Gefördert durch:
Bundesministerium für Wirtschaft und Energie
aufgrund eines Beschlusses des Deutschen Bundestages

From AutomationML to ROS:

A Model-driven Approach for Software Engineering of Industrial Robotics

4th AutomationML User Conference

Yingbing Hua, Björn Hein

Karlsruhe Institue of Technology
Engler-Bunte-Ring 8
76131 Karlsruhe
Tel. +49 721 608-47122
E-Mail: yingbing.hua@kit.edu

The ReApp Project

Gefördert durch:

Bundesministerium für Wirtschaft und Energie

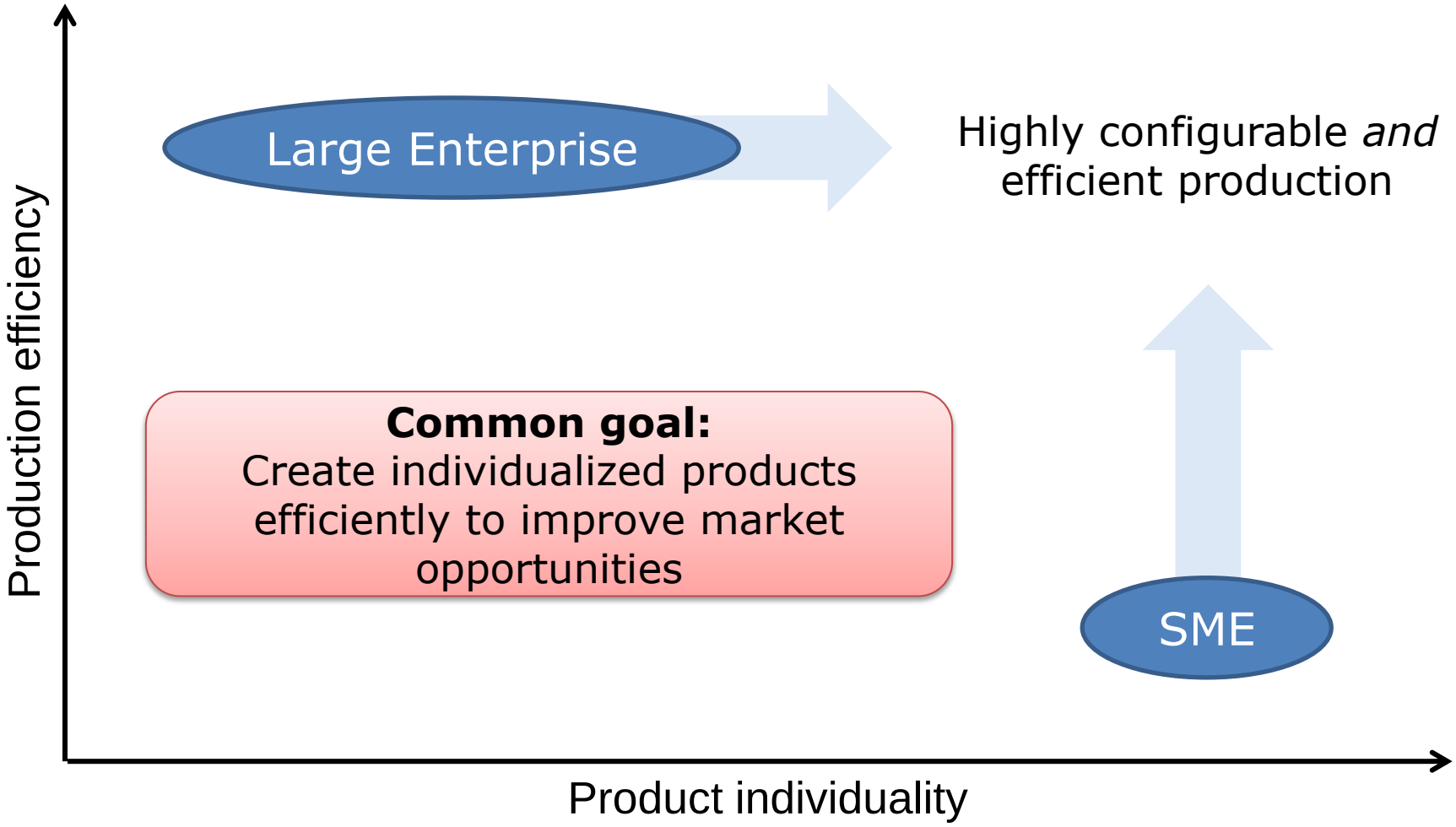
aufgrund eines Beschlusses des Deutschen Bundestages

AUTONOMIK FÜR INDUSTRIE 4.0

- **Re**usable Robot **App**lications for Flexible Robot Systems based on ROS-Industrial
- Program: Autonomic for Industrie 4.0
- Duration: 01.2014 – 12.2016

- 12 Partners
  - 3 End users
  - 3 Component providers
  - 2 System Integrators
  - 4 Research organizations

The ReApp Perspective

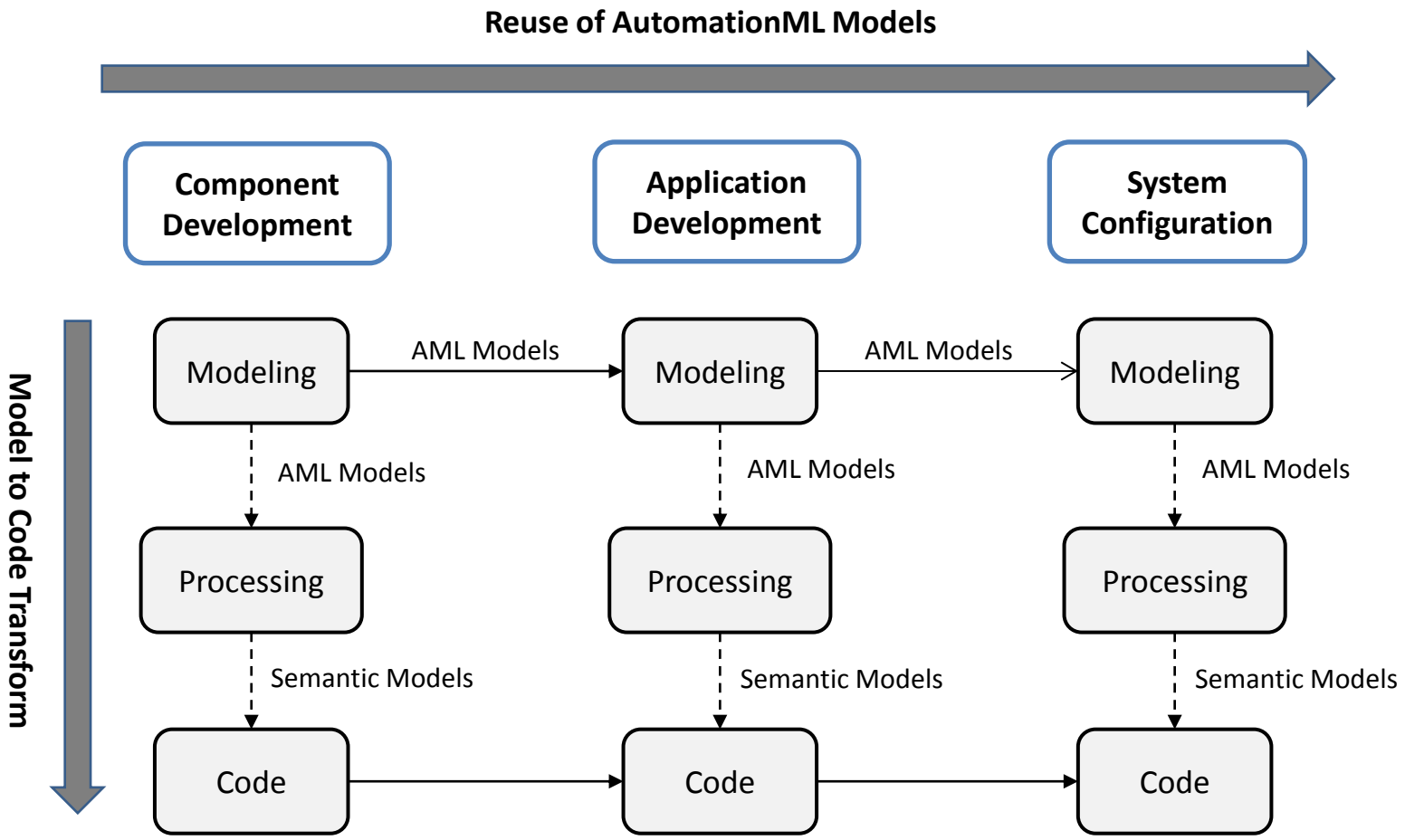


The Model-driven Approach

- On the shoulder of giants
  - AutomationML (AML)
  - Semantic Web
  - Robot Operating System (ROS)

- Role separation
  - Component developer
  - Application developer
  - System developer

The Model-driven Approach



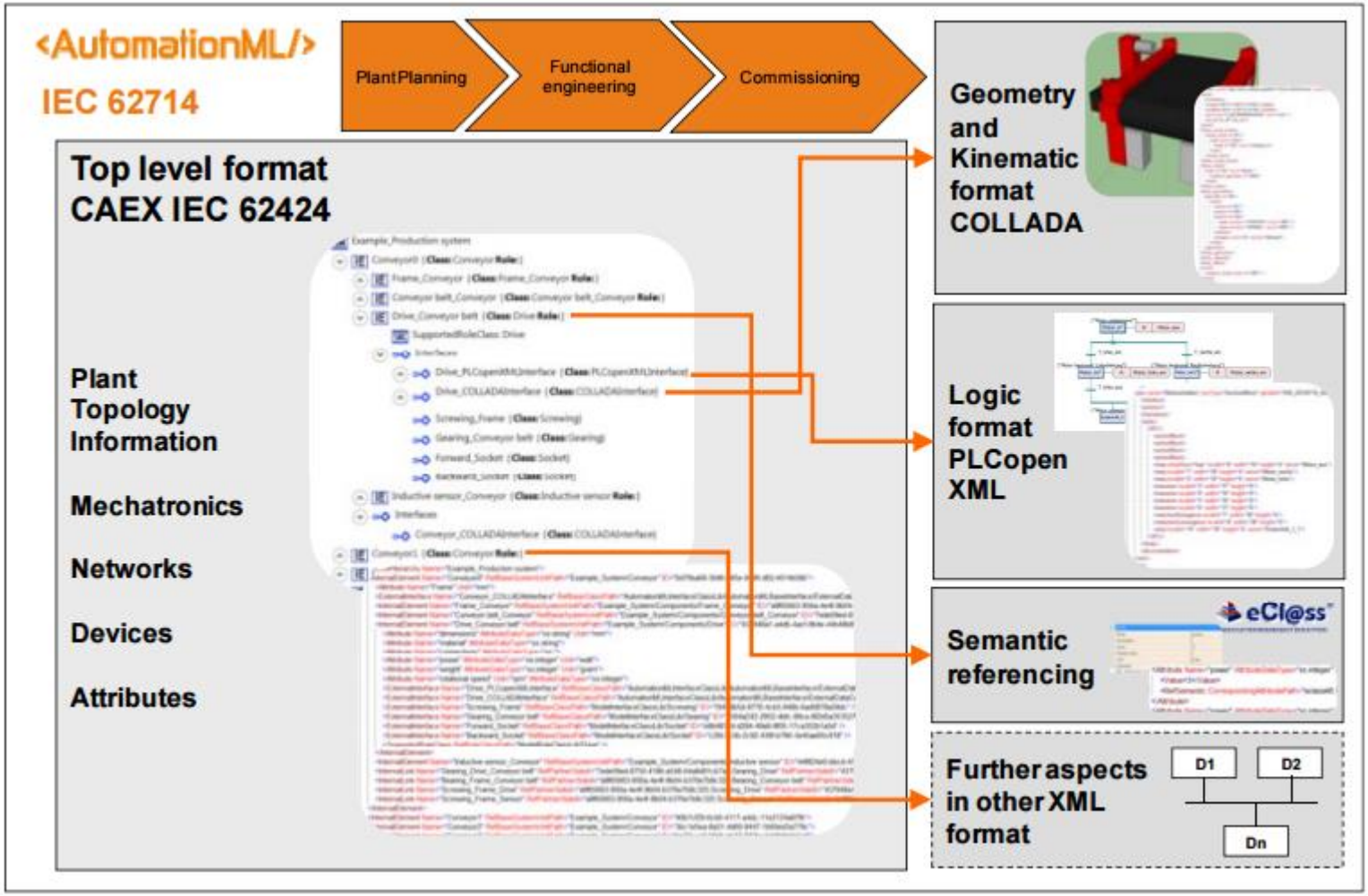
How do we generate Models for the

software engineering

of

industrial robotics?

AutomationML as Modeling Framework



Software Engineering of Robotics

- Component Development
  - Physical: dimension, weight, payload, ...
  - Geometry, kinematics
  - Interfaces
  - Functionalities

- System Development
  - Actuators
  - Sensors
  - Transports
  - Connections
  - Behaviour

ReApp Libraries

- AutomationMLBaseRoleClassLib
- AutomationMLBMIRoleClassLib
- AutomationMLCMIRoleClassLib
- AutomationMLCSRoleClassLib
- AutomationMLDMIRoleClassLib
- AutomationMLExtendedRoleClassLib
- CommunicationRoleClassLib

- AutomationMLInterfaceClassLib
- CommunicationInterfaceClassLib

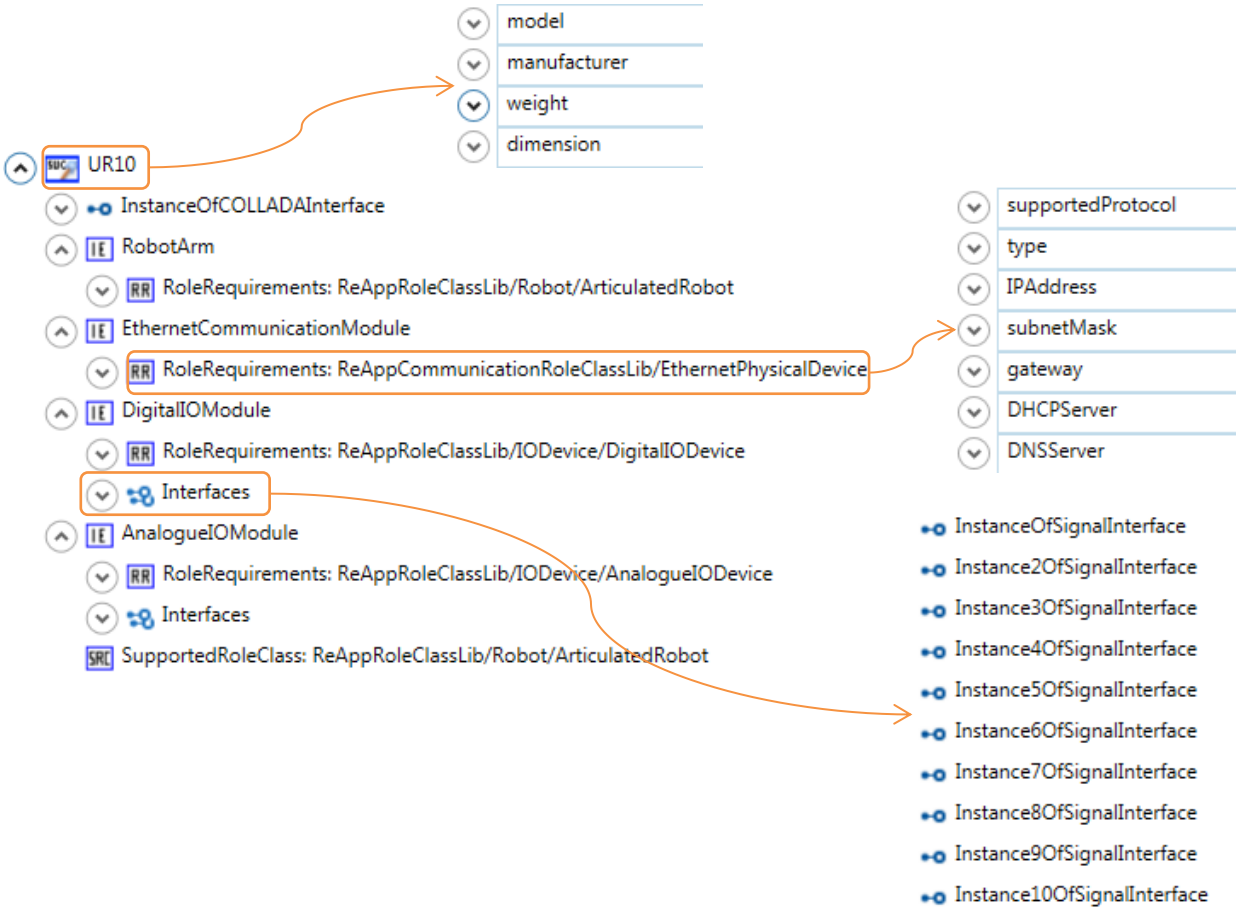
- ReAppCommunicationRoleClassLib
- ReAppRoleClassLib
  - Robot {**Class:** Robot}
  - EndEffector {**Class:** Tool}
  - Actuator {**Class:** Actuator}
  - HMI {**Class:** HMI}
  - Conveyor {**Class:** Conveyor}
  - IODevice {**Class:** IODevice}
  - Object {**Class:** AutomationMLBaseRole}
  - Sensor {**Class:** Sensor}

- ReAppCommunicationInterfaceClassLib
  - SignalInterface {**Class:** SignalInterface}
  - EthernetPhysicalPlug {**Class:** PhysicalEndPoint}
  - EthernetPhysicalSocket {**Class:** PhysicalEndPoint}
  - USBPhysicalPlug {**Class:** PhysicalEndPoint}
  - USBPhysicalSocket {**Class:** PhysicalEndPoint}
  - SerialPhysicalPlug {**Class:** PhysicalEndPoint}
  - SerialPhysicalSocket {**Class:** PhysicalEndPoint}
  - CANPhysicalPlug {**Class:** PhysicalEndPoint}
  - CANPhysicalSocket {**Class:** PhysicalEndPoint}
  - WLANPhysicalPlug {**Class:** PhysicalEndPoint}
  - WLANPhysicalSocket {**Class:** PhysicalEndPoint}
  - OpticalFiberPhysicalPlug {**Class:** PhysicalEndPoint}
  - OpticalFiberPhysicalSocket {**Class:** PhysicalEndPoint}

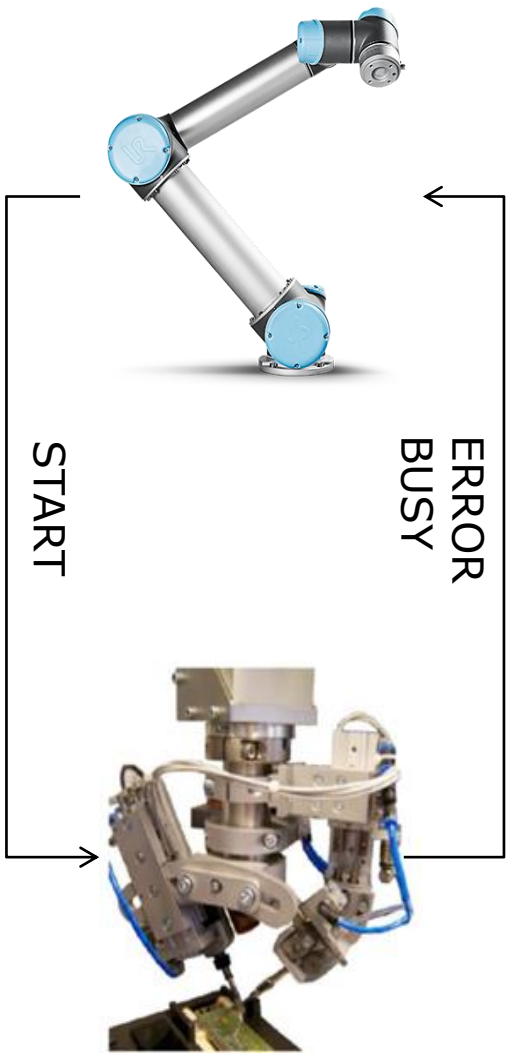
Component Modeling

Devide the component into **functional parts** → `InternalElements`

- Technical specification
- Reusable
- Configurable



Application Modeling



- DDECell
  - InstanceOfUR10 {**Class:** UR10 **Role:**}
  - InstanceOfATN-LK-R3 {**Class:** ATN-LK-R3 **Role:**}
  - InstanceOfSICK-OD1-B015H05I14 {**Class:** SICK-OD1-B015H05I14 **Role:**}
  - InstanceOfSICK-PIM60 {**Class:** SICK-PIM60 **Role:**}
  - InstanceOfMayser-EKS-026 {**Class:** Mayser-EKS-026 **Role:**}
  - InstanceOfBR-PC-910 {**Class:** BR-PC-910 **Role:**}
  - SolderRobotConnections {**Class:** **Role:** IOPhyiscalNetworkRoleClass}
    - START {**Class:** **Role:** IOPhyiscalConnectionRoleClass}
    - BUSY {**Class:** **Role:** IOPhyiscalConnectionRoleClass}
    - ERROR {**Class:** **Role:** IOPhyiscalConnectionRoleClass}
  - BumperRobotConnections {**Class:** **Role:** IOPhyiscalNetworkRoleClass}
    - TriggerEmergencyStop {**Class:** **Role:** IOPhyiscalConnectionRoleClass}
  - RobotPCConnections {**Class:** **Role:** EthernetPhysicalNetworkRoleClass}
    - RobotControl {**Class:** **Role:** EthernetPhysicalConnectionRoleClass}
  - PimPCConnections {**Class:** **Role:** EthernetPhysicalNetworkRoleClass}
  - ODMiniPCConnections {**Class:** **Role:** SerialPhysicalNetworkRoleClass}

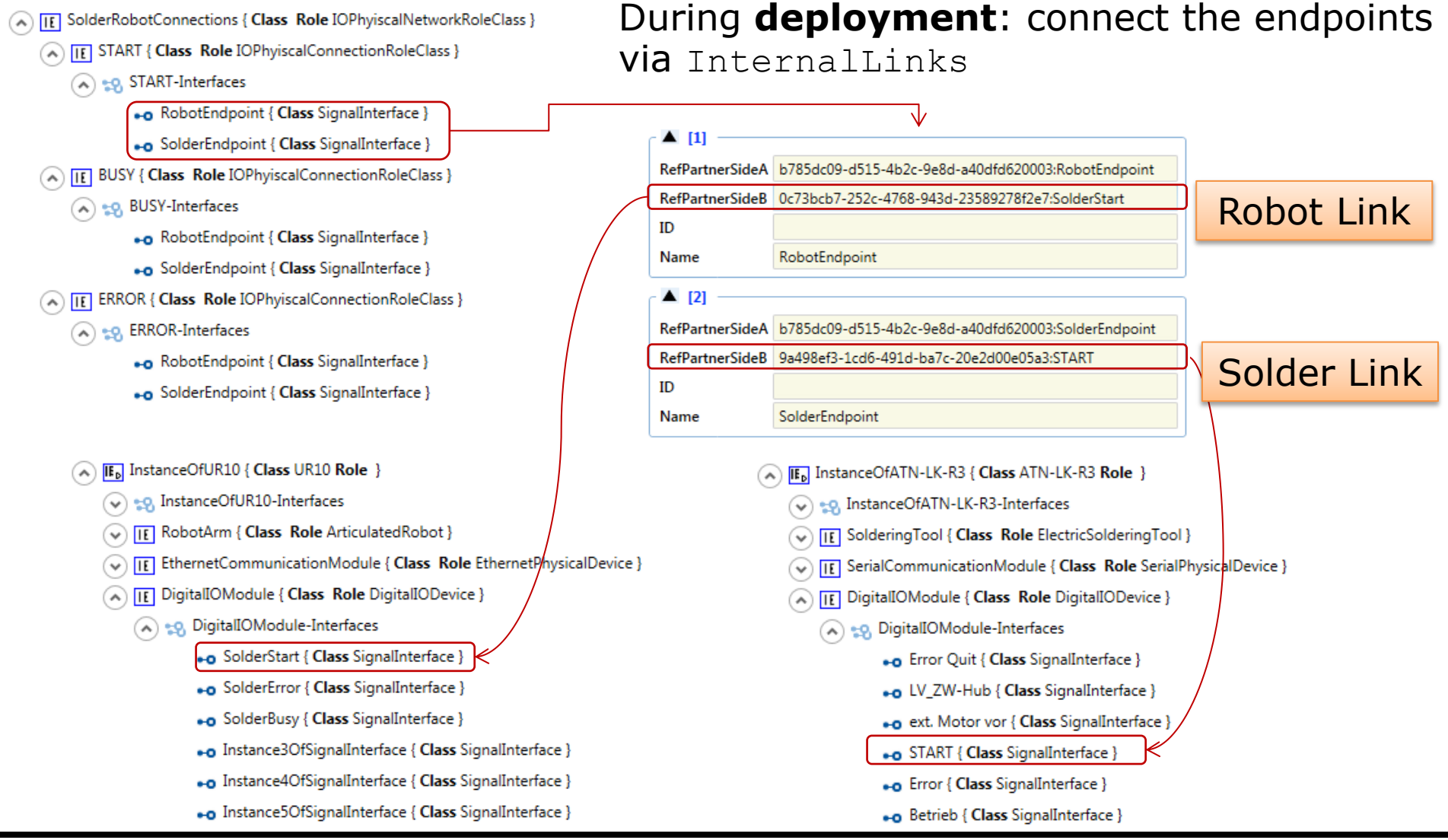
Component Instances

Required connections

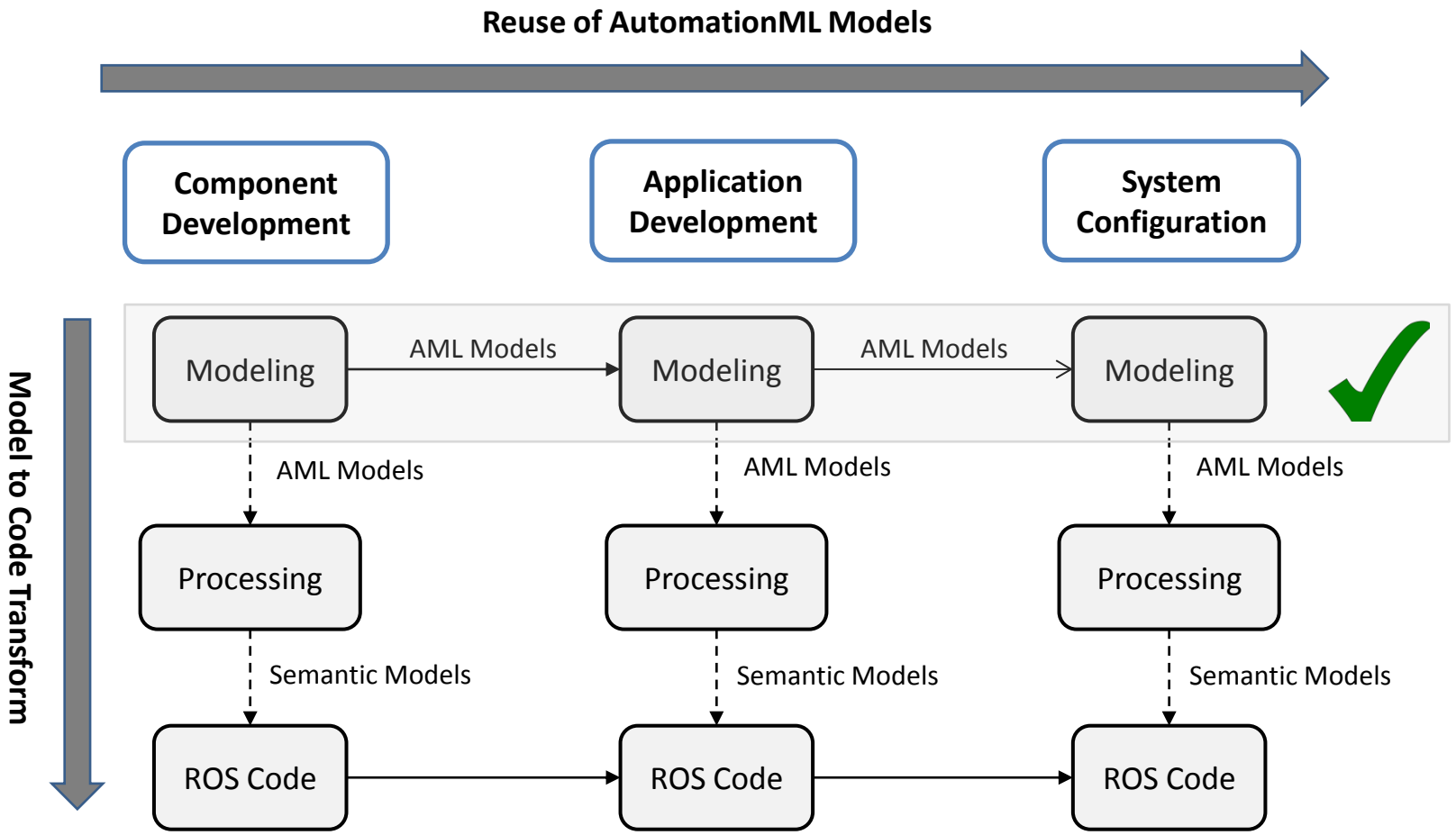
It is only a **functional modeling**: concrete connection points not known yet

System Configuration

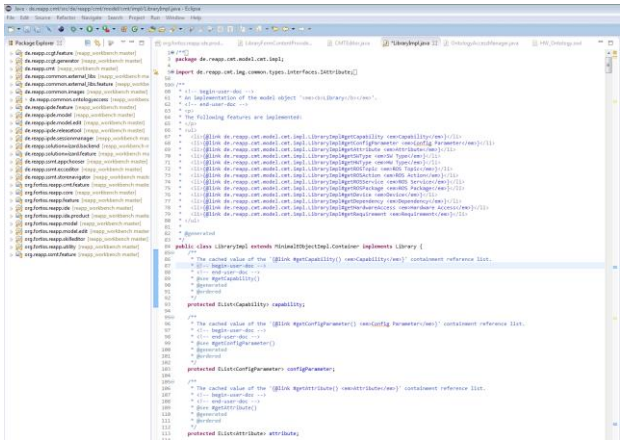
During **deployment**: connect the endpoints via `InternalLinks`



The Engineering Pipeline



Model-driven Software Engineering

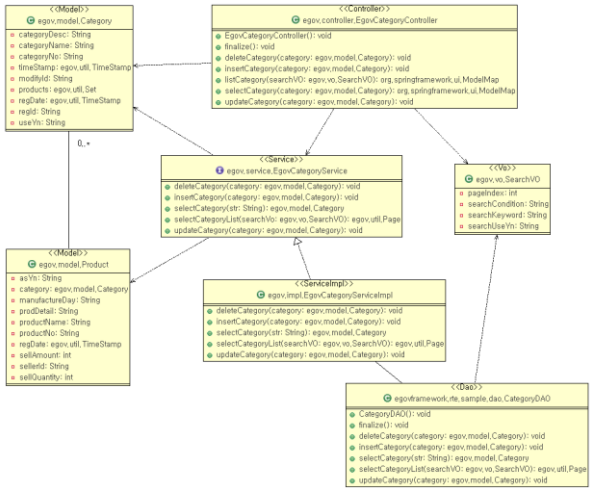


Abstraction

Code

```
<SystemUnitClass Name="UR10" RefBaseClassPath="ReAppHardwareClassLib/HardwareBaseClass">
    <Attribute Name="model" AttributeDataType="xs:string" ChangeMode="create">
        <Description>product model of this hardware</Description>
        <Value>UR10</Value>
    </Attribute>
    <Attribute Name="manufacturer" AttributeDataType="xs:string" ChangeMode="create">
        <Description>manufacturer of this hardware</Description>
        <Value>Universal Robots</Value>
    </Attribute>
    <Attribute Name="weight" AttributeDataType="xs:double" Unit="kg" ChangeMode="create">
        <Description>weight of this hardware</Description>
        <Value>28.9</Value>
    </Attribute>
    <Attribute Name="dimension" AttributeDataType="xs:double" ChangeMode="create">
        <Description>physical dimension of this hardware</Description>
        <Attribute Name="length" AttributeDataType="xs:double" Unit="m" ChangeMode="create">
            <Description>length of this hardware</Description>
        </Attribute>
        <Attribute Name="width" AttributeDataType="xs:double" ChangeMode="create" Unit="m">
            <Description>width of this hardware</Description>
        </Attribute>
        <Attribute Name="height" AttributeDataType="xs:double" Unit="m" ChangeMode="create">
            <Description>height of this hardware</Description>
        </Attribute>
    </Attribute>
    <ExternalInterface Name="InstanceOfCOLLADAInterface" RefBaseClassPath="AutomationMLInter
        <Attribute Name="refType" AttributeDataType="xs:string" ChangeMode="create">
            <Description>Reference type of the ColladaInterface</Description>
            <Value>explicit</Value>
            <Constraint Name="refTypeConstraint">
                <NominalScaledType>
                    <RequiredValue>explicit</RequiredValue>
                    <RequiredValue>implicit</RequiredValue>
                </NominalScaledType>
            </Constraint>
        </Attribute>
        <Attribute Name="refURI" AttributeDataType="xs:anyURI" ChangeMode="create">
            <Description>Reference URI of the ColladaInterface</Description>
            <Value>./UR10.dae</Value>
        </Attribute>
    </ExternalInterface>
```

Model to Model Transformation



Formal Semantics in MDSE

- Benefits of applying **formal semantics** in MDSE:
  - Externalize knowledge into ontologies
  - Sharable, extendable, adoptable
  - Extend existing descriptions by reasoning techniques
  - Formal verification and validation

Domain Knowledge

ReApp

Application Knowledge

ROS industrial

Component to ROS propagation via **complex role inclusion axioms**:

$\exists hasFunctionalPart \circ hasROSTopic \sqsubseteq hasROSTopic$

$\exists hasFunctionalPart \circ hasROSAction \sqsubseteq hasROSAction$

$\exists hasFunctionalPart \circ hasROSService \sqsubseteq hasROSService$

$UR10 \sqsubseteq \exists hasROSTopic.JointState$

$UR10 \sqsubseteq \exists hasROSAction.FollowJointTrajectory$

$UR10 \sqsubseteq \exists hasROSTopic.IOStates$

$UR10 \sqsubseteq \exists hasROSService.SetIO$

Component Compatibility

SignalInterface "In"

IOConnection ⊑

(∃!hasEndPoint.(SignalInterface ⊓ !hasValue."In")) ⊓

(∃!hasEndPoint.(SignalInterface ⊓ !hasValue."Out"))

SignalInterface "Out"

Code Generation

- Java engine to convert RDF Graph to Ecore
- Eclipse based model-to-text transformation

ROS industrial

</AutomationML>

RDF Graph

Ecore

Some Code Examples

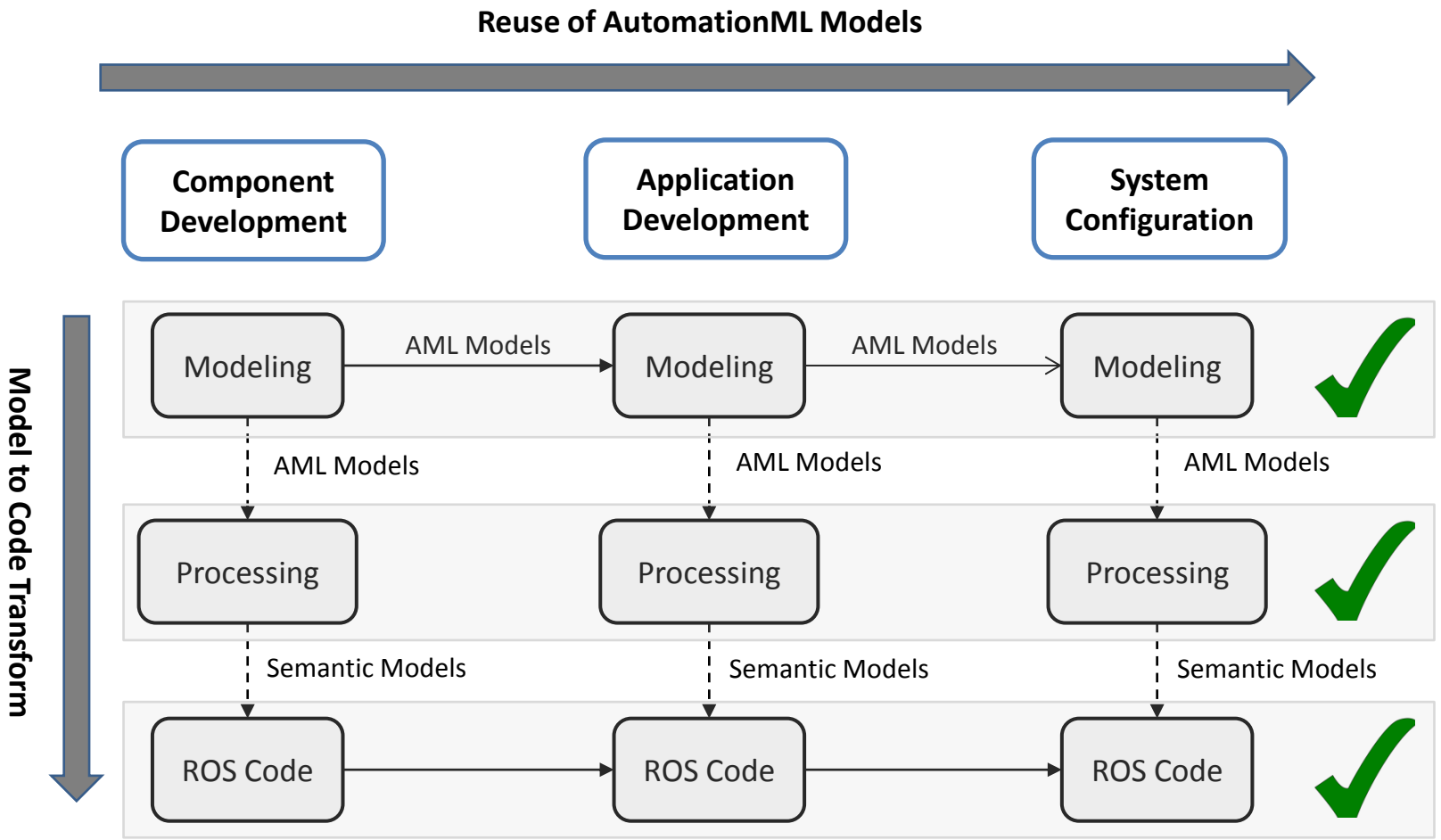
```cpp
// ROS includes
#include <ros/ros.h>
#include <dynamic_reconfigure/server.h>
#include <actionlib/server/simple_action_server.h>
#include <reapp_ur10/reapp_ur10Config.h>

// ROS message includes
#include <sensor_msgs/JointState.h>
#include <control_msgs/FollowJointTrajectoryAction.h>
#include <ur_msgs/IOStates.h>
#include <ur_msgs/SetIO.h>

// other includes
#include <reapp_ur10_common.cpp>

class reapp_ur10_ros
{
 public:
   ros::NodeHandle n_;
   ros::NodeHandle np_;

   reapp_ur10_data component_data_;
   reapp_ur10_config component_config_;
   reapp_ur10_impl component_implementation_;

   ros::Publisher joint_states_;
   ros::Publisher io_states_;
   ros::ServiceServer set_io_;
   ros::SimpleActionServer<control_msgs/FollowJointTrajectoryAction> as_;

   void configure()
   {
     component_implementation_.configure(component_config_);
   }

   void actionCallback_joint_trajectory(const control_msgs::JointTrajectoryGoalConstPtr &goal)
   {
     // Please insert your code here!
   }

   void update()
   {
     // Please insert your code here!
   }
}
```

Part of the code for robot control

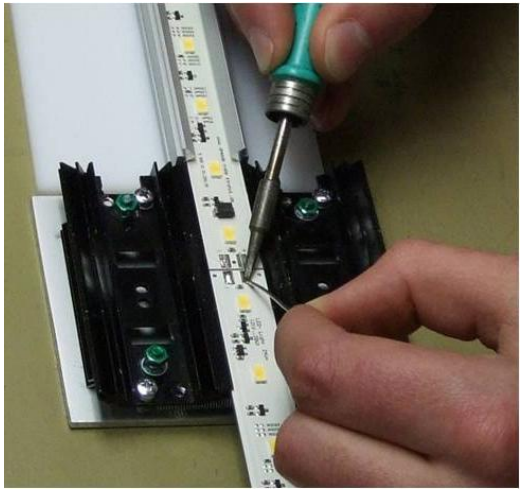
```xml
<launch>

  <node name="ur10_node" package="ur10_driver" type="ur10_node" ns="robot">
    <param name="trigger_solder_start_out" value="ur_digitalOut1"/>
    <param name="trigger_solder_stop_out" value="ur_digitalOut2"/>
    <param name="read_soler_status_in" value="ur_digitalIn1"/>
  </node>

  <node name="atn_lkr3_node" package="atn_lkr3_driver" type="atn_lkr3_node" ns="solder">
    <param name="trigger_solder_start_in" value="atn_digitalIn1"/>
    <param name="trigger_solder_stop_in" value="atn_digitalIn2"/>
    <param name="read_solder_status_out" value="atn_digitalOut1"/>
    <remap from="io_states" to="/robot_driver/io_states"/>
    <remap from="set_io" to="/robot_driver/set_io"/
  </node>

</launch>
```

Part of the code for system configuration

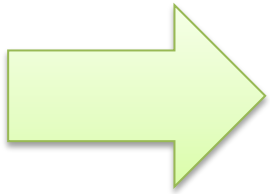
The Engineering Pipeline

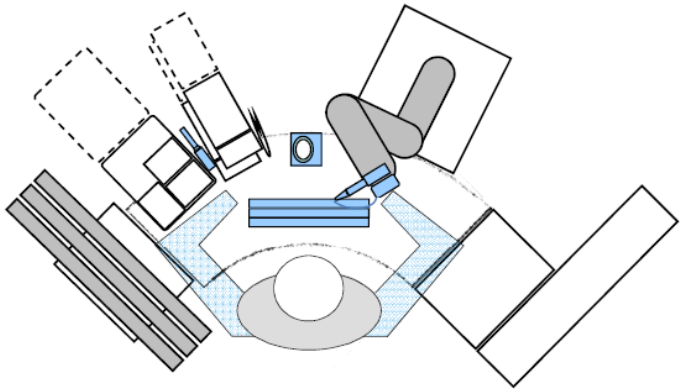


Demonstrator

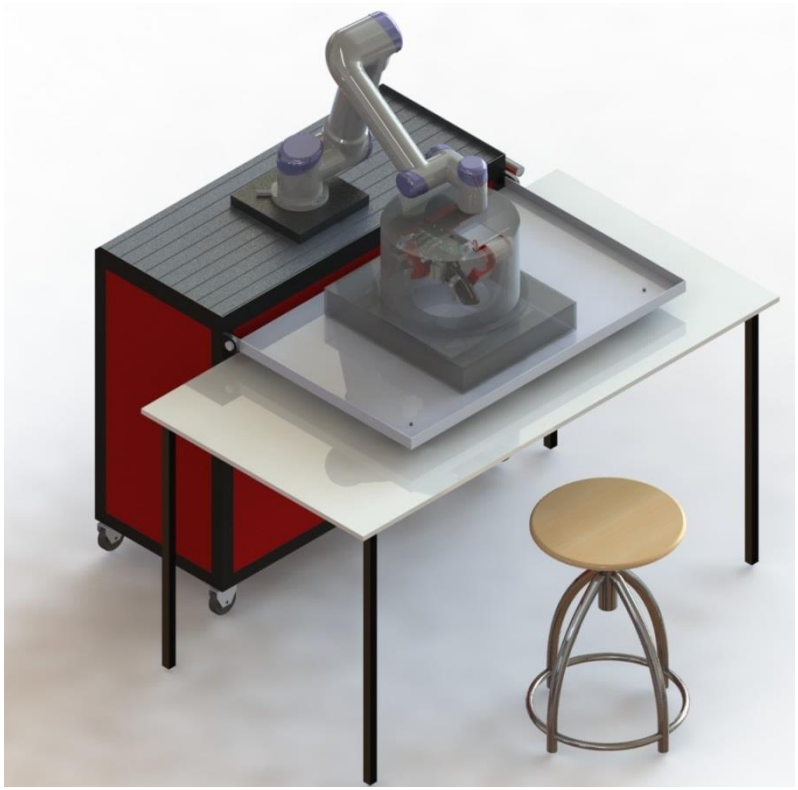


- Small lot size
- Stable quality
- human resource costs
- Flexible reconfiguration





The AutomationML Model



- ReAppHardwareClassLib
  - HardwareBaseClass { **Class** }
  - UR10 { **Class** HardwareBaseClass }
  - SICK-PIM60 { **Class** HardwareBaseClass }
  - SICK-OD1-B015H05I14 { **Class** HardwareBaseClass }
  - Mayser-EKS-026 { **Class** HardwareBaseClass }
  - ATN-LK-R3 { **Class** HardwareBaseClass }
  - BR-PC-910 { **Class** HardwareBaseClass }

- DDECell
  - InstanceOfUR10 { **Class** UR10 **Role** }
  - InstanceOfATN-LK-R3 { **Class** ATN-LK-R3 **Role** }
  - InstanceOfSICK-OD1-B015H05I14 { **Class** SICK-OD1-B015H05I14 **Role** }
  - InstanceOfSICK-PIM60 { **Class** SICK-PIM60 **Role** }
  - InstanceOfMayser-EKS-026 { **Class** Mayser-EKS-026 **Role** }
  - InstanceOfBR-PC-910 { **Class** BR-PC-910 **Role** }
  - SolderRobotConnections { **Class** **Role** IOPhyiscalNetworkRoleClass }
  - BumperRobotConnections { **Class** **Role** IOPhyiscalNetworkRoleClass }
  - RobotPCConnections { **Class** **Role** EthernetPhysicalNetworkRoleClass }
  - PimPCConnections { **Class** **Role** EthernetPhysicalNetworkRoleClass }
  - ODMiniPCConnections { **Class** **Role** SerialPhysicalNetworkRoleClass }

Karlsruhe Institute of Technology

ReApp - PD3

Automated Soldering with Reusable programing components

Wiederverwendbare Roboterapplikationen für flexible Roboteranlagen basierend auf ROS-Industrial

Summary

- ✓ Modeling of robot components and systems using AML
- ✓ Semantic uplifting of AML models
- ✓ Utilizing formal semantics and automated reasoning in MDSE
- ✓ Generate ROS code to ease implementation
- ➢ Extend AML modeling with software components
- ➢ Incorporate PLCopenXML to generate process logic
- ➢ Improve generating geometric and kinematic configurations from COLLADA

Karlsruhe Institute of Technology

Thank you for the attention!

Any Questions?