

Data Modeling with AutomationML

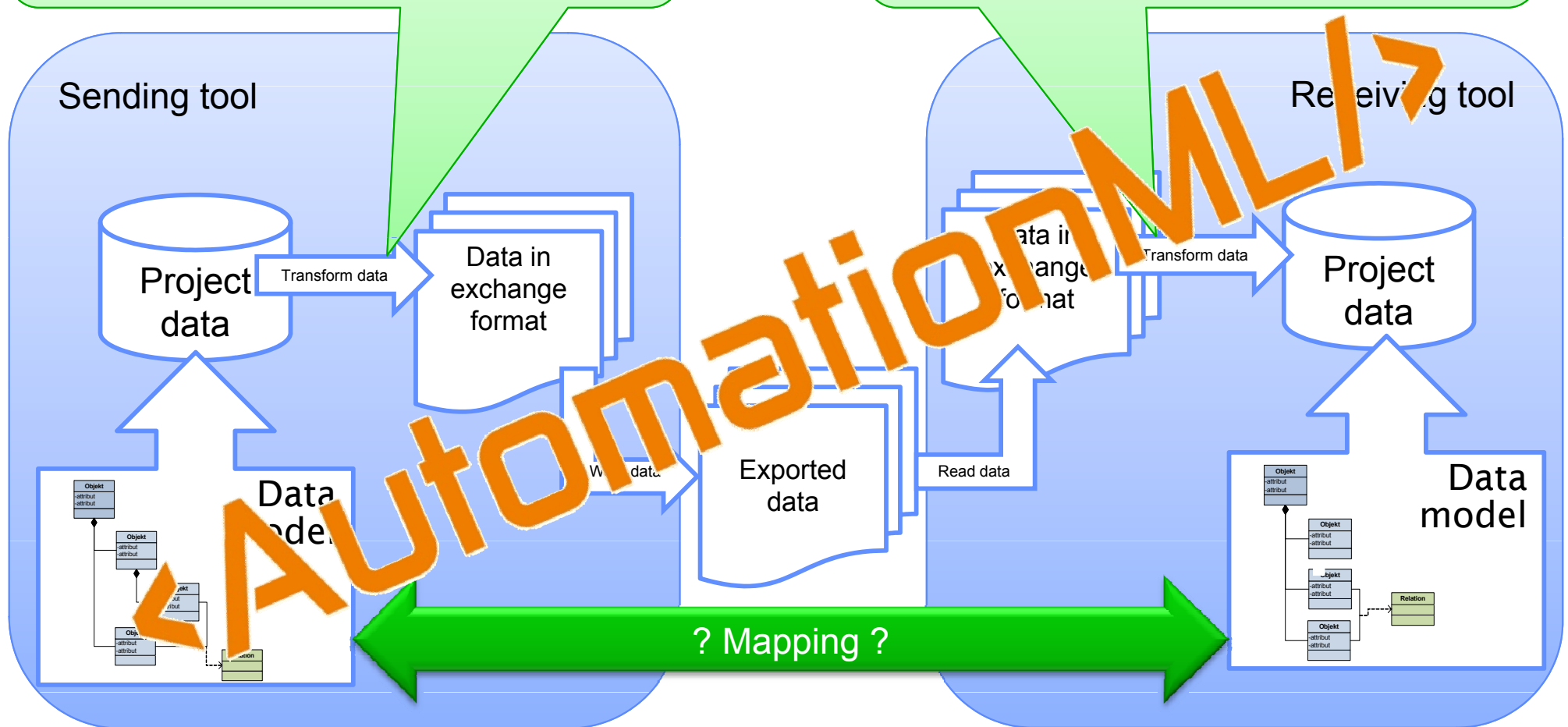
Mapping of data into data exchange format AutomationML

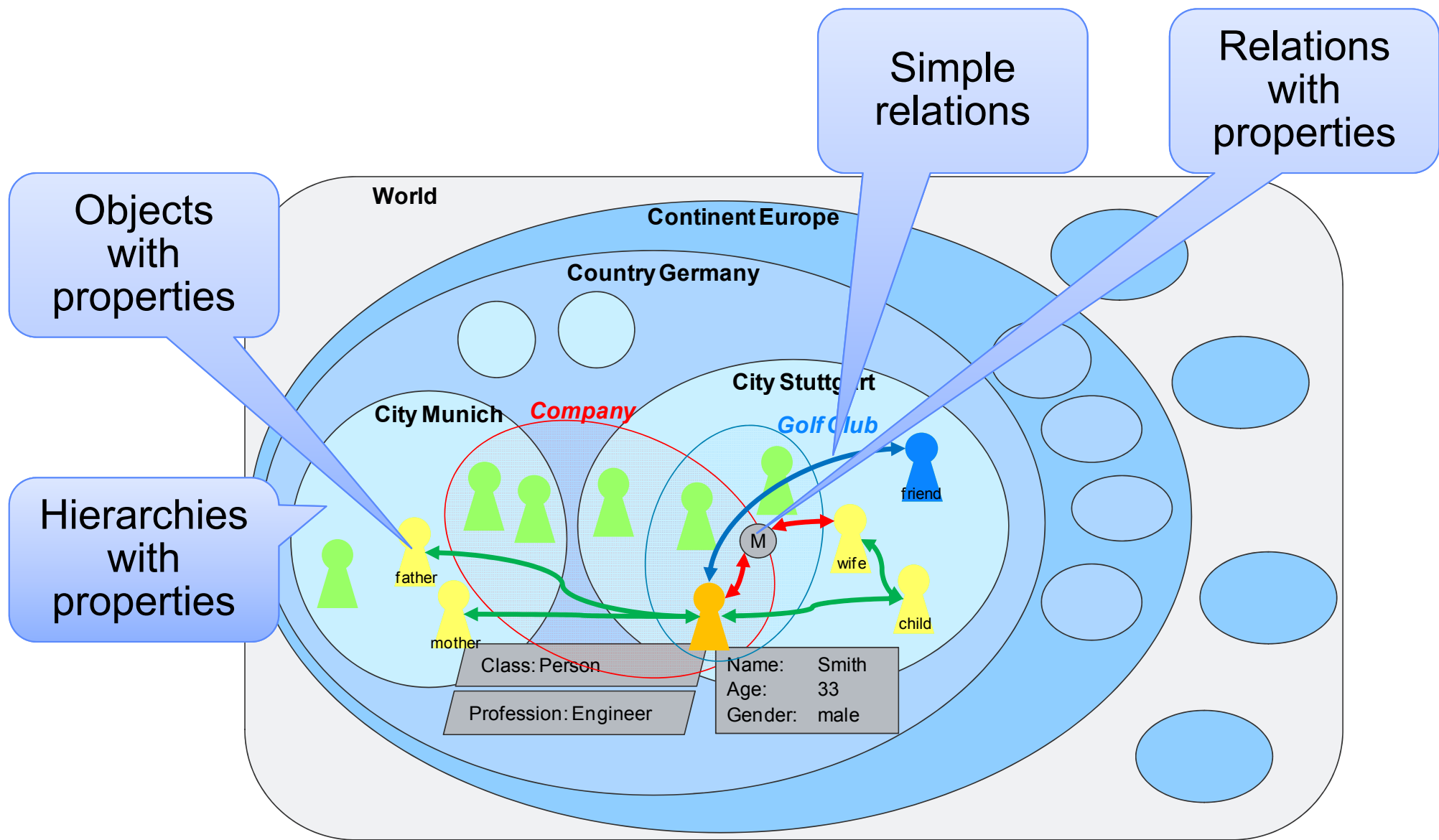
Arndt Lüder

Otto-von-Guericke University Magdeburg

Preconditions: Definition of data elements to be exported including syntax and semantics

Preconditions: Knowledge of received data elements including syntax and semantics and **knowledge about dependencies to internal data model**





Top level format CAEX IEC 62424

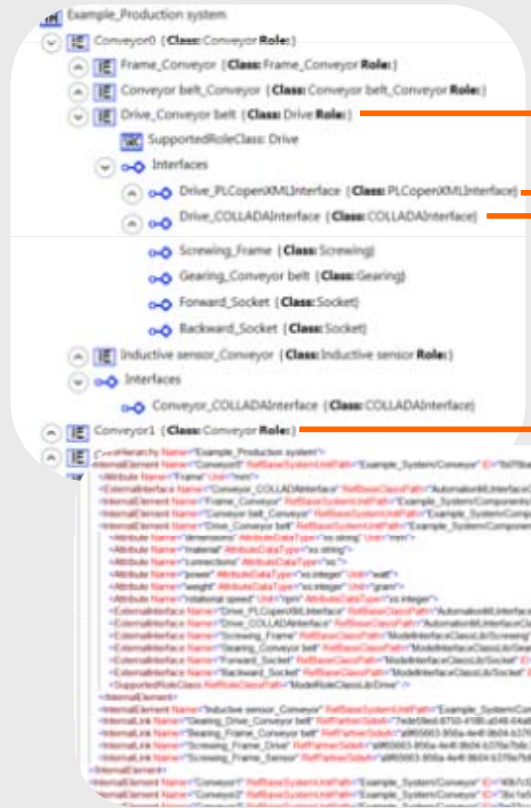
Plant
Topology
Information

Mechatronics

Networks

Devices

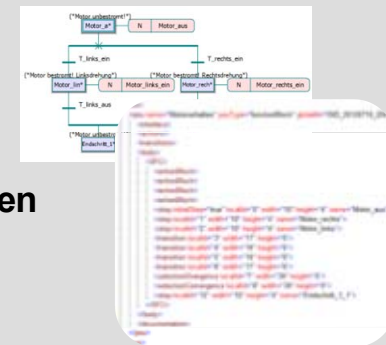
Attributes



Geometry
and
Kinematic
format
COLLADA



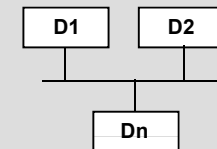
Logic
format
PLCopen
XML

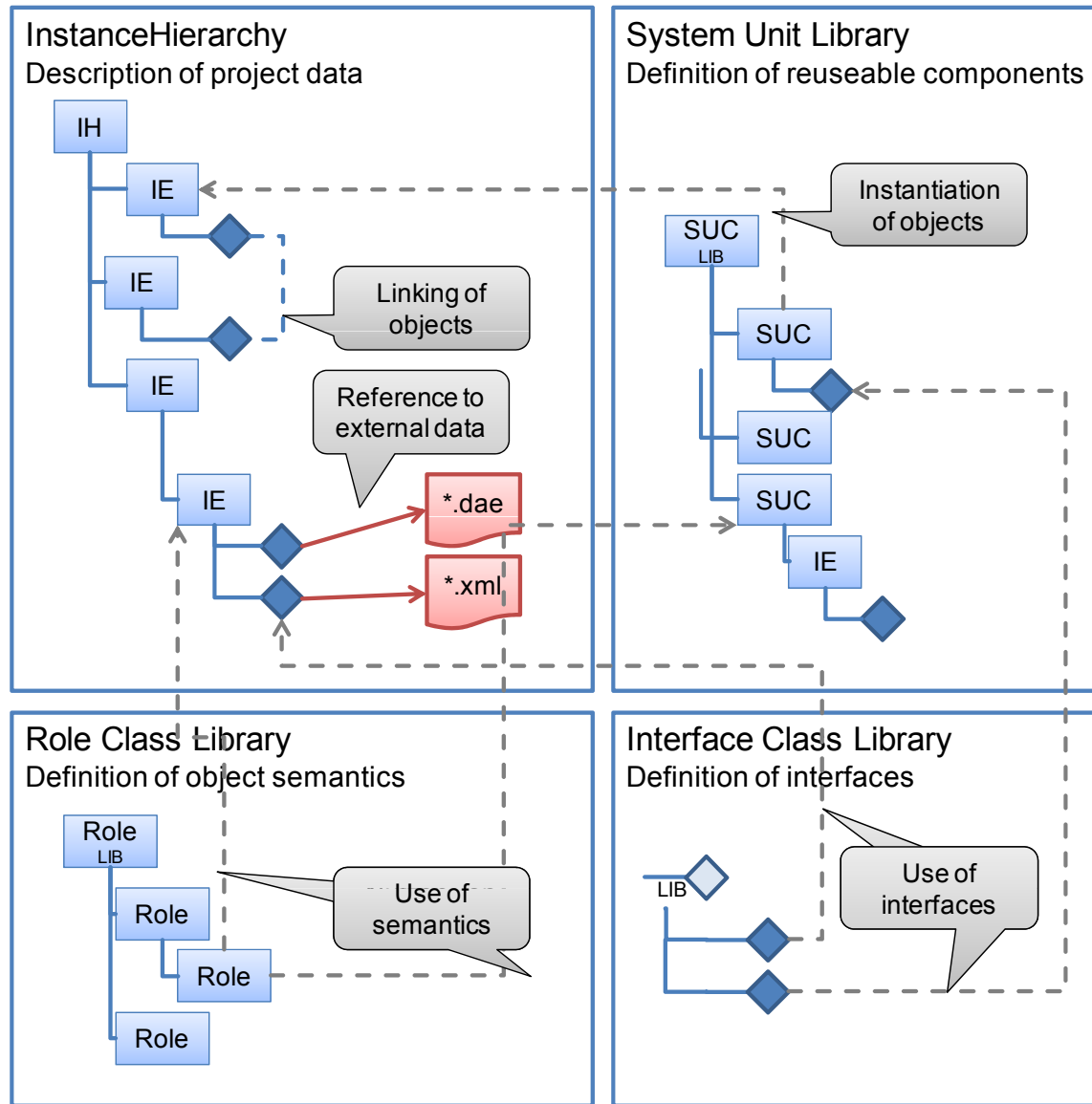


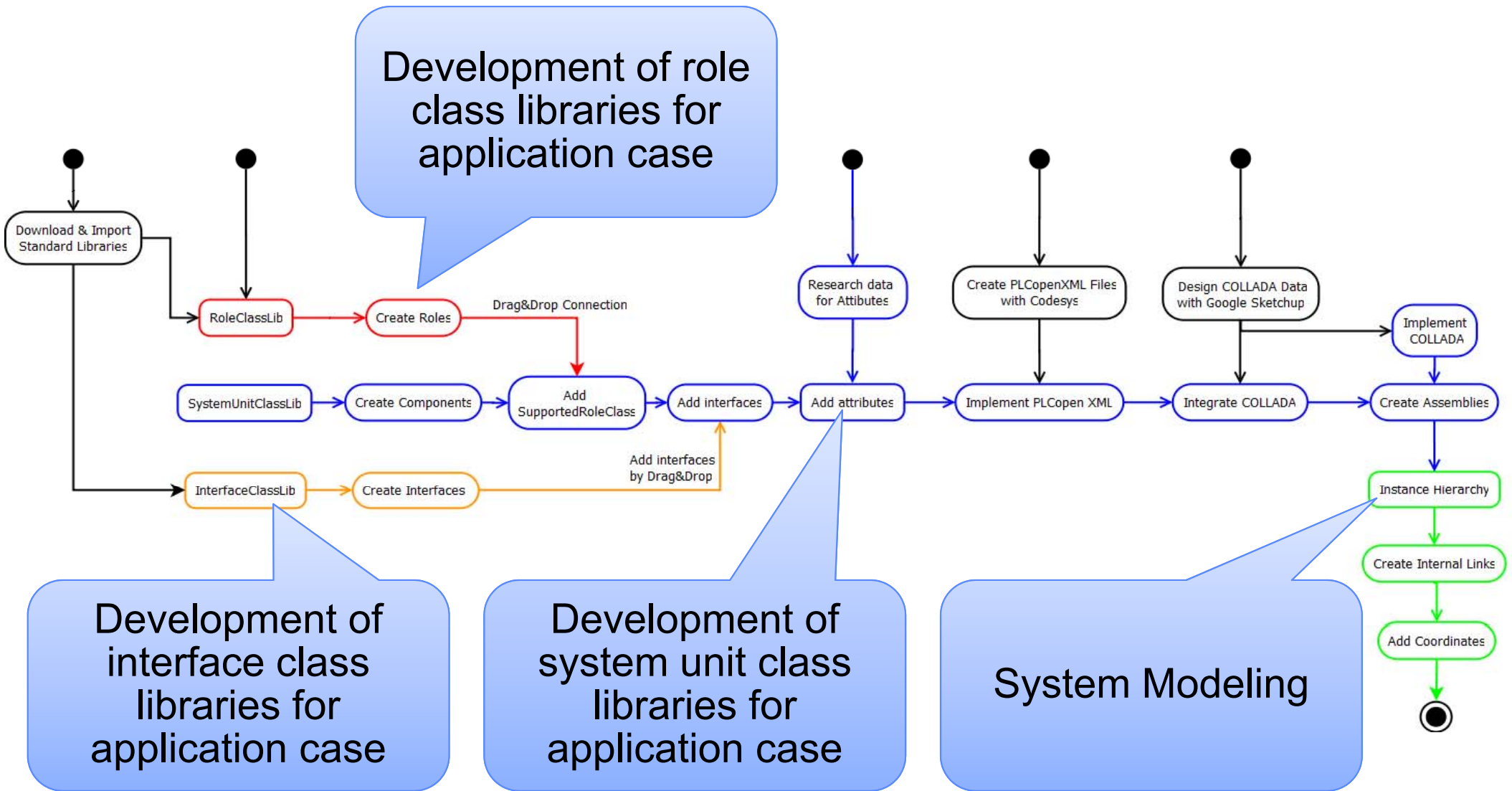
Semantic
referencing



Further aspects
in other XML
format







Arndt

Mona

Hermann

Jutta

Beate



■ Role classes are

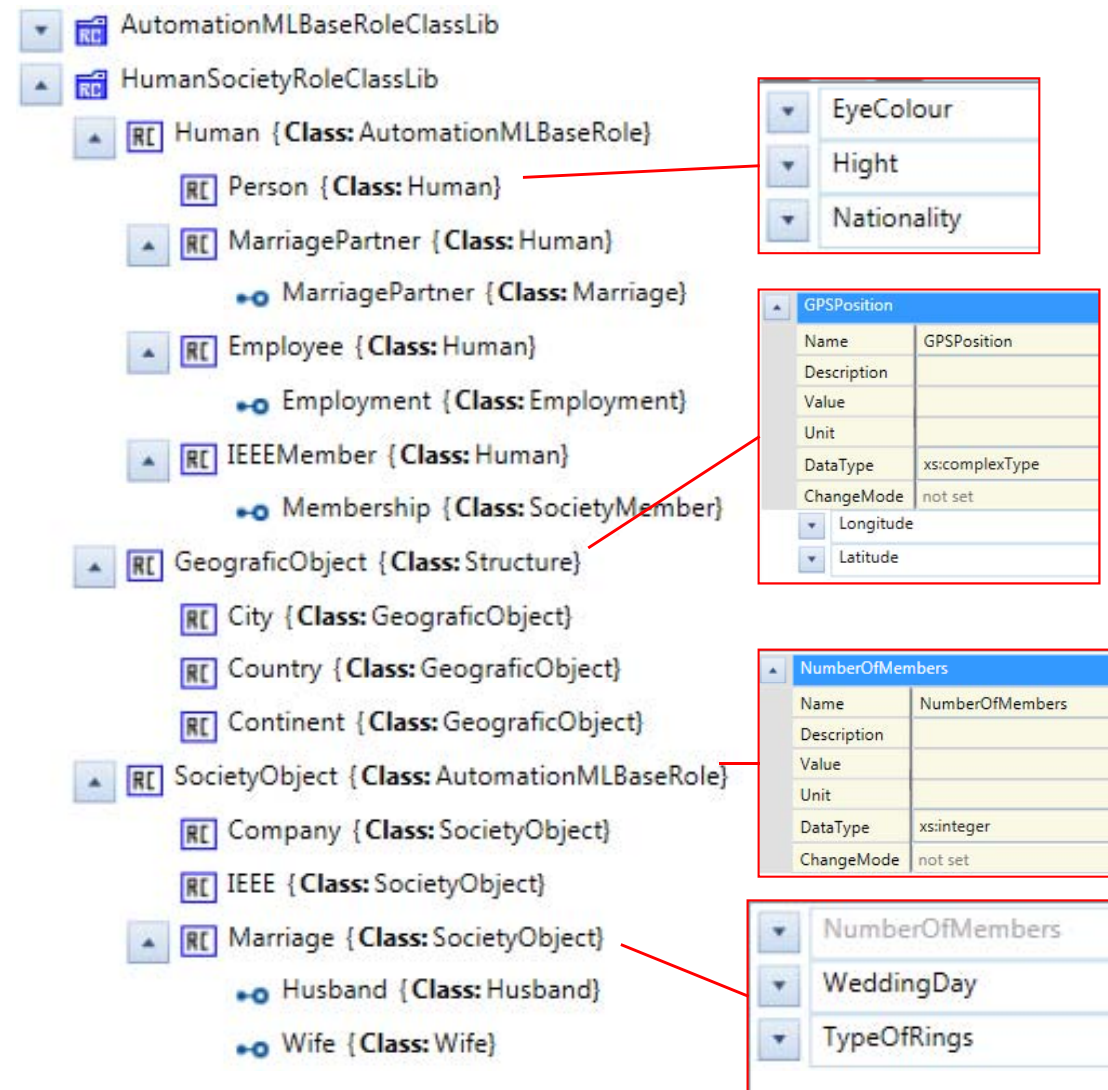
- Definition of the abstract semantics of information objects exchanged within an project.
- Seen as a
 - Requirement to an exporter to provide specified information (attributes and interfaces) and
 - Assurance to an importer to receive specified information (attributes and interfaces).

1. Identify necessary concepts

- What are the objects relevant in the modeled system?
- What are the concepts addressed within the involved tools

2. Create relevant role classes

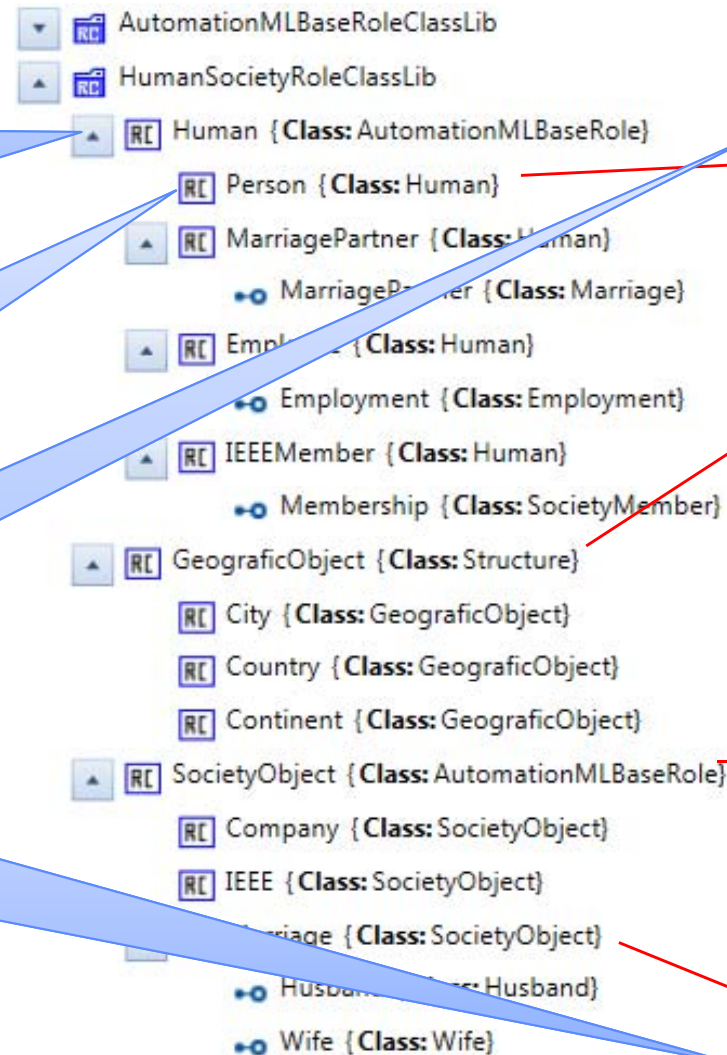
- Set up the role
- Derive role from relevant roles of higher abstraction level
- Add attributes
- Add interfaces



Definition of generic
role classes

Definition of
specialized role
classes

Definition of
appropriate
attributes



EyeColour
Hight
Nationality

GPSPosition	
Name	GPSPosition
Description	
Value	
Unit	
DataType	xs:complexType
ChangeMode	not set
Longitude	
Latitude	

NumberOfMembers	
Name	NumberOfMembers
Description	
Value	
Unit	
DataType	xs:integer
ChangeMode	not set

NumberOfMembers
WeddingDay
TypeOfRings

■ Interface classes

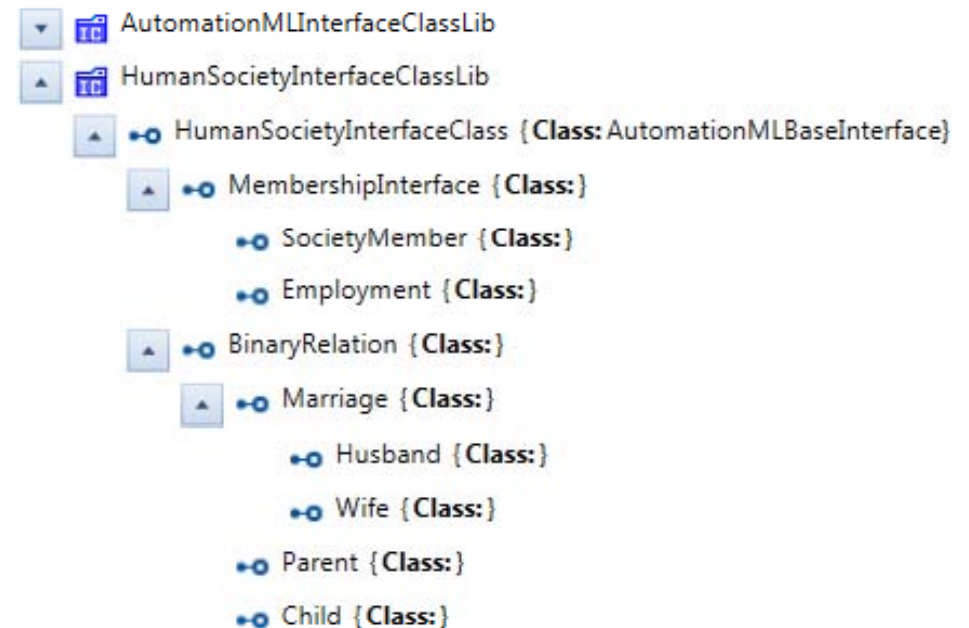
- Are abstract concepts for the relation between objects
 - Within an project
 - To external data
- Describe contact points for relations (describe only the one side of the relation)
- Can specify properties of the contact point
 - Like direction or type

1. Identify necessary concepts

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

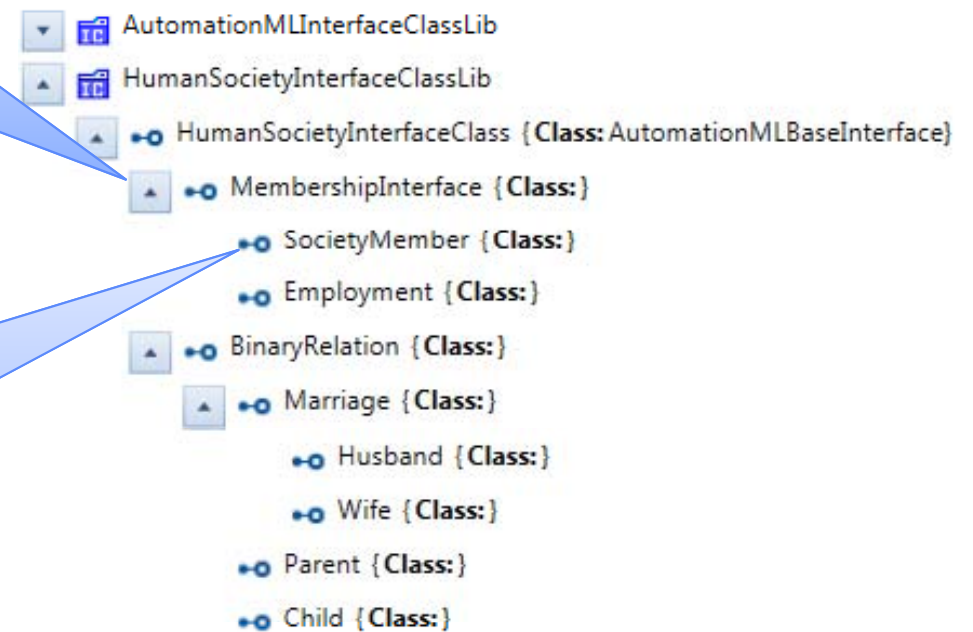
2. Create relevant interface classes

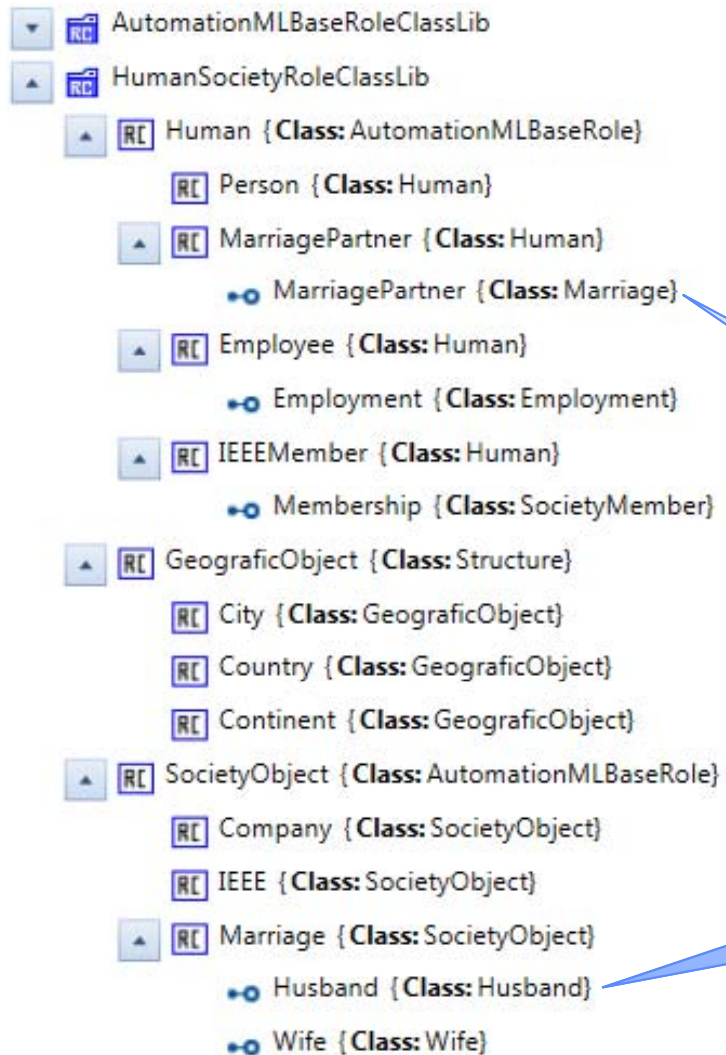
- Set up the interface class
- Derive interface class from relevant interface class of higher abstraction level
- Add attributes



Definition of generic
interface classes

Definition of
specialized
interface classes





Add interfaces to
role classes

■ System unit classes

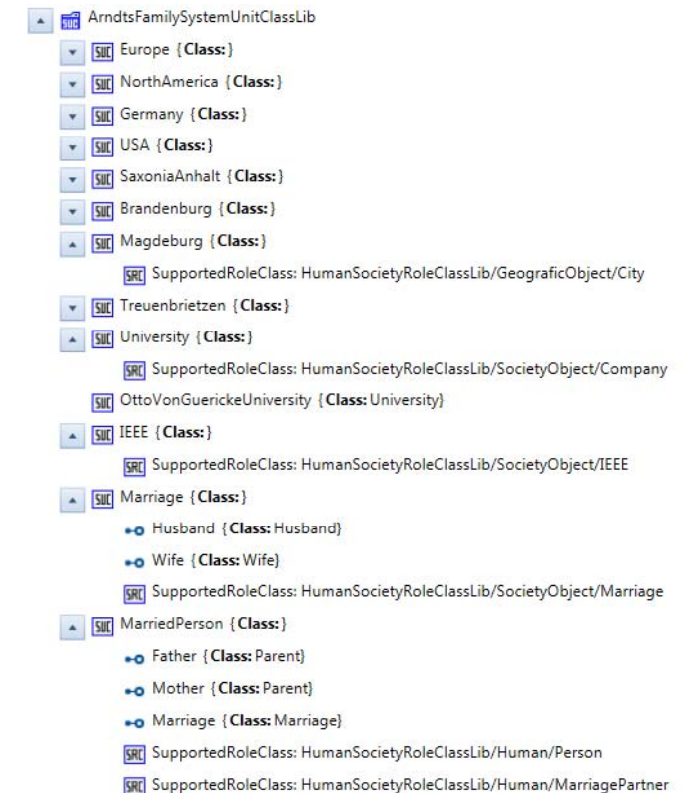
- Describe concrete types of objects reused in the engineering of production systems
 - Like married humans or cities
- Specify the necessary properties (attributes) and relations (interfaces) of the objects to be modeled
- Specify roles (possibly more than one) they “implement”
- Specify their internal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

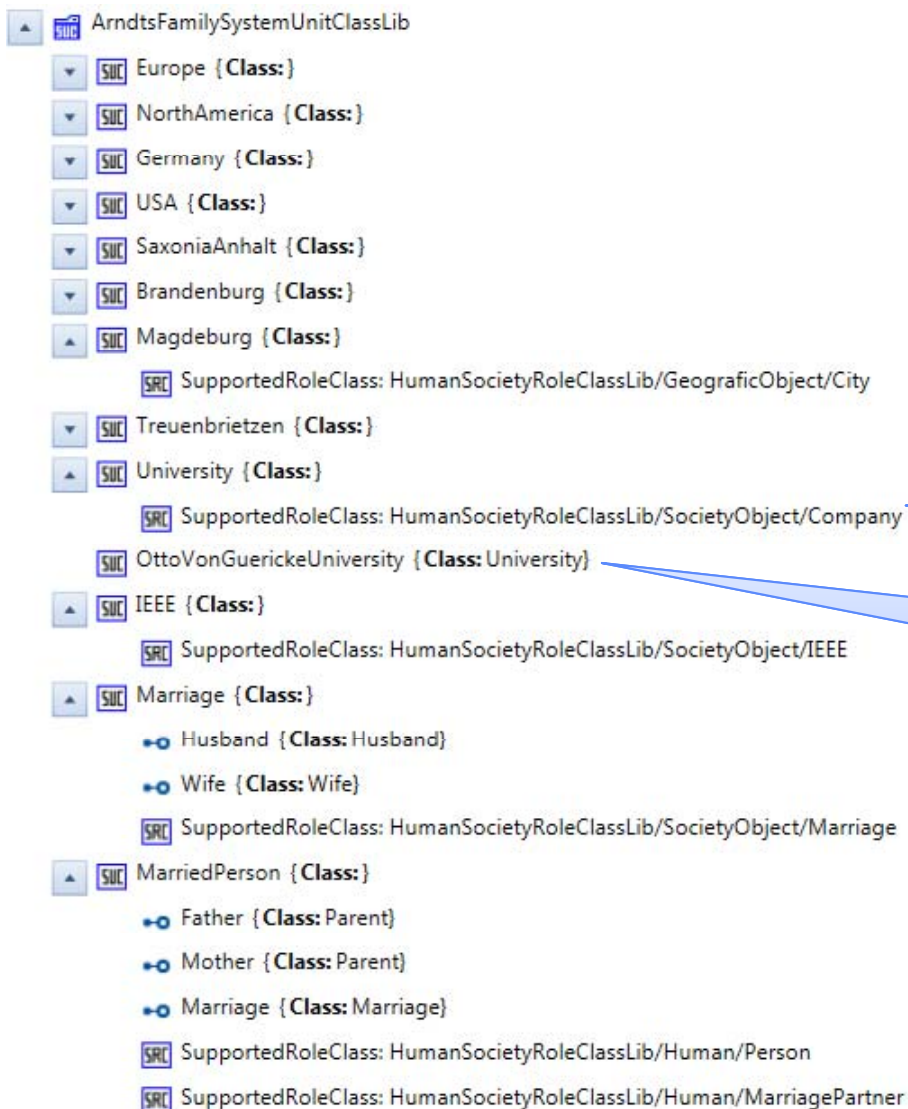
1. Identify necessary reusable objects

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

2. Create relevant system unit classes

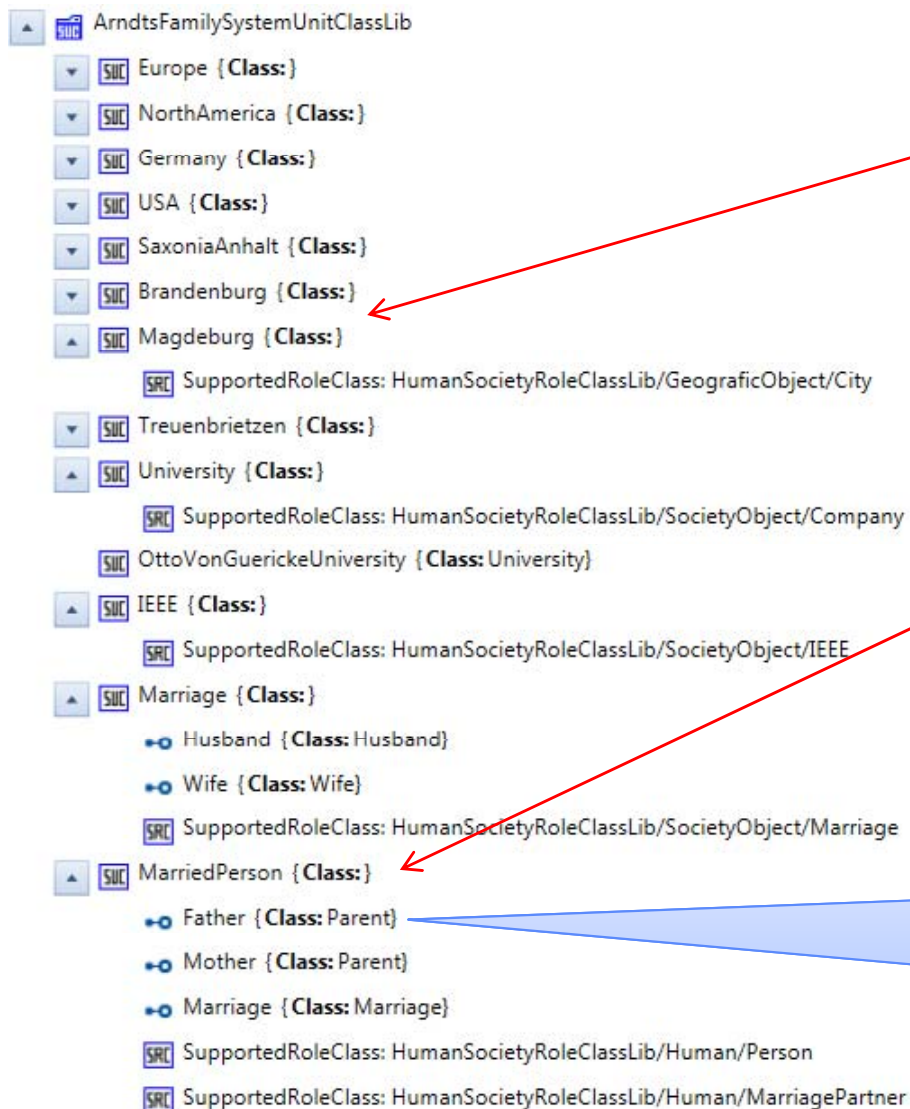
- Set up the system unit class
- Derive system unit class from relevant system unit class of higher abstraction
- Add relevant attributes
- Add relevant interfaces
- Integrate relevant substructures
- If relevant add geometry / kinematics model
- If relevant add behavior model





Give system unit class a relevant role

Derive system unit class from higher abstraction levels



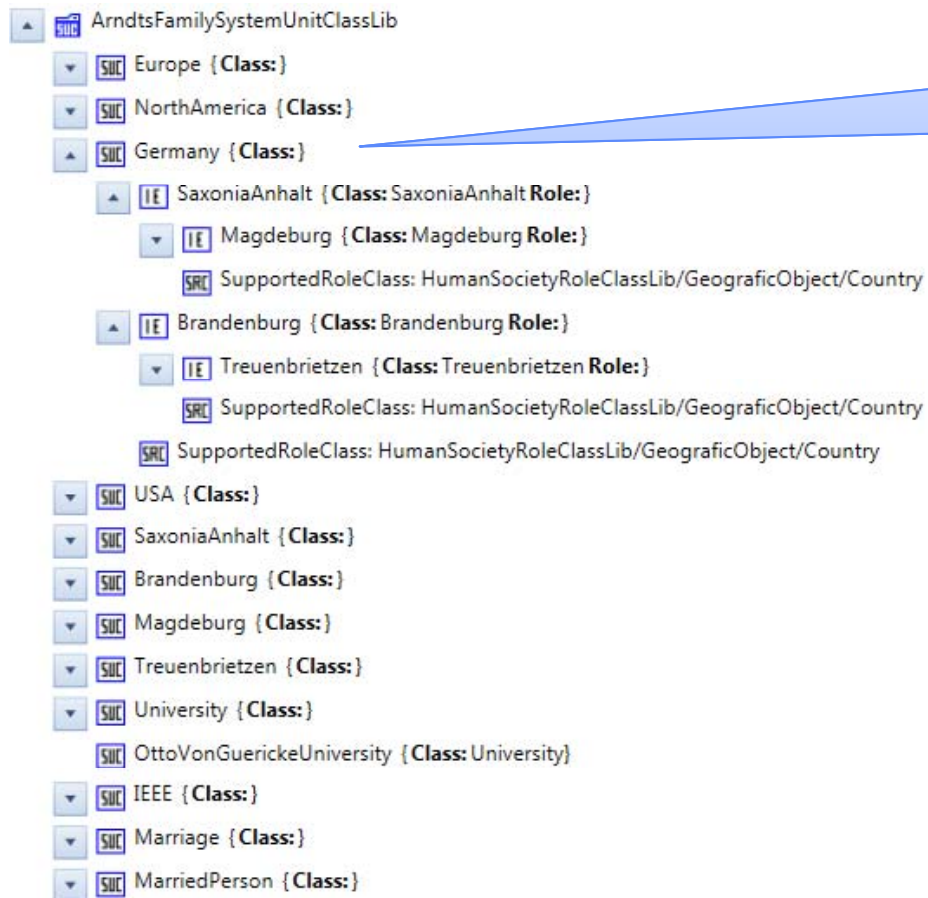
▼	Longitude
▼	Latitude

Add attributes

▼	EyeColour
▼	Hight
▼	Nationality
▼	PreferedTypeOfFlower

Add attributes

Add interfaces



Integrate
substructure

■ Instance Hierarchy

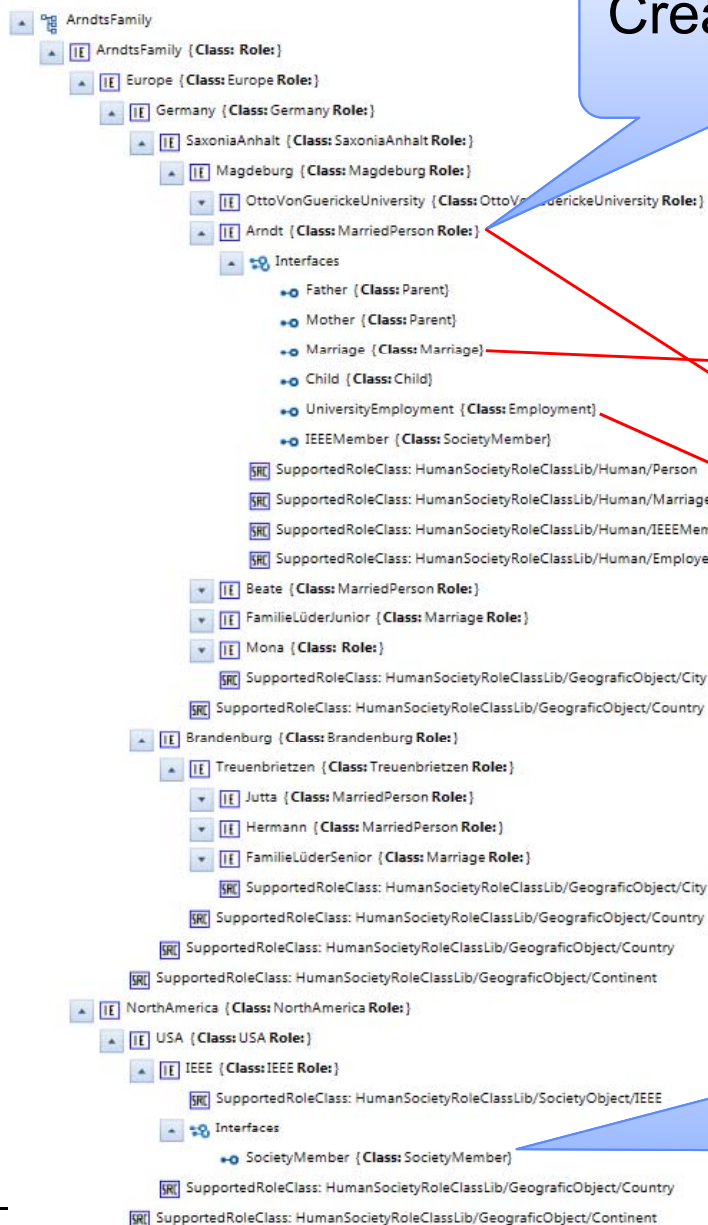
- Is the area to model the data to be exchanged (whole production system with all elements)
 - Consists of a hierarchy of internal elements with interfaces and attributes
 - Can be modeled without using created libraries
 - For reasons of semantical clearness that should be avoided
- Before starting 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

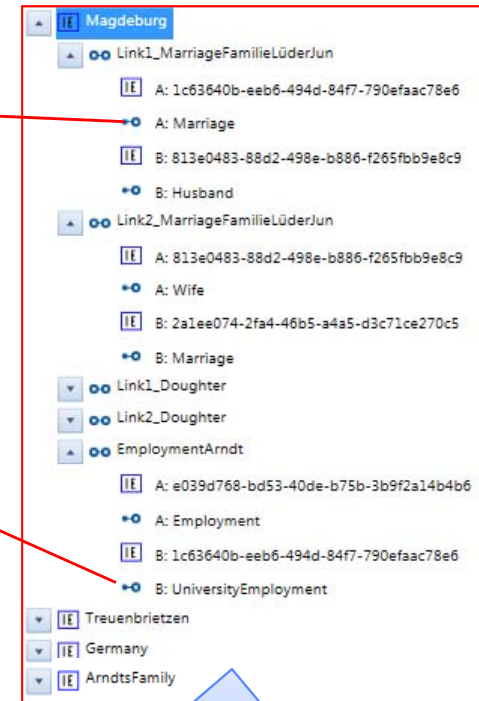
- Set up the internal element by either
 - Instantiate relevant system unit class
 - Create new internal element
- Complete / add relevant attributes
- Complete / add relevant interfaces
- Complete / add relevant substructures
- If relevant add geometry / kinematics model
- If relevant add behavior model
- Add necessary links



Create IE in
IH

Complete / add
relevant attributes
→ Each attribute
shall have a value

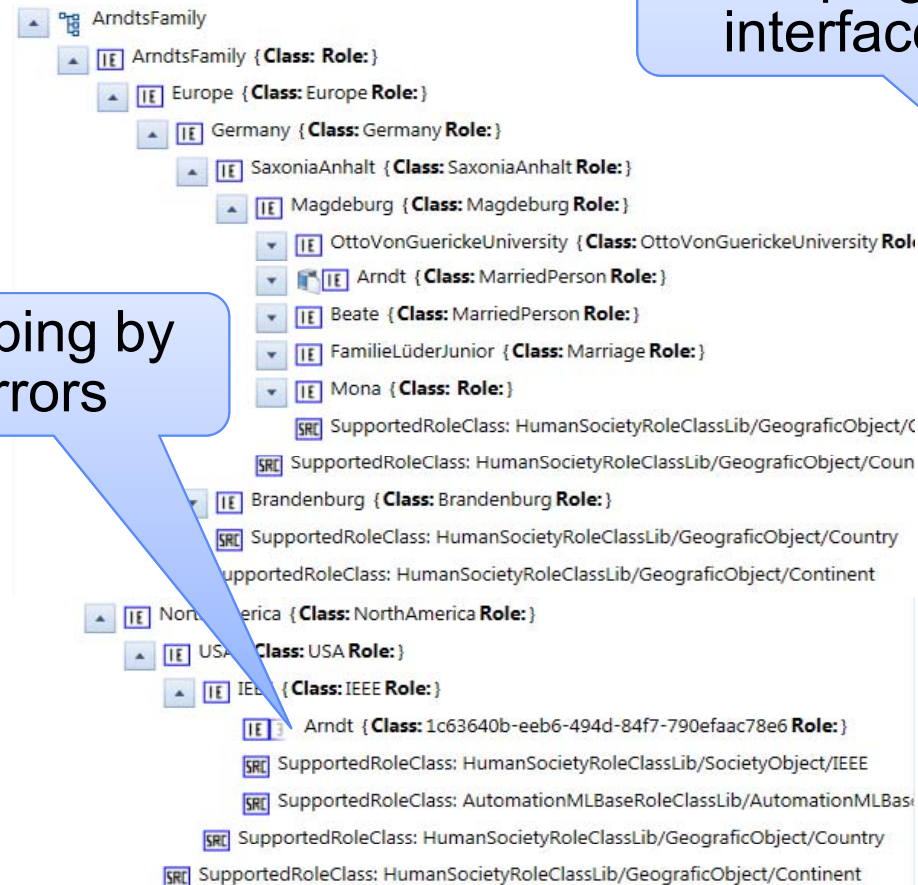
Name	Salary
Description	Amount of money received based on the employment
Value	to less
Unit	
DataType	xs:string
ChangeMode	not set
EyeColour	
Name	EyeColour
Description	
Value	brown
Unit	
DataType	xs:string
ChangeMode	not set
Hight	
Name	Hight
Description	
Value	178
Unit	
DataType	xs:integer
ChangeMode	not set
Nationality	
Name	Nationality
Description	
Value	German
Unit	
DataType	xs:string
ChangeMode	not set
PreferredTypeOfFlower	
MemberNumber	



Complete / add
relevant interfaces
Complete inter-
face attributes

Add necessary
links

■ Playing with alternatives: Example IEEE



Grouping by
Mirrors

Grouping by
interfaces

