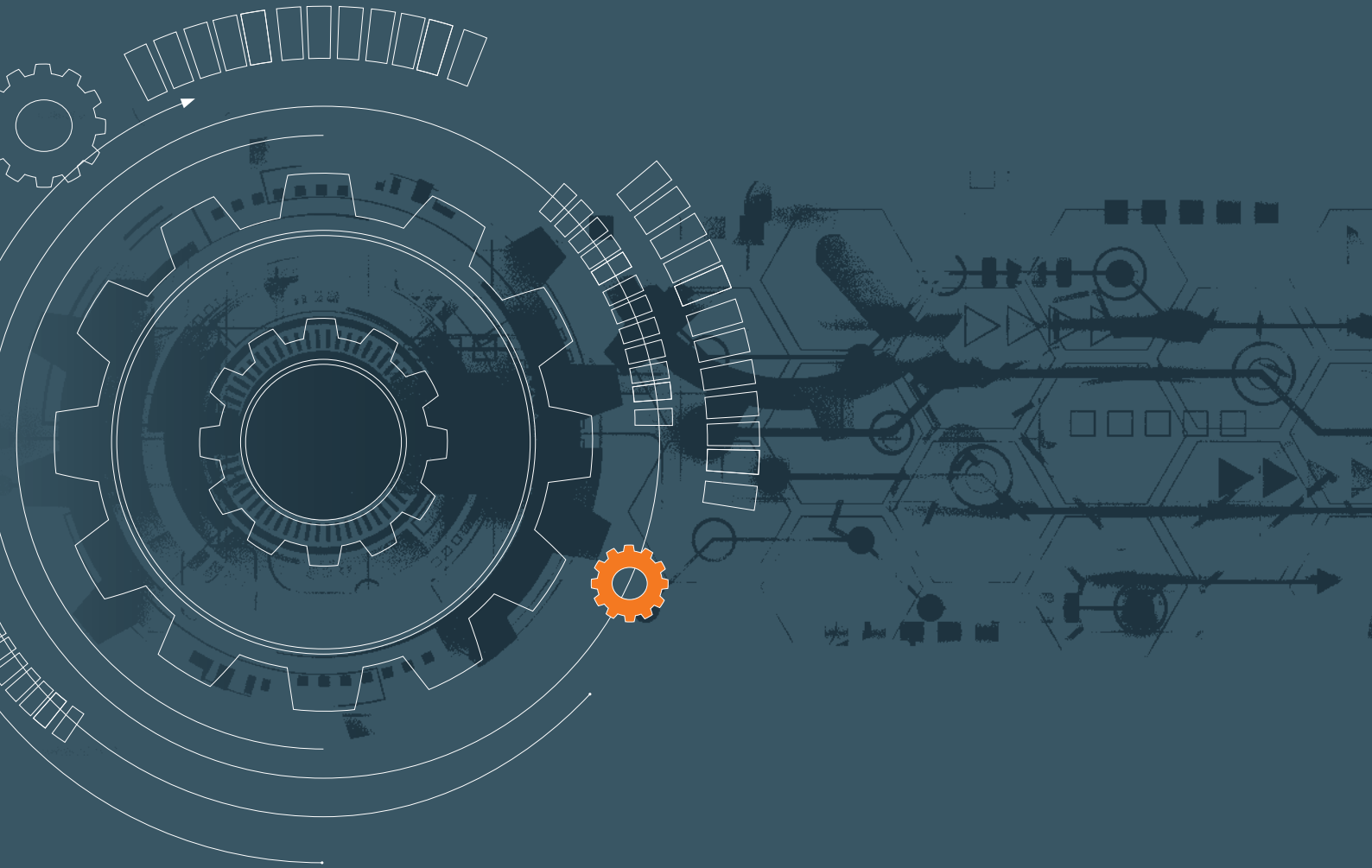


# <AutomationML/>

standardized data exchange in the  
engineering process of production systems



# CONTENT

|  |    |
|--|----|
| Preface  | 03 |
| Motivation behind AutomationML                               | 04 |
| AutomationML Basic Architecture                              | 06 |
| Modelling Topologies with AutomationML                       | 08 |
| Modelling Geometry and Kinematics with AutomationML          | 10 |
| Modelling Behaviour with AutomationML                        | 12 |
| Technological Innovations based on CAEX 3.0                  | 15 |
| Combining AutomationML and OPC UA                            | 16 |
| Communication Networks in AutomationML                       | 17 |
| Application and Best Practice Recommendations                | 18 |
| Defining Semantics in AutomationML                           | 20 |
| Implementing AutomationML Tool Interfaces                    | 22 |
| Application Recommendation: Automation Project Configuration | 25 |
| Virtual Commissioning using AutomationML                     | 26 |
| How to get started with AutomationML?                        | 28 |
| Current Research Projects around AutomationML                | 29 |
| The AutomationML Association                                 | 30 |
| AutomationML's Standardization                               | 31 |
| Contact  | 32 |

## THANKS TO OUR AUTHORS:

Rainer Drath  
Michael John  
Arndt Lüder  
Johanna Pauly  
Josef Prinz  
Markus Rentschler  
Matthias Riedl  
Miriam Schleipen  
Nicole Schmidt  
Jannis Stecken  
Anton Strahilov  
Bernhard Wally



Follow the link to our  
AutomationML Homepage  
to have a look on example files  
and find further information.

## PREFACE //

**The relevance of the engineering phase is continuously increasing within the life cycle of production systems. Users, developers and researchers therefore discuss methodologies applicable to improve efficiency, effectivity and sustainability of the engineering of production systems.**

Basic pre-condition for these efforts is the meaningful treatment of the engineering artifacts required or developed within the engineering activities. Following the idea of information logistics the engineering artifacts have to be available at the right location at the right costs, at the right time, and the right quality and amount. But the engineering processes are usually characterized by a larger amount of engineering activities with different engineering tools, different involved humans, and (partially) different companies.

### **Enabling lossless data exchange**

One means to manage the data logistics are data exchange formats enabling a lossless, consistent, and technically high-quality data exchange. On the one hand they can be applied for bilateral data exchange between different engineering activities, engineering tools, and humans. On the other hand they can be utilized to realize the interaction of sets of engineering activities, engineering tools, and humans with centralized data management systems.

AutomationML is a data format like this which has been tailored to the needs of production system engineering.

Starting in 2006 and utilizing the experiences gained within the development of data exchange formats like STEP and XML, AutomationML is and will be developed as a neutral, vendor independent, open, and freely accessible standard by an association of companies, research entities and universities that will be standardized within the International Electrotechnical Commission (IEC).

The present document is supposed to allow users, developers, and researchers, decision makers and experts, companies and public entities as well as further interested parties of various industrial and public areas to find an entrance point and basic information to understand the structure, content, application and development of the AutomationML data exchange format.

We intend to not only enable knowledge transfer but also to facilitate a broad discussion on application, applicability, and further development of AutomationML within various relevant fields and industries. The provision of this document shall thereby enforce the distribution of AutomationML.

### **Spread the word**

In this spirit, we intend to request all readers of this brochure to express their ideas, wishes, and recommendations related to the use and development of AutomationML towards the AutomationML association and its members to cooperatively develop the future of data exchange within the engineering of production systems.

### **The board of directors**

Engineering artifacts have to be available at the right location at the right costs, the right time point, and the right quality and amount.

## » MOTIVATION BEHIND AUTOMATIONML //

**The life cycle of production systems is and will be more and more digitalized, independent of the invention of the „Industrie 4.0“. Data has been playing an important role within the life cycle of production systems for ages.**

In the context of the **engineering of production systems**, design data is specified step by step. Based on the first draft planning, design data is specified or enriched until it has reached a sufficient level of detail to physically realize the production system.

Within the **phase of the physical realization** of the production system the developed digital artifacts are applied and the realization results (and partially the required changes) are documented. A system documentation is thereby achieved that covers the complete system in a detailed structure, behavior and property representation.

The **use phase** of the production system is characterized by the collection and processing of data that are either used to control the system/system element utilization or to document the system use (for example with regard to product or process quality achieved).

**Data flow** along the entire life cycle of a production system has increasingly been identified as bottleneck in terms of achieving good engineering efficiency and quality. This is one of the main drivers for the development of ideas in Industrie 4.0. Engineering constitutes one example of that.

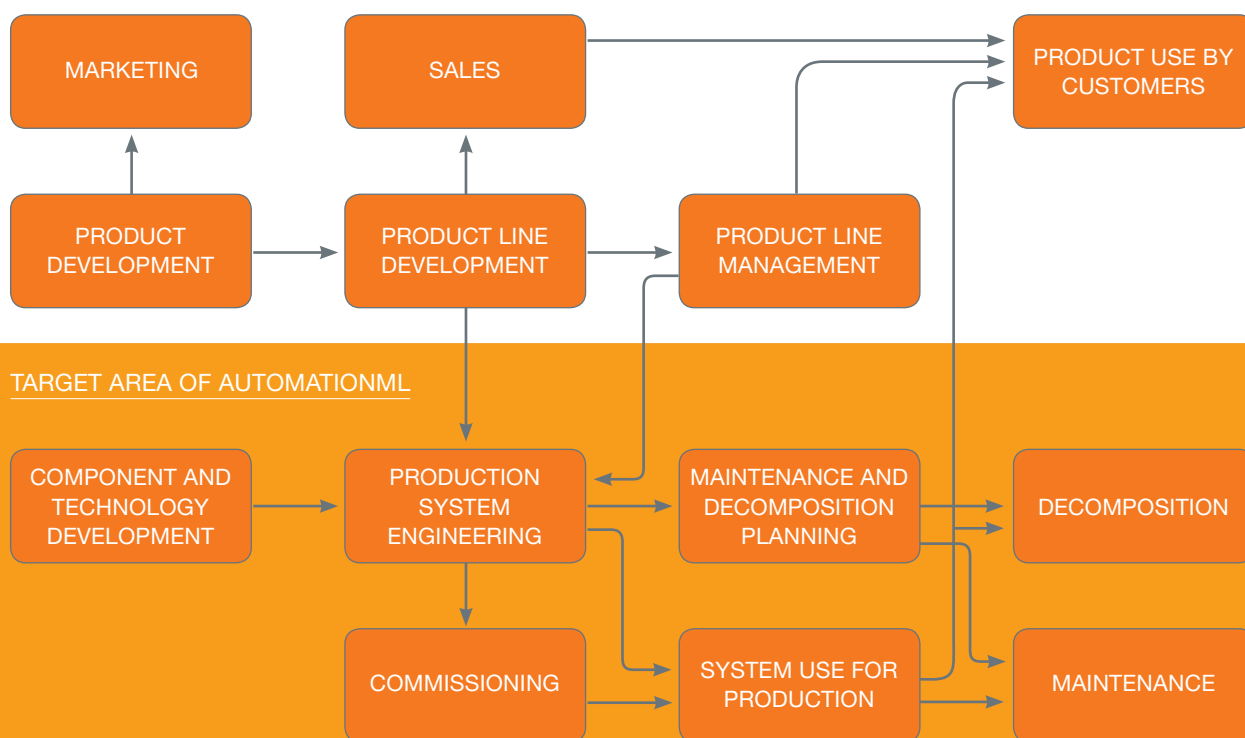
According to several studies, manual engineering activities make up more than 50 % of the overall costs of the design and realization of production systems, i.e. the overall investment. The required activities for data provision and preprocessing for engineering activities account for an essential share of these manual activities.

This is where AutomationML comes in. It has been developed as a data exchange format applicable for all production system engineering data relevant within the whole life cycle of production systems. Therefore, AutomationML provides concepts for system modelling following object oriented paradigms enabling lossless bilateral data exchange and furthermore the development of systems for centralized data management and engineering artifact libraries.

**AutomationML: vendor-independent and industrial area neutral**  
The data format can be applied for lossless data exchange along various chains of data processing systems (including engineering tools) of all industrial areas and beyond without any limits related to licensing and application costs.

Practical application experiences of the use of AutomationML are available for discrete manufacturing systems, the process industry, in large enterprises and small and medium-sized companies, in basic and detailed engineering as well as virtual commissioning, physical realization, and commissioning. In addition to that, the interaction of different companies are realized, including OEM's, system integrators and component/device vendors.

### WHERE TO USE AUTOMATIONML?



### AutomationML: Exchange engineering data along the entire life cycle of production systems.

The XML-based data format enables the vendor-independent exchange of design data between various discipline-specific design tools. It depicts system structures and system behavior for/in production systems in the usage phase and allows the modeling of plant components as data objects that combine various aspects.

AutomationML focuses on combining and adapting existing industry formats designed to interchange and drop different design issues, and provides a coherent, distributed document architecture that handles large volumes of data and offloading libraries to external files facilitated.

In doing so, AutomationML addresses the most important requirement for consistent and lossless data exchange, namely the mapping of all design data that is relevant for at least two partners involved in the data exchange. Summarizing this data results in design information which AutomationML must be able to map.

- **Topology information:** This information set describes the hierarchical structuring of the production system from the production system level, through the cell and resource levels down to the levels of the equipment and mechanical components. In addition to the individual hierarchy elements, it also covers their relations and descriptive properties.
- Information related to the **mechanical properties:** This amount of information describes the mechanical design of the production system including its geometric and kinematic properties. It is usually developed as a technical drawing of a MCAD tool. In addition, this information set contains physical properties of the production system and its parts such as forces, velocities and angles of rotation as well as chemical properties such as material information.

- Information related to **electrical, pneumatic and hydraulic properties:** This information includes the electrical and fluidic design of the production system, including wiring and piping, as it might be produced with ECAD and FCAD tools. For this purpose, it includes the connected components on the one hand and, on the other hand, the connections between them with their interfaces and the respective electrical and fluidic related properties.

- Information related to the **functions of the production system:** This information set serves to characterize the functions of the production system and its components. It includes behavioral models of uncontrolled (physical, chemical, etc.) behavior and controlled behavior. These are the functional parameters and the technical parameters. The functions considered by this information set do not only relate to the functions required in production but also to all auxiliary, diagnostic, maintenance, and other functions.

- Information related to the **control of the production system:** This information contains all the information related to the control unit. These are the hardware configuration, control code and control parameters in particular.

- **Further information:** This information set subsumes further necessary information such as relevant business information like the manufacturers' article numbers or prices, organizational information such as installation and maintenance instructions, manuals, etc.

The generic structure of AutomationML allows the integration of additional sets of information, if relevant to the representation of a production system. Examples include information regarding production planning for a production system.

### Compatible with the PPR-Concept

In addition to the information given above, AutomationML also allows the application of the so-called Product Process Resource (PPR) concept for the consistent presentation of the relations between products, the production processes necessary for their production and the resources that carry out these processes. The functional, technical and economic relationships between the PPR elements can be mapped.

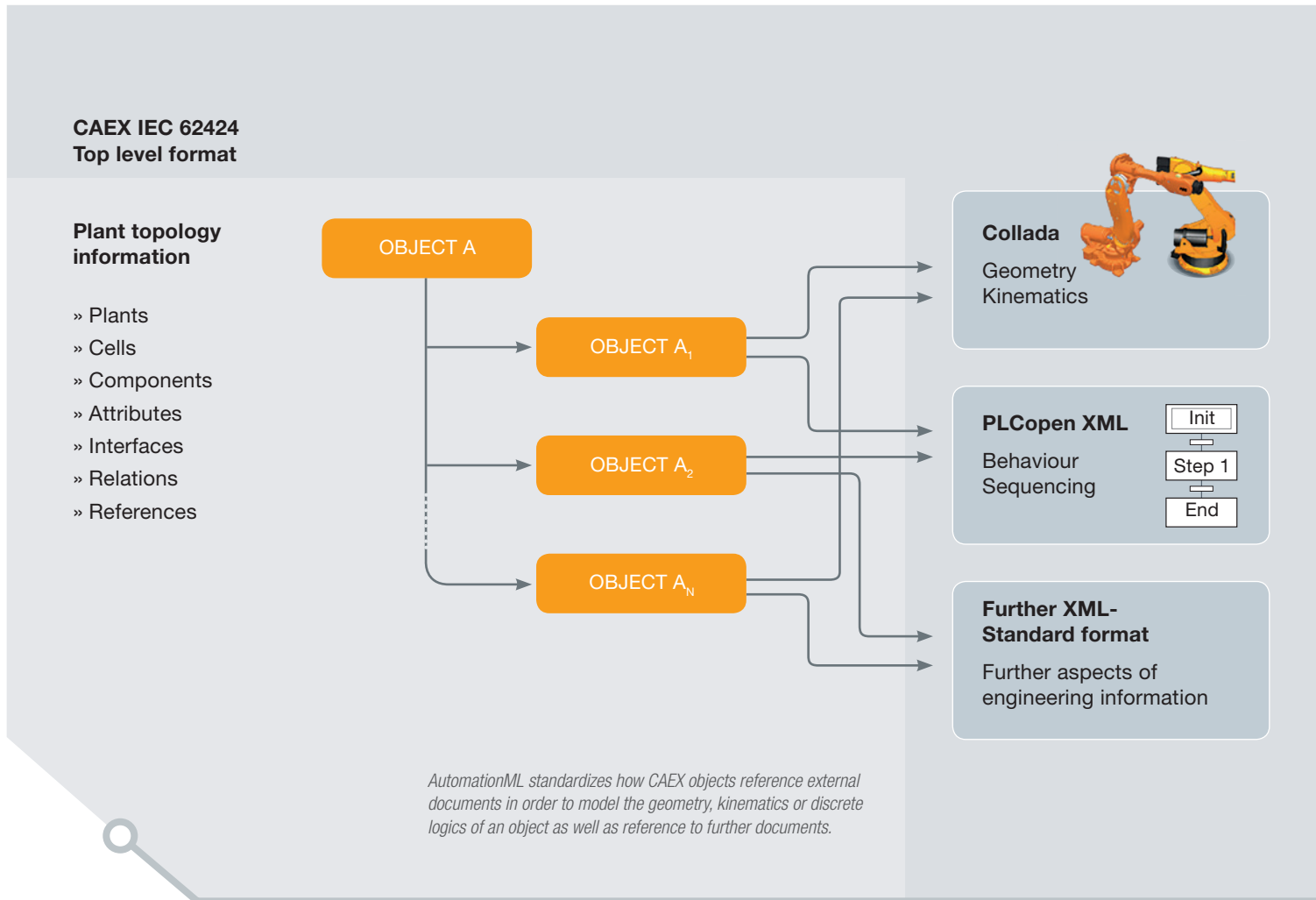
## » AUTOMATIONML BASIC ARCHITECTURE //

AutomationML has a lean and distributed file architecture. It does not define any new file format but combines existing established XML data formats which have been proven in use for their specific domain. This is why the normative part of the IEC62714-1:2018 document consists of 32 pages only. The data formats for the following modelling domains are:

- object topologies including hierarchies, properties and relations of objects: CAEX according to IEC 62424 (see page 8)
- geometries and kinematics of objects: COLLADA 1.4.1 and 1.5.0 (ISO/PAS 17506:2012) (see page 10)
- discrete behavior of objects: PLCopen XML 2.0 and 2.0.1; in addition, IEC62714-4 will allow the usage of IEC61131-10 (see page 12)

CAEX according to IEC62424 forms the base of AutomationML. It stores object-oriented engineering information, e.g. a plant hierarchy structure (see AutomationML topology). Each CAEX object can contain properties and reference geometry, kinematics or logics information stored in third party XML files. This enables cross-domain modelling and is designed for future extension.

### AUTOMATIONML BASIC ARCHITECTURE



AutomationML combines existing established XML data formats which have been proven in use for their specific domain.

### Essential features of AutomationML

- AutomationML allows the modelling of a wide range of engineering data.
- AutomationML is object-oriented and supports the modelling of object networks.
- AutomationML does not provide domain-specific semantics, but allows the modelling of existing and future semantic standards. This is a result of a strict separation of syntax and semantics.
- AutomationML allows the modelling of heterogeneous data (a mix of proprietary and standardized data).
- AutomationML provides strict identification capabilities for objects based on UUIDs.
- AutomationML supports storage of version information including version history information.
- AutomationML provides data source information in high granularity.
- AutomationML has a built-in ability to be extended by new libraries, new external documents and new concepts.
- AutomationML has an active community that continuously develops best practice recommendations and software support.
- AutomationML requires low development effort by the AutomationML engine.



# Lenze

## AutomationML – Efficiency in the engineering process

The development of a machine or of a production system requires us to carry out a great number of tasks in a wide variety of disciplines. In order to optimise the amount of time spent and the system produced while at the same time reducing the costs and the risks, the engineering processes are increasingly being carried out virtually and supported by the use of a wide variety of tools.

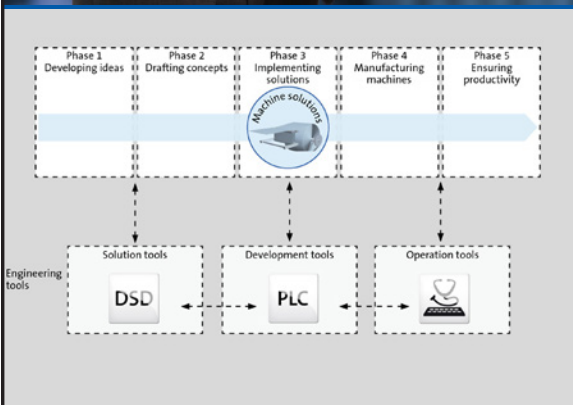
As a provider of drive and automation solutions, Lenze offers tools for the optimal drive design, and for the programming, commissioning and diagnosing of machines and production systems.

Efficiency, though, can only be achieved if data can be exchanged between the different software tools that are used in the different phases of the machine's life cycle. To make this possible, we need access to the data models and interfaces of the Lenze module system and to those of the software tools from third-party suppliers. When speaking of this data and these data models and interfaces in the context of Industrie 4.0, we talk of digital twins, which display the contents of mechatronic domains, and also of a standardised exchange format.

An exchange of data in the tool chain between one's own software tools and third-party software tools used to be almost impossible with the solutions that were available. Now, though, AutomationML provides us with a central data exchange format that is standardised (IEC62714) and therefore offers a clear structure for data transfers. It also includes established standards such as CAEX, COLLADA or PLCopen.

The time and effort involved in the initial set-up of AutomationML should not be underestimated, but it is outweighed by the benefits, especially when several tools are involved or have to be introduced later in the process.

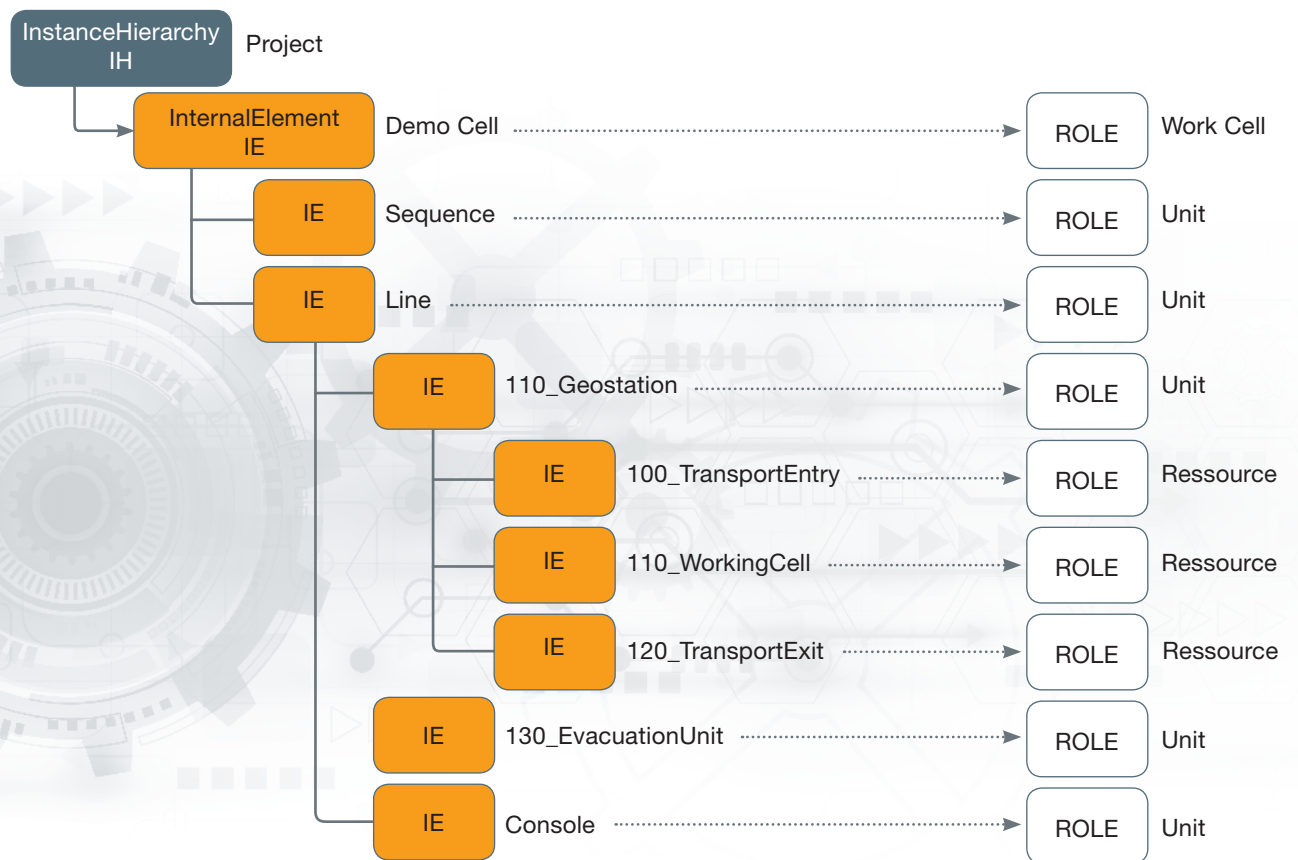
Lenze has already gained positive experience of this format in optimising the planning process for drive solutions in the engineering chain. Prototypal work done with the DSD drive designer tool showed that the AutomationML format can be used to import data from a CAD tool, process it, and then transfer it to simulation tools.



## » MODELLING TOPOLOGIES WITH AUTOMATIONML //

Object hierarchies in CAEX form the core of AutomationML. A CAEX object is a data representation of any asset. It can model physical assets, e.g. a motor, a robot, a tank; or abstract assets like a function block, a model or a folder. CAEX allows to link those objects to systems, since every physical or logical system is characterized by internal elements (objects) which may contain further internal elements, and all elements may have interfaces, attributes and connections with each other. Finally, CAEX allows the modelling of any plant topology, communication topology, process topology, resource topology etc., (see figure below).

### EXEMPLARY PLANT TOPOLOGY

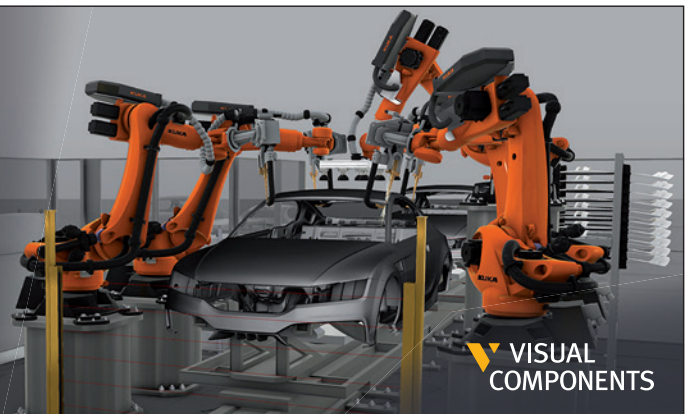


### DESIGN THE FACTORIES OF THE FUTURE

Visual Components 4.0 - The next generation of 3D manufacturing simulation

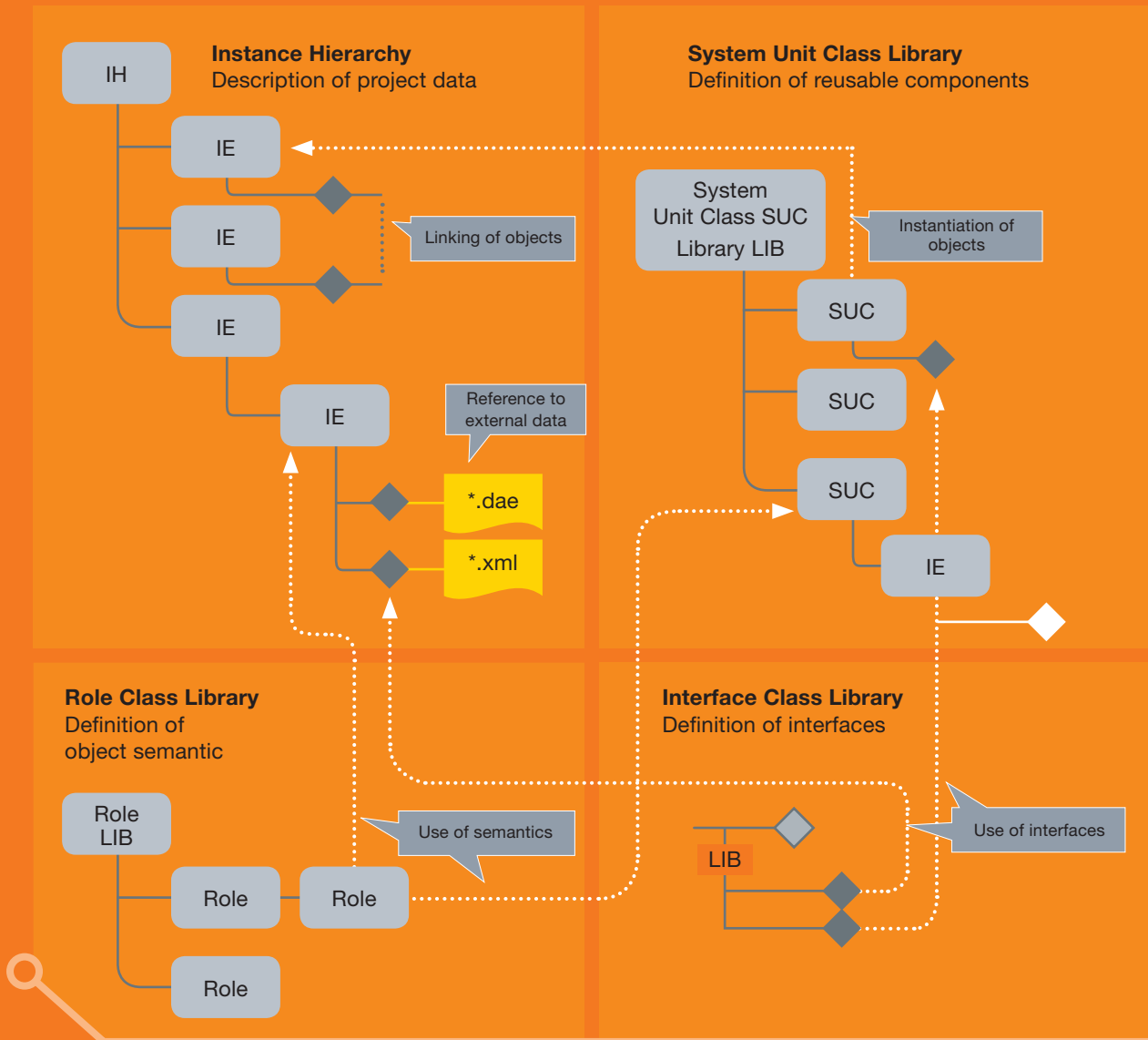
- ▶ Layout Planning
- ▶ Production Optimization
- ▶ PLC Verification
- ▶ Offline Programming
- ▶ Virtual Reality

[www.visualcomponents.com](http://www.visualcomponents.com)





## AUTOMATIONML CORE MODELLING CONCEPTS



Object-oriented modelling is a proven and powerful approach to manage bulky and complex information. CAEX supports object-oriented modelling based on four types of libraries (see figure above): a) neutral object semantics (role class library), b) interface class libraries (interface class library), c) libraries for attribute types (attribute type library), and d) proprietary vendor-specific libraries (system unit library). This makes CAEX applicable in all industrial domains.

Furthermore, CAEX provides extended concepts which are essential for the data exchange of object structures in a heterogeneous engineering tool landscape: UUID-based object identification, version information, data source information, and the capability to model mixed semantics.



[iipr.kit.edu](http://iipr.kit.edu)

Karlsruhe Institute of Technology

**Institute for Anthropomatics and Robotics (IAR)**  
**Intelligent Process Automation and Robotics Lab (IPR)**

„Interactive and Learning Production Systems Based on AutomationML and OPC-UA“

## » MODELLING GEOMETRY AND KINEMATICS WITH AUTOMATIONML //

As mentioned above AutomationML exploits the international standard COLLADA 1.4.1 and 1.5.0 for the representation of geometry and kinematics information which is standardized as ISO/PAS 17506:2012. Therefore, AutomationML has developed a two-stage process:

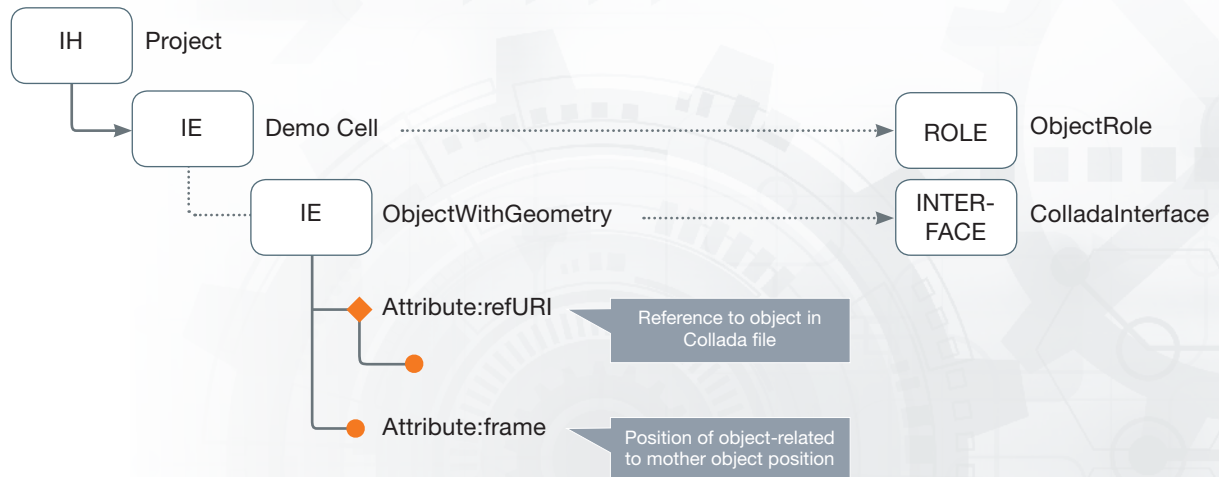
1. relevant geometries and kinematics are modelled as COLLADA files.
2. these files and the data objects within them are referenced out of the CAEX file.

COLLADA stands for COLLABorative Design Activity. It was developed by the KHRONOS association under the leadership of Sony as an intermediate format within the scope of digital content creation in the gaming industry.

It is designed to enable the representation of 3D objects within 3D scenes covering all relevant visual, kinematic, and dynamic properties needed for object animation and simulation.

COLLADA is an XML-based data format with a modular structure enabling the definition of libraries of visual and kinematic elements. It can contain libraries for the representation of geometries, materials, lights, cameras, visual scenes, kinematic models, kinematic scenes, and others.

### DEFINITION OF COLLADAINTERFACE INTERFACE CLASS



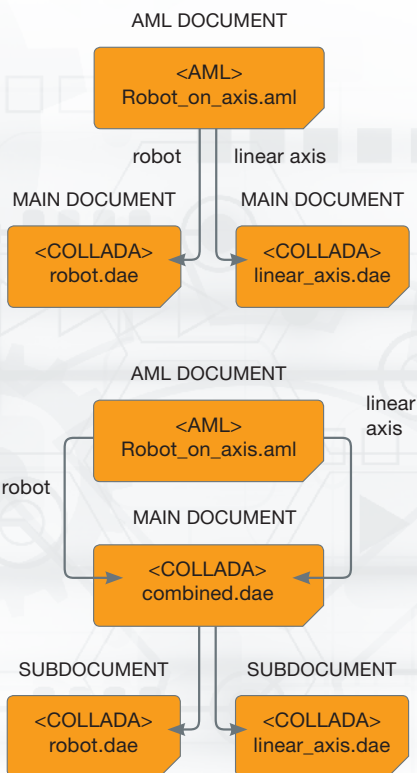
COLLADA is an XML-based data format with a modular structure enabling the definition of libraries of visual and kinematic elements.

The most important feature of AutomationML is the clear identifiability of objects in COLLADA files, which allows the integration of these files into AutomationML. Several data objects within a COLLADA file have a unique identification (ID) like geometries, visual scenes, kinematic models and kinematic scenes.

In order to reference these objects, AutomationML has defined a special interface class within the AutomationMLInterfaceClassLib named COLLADAInterface which shall be applied to derive the needed interfaces for geometry integration.

This interface class itself is derived from the interface class ExternalDataConnector and therefore has an attribute refURI. This attribute can be used to reference into a COLLADA file, thereby referring to an ID of an object modelled in the COLLADA file. Thus, the value of the refURI attribute shall contain a string structured like file:///filename.dae#ID. The attribute refType is used to differentiate between various ways of embedding objects in a modeled scene. It can provide information on how static an object in the scene is in relation to other objects, e.g. whether a work piece and the conveyor belt move at the same time.

### STRUCTURE OF REFERENCING COLLADA SUBDOCUMENTS



Naturally, an InstanceHierarchy may contain more than one InternalElement with a geometry assigned to it. There are different ways to define the relationship between different objects. The two possibilities are shown in the figure on the left. The relationship between the linear axis and the robot can be defined in a common COLLADA file and then referenced. The other option is to reference two separate COLLADA files and define the relationship in the AutomationML file.

To enable the proper positioning of these geometries within the overall set of geometries each InternalElement may have an attribute that can be used to specify the position of the assigned geometry related to the coordinate system of the parent InternalElement. This frame attribute depicted in 4 enables the specification of the offset of the geometry in the Cartesian directions x, y, z as well as its rotation around these axes.

Exploiting the BPR – External Data Reference it is possible to add additional documentations to AutomationML objects. This enables, for example, the use of JT to document the geometry of an object by referencing this geometry as an external data files and providing the geometry semantics using an appropriate object role (see page 18f).

## » MODELLING BEHAVIOUR WITH AUTOMATIONML //

**AutomationML with its object-centric modelling approach enables a dedicated storing of logic information on object level. For this purpose, certain object semantics as well as object interface semantics are developed. In addition to that, logic models are identified, commonly used in the engineering process of a production system, and made exchangeable, but also transformable among each other. This allows and supports an information enrichment process in terms of logic information that is required for the scope of AutomationML. All concepts are going to be standardized in IEC 62714-4.**

Logic information is an important aspect for raw system planning, electrical design, HMI development, PLC and robot control programming, for simulation purposes, and virtual commissioning. To support the different phases in the iterative production system engineering process covering different levels of detail, AutomationML needs to be able to store logic information from different tools and disciplines.

### **Three types of logic information: sequencing, behaviour, and interlocking information**

While sequencing information describes the instructions to the controlled object, behaviour information describes the possible responses of the controlled object to the sequencing information and to other external interactions. The third type is interlocking information. This describes the instructions to the controlled object to avoid the object from causing harmful effects on humans and environment. Therefore, an object of the production system structure can have up to three types of logic information assigned to it since an object can be composed of other objects. These objects can then be composed of other objects and so on.

Logic information is generally organized and stored in logic models. To be applicable within the engineering process of production systems, AutomationML supports common logic models which cover different phases of the engineering process. A common basis named Intermediate Modelling Layer (IML), is defined for the following three logic models. With this, it is possible to transform one logic model into another one. A forward transformation supports the information enrichment process and reduces or avoids a re-entry of information between the exchanging engineering tools. Theoretically, this workflow can be carried out until an executable program. But the backwards transformation is also reasonable as the engineering process is not a linear process. It has iterations and cycles due to possible requirements, changes or errors made in engineering. However, since the logic models have different modelling powers, backward transformation may result in information loss when not handled appropriately by the importers and exporters of software tools.

### **IML is the common basis for these logic models:**

- Gantt charts are mainly used in early phases of the engineering process to represent the timing and duration of manufacturing processes on a high level of abstraction.
- Activity-on-node networks are primarily applied in early phases of the engineering process to represent the timing and interdependence of manufacturing processes with the capability to store timing information.
- Timing diagrams are mostly used in the later phases of the engineering process to describe the devices with their states and timing relationships. Real signals are introduced and it is possible to express sequencing information.

A mapping of other logic models, e.g. event-driven logic models like state charts, onto IML is possible.

Beyond those IML-based logic models, AutomationML enables the usage of other essential logic models as well. But those come with an own formal, machine readable and processable representation and do not need to be mapped onto IML. These models are:

- Sequential Function Charts (SFC) are one of the programming languages of the IEC 61131-3, mainly applied in later phases of production system engineering. Modelling of sequencing information as well as behaviour information is possible.
- Function Block Diagrams (FBD) are one of the programming languages of the IEC 61131-3, mainly applied within later phases of production system engineering. Modelling of interlocking information as well as behaviour information is possible.
- Mathematical expressions are primarily used in the later phases of the engineering process to describe mathematical, physical, chemical, or logical relationships of a device.

For the interlocking information, AutomationML provides a concept for how cause and effect matrices (logic model) can be stored using FBD.



**IEC 61131-10 is the basis for storing the logic models.**

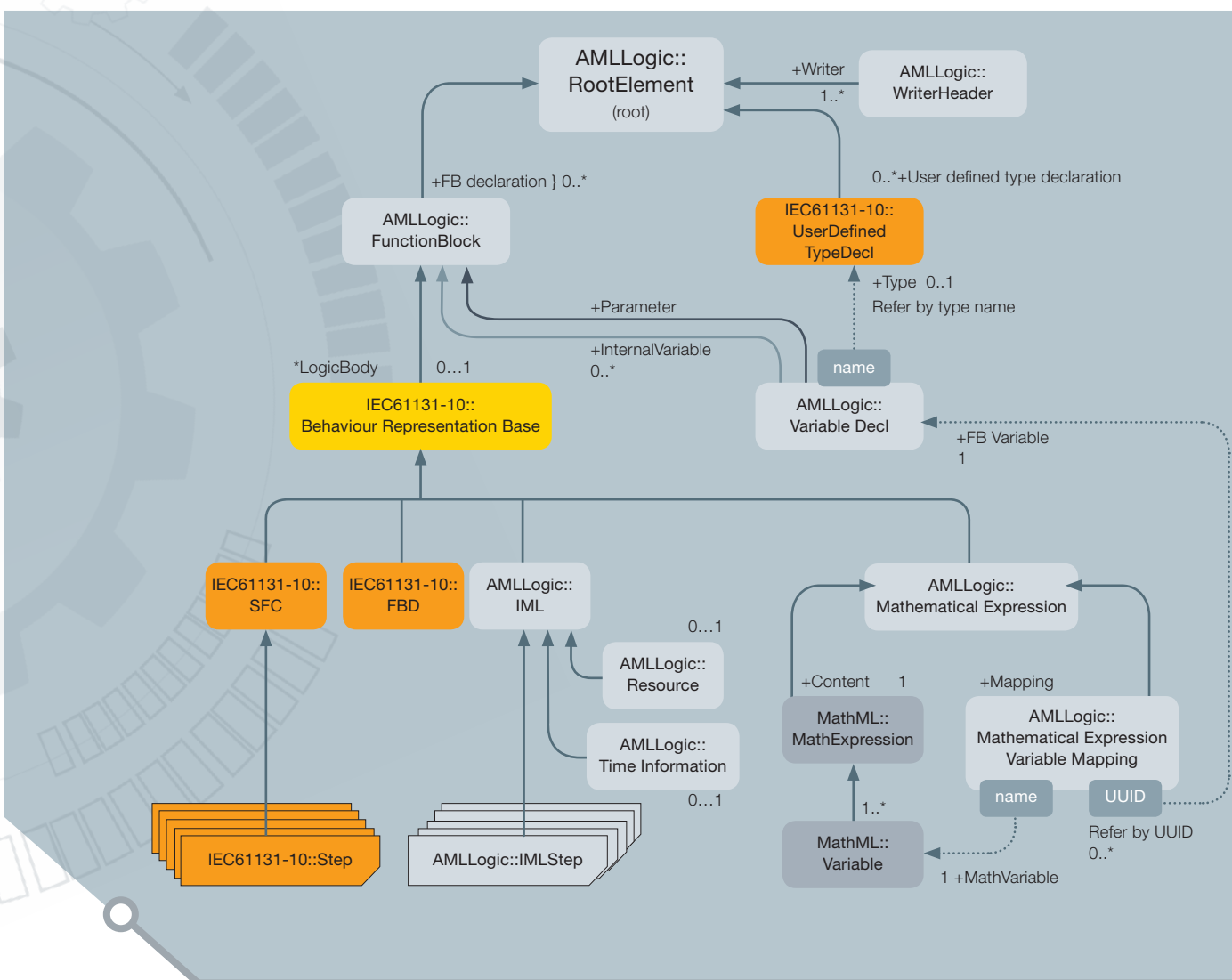
The scope of IEC 61131-10 is the exchange of PLC programming projects as XML. However, AutomationML takes more logic information into consideration than PLC programs do and adapts the usage of IEC 61131-10 by defining the AutomationML logic XML schema. This schema is used to store the mentioned logic models by mapping IML to IEC 61131-10. The modelling elements of AutomationML logic XML schema are shown in the figure below.

In the multi-document architecture of AML, logic information is stored in separate AutomationML logic XML documents based on the IEC 61131-10 data format. Modelling logic information is therefore divided into two parts. In the first step, an object with logic information is modelled in CAEX, the top-level format of AML. In the second step, an AutomationML logic XML document has to be provided containing this logic information. Both documents need to put in a relation to each other.

Therefore, a reference between the AutomationML object and the AutomationML logic XML document or an element within it is created. The second part is extensively described above. For the first part, the mentioned object semantics and object interface semantics is needed. To identify an object as a logic model-relevant object, the RoleClass "LogicModelObject" can be used. For building up the relation between both documents or parts of it, the derivations of the InterfaceClass "PLCopenXMLInterface" are used. The main derivations are the InterfaceClass "LogicModelInterface" and "LogicModelElementInterface". Derivations of the first one are used to classify the type of logic information that is stored in the external AutomationML logic XML document, such as sequencing, behavior or interlocking information. Derivations of the second InterfaceClass are used to identify the type of logic model element that is referenced by the AutomationML object, e.g. variables.

» For more details, IEC 62714-4 can be consulted.

**BEHAVIOUR MODELLING ELEMENTS**



# PLCopen® Japan

*for efficiency in automation*



PLCopen Japan is the branch office of PLCopen headquartered in the Netherlands. PLCopen and AutomationML e.V. adapted the PLCopen XML format to be useable for all Logic descriptions which are defined in AutomationML Part 4.

Some engineering tools already provide it and the underlying PLC's programming standards (IEC 61131-3) are field-proven in most Japanese manufacturers.

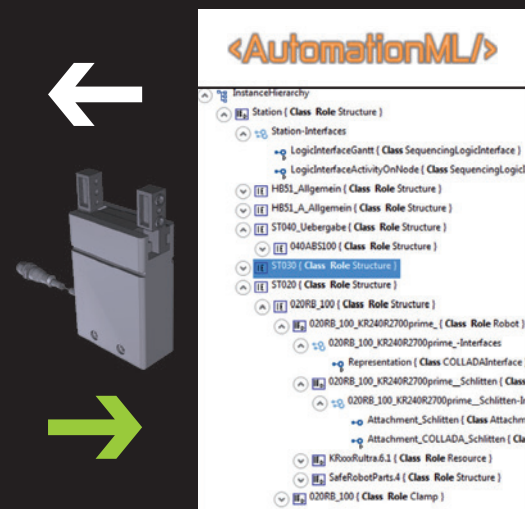
PLCopen XML format will soon be adopted as international standards (IEC 61131-10).



PLCopen Japan :  
<http://www.plcopen-japan.jp/>

PLCopen Global :  
<http://www.plcopen.org/>

## Higher quality data exchange for efficient manufacturing engineering processes



## » TECHNOLOGICAL INNOVATIONS BASED ON CAEX 3.0 //

In 2016, the new version 3.0 of CAEX has been internationally standardized - IEC62424:2016 replaces the previous version of CAEX known as version 2.15. In parallel, the AutomationML community started the international standardization process to give AutomationML an upgrade towards CAEX 3.0. In 2018, the new IEC62714-1:2018 was published, replacing the previous version of the AutomationML architecture.

### New features of CAEX 3.0/ AutomationML Edition 2.1:

- CAEX 3.0 provides a new attribute type library: this allows the modelling of proprietary or standard attribute types in libraries, or both together in different libraries. Any class or instance attribute can now reference attribute types. This allows the user to transparently model the semantics of attributes, or to simplify multi-language support. Any engineering tool can now name its attributes individually according to language settings or company standards. It is the reference to a common attribute type library that determines the meaning.
- CAEX 3.0 provides native support of multiple role references. If a multi-functional device can assume the role of a fax, printer or scanner, CAEX 3.0 simplifies the assignment of multiple roles.
- CAEX 3.0 natively provides nested interfaces. This allows the modelling of complex plugs with internal interfaces, e.g. for power supply, data connection and pneumatic interfaces.
- CAEX 3.0 provides a field to store the version of a superior standard as it can now be embedded in superior standards. This simplifies the adoption of CAEX 3.0 in AutomationML.
- CAEX 3.0 now has an own name space. This helps to differentiate CAEX from other standards if CAEX 3.0 is combined with other standards in a superior standard.
- CAEX 3.0 natively provides fields to store meta information about the data sources. This allows the user to see which tool created the file. Even multiple data sources can be modelled. Every importer can now understand whether the file comes from EPLAN version x, Siemens COMOS version y or ABB 800xA version z. This significantly helps to develop data exchange solutions without waiting for global engineering semantic standards.
- CAEX 3.0 provides the ability to store source information for each object down to attributes. This allows any importer to investigate which tool has changed which object or attribute of an object.
- CAEX 3.0 now requires that every interface is identified by a unique identifier. All links between interfaces now base on IDs, this simplifies the modelling of interconnections.
- CAEX 3.0 extends the mirror concept to attributes and interfaces and allows the modelling of shadow copies of objects or attributes in different topologies and extends the modelling of multi-topologies. This is very helpful in case a CAEX structure models a plant topology, process topology, communication topology, logic topology, location topology etc. in combination.

"The AutomationML association will maintain each document step by step and adapt it to the new CAEX standard, independent of maintenance cycles of the IEC standard. Both CAEX versions will be supported due to this procedure."

## » COMBINING AUTOMATIONML AND OPC UA //

Manufacturing systems change continuously and therefore need to be adaptive. Changes/re-engineering in manufacturing systems lead to reconfigurations of hardware and software components. This affects all planning phases, disciplines and corresponding tools and systems.

In this environment, AutomationML is conducive to the modelling of linked production systems. This enables a transfer of engineering data of these systems across domains and companies in a heterogeneous engineering tool landscape.

OPC UA is used for the interconnection of components in these systems. It defines platform-independent communication mechanisms for online data exchange and generic, extensible, and object-oriented modelling capabilities for the information a production system wants to expose.

**AutomationML describes which data and information is exchanged while OPC UA determines how the exchange of data and information takes place.**

Thus, AutomationML can be classified as standard on the information layer of the reference architecture model Industrie 4.0 (RAMI4.0), while OPC UA can be classified as standard on the communication (and partly also the information) layer of the RAMI4.0.

For this reason, it is logical to combine those two. The usage of both standards in combination and collaboration creates synergy and leads to a wider acceptance and usability of both standards.

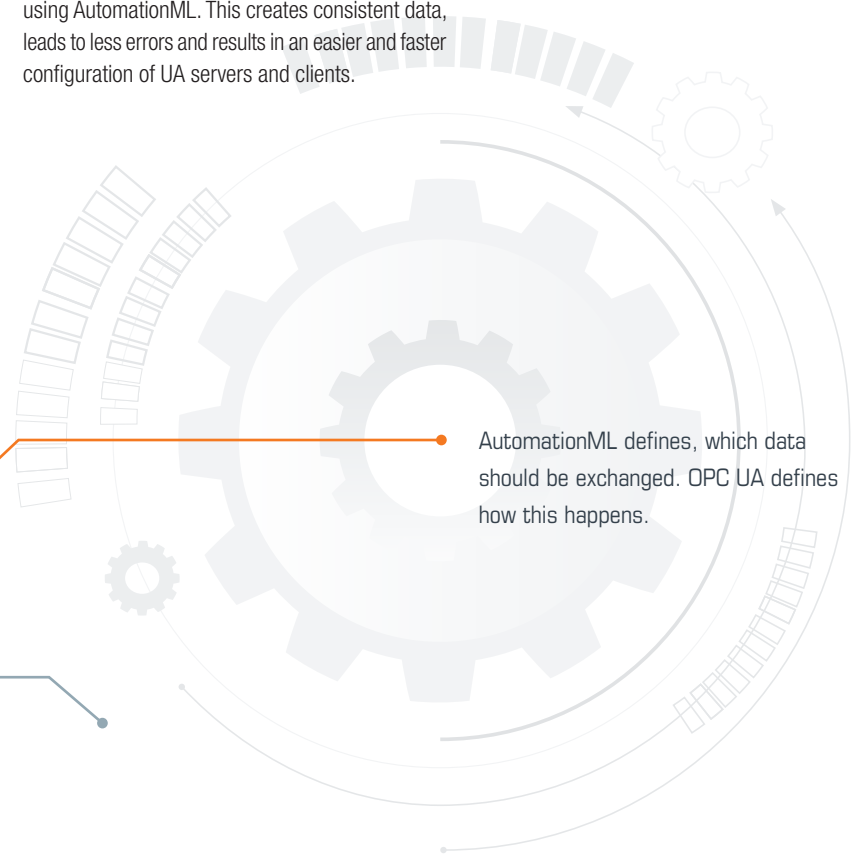
A joint working group of the AutomationML e. V. and the OPC Foundation have been dealing with this topic since 2014. A first result of this joint working group was the Companion Specification 'AutomationML for OPC UA'. It describes rules to transform AutomationML models into OPC UA information models, and has been expanded to DIN SPEC 16592 „Combining OPC Unified Architecture and Automation Markup Language“ which was published in December 2016.

**The integration and harmonization are done vice versa and focusses on two main use cases:**

- AutomationML integration in OPC UA: Combining AutomationML and OPC UA offers the possibility to communicate and operationalize AutomationML by means of OPC UA. It is possible to simplify the creation of OPC UA information models based on existing AutomationML data. AutomationML models can be exchanged and managed by OPC UA which makes an up-to-date description of the as-is state of the system possible.
- OPC UA integration in AutomationML: Another possibility is the lossless exchange of OPC UA system configuration within AutomationML models. The manual exchange of OPC UA server configuration data will be replaced by standardized/specified description in AutomationML. Parameters used to set up OPC UA communication between tools can be exchanged by using AutomationML. This creates consistent data, leads to less errors and results in an easier and faster configuration of UA servers and clients.

As AutomationML is mostly used during the planning phase and OPC UA during the operation phase, this combination presents a holistic solution to create and extend the Digital Twin along the entire life cycle of the production system.

In the context of Industrie 4.0, the 'Industrie 4.0 component' and its services as well as the corresponding administration shell including component manager and sub-models must be 'somehow' implemented. The obvious way is to apply already existing standards of the automation domain. One possibility is a technology combination of OPC UA and AutomationML. OPC UA can provide all accessible services of the component manager by means of basic services of OPC UA or specific OPC UA methods. AutomationML can describe the data content of the component manager and is able to separate this data and include internal and reference external sub-models. Both standards are explicitly open for the combination and harmonization with other standards. Therefore, they are capable of fulfilling the requirements of the asset administration shell, its component management and the Industrie 4.0-compliant communication!



AutomationML defines, which data should be exchanged. OPC UA defines how this happens.



## » COMMUNICATION NETWORKS IN AUTOMATIONML //

The modelling of communication networks is one important task in the engineering of automation systems. Depending on the respective communication system, different participants can be considered. Considering it from the automation perspective there are controllers, sensors or actuators. They act in different roles. In addition, infrastructure devices like bridges or gateways are also involved to model a communication network. As a rule, such devices do not perform any automation task.

The design goal for regarding the development of modelling rules for communication networks in AutomationML was the need to be able to handle all types of networks. Therefore, there are two main entry points: the topology which can be viewed from the physical and from the logical point of view.

### Furthermore, the model shall provide following information:

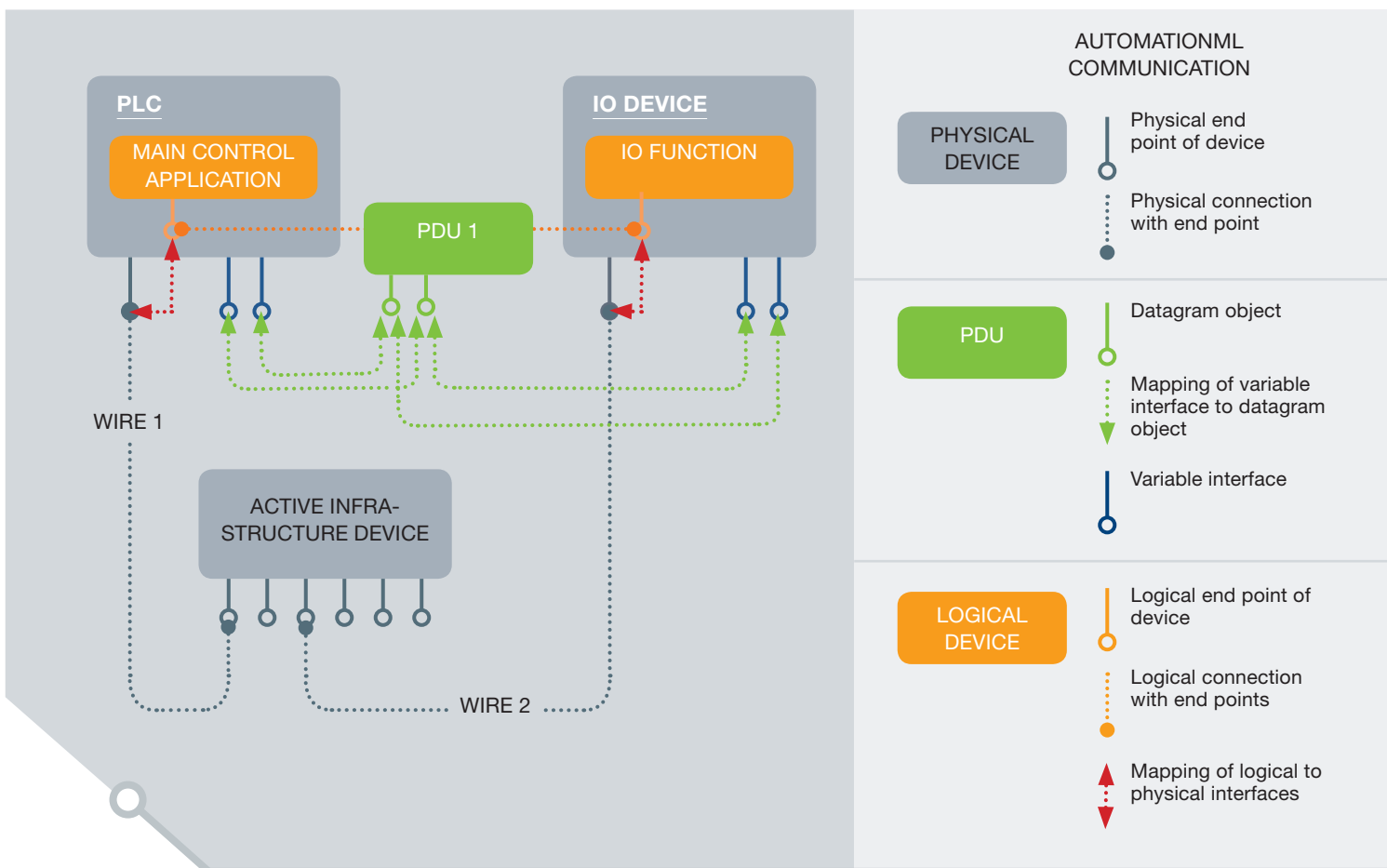
- Description of device and the used cabling or wireless channels
- Ports where the cables are plugged or the wireless channels are pinpointed
- Representation of virtual devices/applications running on the same device
- Representation of the data flow by means of the logical topology
- Relation between physical and logical items
- Internal relations between physical sockets or antennas and logical end points

These aspects are depicted in the following figure. The two involved automation devices are connected by means of Ethernet cables via a switch. Logically, there is a connection between the Programmable Logic Controller (PLC) and the Input-Output (IO) Device.

It is possible to model the Protocol Data Unit exchanged between the automation devices in particular. The level of detail of the modelling can be determined by the tool. In principle, the structure and layout of the Protocol Data Unit (PDU) can be described. Using this information, relationships to the device internals, e.g. the PLC program variables, can be modeled.

The modelling strategy follows the AutomationML approach to define Role- and Interface classes. Both allow the definition of basics. Existing types of different communication systems, communication patterns or communication system specific standard interfaces can be derived and may be maintained from dedicated user organizations. Furthermore, product type specific communication interfaces can be provided by manufactures. Such information is normally provided in special libraries for System Unit Classes. Finally, instances of these System Unit Classes will be created in order to model the real industrial communication network.

### STRUCTURE OF COMMUNICATION SYSTEM MODELS



## » APPLICATION AND BEST PRACTICE RECOMMENDATIONS //

**Application recommendations (AR)** are developed by the AutomationML association to provide guidelines for the application of AutomationML in special industries or for special information sets relevant within the life cycle of production systems.

**Currently, there are two ARs published by the AutomationML association: the AR Automation Project Configuration, which has been presented in this document, and the AR Provisioning for MES and ERP – Support for IEC 62264-2 and B2MML (AR-MES-ERP).**

This second AR covers the modelling of information exchanged between Enterprise Resource Planning systems (ERP) and Manufacturing Execution Systems (MES) in a fully integrated business information system. For this information exchange a data standard has been specified in the IEC 62264 series, based on a standard that has been standardized by the American National Standards Institute (ANSI) and the International Society of Automation (ISA): ANSI/ISA-95. As IEC 62264 enables the linking of the business layer (ERP and supply chain management) to the manufacturing layer, linking objects from AML to elements of IEC 62264 further fosters the integration of manufacturing layers.

**IEC 62264-2 defines five types of main information objects that are mainly relevant for representing the interaction between ERP and MES layers:**

- *Personnel*: comprising the actors that are required to operate manufacturing processes,
- *Equipment*: representing the types of equipment of an organization in form of a role based model,
- *Physical assets*: represent the used physical pieces of equipment,
- *Material*: representing raw, finished, and intermediate materials, as well as consumables, and
- *Process segments*: resembling elements of manufacturing activities that are visible to business processes

There may be properties and relations between the individual objects. AutomationML is mapping these entities to role and interface classes in a way that allows a unique identification and a unique mapping of object properties.

The developed recommendations are based on our members' use cases, interests, know-how, and practical experiences with AutomationML.

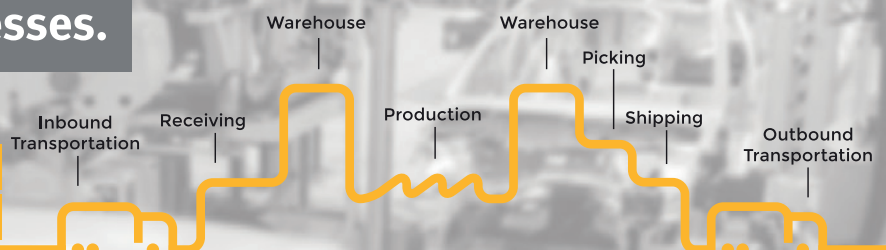
Experts for AutomationML and system integration

**[SO:IT]**<sup>®</sup>  
SALT SOLUTIONS

IT solutions for  
business-critical processes.

AutomationML kernel development

AutomationML based applications

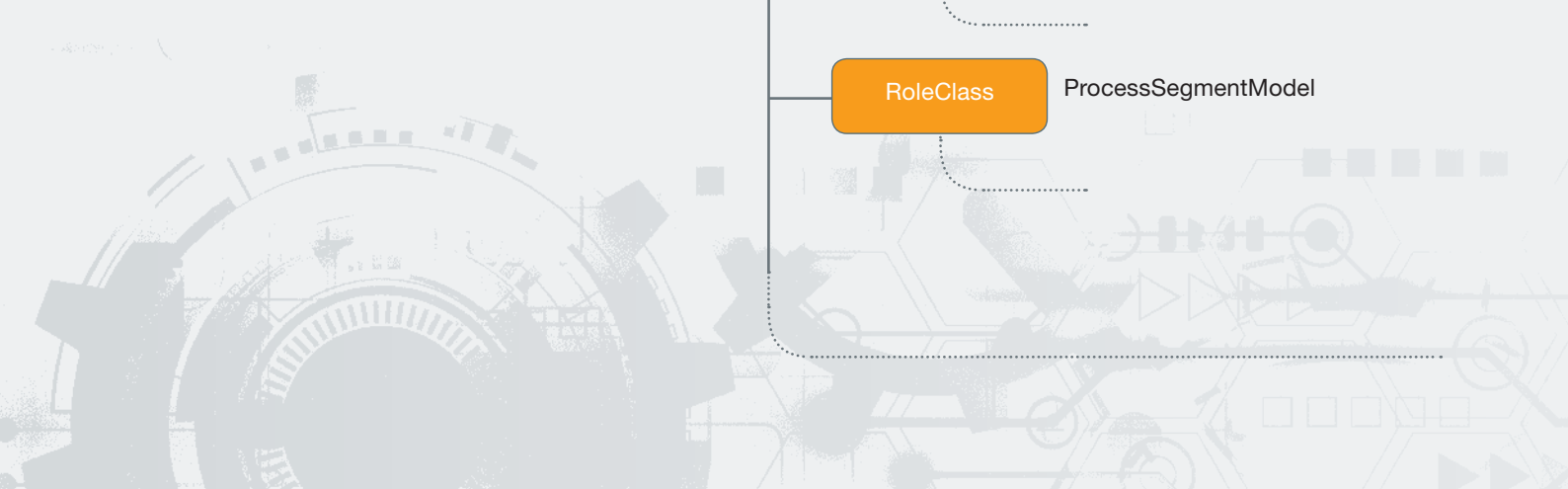
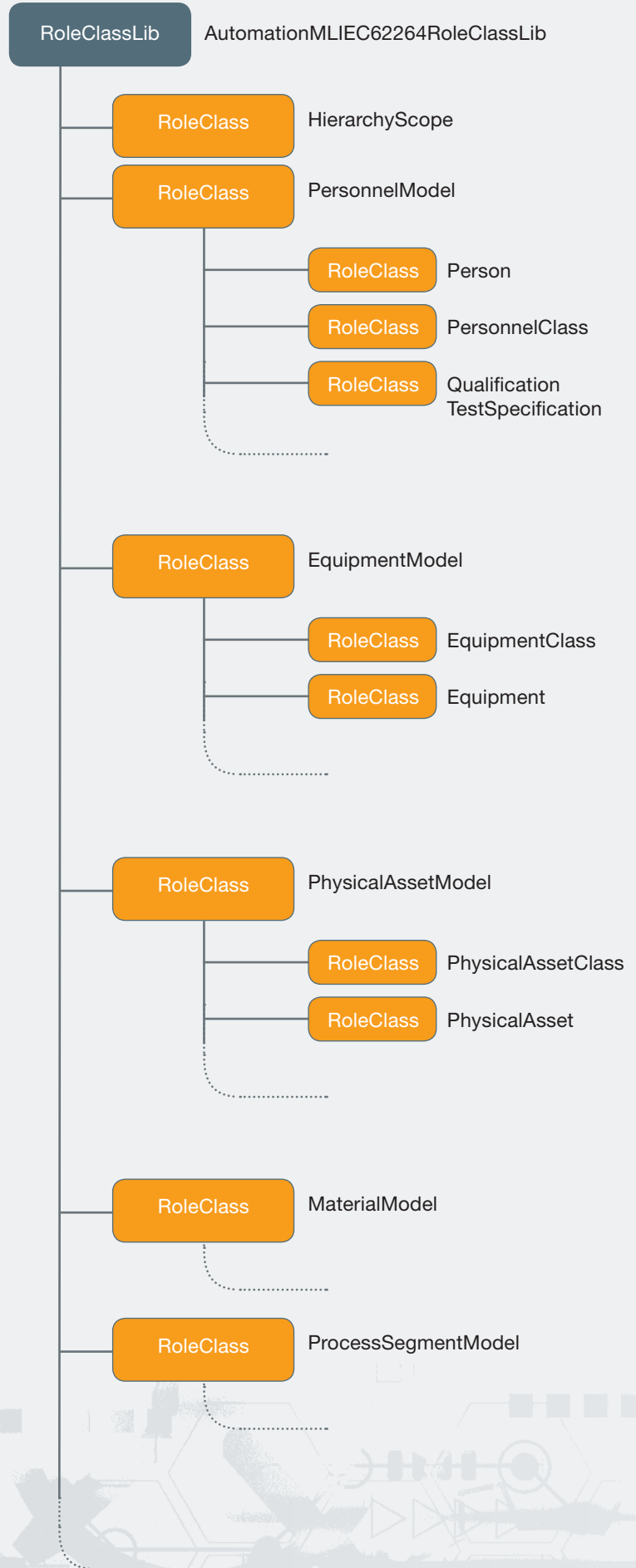


> [gosalt.de/automationml](https://gosalt.de/automationml)

- **Best practice recommendations (BPR)** are a specification document type applied by the AutomationML association for highly specialized problems regarding the modelling of production systems that define standard solutions. As the AutomationML association has developed an increasing set of BPRs, only five of them will be described here briefly and serve as selected examples. The complete set can be found at the AutomationML homepage.
- The **BPR – Reference Designation** discusses the need of reference designations following IEC 81346 as an identification methodology of structured objects. It defines specific role and interface classes that contain specialized attributes. They can be used to identify objects carrying reference designations and, therefore, obtain their specific values.
- Within one white paper AutomationML provides modelling means to describe communication networks in production systems. This includes the description of logical devices responsible for the communication. The **BPR – Data Variable** is based on this fact and provides a modelling mechanism representing the communication configuration/access to the values of the communication devices. This modeling mechanism is applicable to automatically access these communication values enabling, for example, the coupling of engineering data and runtime data within an AutomationML OPC UA system.
- The **BPR – External Data Reference** considers the need to add additional documentations to AutomationML objects. Plant components, e.g. devices, are usually accompanied by manuals, mounting guidelines, warranty documents, etc. as well as special documentations applicable in special tools. The named BPR provides a role and interface-based methodology in order to reference external data files and provide them with a semantics within the overall AutomationML model.

While the three BPRs mentioned address specific engineering information directly, BPRs that relate to the improvement of information management are available as well. Here are two examples:

- The **BPR – Multi-Language** specifies the way different spoken/written languages can be integrated into AutomationML projects to enable the correct naming / representation of objects in different countries.
- The **BPR – AutomationML Container** defines the application of compressing mechanisms to AutomationML projects.



## » DEFINING SEMANTICS IN AUTOMATIONML //

One key to efficiency is the interoperability and seamless interaction across interfaces, tool barriers, and planning phases. The transfer of information does not create added value. The full potential can only be tapped if the exchanged information can be understood by all partners, e.g. based on common semantics or information models.

A critical point during the import of data into engineering tools is the mapping of the incoming data to the data model of the importing tool. Therefore, the importing tool must gain the semantic of each imported data element.

Due to the heterogeneity of the different domains, it is not possible to create one unified information model. Therefore, general concepts must be defined, but specific information models including domain semantics must be possible as well.

### AutomationML separates syntax and semantics of data elements.

Even though AutomationML enables an object-oriented approach, elements cannot only reference element types, but they can also assign one or multiple roles to the element. A role describes an abstract functionality without defining the underlying technical implementation. Thus, it has to be seen as an indicator for the semantics of an object which can be described in an abstract way and independent of the manufacturer. Roles can include general attributes (size, number of

axes, etc.) and interfaces (PPRConnector, ...) to describe the interaction possibilities of the element which assigns this role. Roles are organized hierarchically in libraries and can have interrelations to other roles and further elements to describe their dependencies. If a certain role is too specific, the next parent should be referenced.

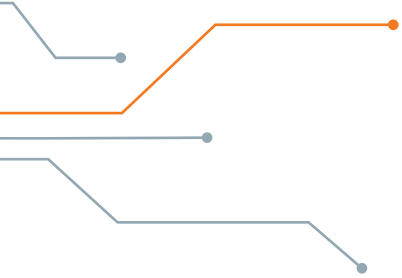
### Defined role sets and role class libraries

AutomationML defines different role sets in its various white papers, application recommendations, and best practice recommendations. One set of role class libraries is given in Part 2 of the AutomationML standard. These roles are based on the general role class library (AutomationMLBaseRoleClassLib) defined in Part 1 of the AutomationML standard. The role class "AutomationMLBaseRole" in the AutomationMLBaseRoleClassLib is a basic abstract role type and the base class for all standard or user-defined role classes. All AML objects shall be associated directly or indirectly to the role class "AutomationMLBaseRole" to have a common basis, e.g. for simplifying the implementation. MechanicalPart, Conveyor, Robot, or Device are example role classes used in AutomationML.

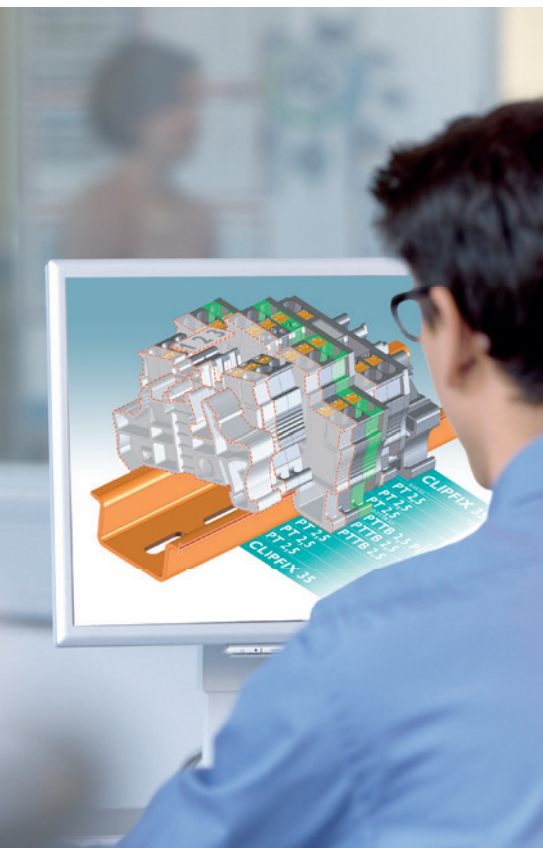
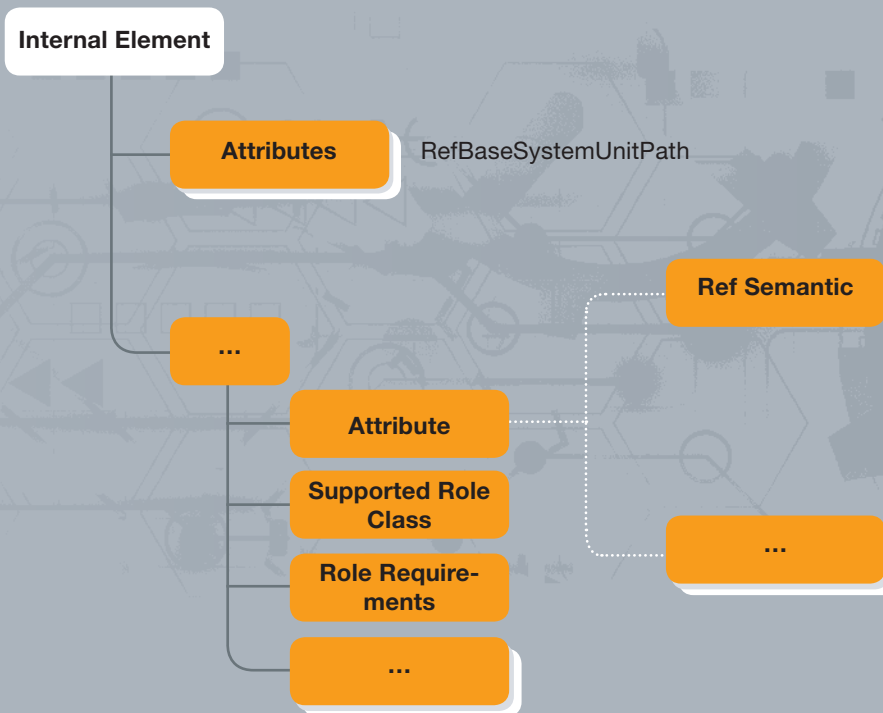
Role class libraries can be defined for general use cases but can also relate to specific domains. They don't need to be standardized or defined by AutomationML but can also be created by user groups or single users of AutomationML.

### AUTOMATIONML ROLE CLASS LIBRARIES

|   |  |
|---|--|
| AutomationMLBaseRoleClassLib                    | IEC 62714-1 – normative  |
| AutomationMLDMIRoleClassLib                     | IEC 62714-2 – normative  |
| AutomationMLCMIRoleClassLib                     |  |
| AutomationMLBMIRoleClassLib                     |  |
| AutomationMLCSRoleClassLib                      |  |
| AutomationMLExtendedRoleClassLib                | IEC 62714-2 – informative  |
| UserDefinedRoleClassLib_RedBookVDMA             | IEC 62714-2 – informative, user-defined examples                 |
| UserDefinedRoleClassLibCompanySpecificStructure |  |
| UserDefinedRoleClassLib_FoodAndBeverage         |  |
| UserDefinedRoleClassLibPandixPCE                |  |
| UserDefinedRoleClassLibPandixPPE                |  |
| ...   |  |
| <Multiple additional RoleClassLibs>             | BPR/AR/user documents – concept-related or user-defined examples |
| ...   |  |



## MEANS FOR SEMANTIC INTEGRATION IN INTERNAL ELEMENTS AND ATTRIBUTES



## AutomationML and eCI@ss Advanced

Every development of a new product has an engineering phase, regardless of whether the product is a car, a TV or a switch cabinet. With the combination of the well-established standards AutomationML and eCI@ss Advanced a continuous process can be realised, which not only reduces errors, but also the effort in engineering and production.

Most products will not be designed by a single engineering tool. Instead, an engineering tool chain is used, consisting of various tools for different purposes. Because each of these tools works in another functional domain, the tools 'speak' different and domain specific languages. As a result, the data formats used are not compatible to each other and the data content has no common semantics.

Knowing this, it is more purposeful to have one common data format with well-defined semantics for all engineering tools. Such a data format would allow for continuous engineering and contain all information arising in the engineering process. It would also enable modern approaches like 'order-based production', where a digital description of the product to be manufactured is analysed to derive necessary machine capabilities.

For this reason Phoenix Contact is a driving force in the joint working group of the AutomationML and eCI@ss associations to define a synergetic combination of both data formats for the above mentioned purposes.

[phoenixcontact.com/industrie40](http://phoenixcontact.com/industrie40)

# » IMPLEMENTING AUTOMATIONML TOOL INTERFACES //

AutomationML can be applied to various data exchange scenarios in design processes. To be applicable, the used engineering tools shall be equipped with AutomationML interfaces to write and/or read AutomationML data.

### A two-phase design process is recommended in order to implement an AutomationML interface:

1. The data models of the involved tools and its relevant parts, which are to be transferred to AutomationML, have to be determined. If the exchange partners are known, the decision can be based on the intersection volumes of the data models. Otherwise an assumption must be made, whereby some contents in the model may not be covered by other tools.
2. The defined exchange model has to be modeled in Automation ML using standardized and other published AutomationML role classes and interface classes and other libraries. If the provided libraries don't cover all aspects of the exchange model or don't provide the needed level of detail, specifications have to be defined in additional Automation ML libraries.

Interface implementation can be based on design patterns, suitable for AutomationML export and import interfaces.

The interface components are similar in both patterns but the data-flow differs. The generator in the export pattern has to create AutomationML data whereas the generator in the import pattern generates system-specific data. The export generator is responsible for the

attachment of model references like RoleClass and InterfaceClass references to generated instance data. Only if the operator assigns precise definitions to data objects the import interpreter will be able to understand the data objects and invoke a suitable generation method to transform AutomationML data into system specific data.

### The following pseudo code fragments can illustrate the principal procedure:

```

Interpretation and generation in export interfaces:
for any data object a: if interpret(a) then create instance of Role: „./././“

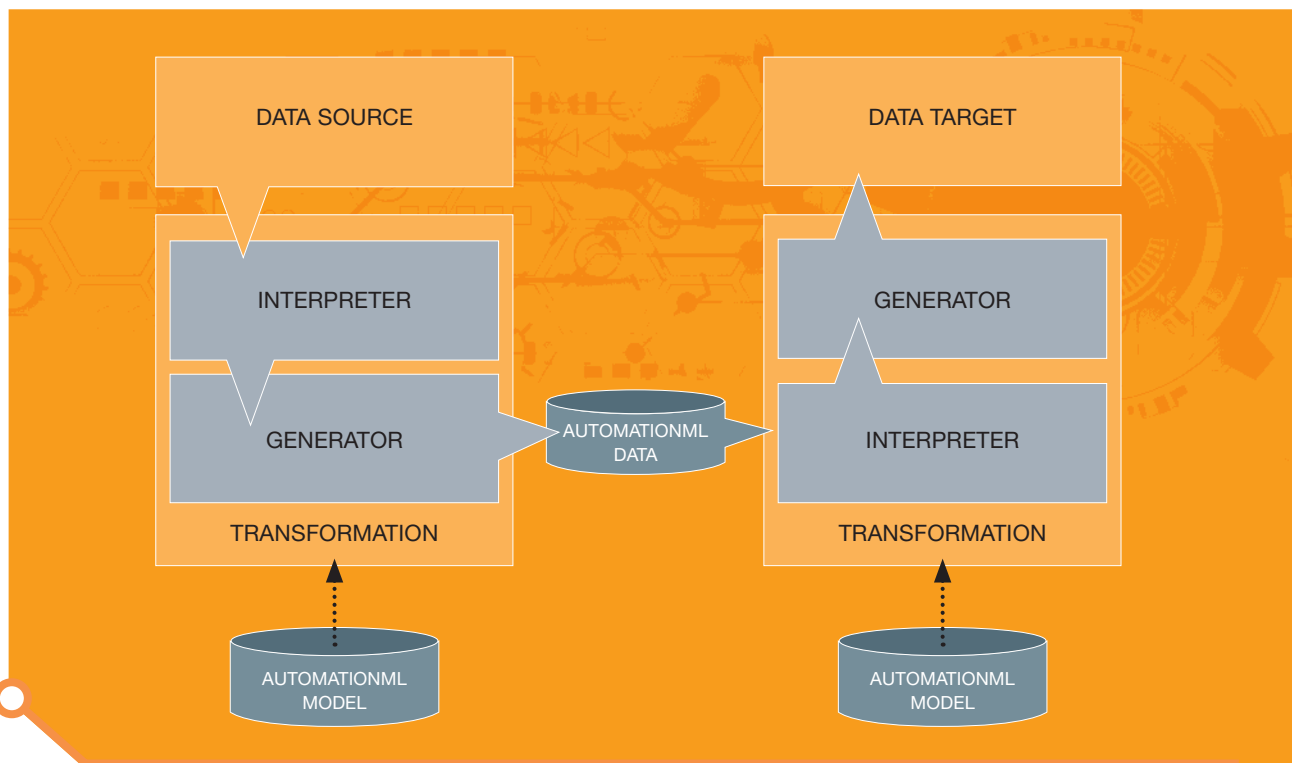
Interpretation and generation in import interfaces:
for any instance a: if „Role“ == „./././“ then create (a)

```

If import interpreters, based on "RoleClasses", are confronted with unknown classes (which don't have an assigned generator method) a fallback strategy is needed:

1. A generator method of the generic RoleClass, from which the found role class has derived, may be applicable. Using a more generic method could result in loss of information but can be uncritical if the application doesn't need the detailed information.
2. A protocol listing the interpretation lacks can be generated. The protocol can be used later to modify the interface.

## BASIC FUNCTIONAL ELEMENTS WITHIN DATA EXCHANGE PROCESS





Source: Mitsubishi Electric Europe B.V., Getty Images

For a faster time to market, Mitsubishi Electric has adopted the AutomationML interface to allow the seamless, bidirectional flow of data between iQ Works and other software tools supporting it.

## Faster time to market

### Consistent data exchange between electrical wiring and control software through AutomationML

Mitsubishi Electric has implemented the AutomationML interface within iQ Works, the integrated software suite for programming Mitsubishi Electric automation products. AutomationML represents an important step-change in the way complex equipment is designed, installed and maintained. Designers, integrators and end users will all benefit from a streamlined approach to system implementation that will drive down costs and shorten time to market. The ability to ensure a seamless flow of information between different systems is also a key driver in the digital transformation of manufacturing on the journey to Industry 4.0.

Consistency of information throughout the engineering, integration, maintenance and ongoing lifecycle management phases of a project improve quality and reduce costs. Frequently, the exchange of information between various engineering disciplines is carried out manually, which is both time consuming and introduces the possibility of errors.

Implementing a universal interface for the automatic exchange of data between software environments ensures consistency of information throughout the lifecycle of a project and fulfils today's requirements to shorten time to market.

Authors:

Matthias Müller, Software Engineer & Marco Hoch, Software Development Engineer – Factory Automation, Mitsubishi Electric Europe B.V

[eu3a.mitsubishielectric.com/fa](http://eu3a.mitsubishielectric.com/fa)

As an example, with an integrated AutomationML interface in Mitsubishi Electric's iQ Works software suite, data can be seamlessly exchanged bidirectionally with ECAD tools that also have an AutomationML interface. The implementation has been built on the AutomationML document "Application Recommendations: Automation Project Configuration."

Showing this in practice, Mitsubishi Electric has worked with e-Factory Alliance partner Eplan to implement data exchange of hardware configurations, I/O tags and network configurations between iQ Works and Eplan Electric P8 using the open standard. This interface ensures interoperability all the way from first design, through engineering processes and commissioning and on through the whole lifecycle management of the automation system.

Mitsubishi Electric is one of the first automation solution providers that has implemented the AutomationML interface with ECAD tools in a commercial product. Furthermore, as an active member of the AutomationML Association, Mitsubishi Electric is contributing to the further development of this open data format.

# Automation Planning

## Integration of planning data from TIA Selection Tool to TIA Portal or Eplan Electric P8

### Seamless and bidirectional transfer of planning and engineering data via AML standard.

AutomationML is the new interface between Siemens TIA Portal, TIA Selection Tool and Eplan Electric P8. Now the advantage is that the AML format is becoming more and more accepted as standard and supports applications suitable for the industry 4.0. So we work on a promising database.

### Significance and advantages of the integration of TIA Selection Tool hardware configuration, TIA Portal Engineering with the electro-CAD world

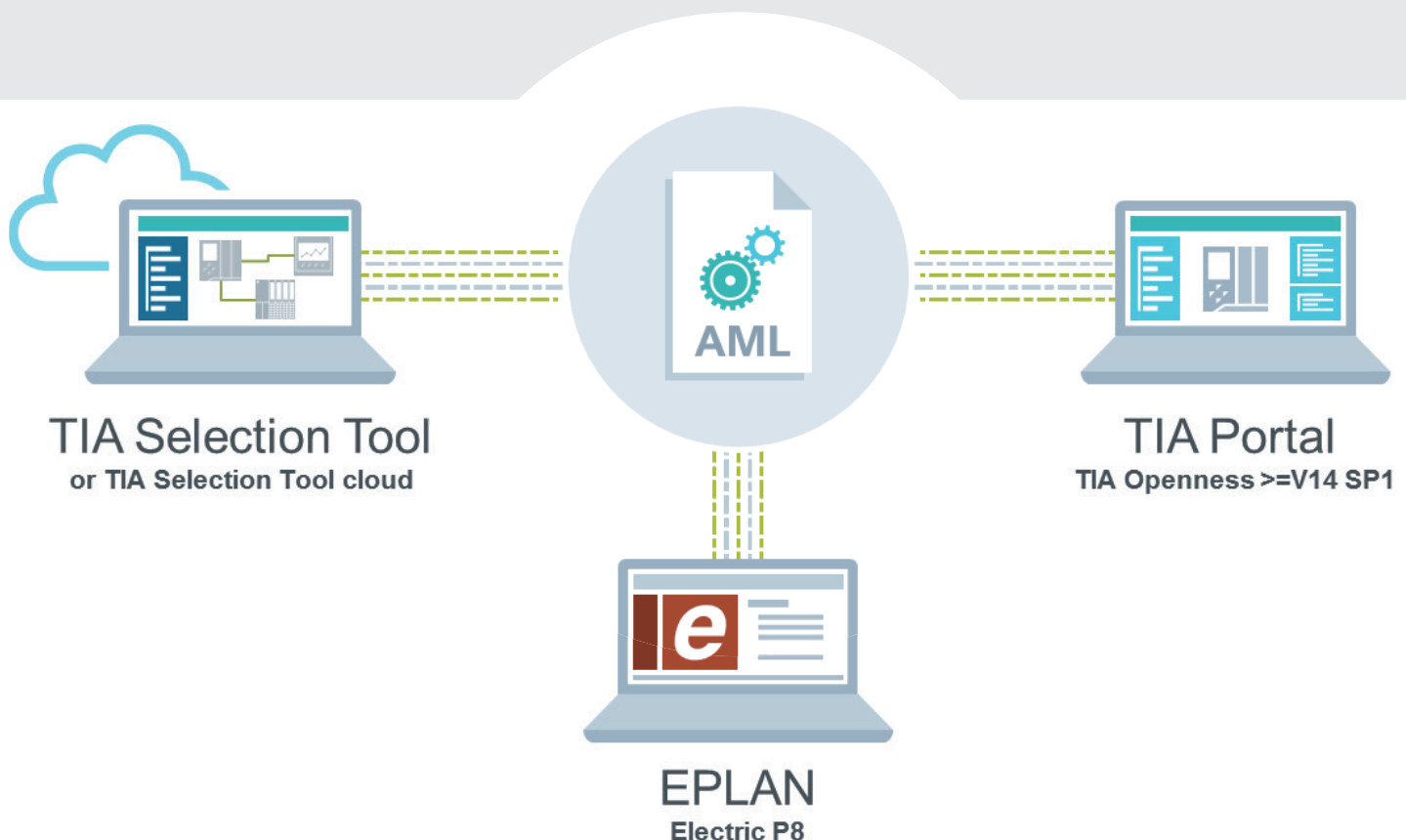
It is the basis for more consistency in the area of hardware design and software development in the automation technology of mechanical and plant engineering. It offers the possibility of exchanging this important information already during engineering, so that both disciplines are always synchronized. In the context of Industry 4.0, the software component in the control of machines and systems continues to increase continuously. In so far, the interdisciplinary comparison is becoming increasingly important.

With the bidirectional connection from Eplan, data is imported or exported into automation and electrical engineering. The relevant data can be easily

exchanged between the different workplaces of the electrical designer and the PLC programmer. Users can exchange, post-process and synchronize data at any stage of the project and in any direction. This replaces the time-consuming, often multiple manual adjustments for changes or an iterative procedure. Users benefit from transparent overviews of the assignment of the PLC modules which can be processed system-supported. The result is a cross-disciplinary engineering process.

### The new workflow creates much savings and improvement potential

Initially, the user selects and configures the PLC components in the TIA Selection Tool (smart way to configure products and entire plant parts) and transfers this hardware configuration to Eplan via the interface. There, in the classic hardware configuration, the wiring diagram shows the assignment of the PLC inputs and outputs with actuators and sensors. The I/O list created automatically on the basis of this project data is then passed on to the TIA Portal. Here the PLC program is created. The trick: The software developer knows exactly which assignment is available and can create his PLC program based on this data. The high quality of the project planning and programming are decisive for a fast and efficient commissioning.





# » APPLICATION RECOMMENDATION: AUTOMATION PROJECT CONFIGURATION //

A very frequently occurring task within the planning process of production and automation systems is the exchange of automation project configuration information of automation system devices between electrical engineering and control programming systems.

As the electrical and control engineer usually attend the project at different times the automation project configuration of the plant must be defined in both tools. It depends on the organization of the project which engineer is taking the lead but in both possible cases the information obtained needs to be exchanged.

Even when electrical engineering and control programming tools have different views of automation system information they have a common ground, e.g. devices which are involved in automation systems and their wiring and relation to control variables.

Beyond the named engineering tools for electrical engineering and control programming, other tools are of interest as well in the common data set of both tools. For instance, tools for mechanical engineering could be interested in the amount of wired devices while documentation tools could be interested in the wiring structure.

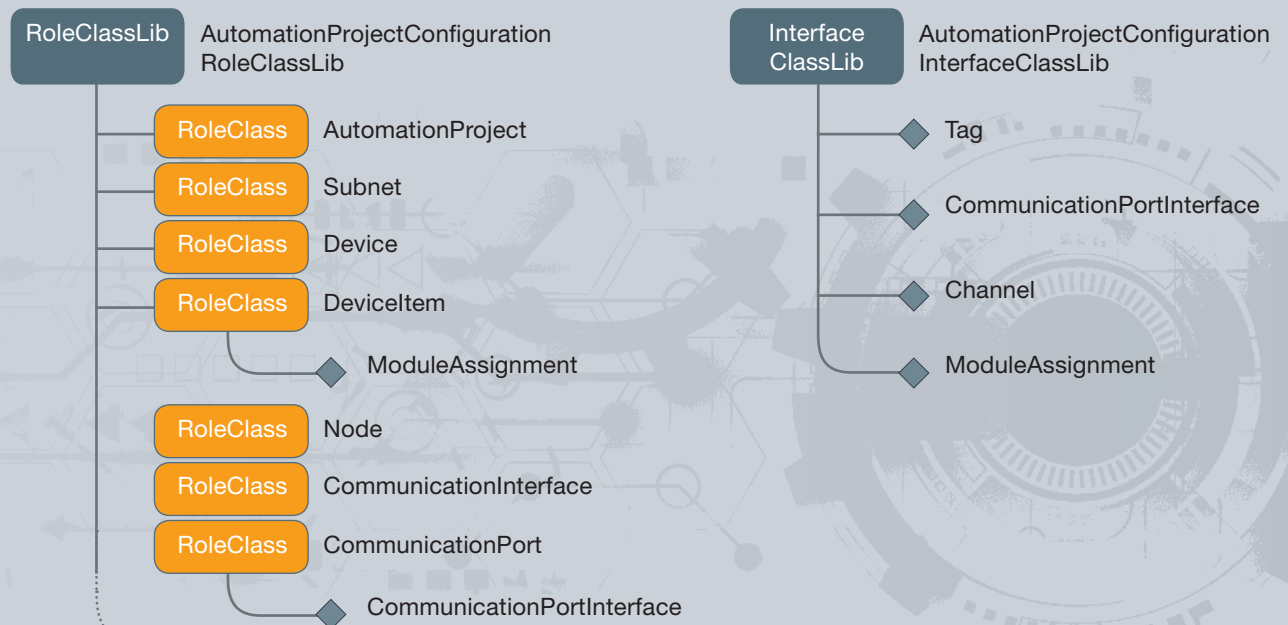
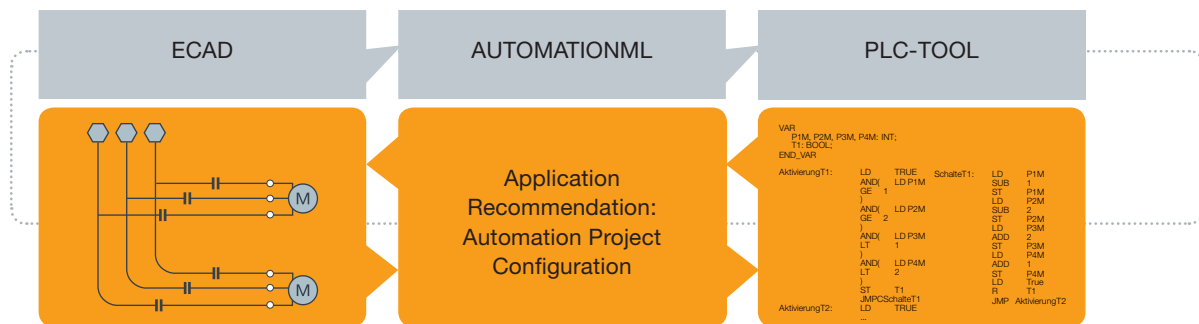
AutomationML has developed an application recommendation taking up the common concepts of automation project configurations and mapping them to AutomationML concepts.

**These are the most relevant concepts:**

- *Automation project* represents the complete, modelled automation system and aggregates all objects that belong to it,
- *Device* represents a collection in which the individual HW objects are brought together,
- *Device item* represents identifiable HW modules and submodules (CPU, I/O module, rack, etc.) of a device,
- *Tag* represents the symbolic name of I/O data and provides the logical view on a channel of a module, and
- *the channel* representing a process interface (e.g. digital or analogue input/output), as well as further data structuring and concepts related to communication systems.

These common concepts are translated into role and interface classes with appropriate attributes compressed in common role and interface class libraries defined in the application recommendation.

## AUTOMATION PROJECT CONFIGURATION BETWEEN ECAD AND PLC TOOL

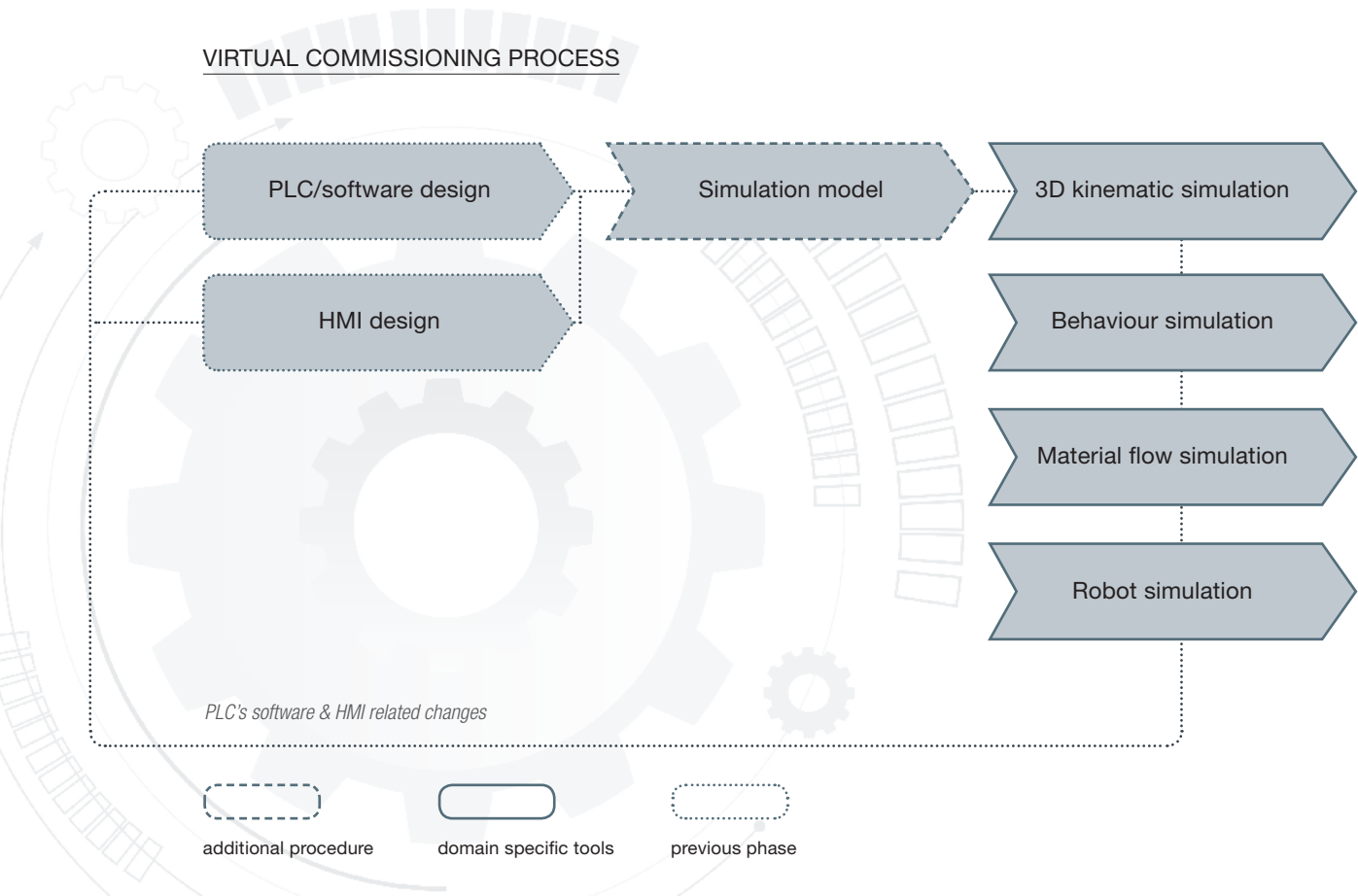


## » VIRTUAL COMMISSIONING USING AUTOMATIONML //

The virtual commissioning can be regarded as a separate set of engineering activities of the development process of production systems. In this phase, all functionalities of the PLC's software are tested based on a simulation model of the production system that doesn't exist in real hardware at this time.

For the purpose of Virtual Commissioning (VC), a first draft of the PLC's software as well as a working and tested simulation of the production system must be available. This requires a simulation model, e.g. a virtual one that contains several submodels, such as 3D simulation model, material flow and robot simulation.

### VIRTUAL COMMISSIONING PROCESS



Based on practical experience, the 3D simulation model created and used by virtual engineering is taken as a base in order to use it for VC. In the market, several tools provide required functionalities to prepare the 3D simulation model and to conduct VC, e.g. DELMIA, Process Simulate, NX-MCD, RF::SGEdit, RF::YAMS etc.

Additionally, the behavior model of the production system is required by VC. This model represents the logical behavior of each component connected to the PLC and communicates with the PLC via signals. For this purpose, the behavior model of each component

installed in the production system has to be created. In practice, each OEM creates this model on their own or assigns a subcontractor. The collected component models are used to build up the entire behavior model of the production system. In this context, each OEM stores behavior models in tool specific data format. As a consequence, fixed tools are used to run the behavior simulation and therefore also VC. Three widely used behavior simulation tools in practice are WinMOD®, SIMIT and RF::ViPer.

Those tools provide several communication interfaces that allow the exchange of simulation data in approximated real-time with other tools, e.g. RF::VIPer with RF::YAMS, WinMOD® with RF::YAMS, SIMIT with NX-MCD, etc. Along with the behavior simulation, robot program simulation is required as well. In this case, robot programs that are detailed during the PLC/ software design are used. To run the simulation of those robot programs, specific tools are required that have to support specific robot programming languages, e.g. KUKA, ABB, Fanuc, Kawasaki, Mitsubishi, etc. For this purpose, most robot manufactures provide a tool to create and simulate robot programs, e.g. RobotStudio, KUKA.Sim, etc. Such tools are standalone solutions and they expect the 3D geometric model of the production system that has to be extended by kinematic information in a previous step.

To achieve greater benefit of VC, modelling effort has to be kept to a minimum. For this reason, the continuous use of a simulation model is essential to avoid unnecessary repeated modelling time.

The use of AutomationML as a neutral data format has proven itself with regard to the realization of model exchange without data loss. Currently, several AutomationML export functions are developed to exchange 3D simulation models between process simulation tools (e.g. DELMIA, Process Simulation and NX-MCD, etc.) and VC's tools (e.g. RF::YAMS and NX-MCD). Some renowned OEMs in the car manufacturing industry use AutomationML export functions of DELMIA and Process Simulate to transfer 3D simulation models to RF::YAMS.

# EKS

InTec GmbH

Virtual Plants  
**Digital Shadow**  
 Workflow by AutomationML

## RF::Suite®

is a software family which enables virtualization and commissioning of virtual plants

-  Modular
-  Short Rump-Up
-  Quality Gate
-  Digital Shadow

[www.eks-intec.de](http://www.eks-intec.de)

# » HOW TO GET STARTED WITH AUTOMATIONML? //

AutomationML presents challenges to new users that are not to be underestimated regarding the knowledge required for the successful application of AutomationML. These relate to very different areas of knowledge that are needed during the implementation of new applications.

### Get to know the architecture

As a basis, a potential user must become familiar with the architecture of AutomationML. They need to understand how AutomationML maps specific data and how it can be applied and expanded. Core concepts are the role classes, the interface classes and the instance hierarchy with their internal elements.

- Role classes should be used to map the relevant concepts within an application domain and to give them the relevant properties in this domain (via attributes) and references to other concepts (via interfaces).
- Interface classes, in this context, depict the possible abstract relations between relevant concepts and their properties (via attributes).
- In the Instance Hierarchy, these role and interface classes are taken into account when creating internal elements and applied as semantic indicator.

### Gain the necessary basic knowledge

Potential users should refer to existing examples as well as the publications of the AutomationML e. V. and its members. On the AutomationML website you will find all documents published by the AutomationML e. V., and a list of specialist publications that provide information on various topics related to AutomationML. It may also make sense to book a training course with one of the

AutomationML members during which not only basic knowledge but also first solution concepts for the potential user's specific application can be discussed.

### Examine the use case

After exploring AutomationML as a technology, a detailed examination of the use case of AutomationML is required. At this stage, the user must be able to understand the application-specific data and, if necessary, to model the relevant data. This modeling forms the basis for the subsequent mapping of the application case using AutomationML.

### Application-specific mapping and modeling

1. The relevant modeling concepts are mapped using role classes that contain all the important properties.
2. At the same time, the relations between the modeling concepts must be mapped via interface classes.
3. Optionally, System Unit classes can now be designed as templates for reusable units.
4. In theory, a validation of the modelling is recommended. The design artifacts that were found in the analysis of the specific application field should be mapped by means of AutomationML in the course of the validation.

It is recommended to execute the four steps in close cooperation with the members of the AutomationML e.V. and the AutomationML workshop team. They are able to identify information about possible inconsistencies and incompatibilities of the developed modeling with the developments in the AutomationML e. V.

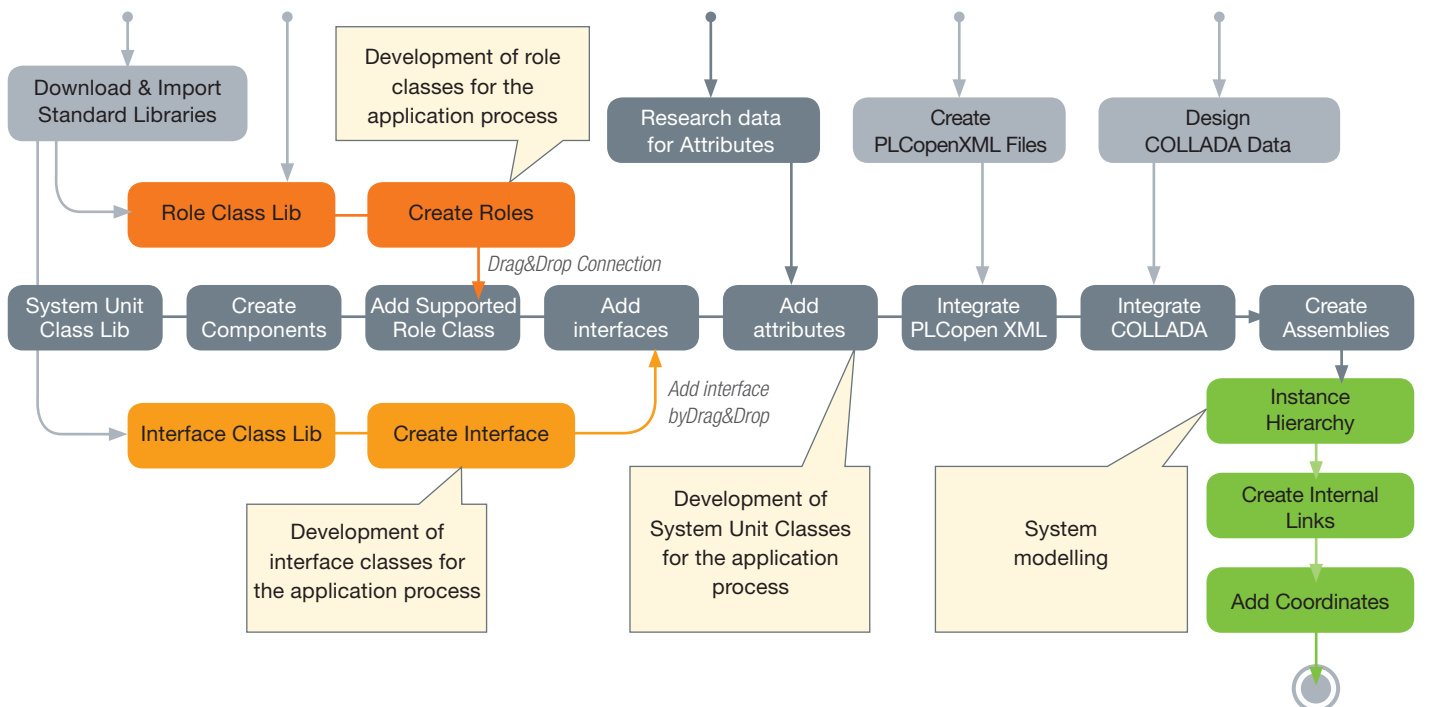
and to make appropriate suggestions for improvement. This is usually done without additional costs for the potential user as the AutomationML e.V. can validate the existing set of specification documents based on the application experience. This application experience can provide the basis for further standardization. This applies in particular to the developed role and interface class libraries.

### Implementation phase

Application-specific modelling is followed by the implementation phase. In this phase, the necessary tool interfaces and other systems required for the desired AutomationML-based data logistics must be created (possibly using the examples created in the modeling). Both the existing and newly created interfaces can be used here. In case of newly created interfaces the AutomationML provides sample software. Being part of AutomationML e.V.'s work, potential users can, of course, contact experienced users and exchange implementation experiences in order to avoid known mistakes.

When using the right approach, the implementation of the necessary software interfaces is one of the smaller problems of a potential user. This is due to the sample software available as well as the standardization of AutomationML Engines, which is promoted by AutomationML, as software systems for reading and writing AutomationML files. Furthermore, providing the AutomationML object model facilitates the implementation. These engines should follow the Common API developed by the AutomationML e. V. as it provides comprehensive object management capabilities for AutomationML based data-systems.

## IMPLICITE AUTOMATIONML APPLICATION PROCESS



## » CURRENT RESEARCH PROJECTS AROUND AUTOMATIONML //

AutomationML is under continuous development by members of the AutomationML association and further interested parties. Particular emphasis is put on extending the field of application and concretizing the application methods while ensuring upward compatibility.

At the time the document at hand was created the following development fields are/ were, among others, under consideration.

In many fields the **realization of a digital representation of existing system components** is currently discussed to ensure knowledge protection and reuse in engineering as well as improvements of system use. In this context, the idea of the Industry 4.0 component with its management shell is a good example.

In cooperation with the Industry 4.0 platform, OPC foundation and further partners, AutomationML is currently developing a component model enabling a **consistent representation of all relevant information on production system components** applicable for automatic data processing during engineering as well as automatic use in production.

AutomationML is continuously observing the development of **international standards** in the field of production system engineering to ensure the compatibility of AutomationML with such standards. One special example is the representation of component geometries in mechanical construction. JT has gained increasing practical interest and reached the state of an international standard. AutomationML will consider this in the context of its component description and, moreover, make efforts to allow general geometry and kinematics information while integrating JT files.

AutomationML has started to provide means for the exchange of data emerging within communication system engineering. Therefore, appropriate modelling means are defined. The next logical step is the **consideration of communication system and device configuration information**. AutomationML is developing new device description methodologies in cooperation with IO link association and ETG as well as system configuration description methodologies with FDT group.

AutomationML provides mechanisms to **reference third party files**, which enables referencing these classical device description files. Classical device description standard can thereby be integrated into AutomationML as is, but further information can be modelled additionally. The idea is to add another AutomationML model to the respective classical device description model to enrich the typical fieldbus-specific information of a

device by, for example, additional model descriptions, classes and also instances with individual configurations, plus the modelling of hierarchies and links between object instances, which allows the representation of modular devices as well.

A similar extension related to the AutomationML application recommendation – **automation project configuration** is under development. In this field, extensions regarding combinations of drive controllers, drives, gearboxes are currently discussed and loads related to drive chains are discussed to enable a simple exchange of drive chain information.

To enable the modelling of high level system behavior, especially with respect to production planning, AutomationML is currently considering the integration of IEC 62264 (based on ISA 95). This can lead to a first consideration of the **modelling of production system runtime data**.

In addition to the association's internal developments the AutomationML association also supervises and accompanies several research and development projects. A list of these projects can be found at the AutomationML homepage.

One development direction considered in different projects is the development of AutomationML-based **centralized data management systems**. This will enable a common knowledge and development space for all engineering data processing units (including tools) which are relevant with regard to the production system life cycle. This will also enable the improvement of project management and consistency management capabilities which leads to higher engineering quality.

Another development direction is the **representation of production system component capabilities** (especially functional capabilities / services). Such models can be applied in engineering to select appropriate components within component libraries and to select and parameterize appropriate execution capabilities at system runtime. One example is the MTP (module type package) to describe modules in the process industry.

The most appropriate way for interested parties to accompany new developments related to AutomationML is to **join the AutomationML e. V.** and, in doing so, to get access to all development data including intermediate specifications and examples.

The AutomationML association is encouraging all interested parties to identify further use cases of interest to enrich and extend the association's work.

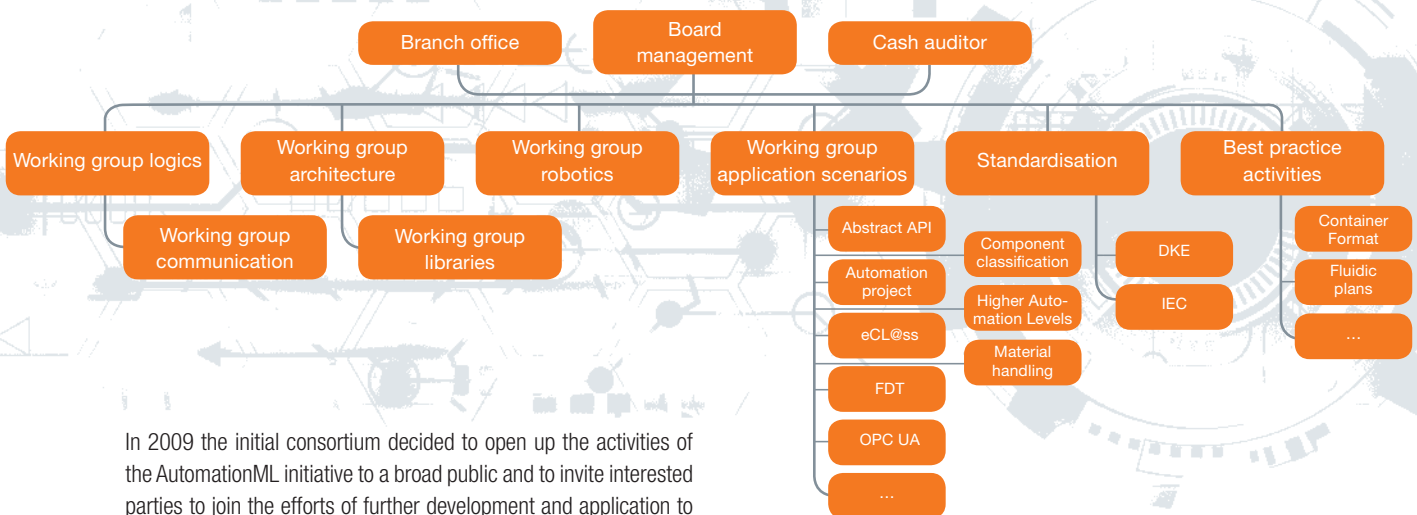
## » THE AUTOMATIONML ASSOCIATION //

In the early 2000s, industrial key players – especially in the automotive industry – have identified the need of more advanced data exchange mechanisms applicable in the engineering of production systems. Consequently, a group of nine volunteering companies and research institutes established the AutomationML initiative in 2006. Its objective is the development and maintenance of an open, neutral, vendor-independent, XML based, and free industry data representation standard which enables a domain independent transfer of engineering data across companies.

Within the first three years this consortium has developed the basic ideas and structures of the AutomationML data exchange format investigating requirements (considering mechatronic thinking in particular), selecting appropriate existing data formats, combining and enhancing them in a way that fulfills the identified requirements.

As a result, the first set of four whitepapers was prepared in 2009 and represents the AutomationML architecture, basic libraries, geometry and kinematics, and logic behavior. The graphic below best illustrates the structure achieved.

### CURRENT ASSOCIATION STRUCTURE



In 2009 the initial consortium decided to open up the activities of the AutomationML initiative to a broad public and to invite interested parties to join the efforts of further development and application to achieve an international standardization within IEC. As a consequence, the AutomationML association was founded as an association of German law in 2009.

To improve the work of the association and to ensure the broad applicability of the achieved developments, the AutomationML association has developed an internal structure with a management board that is elected every two years by the membership assembly and makes decisions about the main development directions, assisted by an office and two cash auditors, and a set of working groups implementing the developments. These members have a regular joint meeting within the AutomationML workshop every two months to ensure the consistent development and to discuss upcoming topics, new developments, standardization, and association issues. Furthermore the AutomationML association has created a network of liaisons with other associations to ensure mutual collaboration and common developments in fields of interest.

In the meantime the AutomationML association has reached a set of more than 50 members from all over the world covering nearly all industrial areas. Leading members with promoter status are ABB, BMW, Daimler, Huawei, KUKA, Siemens, and Volkswagen. Representatives of AutomationML come together in five working groups, four subworking groups and several ad hoc action groups to discuss current and emerging topics, share their ideas and determine the representation of the identified information within the data exchange format. The content worked out by some groups, i.e. the representation of the

information within AutomationML, is getting or will get internationally standardized within the IEC 62714 standard series. The AutomationML association has been cooperating with KRONOS, PLCopen, IEC, DKE, OPC Foundation, eCl@ss, ProSTEP iVIP, FDT Group, VDMA, VDA, ETG, and IO Link associations.

### These are the most important milestones of technical development since 2009:

- Provision and regular improvement of basic software systems like AutomationML Engine, Editor, and Test Center,
- Modelling of communication systems,
- Integration into OPC UA name spaces,
- Integration of eCl@ss and similar classification standards for semantic representation,
- Component modelling related to Industry 4.0,
- Modelling of automation project structures
- And many more

The AutomationML association is constantly increasing their field of work depending on the interest and requirements of its members. Thus, all interested parties that have additional requirements and application fields related to AutomationML are asked to join the association and to enrich the topics discussed in the working groups.

## » AUTOMATIONML'S STANDARDIZATION //

Concepts of AutomationML are standardized by the AutomationML association and published as whitepapers (WP), best practice recommendations (BRP), or application recommendations (AR). But some concepts have the potential to get standardized internationally within the standard family IEC 62714.

AutomationML intends to be applicable in the entire engineering process of production systems. A consistent and lossless data exchange in this heterogeneous software tool landscape of the engineering process means that the syntax, semantics, and structure of a production system model (to be stored in AutomationML) are fully specified. But is it realistic that one association is capable of defining all of this for production systems? Or can it be divided into smaller manageable tasks?

The approach chosen by the AutomationML association was to separate the syntax from the semantics. After having specified and standardized the syntax and basic semantics, a step-by-step identification and specification of discipline-specific semantics as well as production system model structuring can take place.

### Whitepapers specify syntax and basic semantics

Whitepapers containing essential concepts are forwarded to the IEC TC 65/SC 65E/WG 9. This working group within the IEC (International Electrotechnical Commission) is responsible for the international standardization of AutomationML in the standard family IEC 62714 "Engineering data exchange format for use in industrial automation systems engineering – Automation markup language". Currently there are seven whitepapers available.

**Whitepaper part 1** is dealing with AutomationML's architecture and the general requirements for its application. The syntax as well as basic modeling concepts and basic semantics (object semantics as well as object interface semantics) are standardized. AutomationML therefore uses the data exchange format CAEX, which is standardized in IEC 62424. Since AutomationML uses CAEX beyond its original scope to connect all software tools involved in the engineering process of production systems, restrictions and additions needed to be made to CAEX. Whitepaper part 1 has become IEC 62714-1.

**Whitepaper part 2** uses the mentioned basic semantics and extends these to enable a broader standardized basis to semantically describe the data in AutomationML. This part is called "Role class libraries". Role class libraries are one of four concepts of CAEX which AutomationML uses. The meaning or semantics of objects of the production system model are defined neutrally by means of role classes. Whitepaper part 2 has become IEC 62714-2.

If further aspects need to be added to objects, e.g. geometry and kinematics information to enable visualizations and advanced simulations or additionally logic models to enable virtual commissioning, further AutomationML parts are consulted.

**Whitepaper part 3** defines how COLLADA documents (standardized in ISO/PAS 17506) can be integrated into the CAEX part of AutomationML. It also provides extended semantics and modelling examples. Whitepaper part 3 has become IEC 62714-3.

On the one hand **Whitepaper part 4** focuses on how logic models can be modeled in a neutral way so that they become storable in a data exchange format based on PLCopenXML, which is currently under standardization in IEC 61131-10. On the other hand it depicts how those logic descriptions can be integrated into the CAEX part of AutomationML. It also provides extended semantics and modelling examples. Whitepaper part 4 is currently developed to become IEC 62714-4.

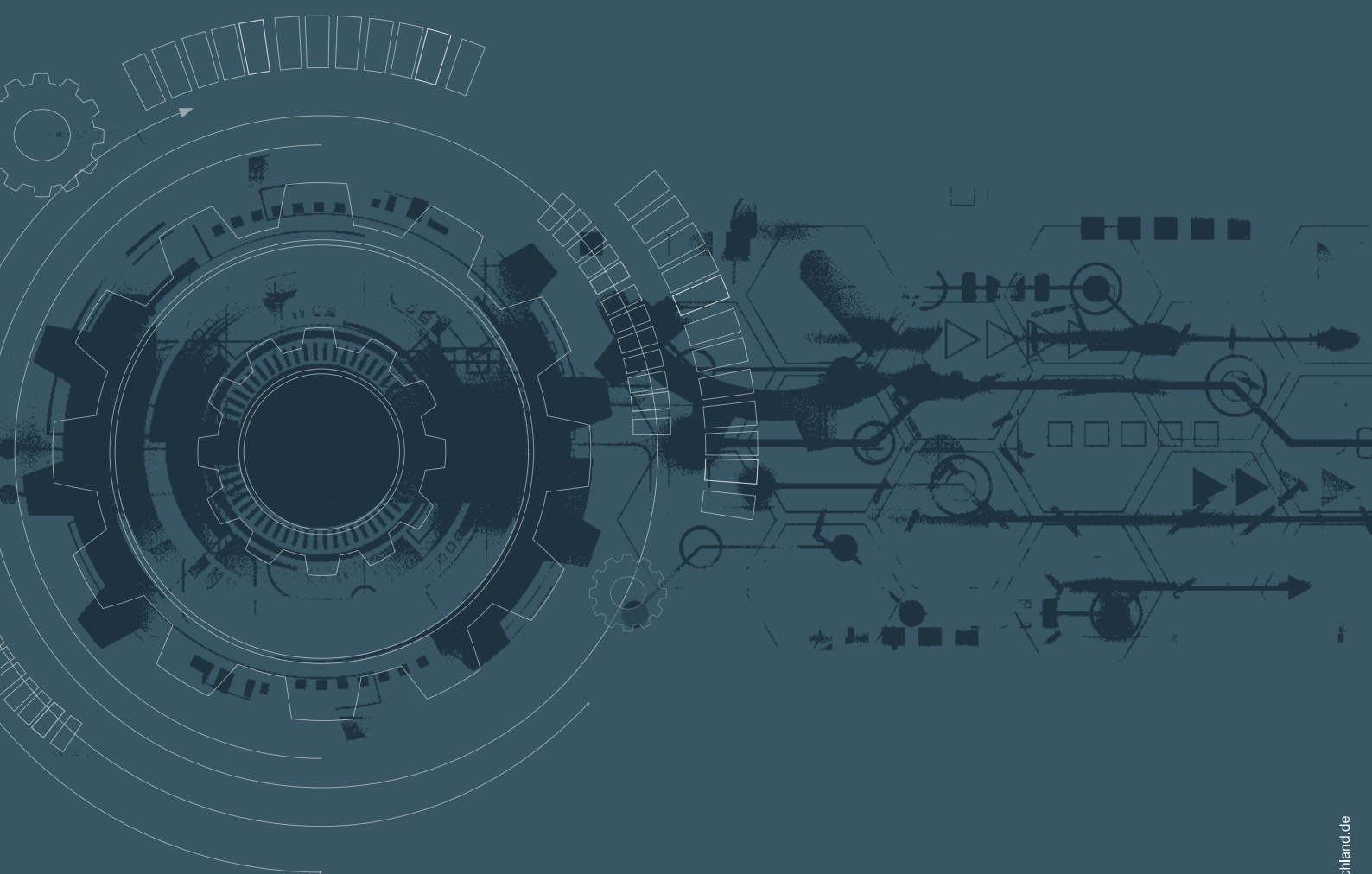
The **communication whitepaper** deals with the modelling of communication networks. The **whitepaper on integrating eCI@ss** in AutomationML describes how to exploit a given eCI@ss catalog to develop a role class library, but also how to exploit the attribute semantics of eCI@ss. The **OPC UA for AutomationML whitepaper** contains the specification of a translation method for mapping AutomationML models to OPC UA node sets of the OPC UA information model and, thereby, making AutomationML models transferable by means of OPC UA.

When misinterpretations, errors, or rework of the concepts are identified through the application and implementation of those whitepapers, they are addressed in BPRs. BPRs are then integrated into the next edition of the corresponding whitepaper.

However, to realize a consistent and lossless data exchange, AutomationML's meta model needs to be used to build up an AutomationML model. Therefore, the mechanism to flexibly extend semantics can be applied and a certain structuring needs to be defined. As the AutomationML association is member-driven and the data exchange format AutomationML is also open and free of charge, companies, committees, and other associations were and are free to develop AutomationML models, which are most beneficial to them, and are free to start specifying – within or outside the AutomationML association. Due to this fact, AutomationML evolves where consistent and lossless data exchange is needed the most. Members also develop AutomationML models in organized working groups within the AutomationML association. Those are published in ARs.

The AutomationML association is no controlling or regulating institution. Instead it is a harmonizing entity that establishes liaisons with other associations to synergize, which provides a panel for collaboration to its members as well as a central publication platform for specifications developed by its members.

# <AutomationML/>



**AutomationML e. V. c/o IAF**

Universitätsplatz 2  
39106 Magdeburg | Germany

E-Mail: [office@automationml.org](mailto:office@automationml.org)



[www.automationml.org](http://www.automationml.org)