# AutomationML in a continuous products life cycle: a technical implementation of RAMI 4.0

**Markus Kiesel**
*Faculty of engineering*
*Albstadt-Sigmaringen University*
Albstadt, Germany
kiesel@hs-albsig.de

Nicolai Beisheim
*Faculty of engineering*
*Albstadt-Sigmaringen University*
Albstadt, Germany
beisheim@hs-albsig.de

***Abstract*—The global market for manufacturing companies is highly competitive, especially the differences between low-wage and high-wage countries lead to various market strategies. The focus of companies in high-wage countries is often on the preservation of a technical advantage, or prime quality products [1]. The technical advantage often come from the core areas of the manufacturers, such as machining technologies, but currently new technologies are increasingly emerging from the information technology sector. In Germany, this trend, which has been going on for several years now, is summarized under the keyword "Industrie 4.0". By setting the focus on such technologies, there is a high chance of increasing key factors e.g. productivity or flexibility [2]. To maintain an interoperability between the, by information technology enhanced, systems there are currently multiple concepts such as the Reference Architecture Model Industry 4.0 (RAMI 4.0) [3] or the Industrial Internet Reference Architecture (IIRA) [4]. While IIRA focuses on interdisciplinary interoperability, the German RAMI 4.0 is particularly suitable for production-intensive industries. The RAMI 4.0 concept requires the introduction of an Asset Administration Shell for each material and immaterial object (asset), which provides several functionalities such as communication or identification. The Asset Administration Shell is connected with the asset throughout the whole product life cycle. Thus it needs to be able to save the heterogeneous data which is generated in each step of the product life cycle. Therefore an approach is necessary which is able to provide such a flexibility within the data structure and also provide the functionality to communicate range of RAMI 4.0. This contribution presents the approach of an AutomationML-based Asset Administration Shell, which is able to handle the heterogeneous data due to the AutomationML basis, provides an RAMI 4.0 compatible communication functionality and is extensible for further features. The implementation of the framework is platform-independent, which enables integration in classic PLC systems (e.g. Beckhoff) as well as several IOT Platforms (e.g. Raspberry PI).**

***Index Terms*—component, formatting, style, styling, insert**

## I. Introduction

In most common development processes, the individual departments are separated according to their specific focus, which leads to various intersections with other departments, especially regarding information and data. Since the respective department communicates predominantly among itself as well as with other departments of similar fields, the data formats are also adapted to this purpose. Due to increasing digitalization, however, the boundaries between the departments are becoming increasingly blurred. In these area of conflict it is difficult to maintain a persistent and complete data model throughout the complete product lifecycle, as the generated data is highly heterogeneous. For this reason the use of a universal data format which can contain the heterogeneous data of the individual domains is therefore increasingly necessary in the future. One example for such a format is AutomationML. However, digitization does not only affect the departments of a company or the cooperation between them. The machines themselves are also increasingly changed by digitization. One example is the distribution of intelligence to individual assemblies in order to achieve the most modular structure possible. Here, too, standards are an important factor, since the integration of external modules can otherwise involve a great deal of effort. Several approaches are currently available to counteract this impending problem as early as possible. One promising approach is the reference architecture model Industry 4.0, which, like AutomationML, enables a very flexible structure.

## II. Relevant Work

For a better understanding of the following explanations, some basic information is discussed in the following section.

### A. Reference Architecture Model Industry 4.0

In order to give Industry 4.0 a tangible structure and to ensure the interoperability of the systems, the German Electrical and Electronic Manufacturers' Association (ZVEI) developed the reference architecture model Industrie 4.0 in cooperation with various industrial companies.

*1) Layer model:* The core of the reference model is a three-axis layer model which is depicted in figure 1. It provides the possibility to represent any state of an arbitrary technical asset within the product life cycle.

*2) CP-Classification:* The CP Classification is intended to enable a simple classification of technical objects in the grid of the reference architecture model Industry 4.0. The matrix of the CP classification is shown in Figure 2. The X-axis shows the communication capability and the Y-axis the recognition in the system.

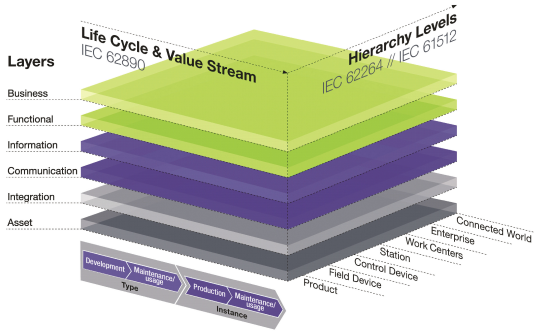


Fig. 1. Layer model of RAMI4.0 [3]

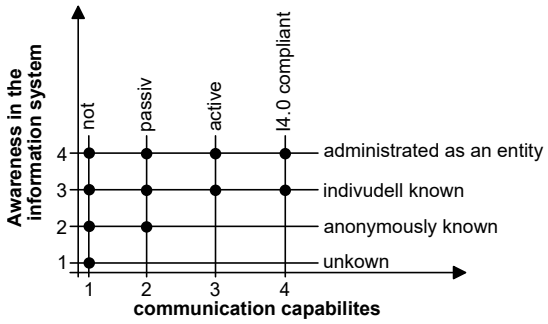


Fig. 2. CP classification of RAMI4.0 [3]

*3) Asset Administration Shell:* In order to depict a technical object in the digital world, the concept of administration shells is introduced in the reference architecture Industry 4.0. The combination of administration shell and technical object is referred to as Industry 4.0 component. According to the CP classification, which was already discussed earlier, industry 4.0 components therefore correspond to a CP classification of CP43 or CP44. In this paper, therefore, only elements of this characteristic are considered. The administration shell not only manages the data of the technical object but can also make its own functions available. These are made available as digital services in accordance with the reference architecture model. An example of such a service can be the execution of a diagnosis of the technical object by the corresponding administration shell. For example, statements about the remaining service life or the next service assignment are then calculated on the basis of the data collected.

### B. AutomationML

Due to the rising complexity of Industry 4.0 based production systems it is obligatory that engineering teams of different departments can exchange information efficiently. One format which can handle heterogeneous data is the XML based data format AutomationML (see e. g. [6], [7]]). It can contain much more information than for example a typical CAD exchange format like STEP or IGES. To make AutomationML easy accessible it incorporates several standards.

The open standards, which are used by AutomationML, are shown in Figure 3. The AutomationML file itself is based on

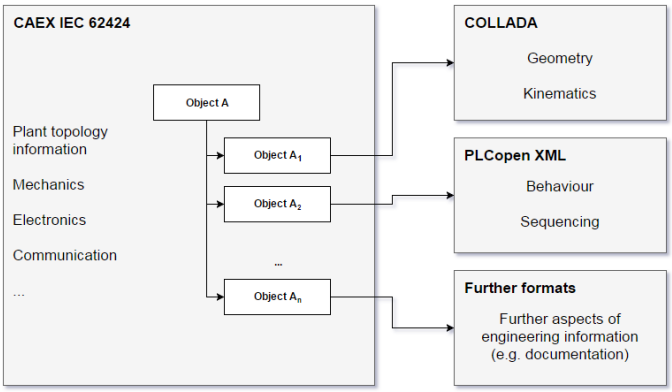


Fig. 3. AutomationML Overview [5]

the CAEX Format (IEC 62424) which is just slightly enriched. As it is XML-based and due to the possibility to reference other files, it is easy expandable. The present components, the hierarchical structure as well as the connection between the components are described with the CAEX Format. The COLLADA standard provides the functionality for the representation of geometry. It is capable of saving geometry as a boundary representation (typically for CAD software) as well as a triangulated mesh representation. Besides the geometry, COLLADA can also contain information about the kinematics and physics of an object, as well as other geometry related information. The PLCopen XML format is also included into AutomationML and makes it especially interesting for virtual commissioning purposes. Since it is based on the IEC61131-3, it adds the functionality to store and transfer programming languages for PLCs, embedded controls and industrial PCs.

This data can be evaluated on software or hardware in the loop systems typically required for virtual commissioning. Also shown in Figure 3 is the ability to incorporate further formats to add special functionality to AutomationML.

## III. AutomationML based Asset Administration Shells

An administration shell accompanies a technical object over the entire duration of the product life cycle. A wide variety of data is generated, in the design phase, for example, this is predominantly planning data such as 3D CAD data. As soon as the technical object is used as an instance, the type of additional data also changes, in this case measurement data, for example, as well as service and service life data is generated. In order to enable the persistent collection of this highly heterogeneous data, it is necessary to select a very flexible system or format for the asset administration shell.

### A. Implementation overview

The reference architecture model Industry 4.0 provides a basic overview of the objectives to be achieved with the model. For the majority of the components, however, no implementation recommendations can be derived. The authors therefore make some assumptions in the following, which serve as a basis for the later implementation.

- runtime environment
  The software-technical execution of the administration shells can be very varying. On the one hand, it is possible to centrally store the data and the runtime environment of the administration shells in a database-oriented system. Depending on the choice of the database, however, restrictions can arise with regard to the type and structure of the data. Another possibility is to embed the administration shells decentralized, for example directly on the managed technical object. As there are plausible use cases for both application scenarios, a possibility should be chosen that enables both scenarios equally.

- data repository
  As already mentioned in the runtime environment, data can be stored central or decentral. In particular, the choice of the data format in which the data is made available plays a central role. A proprietary data format can lead to integration problems with external systems, especially due to there large variance in the software products available. It is therefore advisable to choose an open standard in order not to restrict the use of an administration shell. The chosen data format must be able to contain the already mentioned heterogeneous data, which is generated during the product lifecycle.

- communication
  The communication capability of an administration shell is elementary and should therefore receive special attention. In the reference architecture model, the term „service-oriented architecture" is used at this point. Communication based on such an architecture has proven itself in various software projects in recent years and is therefore also recommended here. However, the authors are of the opinion that a further communication option that is closer to the machine would facilitate the integration of the administration shells at the machine level. Therefore, two forms of communication are considered.

The resulting layer-like structure is shown in Figure 4. In this figure, the individual layers are already occupied with technologies that can fulfil the assumptions made. A fundamental consideration which has to be addressed with the selected programming language is the compatibility with different execution systems. Therefor an approach was selected which allows the execution of the code on different platforms such as Windows or Linux environments. Thus Java as programming language was selected, which allows due to the Java Virtual Machine to run the same code on different platforms.

### *B. Java AutomationML Framework*

The framework provided by AutomationML e.V. is currently only available on the basis of the .Net programming language C#. A use in Java is therefore not possible. For this reason,

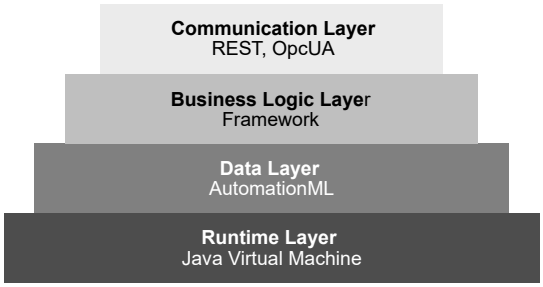


Fig. 4. Implementation Layer Structure

a Java-based AutomationML Framework is required for the approach described above, which allows the effective use of AutomationML under Java. Since this AutomationML framework is to be used in particular for the use in connection with the administration shells, some additional requirements have to be fulfilled.

- Easy integration of additional service life data
  The main task of the framework to be created is the integration of additional data that is generated during the product lifecycle. It should be possible to integrate any kind of additional data into the AutomationML file.

- Complete serialization and deserialization
  In order to make the data more robust against malfunctions and to reduce memory requirements, the data must be able to be both saved and loaded as AutomationML files (*.aml). This requires a serialization and deserialization mechanism.
- Toolkit for mathematical operations based on the FrameAttributeType attributes
  Positions and rotations of individual components can be stored in AutomationML as FrameAttributeType. This FrameAttributeType attribute contains the position and rotation of an element. The rotation is held in Euler angles, which is especially problematic for complex mathematical operations in 3d space. Therefore two new classes are introduced for the arithmetic operations based on the FrameAttributeType attributes.
  The *FramePosition* element contains the position portion of the FrameAttributeType attribute.
  The *FrameRotation* element contains the rotation part of the FrameAttributeType attribute, which is converted into a quaternion FrameRotation In order not to violate the rotation sequence defined by AutomationML e.V. (XYZ), the conversion is performed as shown in equation 1.

$$q_x * q_y * q_z = q_{res} \tag{1}$$

  The indices indicate the rotation around the individual axes. By converting the rotations into quaternions, the required arithmetic operations for spatial calculations are reduced and the mathematical problem of the "gimbal lock" (see also [8]) is avoided. Rotating a position $p_0$

by a given quaternion $q_n$ can then be expressed in the following way.

$$p_1 = q_n * p_0 \tag{2}$$

This system makes it easy to perform complex mathematical operations based on the FrameAttribute type.

- Integrated Toolkit for creating and modifying PlcOpenXML data
  In order to enable an administration shell and thus also the managed technical object to react adaptively to changed boundary conditions, it may be necessary to adapt the PLC program used. The open standard PlcOpenXML is integrated in the AutomationML standard for the purpose of managing PLC programs. In order to simplify the modification of these programs, a toolkit is implemented which enables the semantically and syntactically correct modification of PlcOpenXML data.

### C. Asset Administration Shell Framework

As shown in Figure 4, the administration shell framework is located between the communication layer and the data layer and represents the actual business logic. The mapping of the data to the communication is basically possible in two different forms.

1) Division of an AutomationML structure into individual data elements
2) Mapping of the complete AutomationML structure as a single data element

1 is particularly suitable for communication forms that require such a granular division, e.g. machine controls. Usually this is necessary for runtime-variable data. Variant 2 is e.g. suitable for planning data which should be imported into a software and extended if necessary.

*1) Machine to Machine Communication:* On aspect of industry 4.0 is the relocation of intelligence by embedding control units into subassemblies to form independent objects. This increases the need of a standardized machine to machine communication. Therefor the Asset Administration Shell has to able to communicate in this standardized way. In the recent past the OpcUA standard proves itself as valid competitor for future standardized machine to machine communication. Thus this standard was implement in the Smart Asset Administration Shell Framework.

*2) Human-Machine-Communication:* As even in highly automated processes the influence of an operator is necessary, the Human-Machine-Communication has to be in a comparable quality as the machine to machine communication. To provide a Human-Machine-Communication there are several options available. One typical option nowadays is to embed a display within the technical system, e.g. the control panel at a tooling machine. As this is probably the best option for machines with one single control unit, it can hardly be applied to machines which consist of dozens of control units. Therefor the Human-Machine-Communication is realized comparably to a service-oriented architecture, to enable a user to easily interact with arbitrary control units or Asset Adminstration Shells.

## IV. Conclusion and further research

The acceptance of industry 4.0 components and the reference architecture model industry 4.0 will depend strongly on whether the manufacturers of the systems find a common data technology basis. The combination of AutomationML and the reference architecture model industry 4.0 could represent such a data technical basis and thus contribute to the improved interoperability of these systems. In order to confirm this assumption, however, further research is necessary in the future.

## References


[1] G. Schuh, Ed., „Excellence in Production: Festschrift für Walter Eversheim." Aachen: Apprimus-Verl., 2007.

[2] M. Brettel, M. Klein, and N. Friederichsen, „The Relevance of Manufacturing Flexibility in the Context of Industrie 4.0", Procedia CIRP, vol. 41, pp. 105–110, 2016.

[3] „Referenzarchitektur Industrie 4.0" (RAMI4.0), 91345, 2016.

[4] S.-W. Lin et al., „The Industrial Internet of Things Volume G1: Reference Architecture." [Online] Available: https://www.iiconsortium.org/IIC_PUB_G1_V1.80_2017-01-31.pdf.

[5] AutomationML e.V., "AutomationML data representation," [Online]. Available: http://automationml.org/. [Accessed 18 10 2016].

[6] R. Draht, A. Luder, J. Peschke and L. Hundt, AutomationML - the glue for seamless automation engineering, IEEE International Conference on Emerging Technologies and Factory Automation, 2008.

[7] A. Lüder, L. Hundt and A. Keibel, Description of manufacturing processes using AutomationML, IEEE Conference on Emerging

[8] E. Hemingway, O. Reilly, „Perspectives on Euler Angle Singularities, Gimbal Lock, and the Orthogonality of Applied Forces and Applied Moments", [Online] Available: http://dynamics.berkeley.edu/assets/Gimbal-Lock-Final.pdf [Accessed 11 09 2018].