil. A. Lüder

- Use of library concepts
- Specification of object semantics using role classes
- Specification of interface semantics using interface classes
- Specification of reusable objects using system unit classes
- Modular model structure



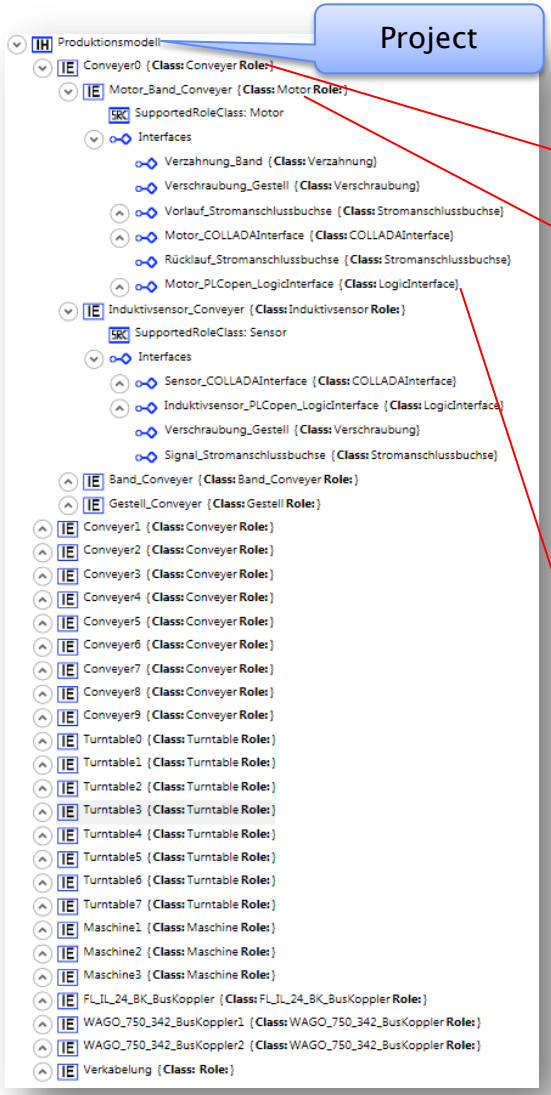
OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

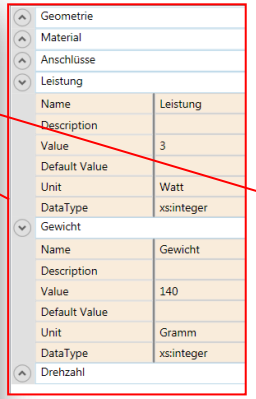
apl. Prof. Dr.- Ing. habil. A. Lüder

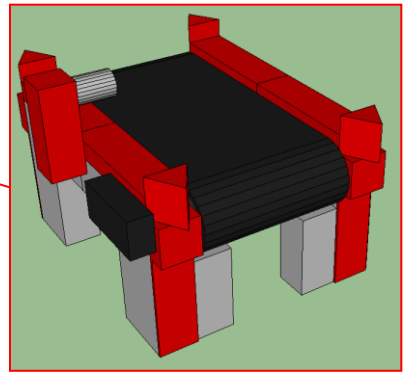


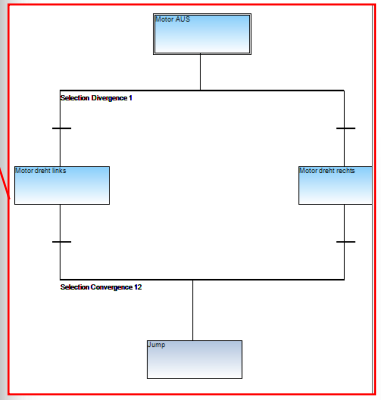
OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

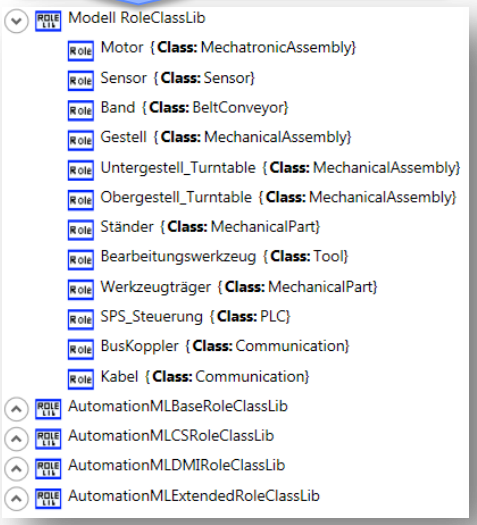




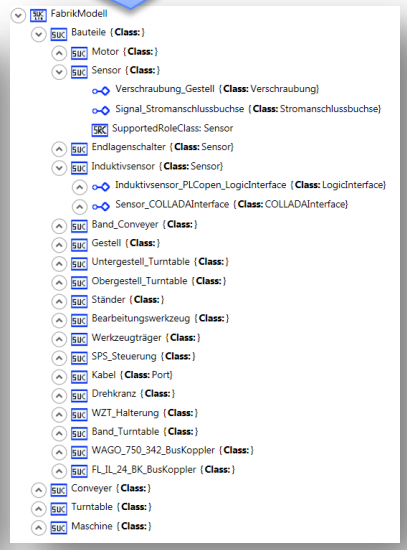




Defined roles as sematic representation



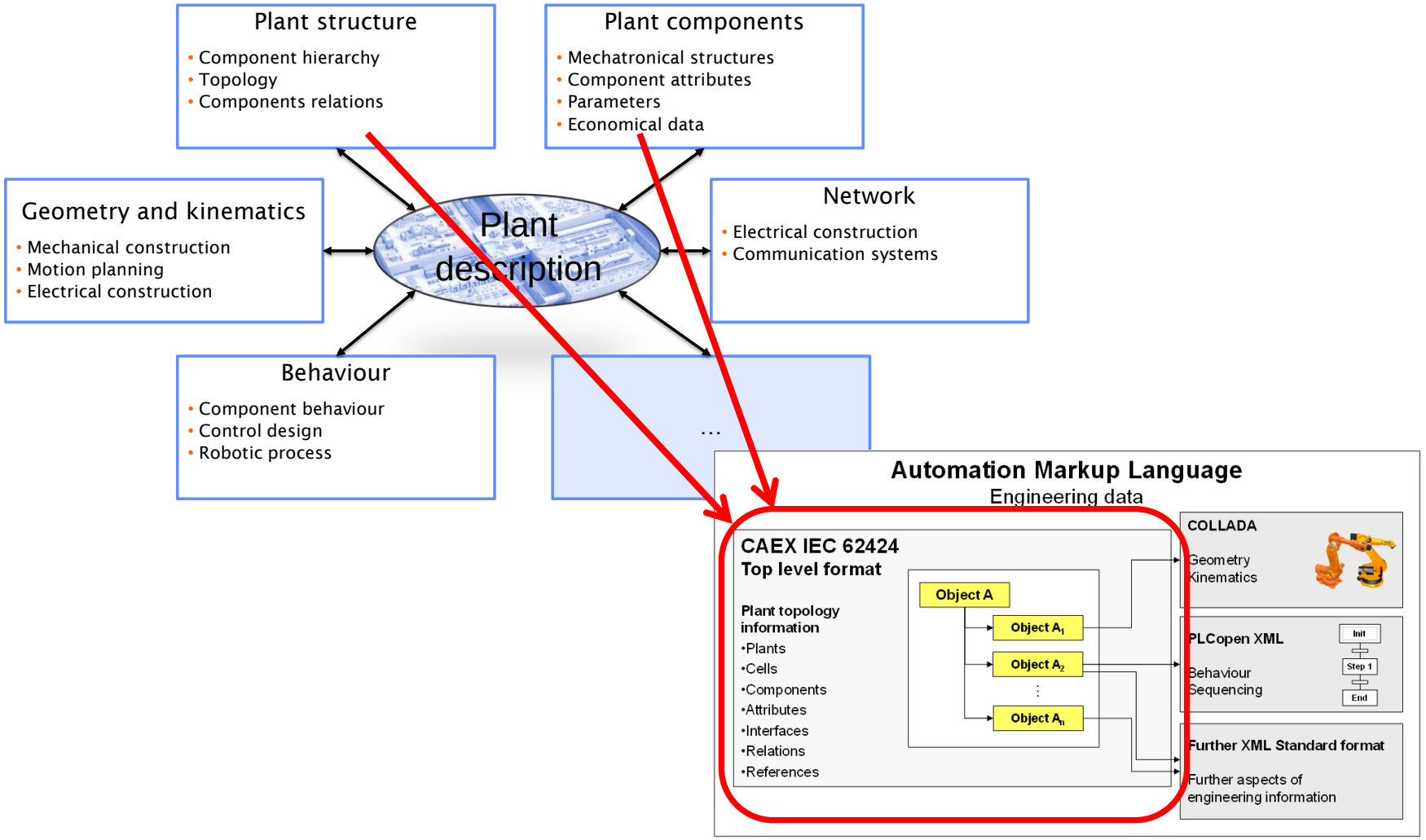
Component library



OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

AutomationML Basics

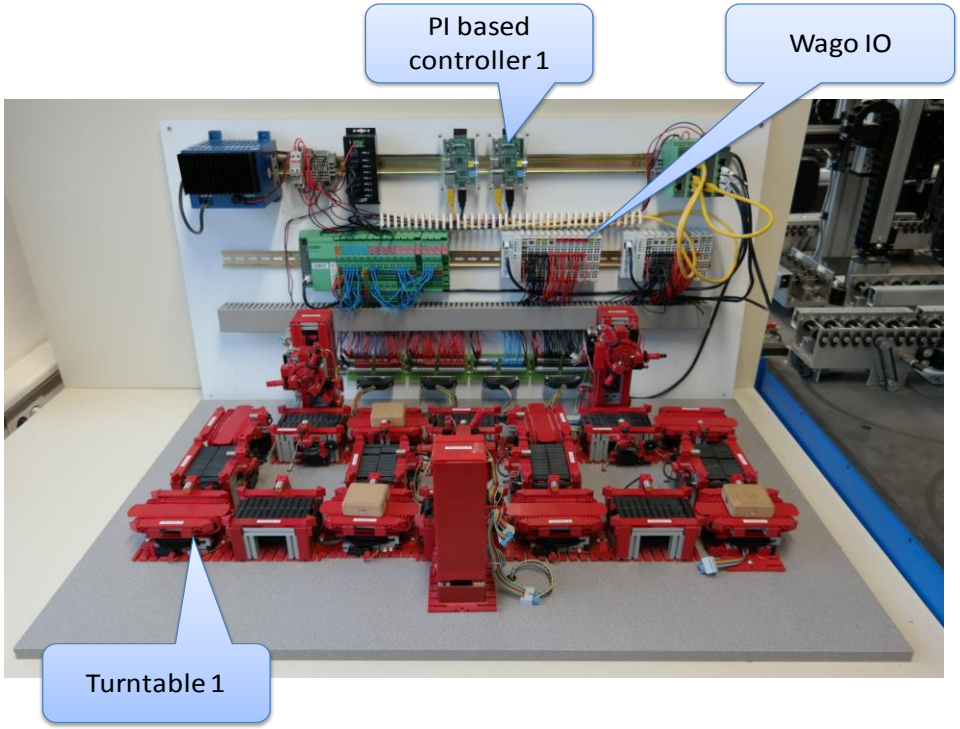


OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

AutomationML Basics

- Modelling of the identified information
- Running example used for a better understanding
  - Consisting set of multipurpose machines, turntables, and conveyers
  - Wired using Field IOs to Raspberry Pi based controllers



OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB
apl. Prof. Dr.- Ing. habil. A. Lüder

Running Example

- AutomationML exploits CAEX for modelling the system topology and the system elements

➔Four main modelling means:

  - Role class
  - Interface class
  - SystemUnitClass
  - InstanceHierarchy

- The modelling means will be described in the following

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

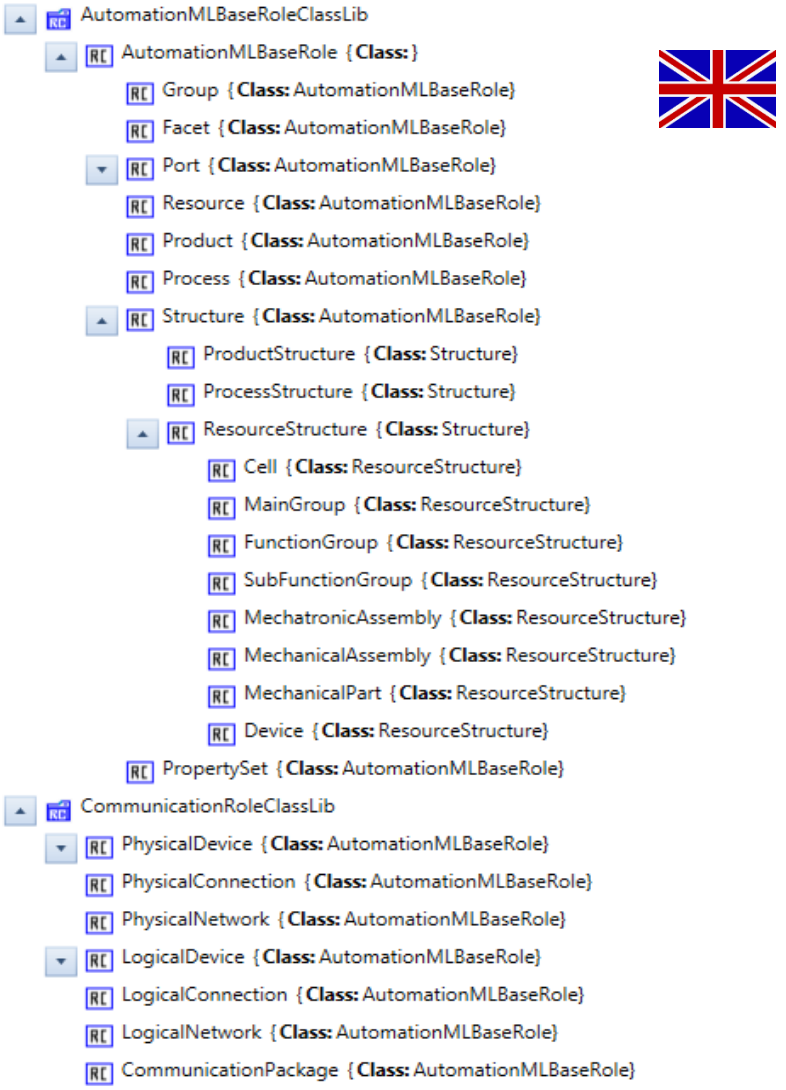
apl. Prof. Dr.- Ing. habil. A. Lüder

- **Role class**
  - Collected within role class libraries
  - Describing abstract functionality without defining the underlying technical implementation
    - ➔As an indicator for the semantics of an object
- **Example:**
  - Classes MechanicalPart and Device indicating system structure semantics
  - LogisticalDevice and PhysicalDevice representing communication system semantics
- **Set of basic role defined by AutomationML**
  - AutomationMLBaseRoleClassLib: fundamental role classes defined in Part 1 of the AutomationML standard
  - CommunicationRoleClassLib defined in the Part 5

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- **AutomationML BaseRoleClass Library and CommunicationRoleClass Library**



OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

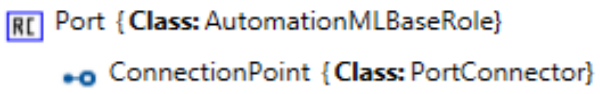
System Topology and System Element Modelling – Role class

- Further role classes defined in Part 2 of the AutomationML standard
- Possible definition of new role classes by every AutomationML user following its use cases and needs for data exchange
- Only some rules for role class definition:
  - Each role class
    - » Shall have an unique name within the role tree of a role class library
      - Unique reference by this hierarchy path
    - » Derived directly or indirectly from AutomationMLBaseRole by using the RefBaseClassPath attribute
    - » May have attributes and interfaces
      - Enabling an importer of an engineering tool to interpret and process incoming information correctly

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- **Port Role Class as Example of Role Definition**
- **Identification path: AutomationMLBaseRoleClassLib/ AutomationMLBaseRole/Port**



Direction	
Name	Direction
Description	
Value	
Default Value	
Unit	
DataType	xs:string
Cardinality	
Name	Cardinality
Description	
Value	
Default Value	
Unit	
DataType	xs:string
MinOccur	
MaxOccur	
Category	
Name	Category
Description	
Value	
Default Value	
Unit	
DataType	xs:string

```
<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="1c6a2bb9-8f93-4394-8fae-ef0e0074716a" RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/PortConnector" />
</RoleClass>
```


OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

System Topology and System Element Modelling – Role class

- **Example of an user defined role class: ModbusTCPPhysicalDevice role class**
  - **With the attributes MACaddress and IPAddress**
    - » **As defined in the running example**
  - **Representing a control device able to communicate over Modbus TCP**

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- **Interface class**
  - Describing an abstract relation between an element to other elements or to information not covered within the CAEX based model
- **Example:**
  - SignalInterface and PhysicalEndPoint indicating provided interfaces for signal processing of cable plugging
  - ExternalDataConnector representing the association to externally stored information
- **Set of basic interfaces defined by AutomationML**
  - AutomationMLInterfaceClassLib with fundamental interfaces defined in Part 1 of the AutomationML standard
  - CommunicationInterfaceClassLib defined in the upcoming Part 5

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- Possible definition of new interface classes by every AutomationML user following its use cases and needs for data exchange
- Only some rules for interface class:
  - Each interface class
    - » Shall have a unique name within the interface class tree of an interface class library
      - Uniquely reference by this hierarchy path
    - » Derived directly or indirectly from AutomationMLBaseInterface by using the RefBaseClassPath attribute
    - » May have attributes
- Attributes
  - Used and filled with values in each occurrence of an instance of the interface class

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

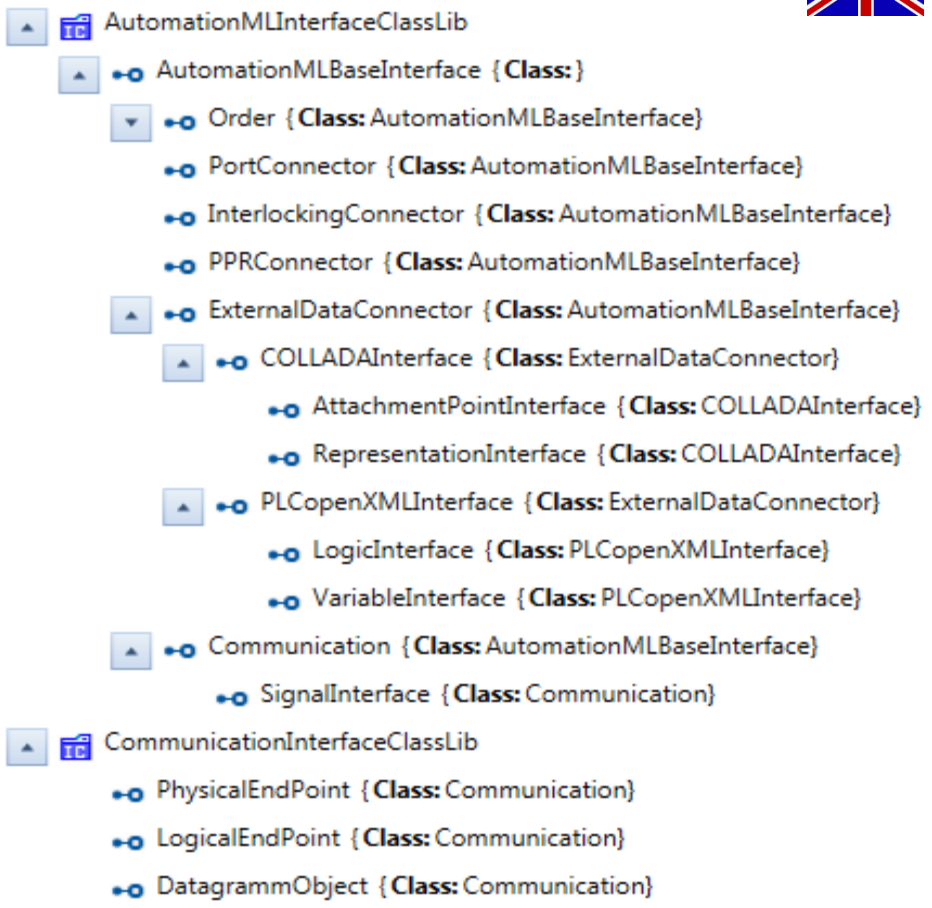
apl. Prof. Dr.- Ing. habil. A. Lüder

- **Example of a user defined interface class: ModbusTCPSocket interface class**
  - **As defined in the running example**
  - **Representing the plugging position for an Ethernet cable within a control device able to communicate over Modbus TCP**

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- **AutomationML InterfaceClass Library and CommunicationInterfaceClass Library**

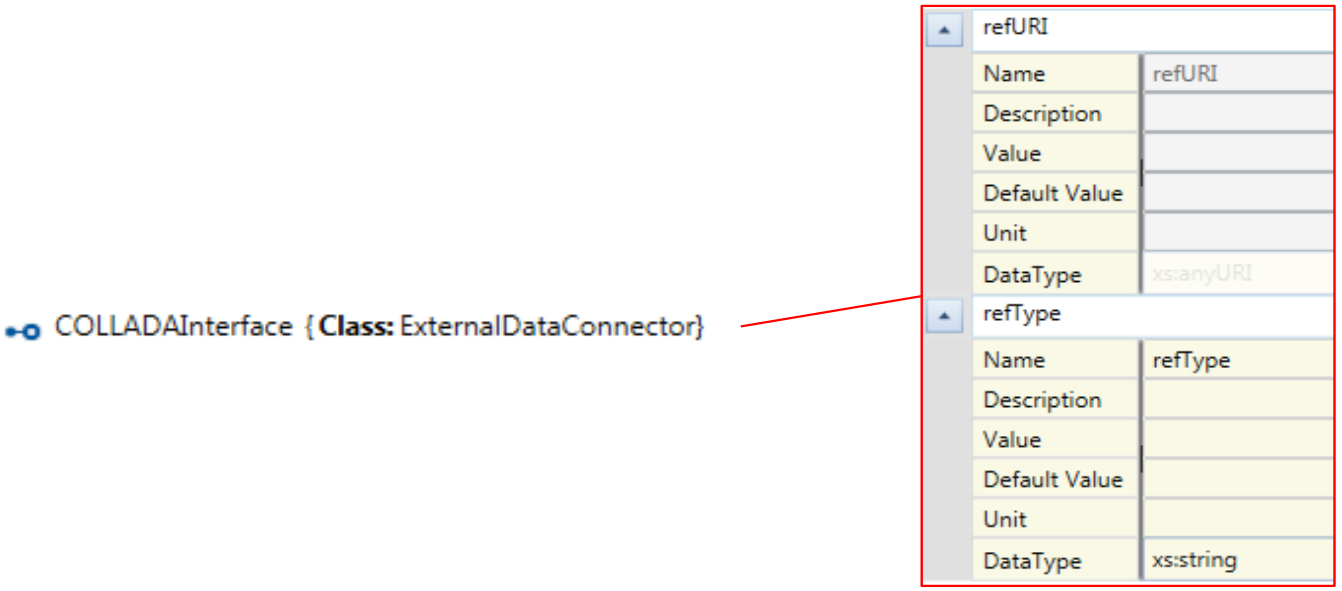


OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

System Topology and System Element Modelling – Interface class

- COLLADAInterface Class as Example for Interface Definition
- Identification path: AutomationMLInterfaceClassLib/AutomationMLBaseInterface/ExternalDataConnector/COLLADAInterface




```
<InterfaceClass Name="COLLADAInterface" RefBaseClassPath="ExternalDataConnector">
    <Attribute Name="refType" AttributeDataType="xs:string" />
</InterfaceClass>
```


System Topology and System Element Modelling – Interface class

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- SystemUnitClasses
  - As reusable system components or
  - As templates for system modelling depending on the point of view
  - Reflection of
    - » Either a vendor dependent library of components or devices or
    - » A set of templates used within an engineering tool to structure discipline dependent model information
- No basic AutomationML SystemUnitClass library within the AutomationML standard
- Definition up to the user of AutomationML

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- **Only some rules for SystemUnitClass definition:**
  - **Each SystemUnitClass**
    - **Shall have a unique name similar to role classes and interface classes**
    - **Shall have at least one role class assigned to it giving the SystemUnitClass a semantic by using the SupportedRoleClass sub-element**
    - **May have sub-objects of the type InternalElement, attributes, and interfaces representing the structure of the modelled class of objects, its properties, and its possible associations**
    - **May be derived from another SystemUnitClass by using the RefBaseClassPath attribute**
      - **Inheritance of all supported role classes, sub-elements, interfaces, and attributes from the parent element**

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

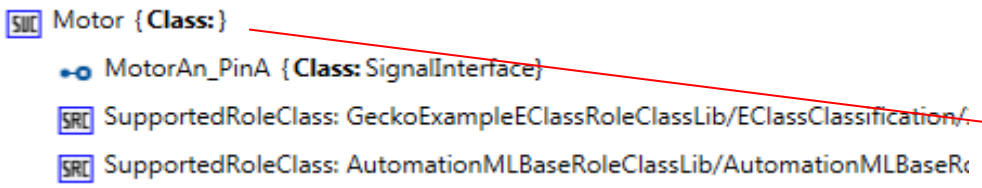
apl. Prof. Dr.- Ing. habil. A. Lüder

Example SystemUnitClass Library

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

Motor SystemUnitClass as Example for SystemUnitClass Definition



eClassReferenceFeatureGroupID2	
eClassReferenceFeatureGroupID	
eClassVersion	
Hersteller-Name	
Herstellerartikelnummer	
Min. Drehzahl	
Name	Min. Drehzahl
Description	Kleinstmögliche Drehzahl, bei der ein Motor im thermisch zulässigen
Value	5800
Default Value	
Unit	1/s
DataType	xs:integer
Nenndrehzahl	
Name	Nenndrehzahl
Description	Drehzahl die den Nenndaten des verwendeten Motors entspricht
Value	8746
Default Value	
Unit	1/s
DataType	xs:float

```
<SystemUnitClass Name="Motor">
  <Attribute Name="eClassReferenceFeatureGroupID2" AttributeDataType="xs:string" />
  <Attribute Name="eClassReferenceFeatureGroupID" AttributeDataType="xs:string">
  <Attribute Name="eClassVersion" AttributeDataType="xs:string">
  <Attribute Name="Hersteller-Name" AttributeDataType="xs:string">
  <Attribute Name="Herstellerartikelnummer" AttributeDataType="xs:string">
  <Attribute Name="Min. Drehzahl" AttributeDataType="xs:integer" Unit="1/s">
    <Description>Kleinstmögliche Drehzahl, bei der ein Motor im thermisch zulässigen Bemessungsbetriebspunkt dauernd betrieben werden kann</Description>
    <Value>5800</Value>
  </Attribute>
  <Attribute Name="Nenndrehzahl" AttributeDataType="xs:float" Unit="1/s">
    <Description> Drehzahl die den Nenndaten des verwendeten Motors entspricht</Description>
    <Value>8746</Value>
  </Attribute>
  <ExternalInterface Name="MotorAn_PinA" RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Communication/SignalInterface" ID="923608d1-6cbb-448b-b739-ff85d11e1958" />
  <SupportedRoleClass RefRoleClassPath="GeckoExampleEClassRoleClassLib/EClassClassification/27 Elektro-, Automatisierungs- und Prozessleittechnik/27-02 Elektrischer Antrieb/27-02-25 DC-Motor/27-02-25-01 DC-Motor (IEC)" />
  <SupportedRoleClass RefRoleClassPath="AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Structure/ResourceStructure/Device" />
</SystemUnitClass>
```


OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- All modelling concepts may have attributes
- Attributes:
  - Seen as properties
  - Possible assignment to role classes, interface classes, SystemUnitClasses, and internal elements
- Some rules for attribute definition:
  - Each attribute:
    - » Shall have a unique name within its parent element
    - » May have a DataType and a Unit attribute and sub-elements for description, default value, value, and semantic referencing

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

Herstellerartikelnummer (vendor article number) Attribute as Example for Attribute Definition

Herstellerartikelnummer	
Name	Herstellerartikelnummer
Description	eindeutiger Produktschlüssel des Herstellers
Value	35481
Default Value	
Unit	
DataType	xs:string
RefSemantic: ECLASS:0173-1#02-AAO676#002	

```
<Attribute Name="Herstellerartikelnummer" AttributeDataType="xs:string">
    <Description>eindeutiger Produktschlüssel des Herstellers</Description>
    <Value>35481</Value>
    <RefSemantic CorrespondingAttributePath="ECLASS:0173-1#02-AAO676#002" />
</Attribute>
```


OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

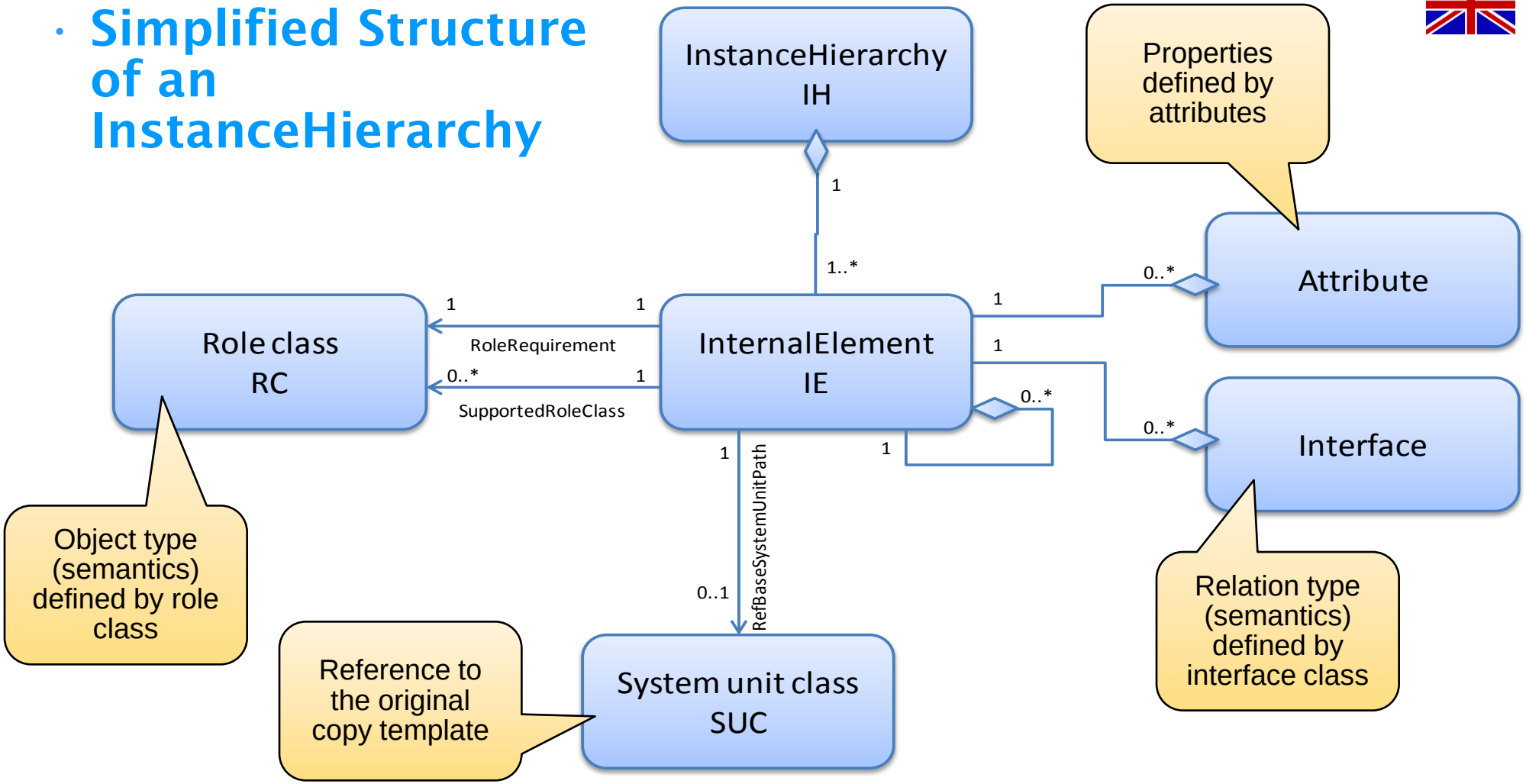
apl. Prof. Dr.- Ing. habil. A. Lüder

- **InstanceHierarchy**
  - With its integrated hierarchy of InternalElements
  - Representation of actual engineering data to be modelled by CAEX following an object oriented and hierarchical structure
- **InternalElement**
  - Workhorse of the representation of actual engineering data
  - Representative of an object in the production system to be modelled
  - Possible representation of physical components like the complete plant, functional component as machines and turntables, a device as a drive or a controller, or just a mechanical part as a conveyer belt or a wire
  - Possible representation of logical components like a PLC program, a product description, or an order

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

Simplified Structure of an InstanceHierarchy



OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- **InternalElements**
  - Generally user defined
  - Containing attributes and interface instances derived from interface classes of any interface class library
  - Able to reference a SystemUnitClass from an arbitrary SystemUnitClass library by using the RefBaseSystemUnitPath attribute
    - » This reference
      - For identification of the corresponding SystemUnitClass as the parent class the InternalElement is derived from
      - For naming the SystemUnitClass as template for the InternalElement

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

➔InternalElement

- Shall have the same substructure, interfaces, and attributes as defined in the SystemUnitClass
- Shall reference at least one (but possible more than one) role class from an arbitrary role class library
  - ➔RoleRequirements and SupportedRoleClass sub-objects shall be applied
  - ➔Definition of the semantics of the InternalElement by the referenced role class

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

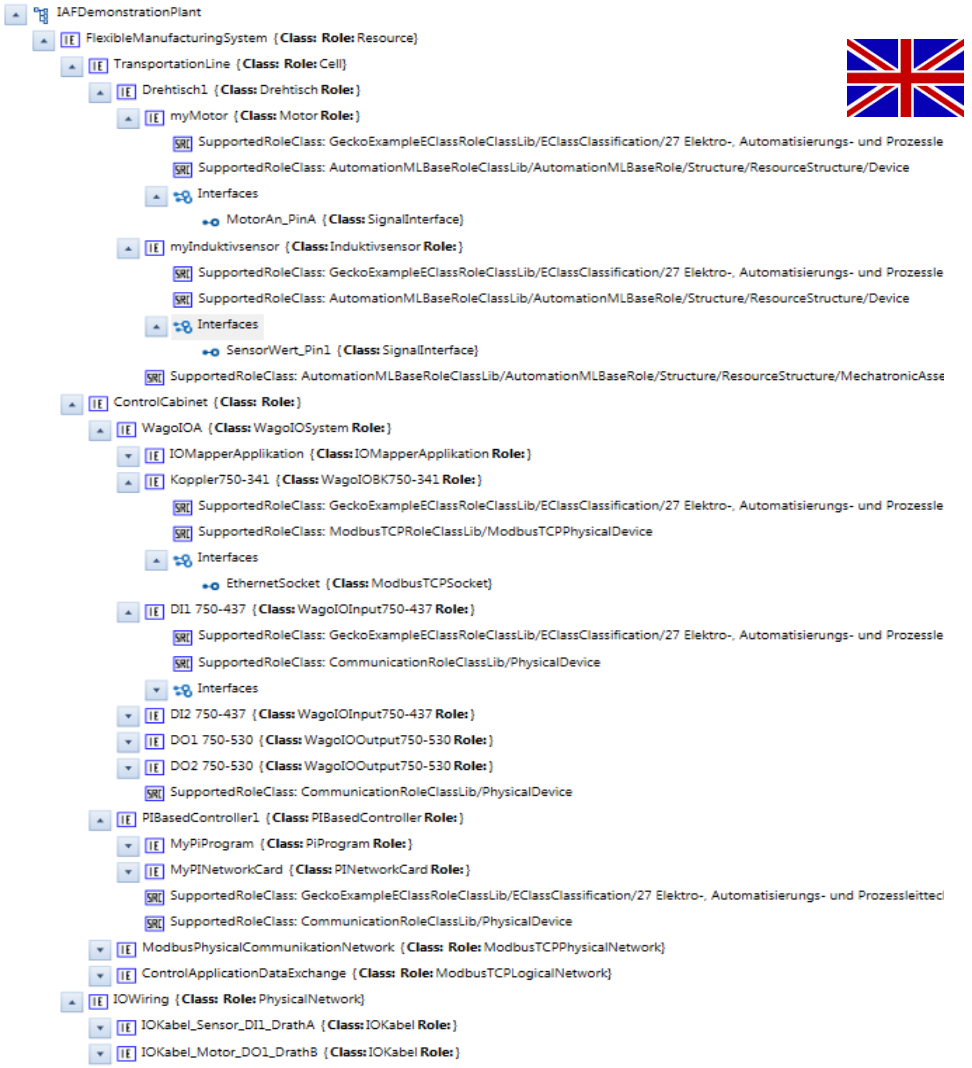
- **Example: InstanceHierarchy modelling the running example**

  - Hierarchy of physical and logical entities ranging from the highest InternalElement FlexibleManufacturingSystem over InternalElements representing
    - » Turntable (Drehtisch1)
    - » IO fieldbus coupler (WagoIOA) and
    - » Controller (PIBasedControllerA)
  - Down to InternalElements representing
    - » Control application (MyPIProgram)
    - » Control devices (myMotor) or
    - » Wires (IOKabel_Motor_DO1_DrathB)

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

InstanceHierarchy of the Example (Extract)



OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

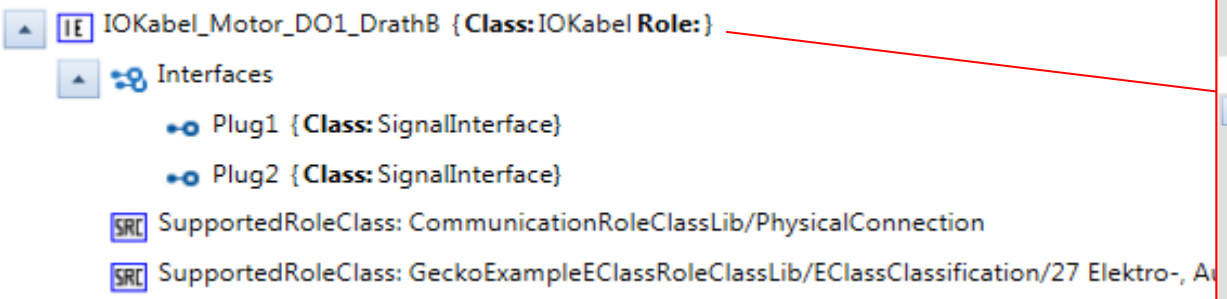
apl. Prof. Dr.- Ing. habil. A. Lüder

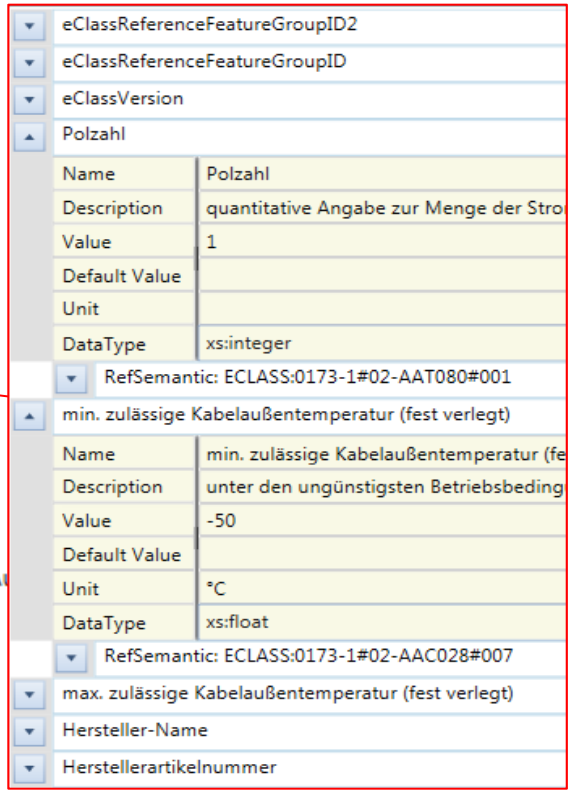
- **Example InternalElement**
  - Depiction of the wire connecting the drive with the field IO
  - With several attributes like
    - Polzahl representing the number of leads in the wire
    - Min. zulässige Kabeaußentemperatur representing the minimal acceptable temperature of the wire surface
  - Two interfaces representing both end points of the wire

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

IOKabel_Motor_DO1_DrathB as Example for an InternalElement





OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

```
<InternalElement Name="IOKabel_Motor_DO1_DrathB" RefBaseSystemUnitPath="GeckoExampleSystemUnitClassLib_PhysicalCommunicationDevices/IOKabel" ID="c154d16f-1ffa-4126-b4ef-b5d3f0e02c5d">
    <Attribute Name="eClassReferenceFeatureGroupID2" AttributeDataType="xs:string" />
    <Attribute Name="eClassReferenceFeatureGroupID" AttributeDataType="xs:string">
    <Attribute Name="eClassVersion" AttributeDataType="xs:string">
    <Attribute Name="Polzahl" AttributeDataType="xs:integer">
      <Description>quantitative Angabe zur Menge der Strompfade (Pole)</Description>
      <Value>1</Value>
      <RefSemantic CorrespondingAttributePath="ECLASS:0173-1#02-AAT080#001" />
    </Attribute>
    <Attribute Name="min. zulässige Kabelaußentemperatur (fest verlegt)" AttributeDataType="xs:float" Unit="°C">
      <Description>unter den ungünstigsten Betriebsbedingungen kleinst zulässiger Grenzwert, der nie überschritten werden darf, um Personen- und/oder Sachschäden zu vermeiden, der Temperatur der außen liegender
ı der die umgebende Atmosphäre gelangen kann, wobei dieses Kabel auf bestimmte Stelle(n) angebracht und befestigt ist</Description>
      <Value>-50</Value>
      <RefSemantic CorrespondingAttributePath="ECLASS:0173-1#02-AAC028#007" />
    </Attribute>
    <Attribute Name="max. zulässige Kabelaußentemperatur (fest verlegt)" AttributeDataType="xs:float" Unit="°C">
    <Attribute Name="Hersteller-Name" AttributeDataType="xs:string">
    <Attribute Name="Herstellerartikelnummer" AttributeDataType="xs:string">
    <ExternalInterface Name="Plug1" RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Communication/SignalInterface" ID="051c2ff5-cfe7-459a-bde3-354f5e988423" />
    <ExternalInterface Name="Plug2" RefBaseClassPath="AutomationMLInterfaceClassLib/AutomationMLBaseInterface/Communication/SignalInterface" ID="be03555c-bf72-4d92-864c-3966a8a983a9" />
    <SupportedRoleClass RefRoleClassPath="CommunicationRoleClassLib/PhysicalConnection" />
    <SupportedRoleClass RefRoleClassPath="GeckoExampleEClassRoleClassLib/EClassClassification/27 Elektro-, Automatisierungs- und Prozessleittechnik/27-06 Kabel, Leitung/27-06-03 Konfektionierte Leitung/27-06-
ıierte Sensor-Aktor-Leitung" />
</InternalElement>
```


System Topology and System Element Modelling – InstanceHierarchy

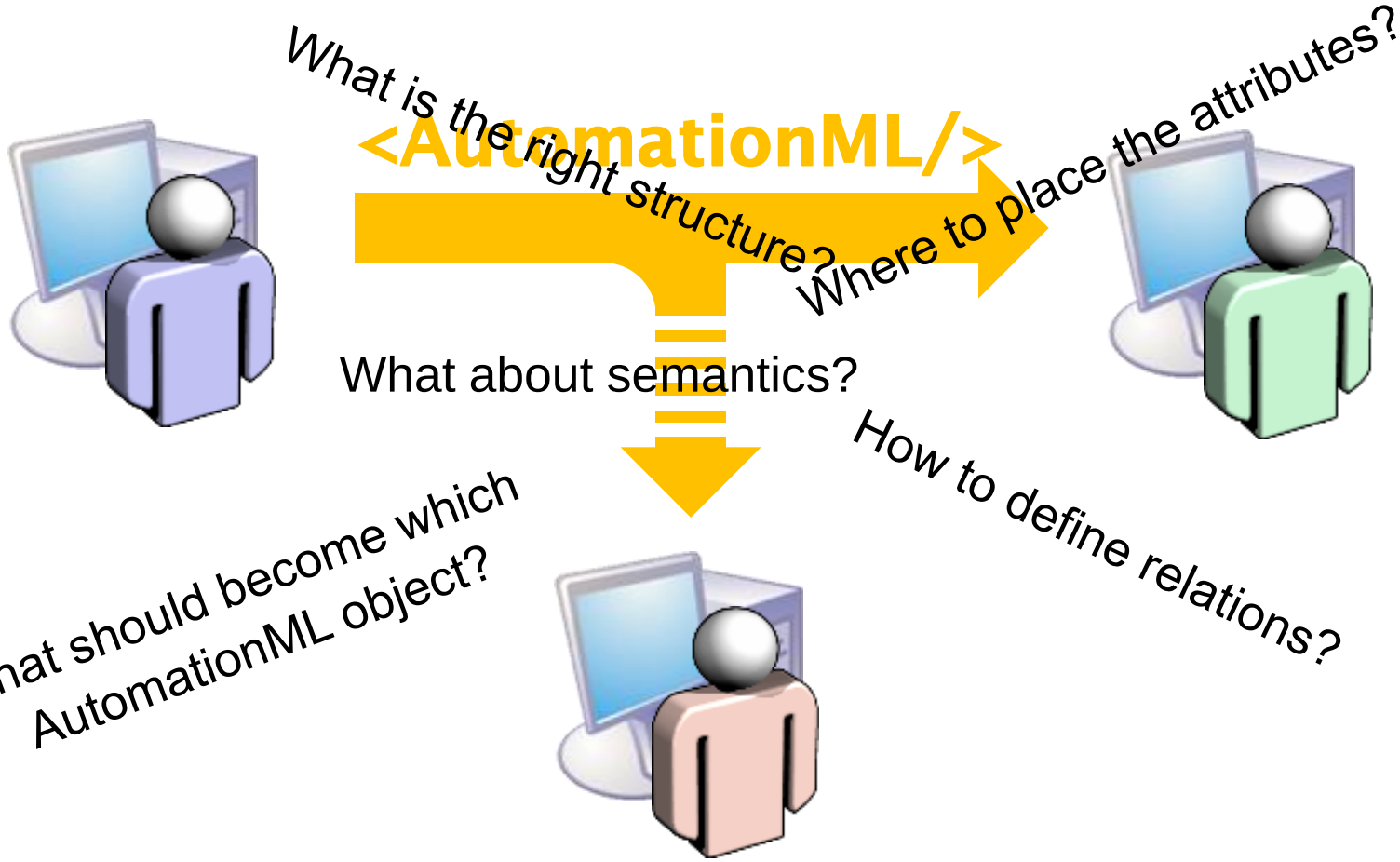
OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

How to use the described architecture?

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

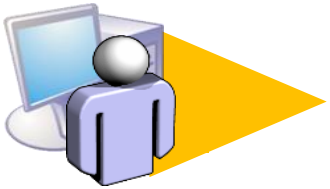
apl. Prof. Dr.- Ing. habil. A. Lüder



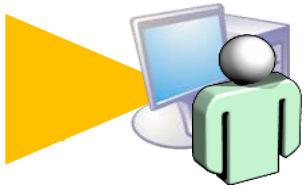
Use process

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

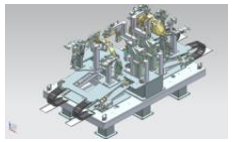


The exporting tool is master for structure and semantics.

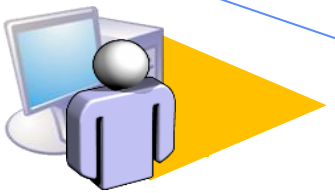


The importing tool is master for structure and semantics.

The export file is master for structure and semantics.



The data itself answers the questions.

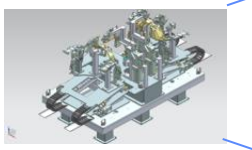


+ less effort, because structure is given

+ the structure works for these use cases

- tool changes can lead to non fitting structure

- more effort to connect further tools



+ more knowledge about the data

+ easier to enrich the data for further use cases

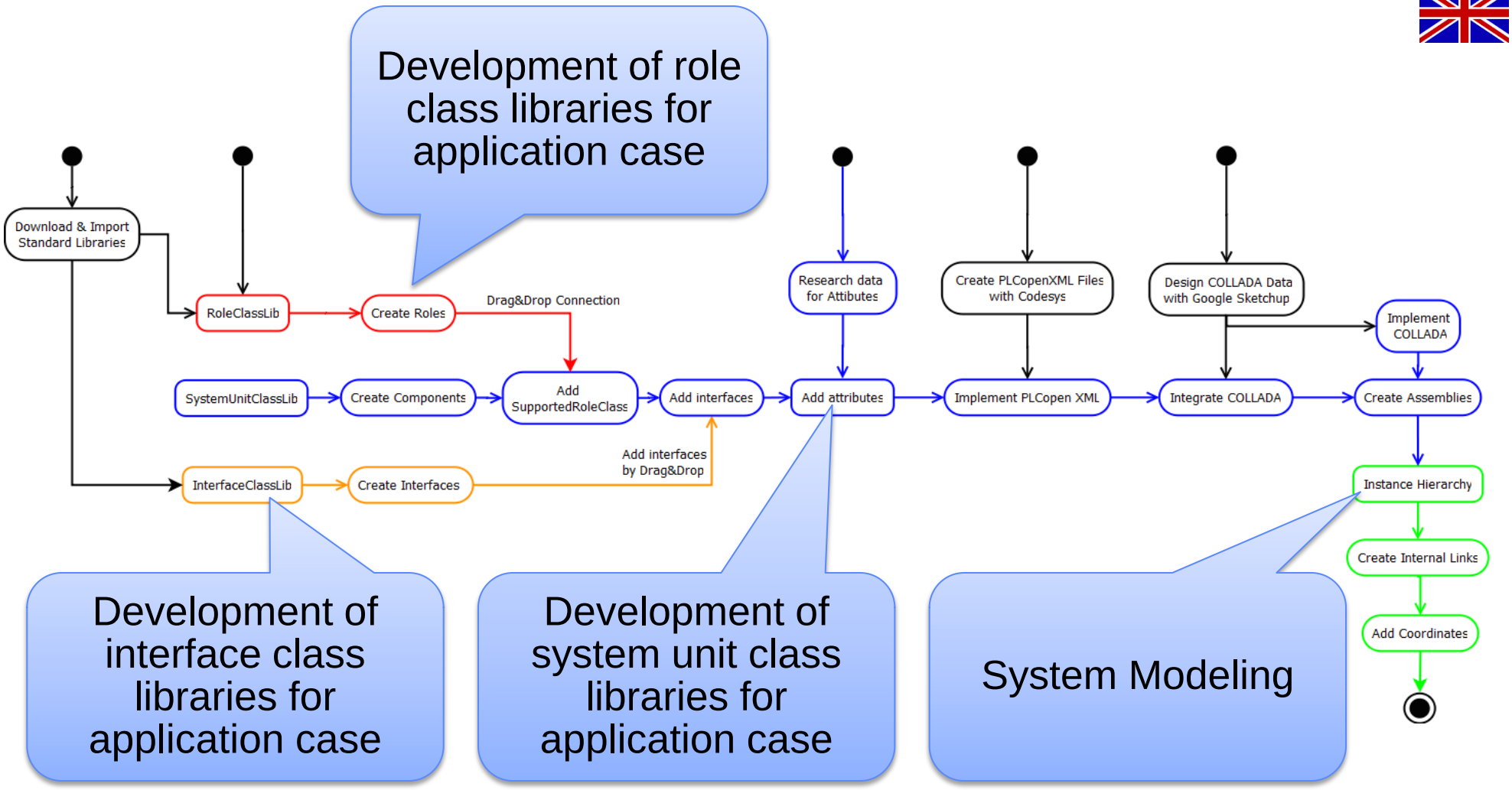
+ data structure fits to all tools

- more effort before implementation of the interface

Evaluation ot the approaches

- What are objects?
- How to structure the objects?
- What are object properties?

- What are relations between objects?
- How to structure the relations?
- What are relation properties?



Modeling process

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

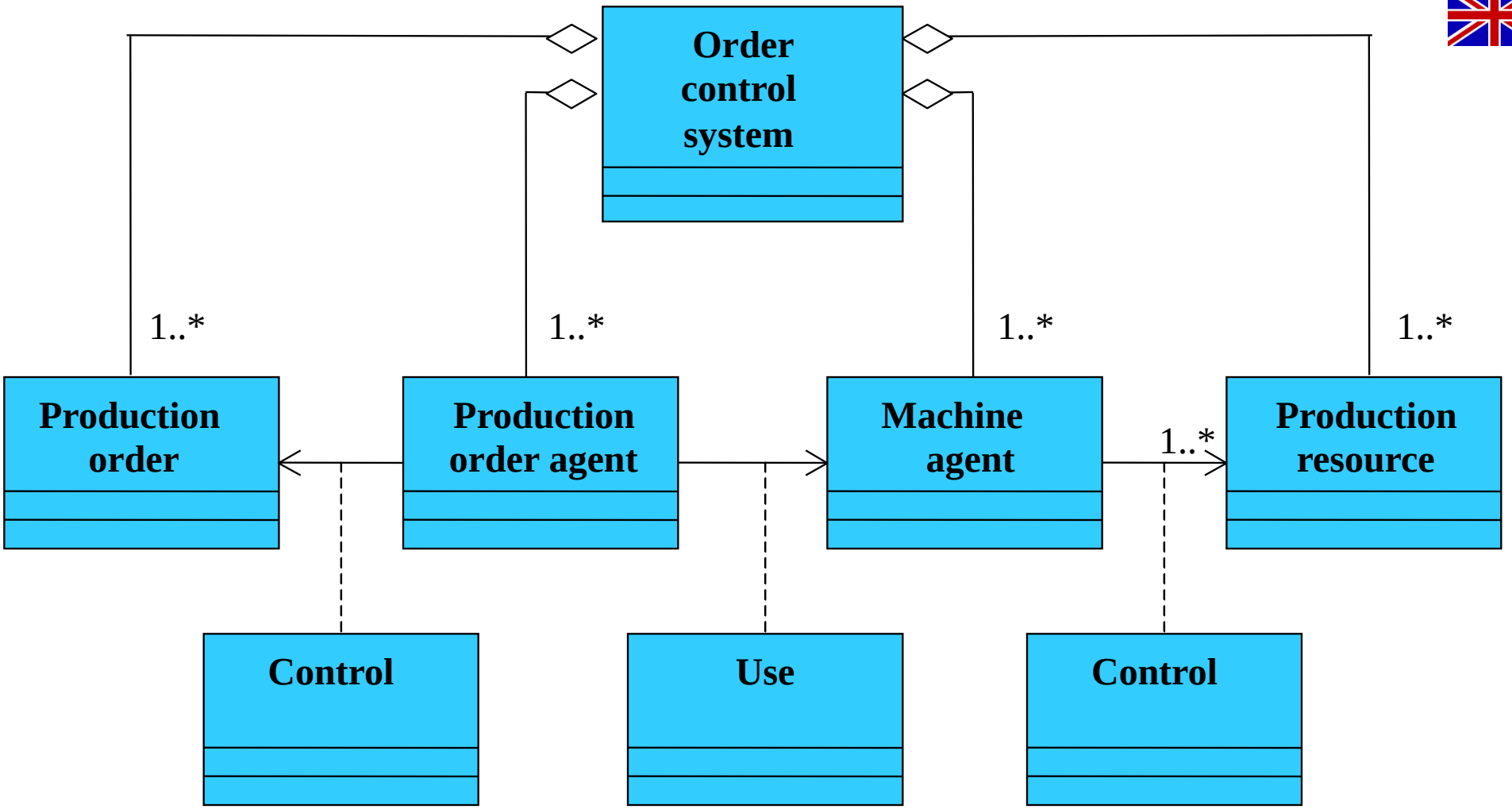
- **Design pattern at production and resource planning layers**
  - **Production orders shall be executed on a set of resources**
  - **For each production order a decision has to be taken at which time point at which resource a process for order execution can/shall be executed**
  - **For each resource shall be decided at which time point which process for order execution can/shall be executed**

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- **Design pattern at production and resource planning layers**
  - Let assume the order scheduling is executed by a distributed optimization system based on agent technology
- **Then:**
  - Production orders will be controlled by production order agents
    - » Have responsibility for planning and execution of production processes required for product creation
    - » Contain all therefore necessary data
  - Resources will be controlled by management agents
    - » Have responsibility for planning and execution of production processes execution by resources
    - » Control resources

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB
apl. Prof. Dr.- Ing. habil. A. Lüder



Practical examples
Control systems
Design pattern

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

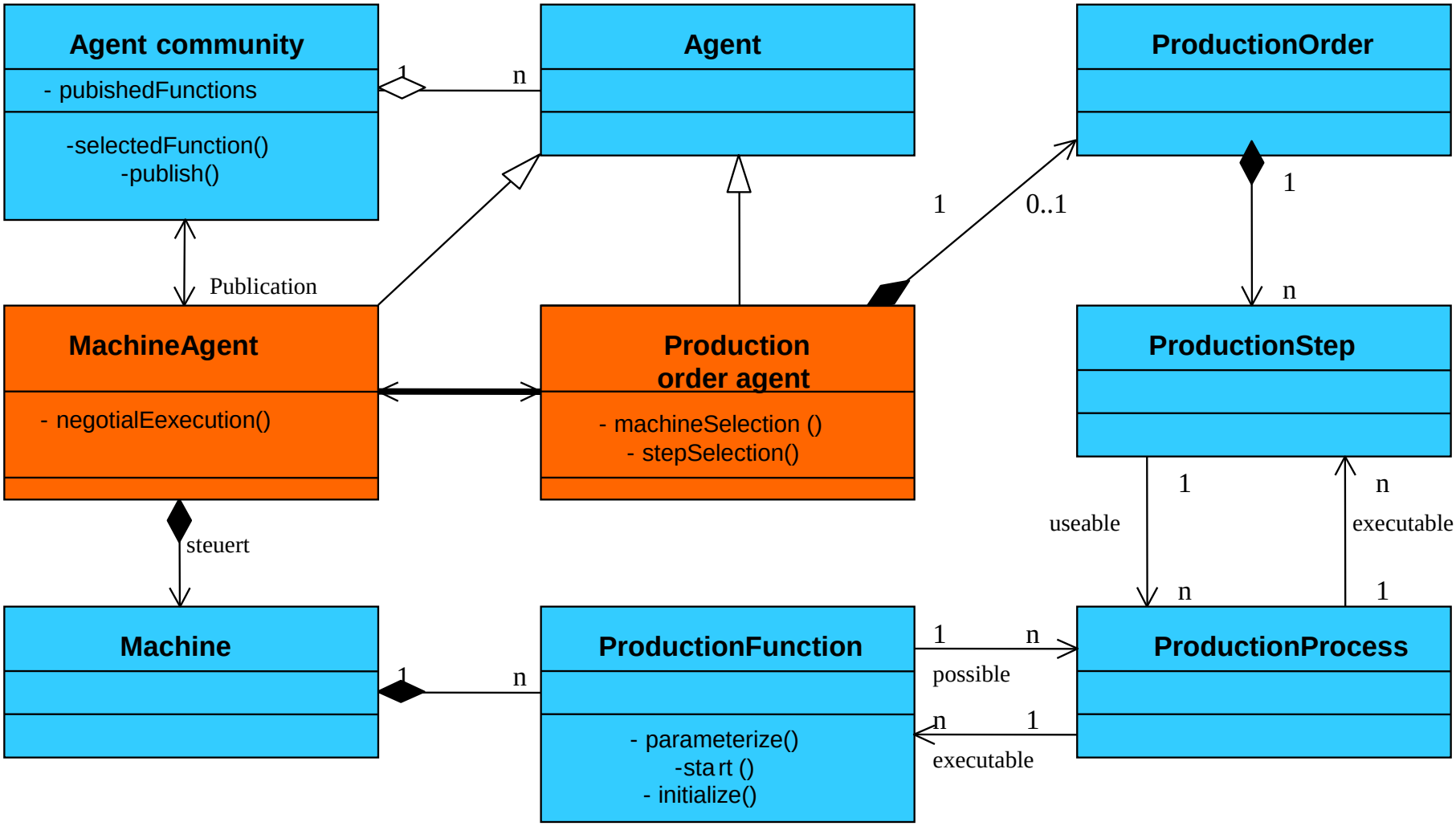
apl. Prof. Dr.- Ing. habil. A. Lüder

Design pattern at production and resource planning layers

- Machine agents
  - » Are able to execute and control the production functions of the resources
  - » Enable the order agent access to production functions
  - » Publish themselves, their access path and the individual production functions within the agent community
- Production order agents
  - » Carry all data required for the production order like required production process sequence and the set of required production functions needed to realize these production processes
  - » Search for required resources
  - » Ensure production process execution at them

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

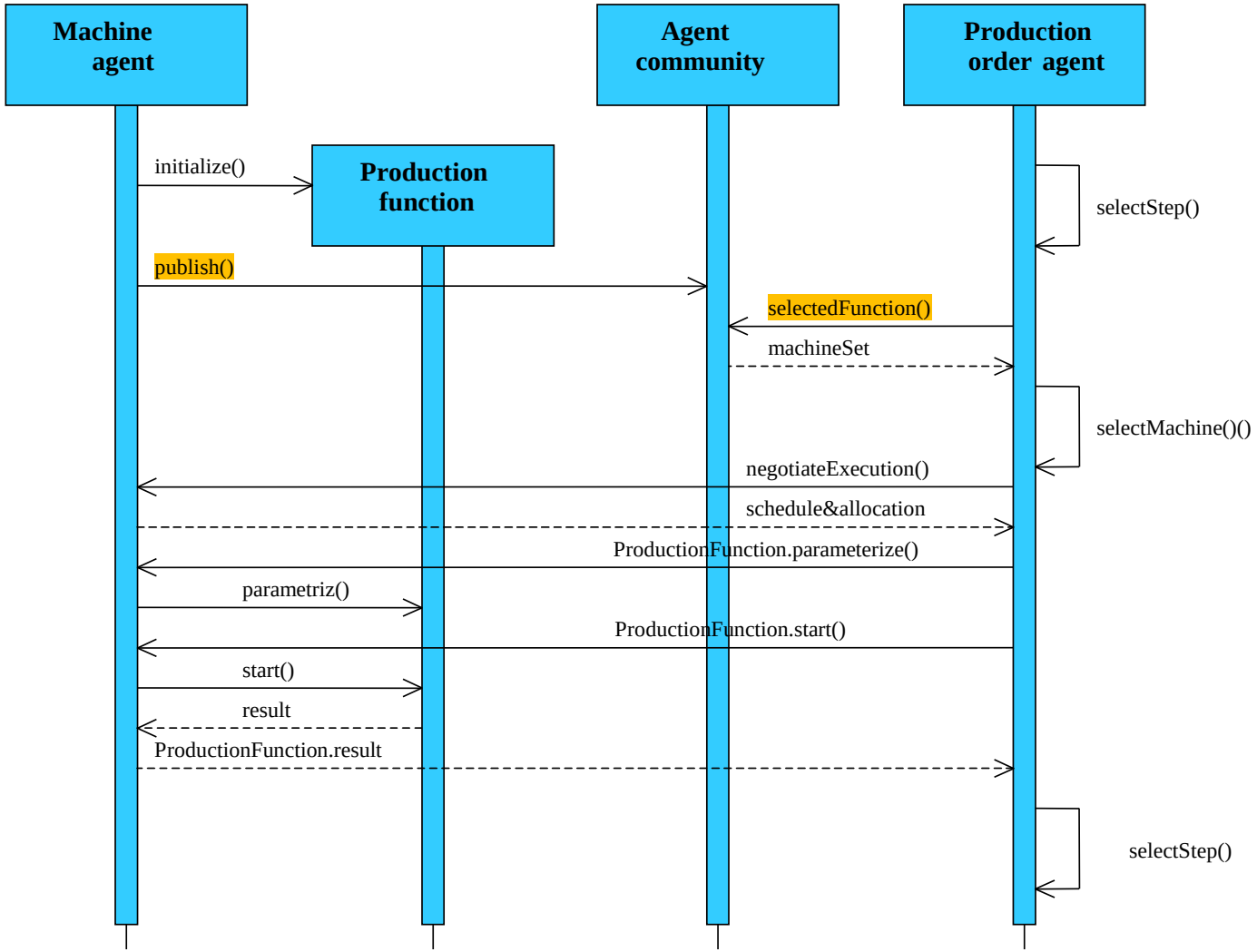
apl. Prof. Dr.- Ing. habil. A. Lüder



Practical examples
Control systems
Design pattern

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

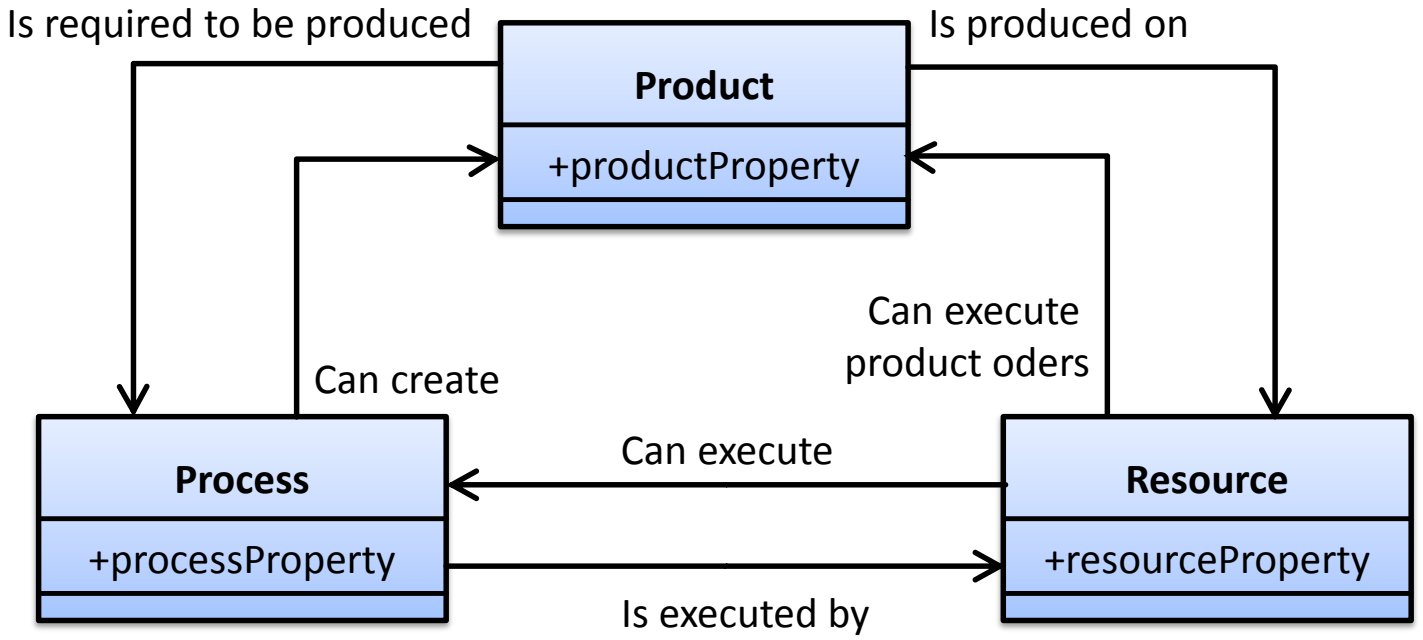
apl. Prof. Dr.- Ing. habil. A. Lüder



Practical examples
Control systems
Design pattern

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG
MB
apl. Prof. Dr.- Ing. habil. A. Lüder

- **Production systems cover three main parts being sources of requirements related to life cycles:**
  - **products**
  - **processes**
  - **resources**

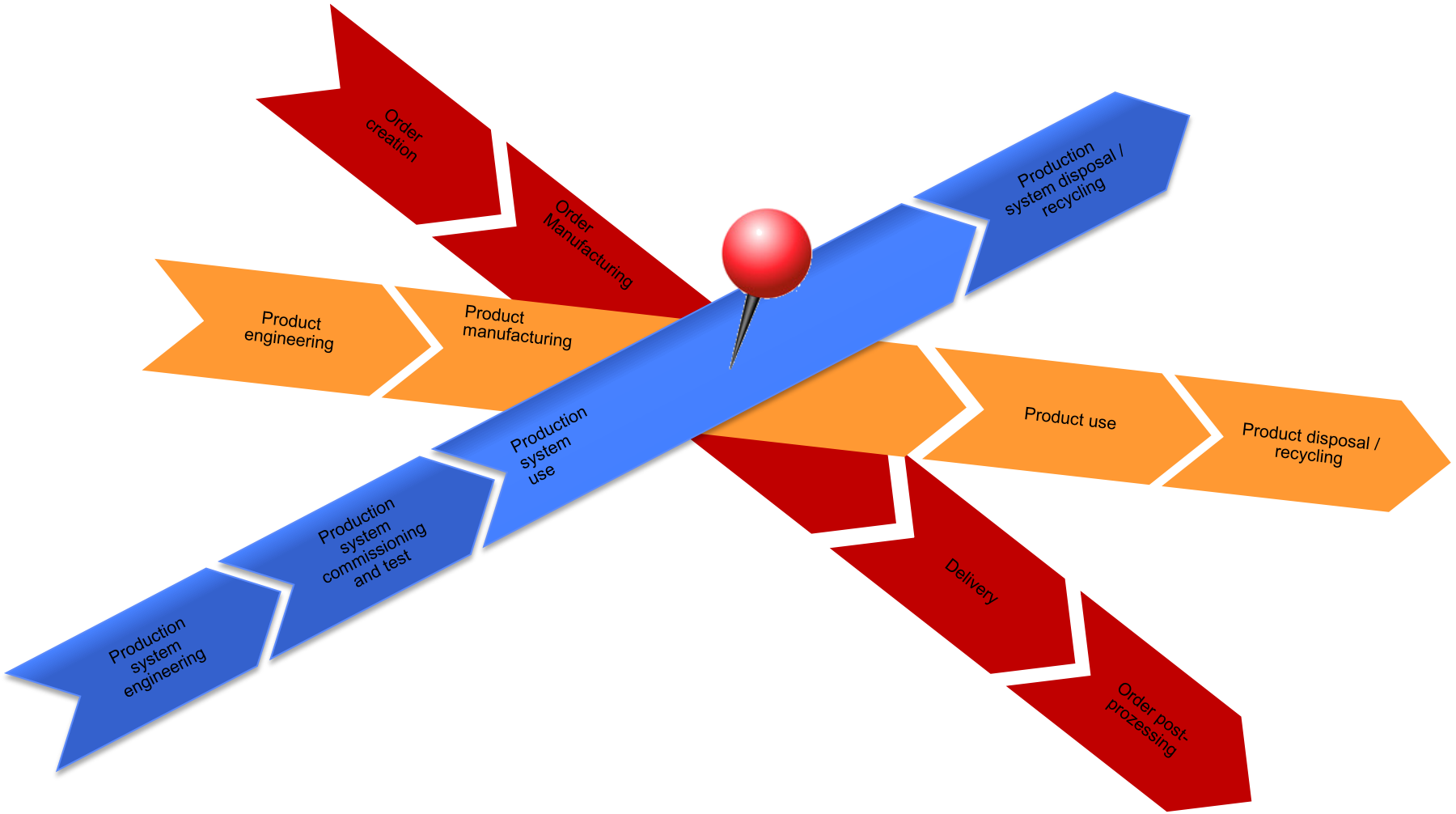


apl. Prof. Dr.- Ing. habil. A. Lüder

- **Products**
  - Are the intended output of the production system
  - Are usually characterized for customers by
    - Product properties
    - Product functionality
  - Are characterized for production system owner by
    - Bill of material
    - Bill of operations (required manufacturing processes based on material use)

apl. Prof. Dr.- Ing. habil. A. Lüder

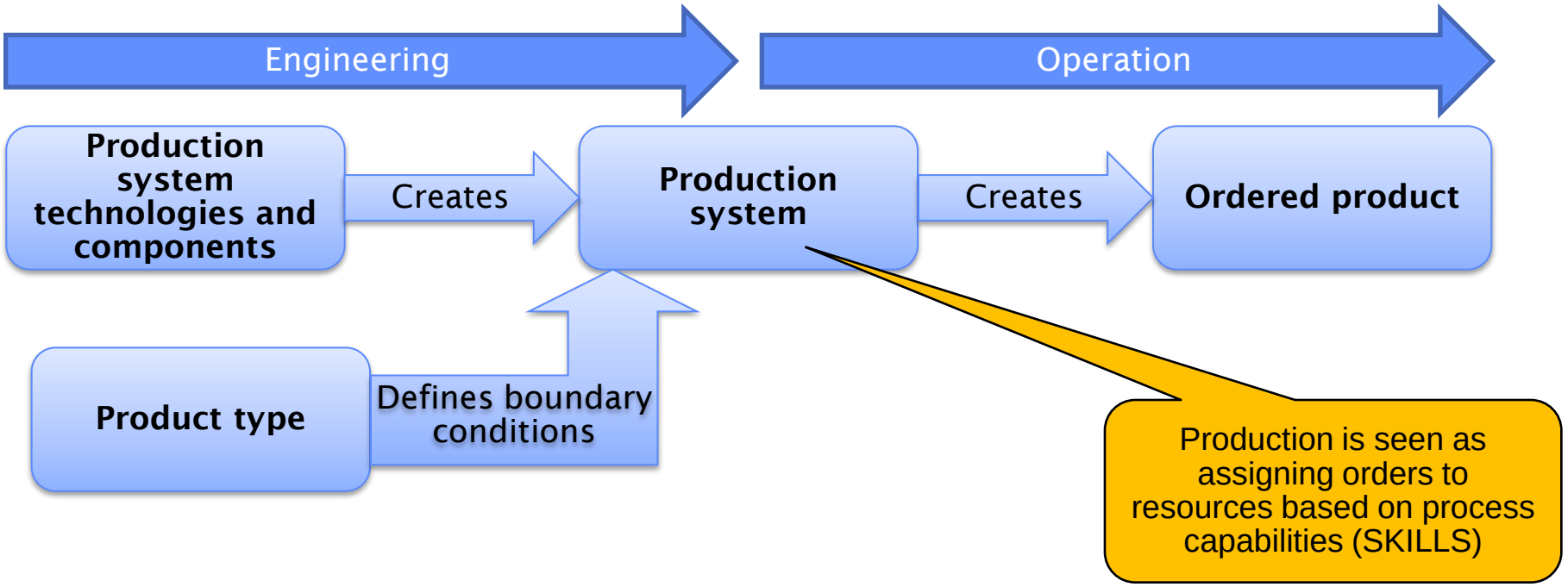
- **Processes**
  - Are required for product processing
  - Are provided by production resources
  - Are characterized by process capabilities and limitations following the involved manufacturing technology or support technology
- **Resources**
  - Are provider of production processes
  - Product creation is scheduled on them
  - Are characterized by resource capabilities related to manufacturing process and resource location



Conclusion

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

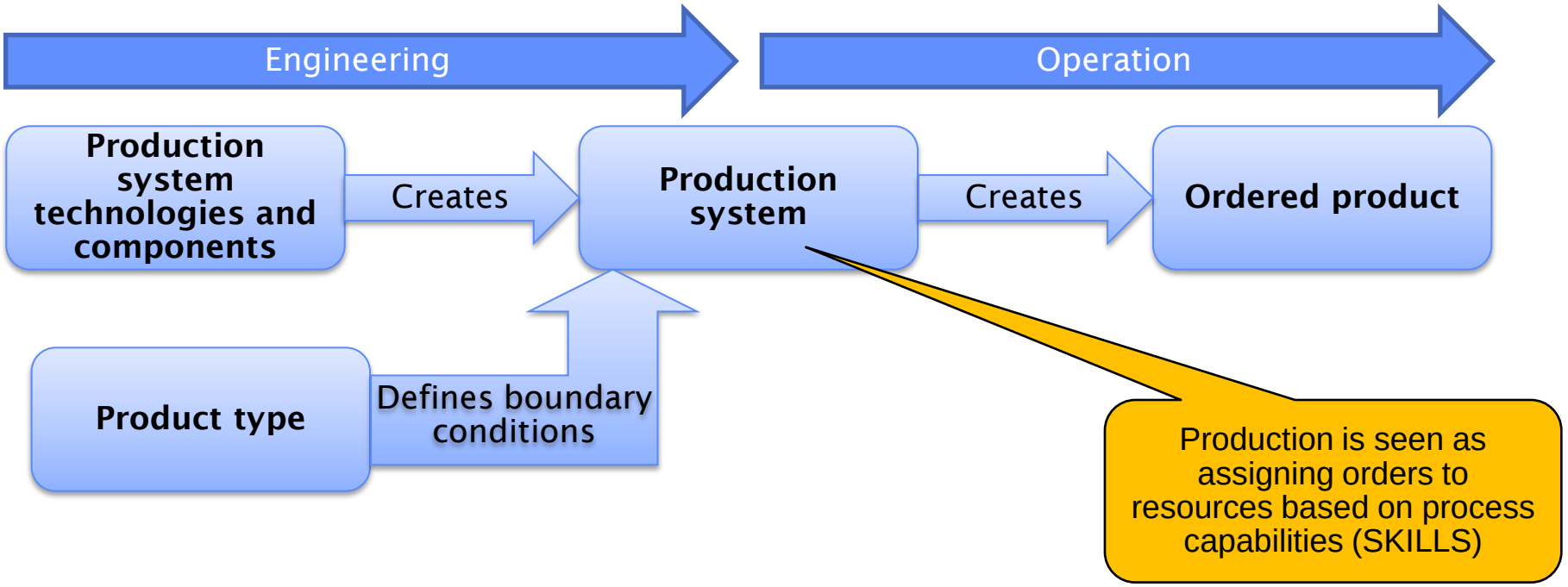
apl. Prof. Dr.- Ing. habil. A. Lüder



Conclusion

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

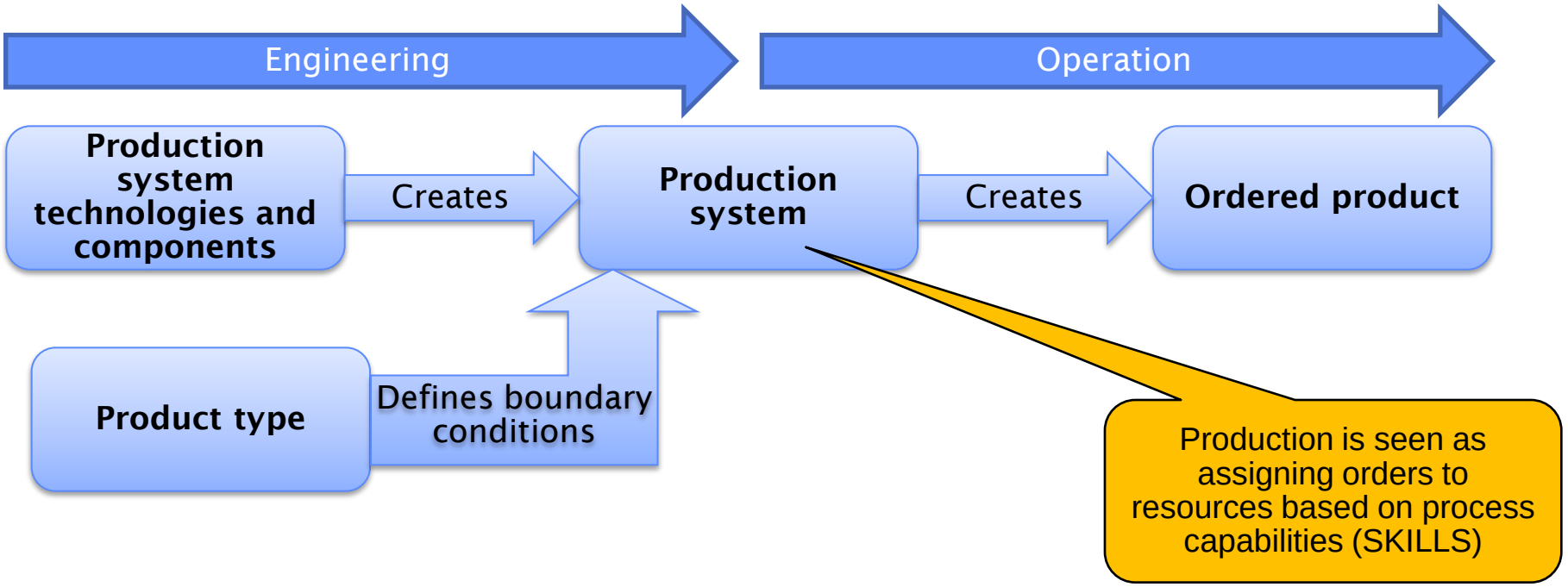
apl. Prof. Dr.- Ing. habil. A. Lüder



Conclusion

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

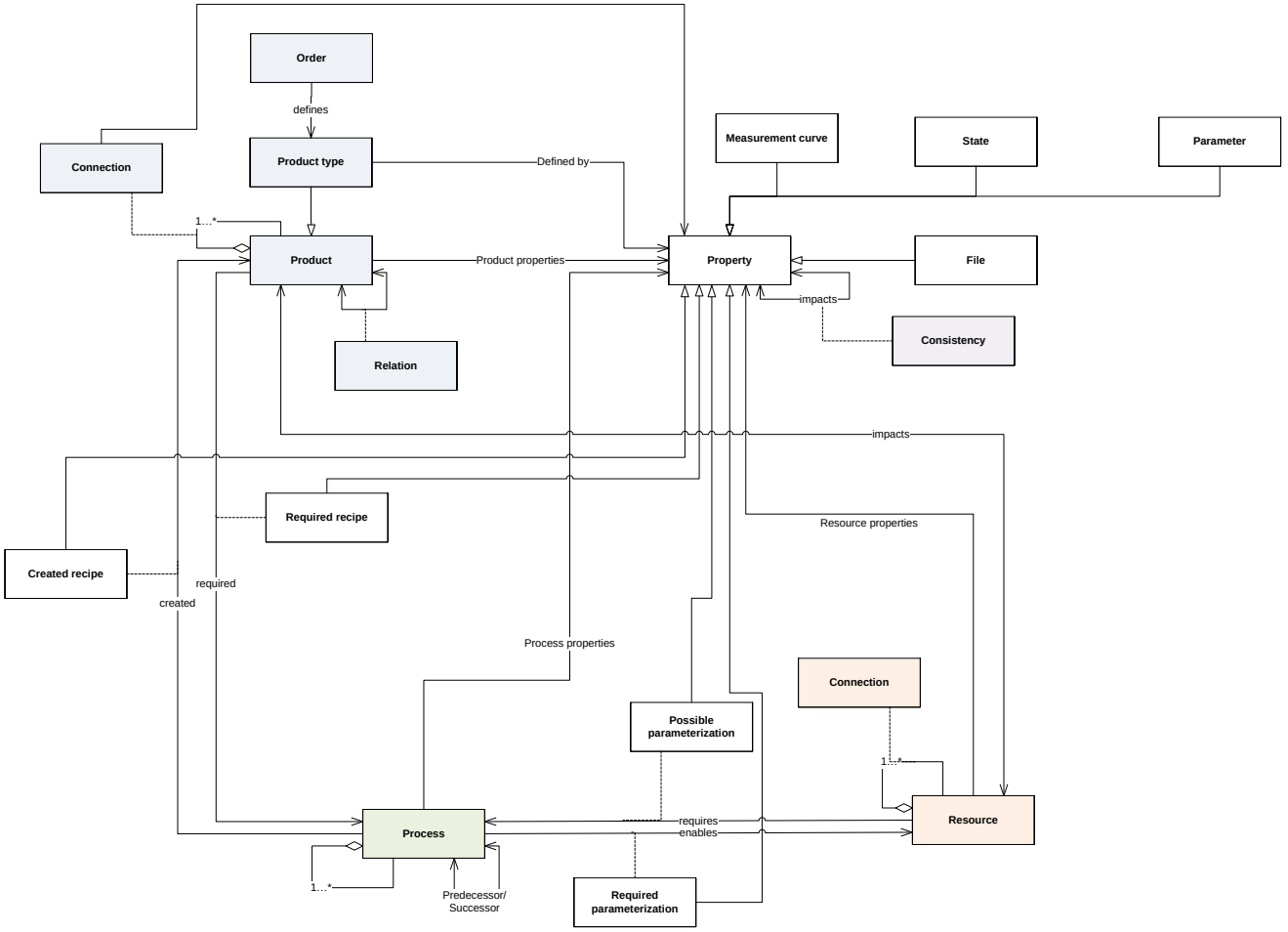


Conclusion

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

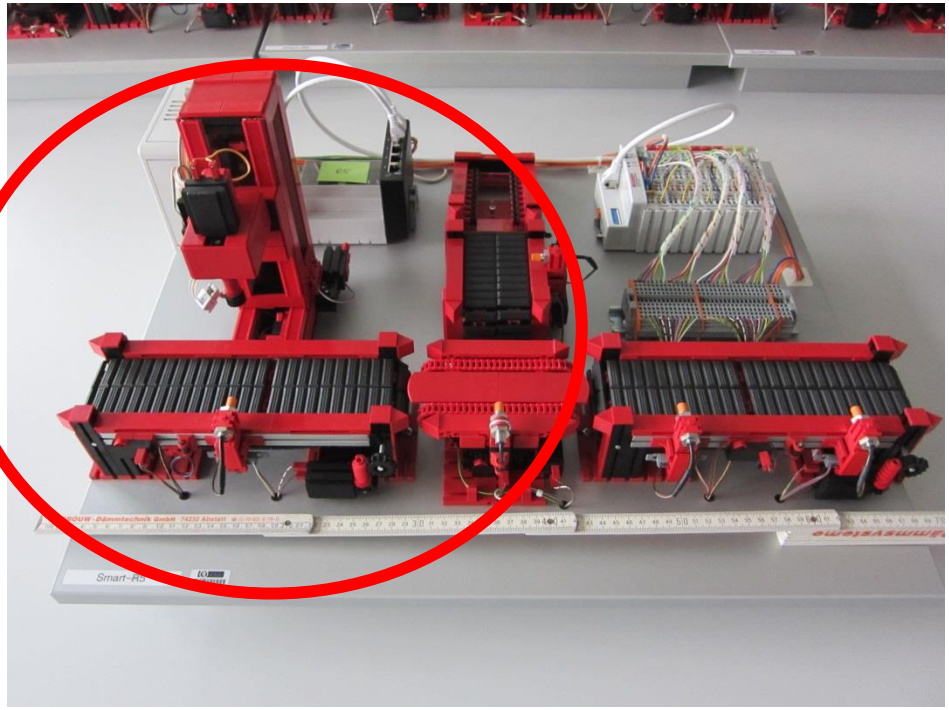
- A skill is seen as a capability
  - required by an order
  - provided by a resource
- A skill is characterized by its properties

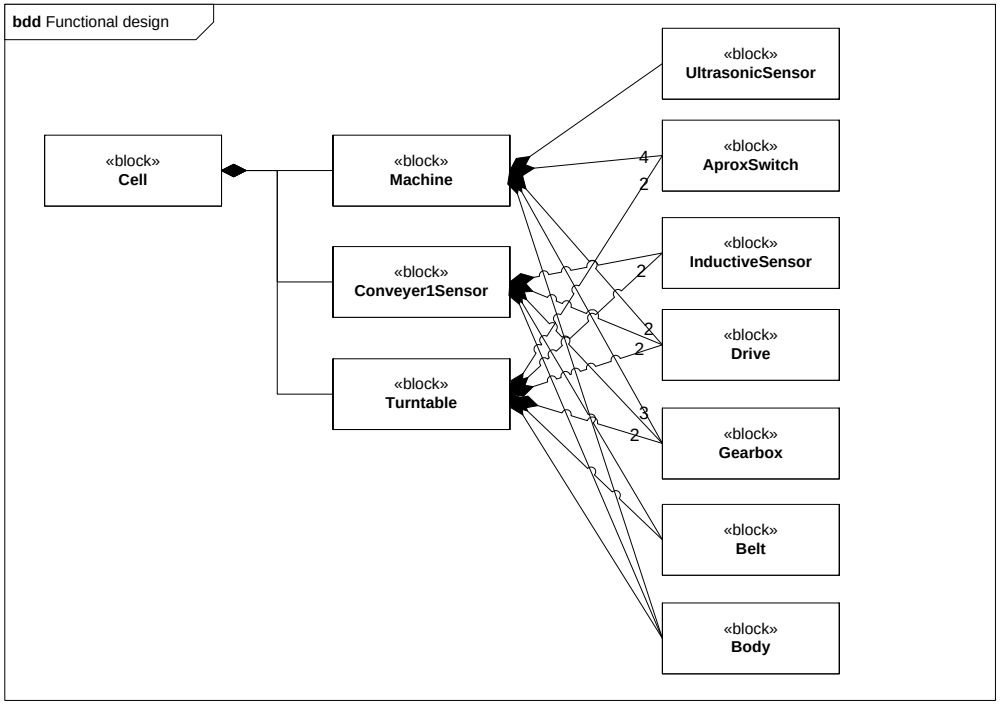


Skill Modelling Example

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder





Example – Ultrasonic measurement cell

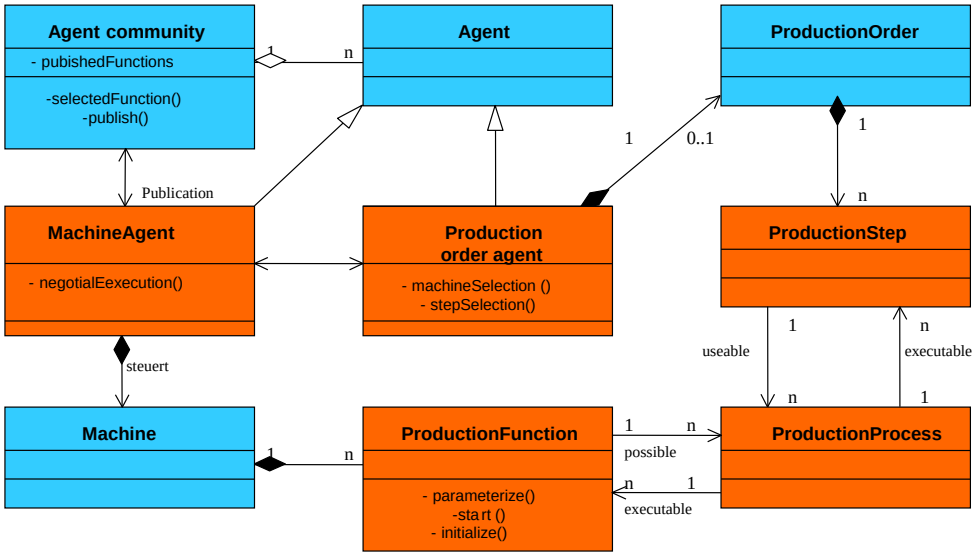
OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

- **Role classes are**
  - **Definition of the abstract semantics of information objects exchanged within an project.**
  - **Seen as a**
    - **Requirement to an exporter to provide specified information (attributes and interfaces) and**
    - **Assurance to an importer to receive specified information (attributes and interfaces).**

1. Identify necessary concepts

- What are the objects relevant in the modeled system?
- What are the concepts addressed within the involved tools and data processing systems?



- Order
- Resource
- Production Process
  - Production Function
  - Production step

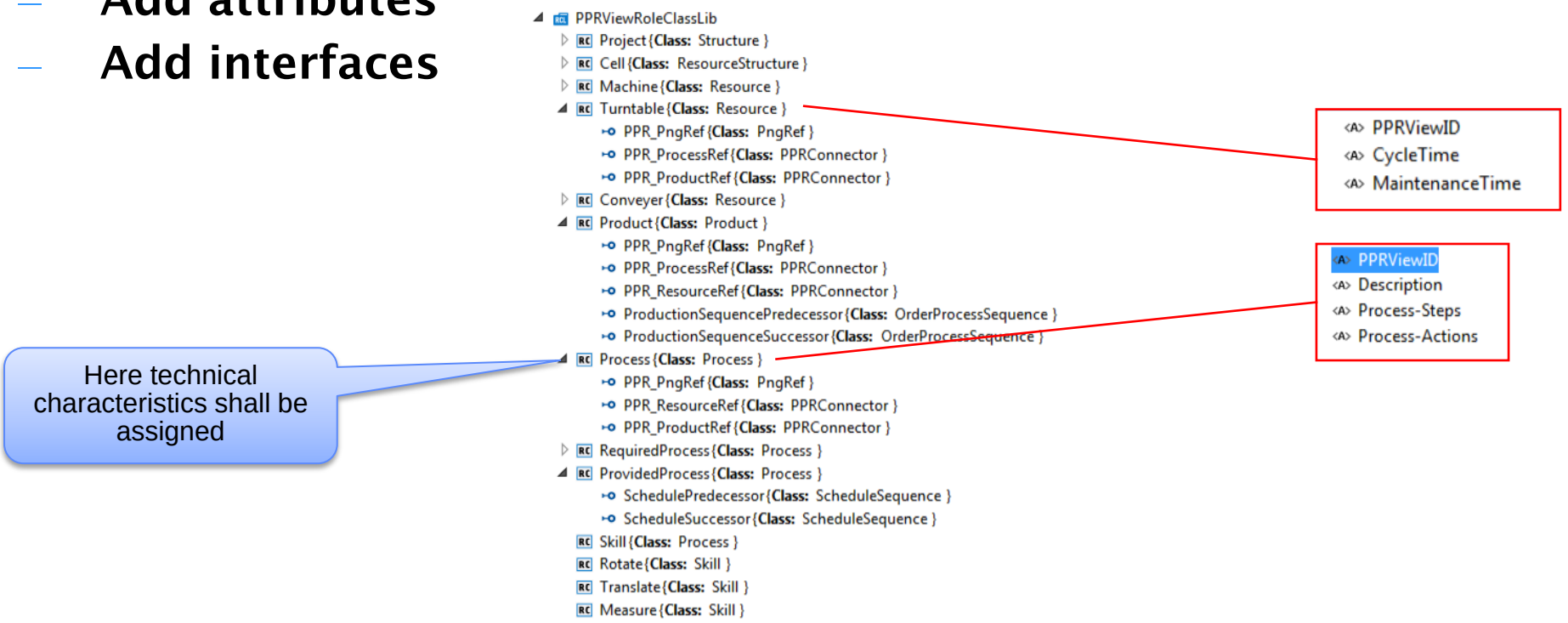
- Technical process characteristics

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

2. Create relevant role classes

- Set up the role
- Derive role from relevant roles of higher abstraction level
- Add attributes
- Add interfaces



OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

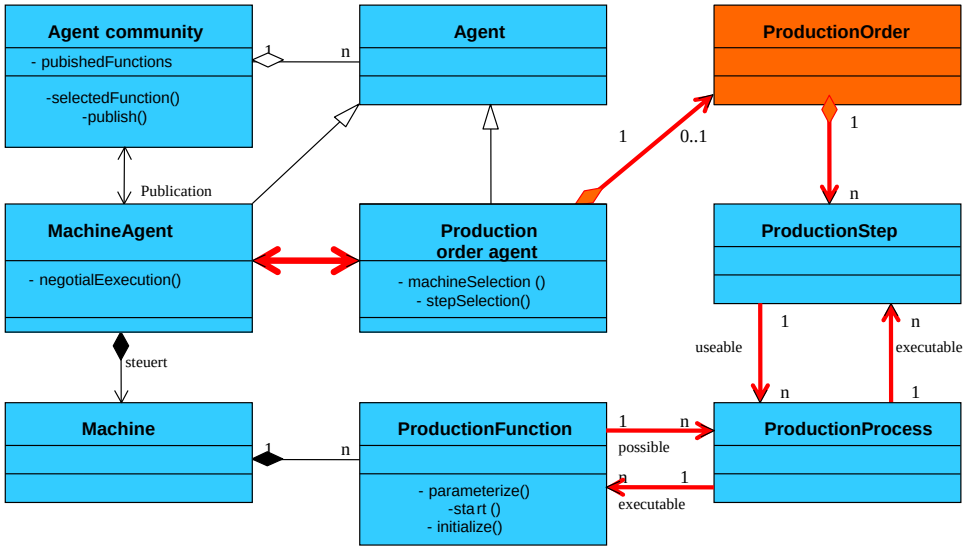
apl. Prof. Dr.- Ing. habil. A. Lüder

Interface classes

- Are abstract concepts for the relation between objects
  - Within an project
  - To external data
- Describe contact points for relations (describe only the one side of the relation)
- Can specify properties of the contact point
  - Like direction or type

1. Identify necessary concepts

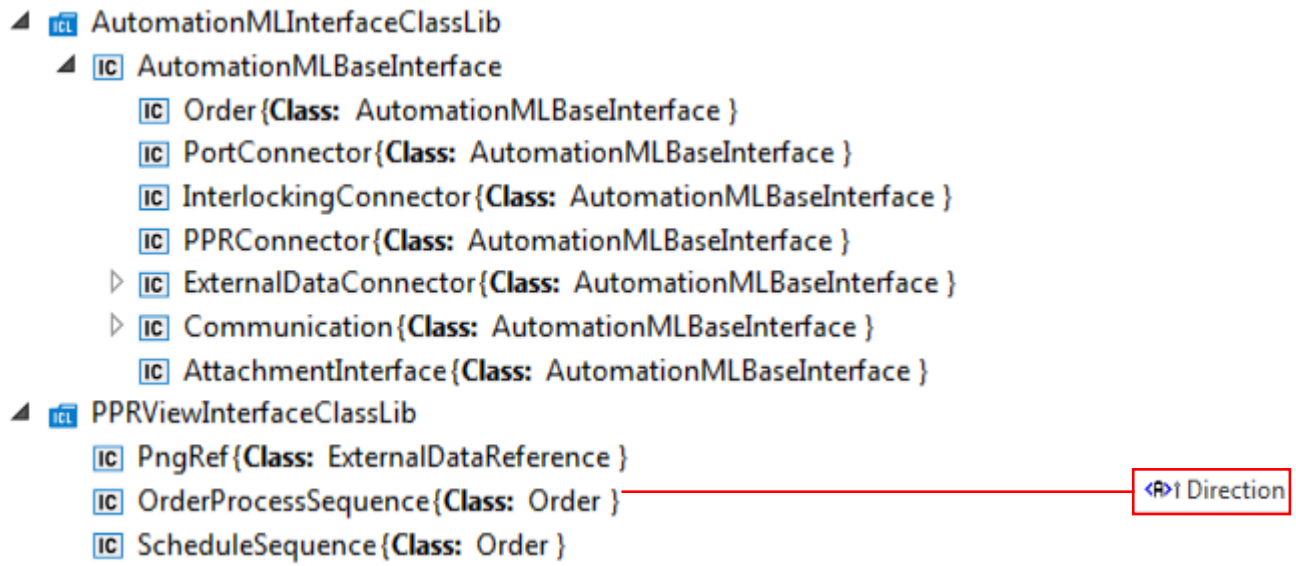
- What are the interfaces between objects an to external data relevant in the modeled system?
- What are the concepts addressed within the involved tools



- Sequence of process steps within an order
- Sequence of scheduled process steps within an machine
- Relation that a process step is scheduled on a machine

2. Create relevant interface classes

- Set up the interface class
- Derive interface class from relevant interface class of higher abstraction level
- Add attributes



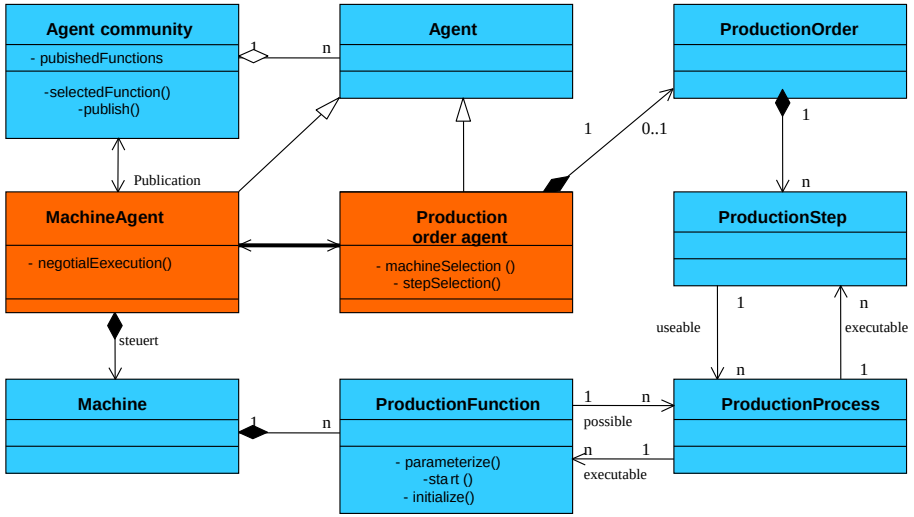
apl. Prof. Dr.- Ing. habil. A. Lüder

System unit classes

- Describe concrete types of objects reused in the engineering of production systems
  - » Like married humans or cities
- Specify the necessary properties (attributes) and relations (interfaces) of the objects to be modeled
- Specify roles (possibly more than one) they “implement”
- Specify their internal structure

1. Identify necessary reusable objects

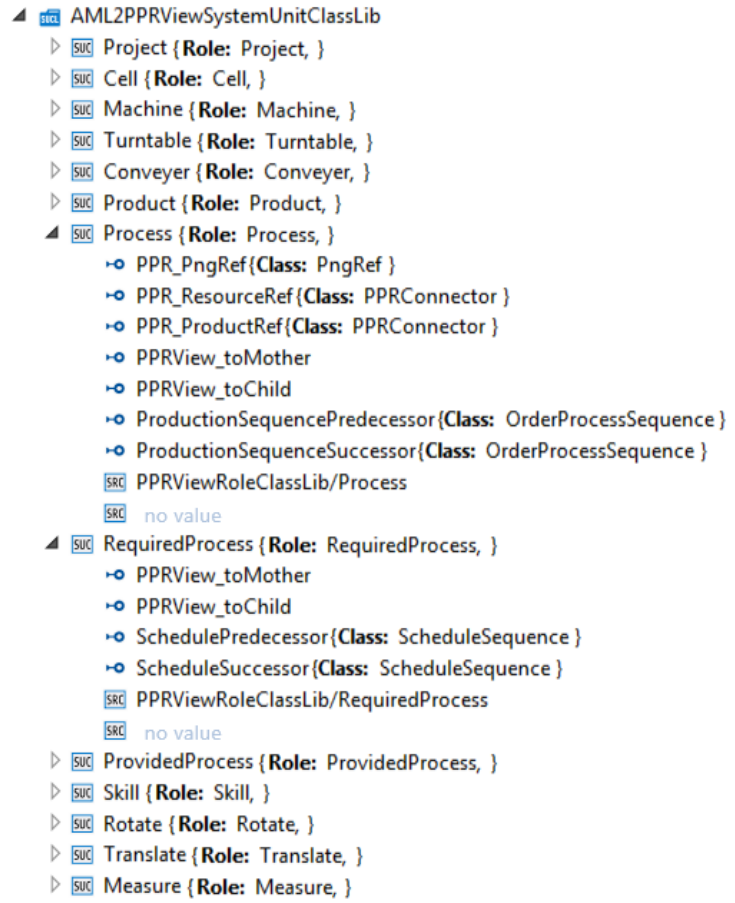
- What are the types of objects relevant in the modeled system?
- What are the hierarchical structures of these objects?



- Required process steps with step sequence
- Provided process steps with step sequence
- Order agent with sequence of required process steps
- Machine agent with scheduled process steps

2. Create relevant system unit classes

- Set up the system unit class
- Derive system unit class from relevant system unit class of higher abstraction
- Add relevant attributes
- Add relevant interfaces
- Integrate relevant substructures
- If relevant add geometry / kinematics model
- If relevant add behavior model



OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB
apl. Prof. Dr.- Ing. habil. A. Lüder

- **Instance Hierarchy**
  - **Is the area to model the data to be exchanged (whole production system with all elements)**
  - **Consists of a hierarchy of internal elements with interfaces and attributes**
  - **Can be modeled without using created libraries**
  - ➢ **For reasons of semantical clearness that should be avoided**
- **Before starting the modeling of the instance hierarchy, the three libraries shall be free of faults to prevent complicated and time-consuming corrections**

1. Select reusable objects from system unit class library
   - What are the objects relevant in the modeled system?
   - Which class do they belong to
   - What are the hierarchical structures of the objects of the modeled system?
2. Create relevant hierarchy of internal elements
   - Set up the internal element by either
     - » Instantiate relevant system unit class
     - » Create new internal element
   - Complete / add relevant attributes
   - Complete / add relevant interfaces
   - Complete / add relevant substructures
   - If relevant add geometry / kinematics model
   - If relevant add behavior model
   - Add necessary links

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG MB

apl. Prof. Dr.- Ing. habil. A. Lüder

Instance hierarchy