Transdisciplinary Engineering for Complex Socio-technical Systems – Real-life Applications
J. Pokojski et al. (Eds.)

© 2020 The authors and IOS Press.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License 4.0 (CC BY-NC 4.0).
doi:10.3233/ATDE200070

Using AutomationML and Graph-Based Design Languages for Automatic Generation of Digital Twins of Cyber-Physical Systems

Nicolai BEISHEIM [a], [1], Markus KIESEL [b], Markus LINDE [a] and Tobias OTT [a]
[a]*Albstadt-Sigmaringen University, Albstadt, Germany*
[b]*CMC-Kiesel GmbH, Hechingen, Germany*

Abstract. The interdisciplinary development of smart factories and cyber-physical systems CPS shows the weaknesses of classical development methods. For example, the communication of the interdisciplinary participants in the development process of CPS is difficult due to a lack of cross-domain language comprehension. At the same time, the functional complexity of the systems to be developed increases and they act operationally as independent CPSs. And it is not only the product that needs to be developed, but also the manufacturing processes are complex. The use of graph-based design languages offers a technical solution to these challenges. The UML-based structures offer a cross-domain language understanding for all those involved in the interdisciplinary development process. Simulations are required for the rapid and successful development of new products. Depending on the functional scope, graphical simulations of the production equipment are used to simulate the manufacturing processes as a digital factory or a virtual commissioning simulation. Due to the high number of functional changes during the development process, it makes sense to automatically generate the simulation modelling as digital twins of the products or means of production from the graph-based design languages. The paper describes how digital twins are automatically generated using AutomationML according to the Reference Architecture Model Industry 4.0 (RAMI 4.0) or the Industrial Internet Reference Architecture (IIRA).

Keywords. AutomationML, Graph-based Design Languages, Digital Twin, Cross-domain Engineering, RAMI 4.0

Introduction

Many products but especially production systems for the industry are mechatronic systems, which are developed by many people working together in interdisciplinary teams. The development process for those systems is nowadays seperated in different concerns, which are handled by the specialist departments such as mechanical, electrical and IT. The generated data however, are passed along the development steps from one department to another. Which leads to a high amount of intersections. The data, which the single departments receive along this process, is often converted in order to make it readable for the used engineering tools. Conversion of data always leads to a reduction

[1] Corresponding Author, Mail: beisheim@hs-albsig.de.

of the original data and causes therefore extra effort in order to restore the missing information. The increasing digitalization of production systems and the associated increase in complexity, will most likely lead to aggravation. Especially the correlation between the IT department and the mechanical department ist critical for digital twins, because the dataformats that are used by commercial CAD software differs from the ones that are mainly used for graphical simulations.
To improve the situation a data format is needed which is capable of maintaining a unified digitalmaster model throughout the whole development process, that can be used by the IT tools off all specialiced departments without losing information that is required to build an accurate cyper physical system. One example of such a data format is AutomationML, which is currently developed by the Automation ML e.V. in cooperation with a large number of companies and universities. It is highly extensible due to its XML-based structure and therefore capable of saving the heterogeneous data, which is generated throughout the development process.

To deal with the rising complexity of production systems it is mandatory to increase the amount of simulation based functional validations. The current workflow for such validations is often based on manual labour, which is needed for the creation of the simulation models, and therefore prohibit fast iterations.

The approach presented in this paper incorporates a combination of the mentioned AutomationML standard along with an automated process of model generation by graph based design langues in order to overcome the limitations in the production process of cyber physical systems.

1. Relevant Work

1.1. Reference Architecture Model Industry 4.0

Industry 4.0 is in broad fields a very abstract concept, which is why the German Electrical and Electronic Manufacturers' Association (ZVEI) is developing the Reference Architecture Model Industry 4.0 (RAMI 4.0) in cooperation with various industrial companies.

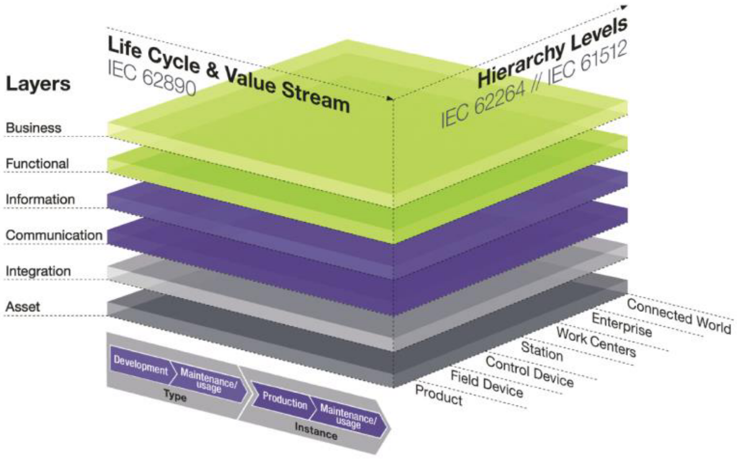

Figure 1. Layer model of RAMI4.0 [1].

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 shown in Figure 2. The x-axis shows the communication capability and the y-axis the recognition in the system.

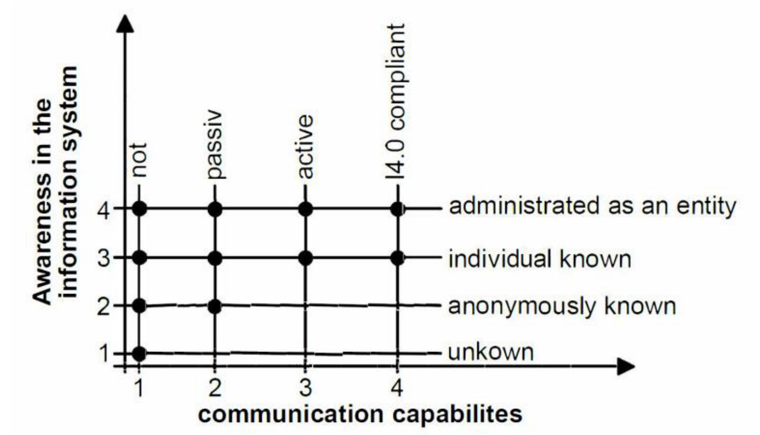
Awareness in the information system
not
passiv
active
I4.0 compliant
4
3
2
1
administrated as an entity
individual known
anonymously known
unkown
1
2
3
4
communication capabilites

Figure 2. CP classification of RAMI4.0 (based on [1]).

3. *Asset Administration Shell*
   In order to depict a technical object in the digital world, the concept of administration shells introduced in the Reference Architecture Model Industry 4.0. The combination of administration shell and technical object 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.

1.2. 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. [3], [4]]). 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.

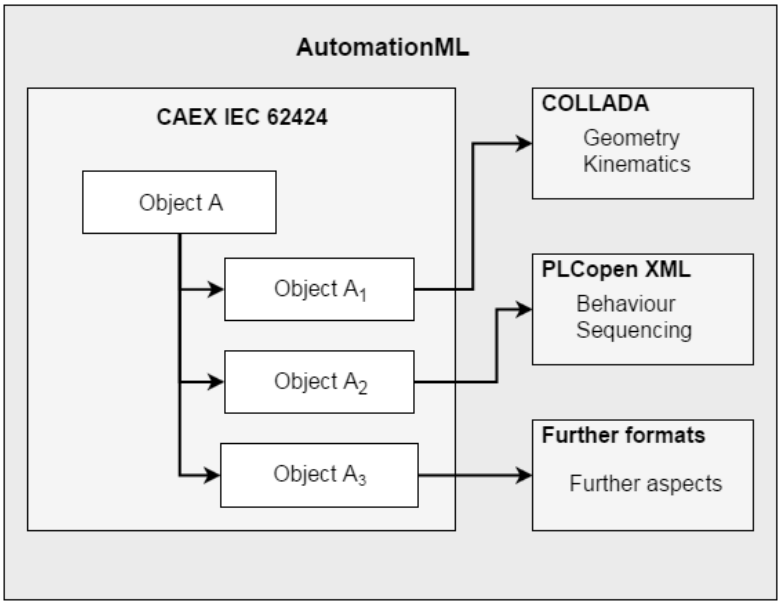

Figure 3. AutomationML Overview [2].

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

2. 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.

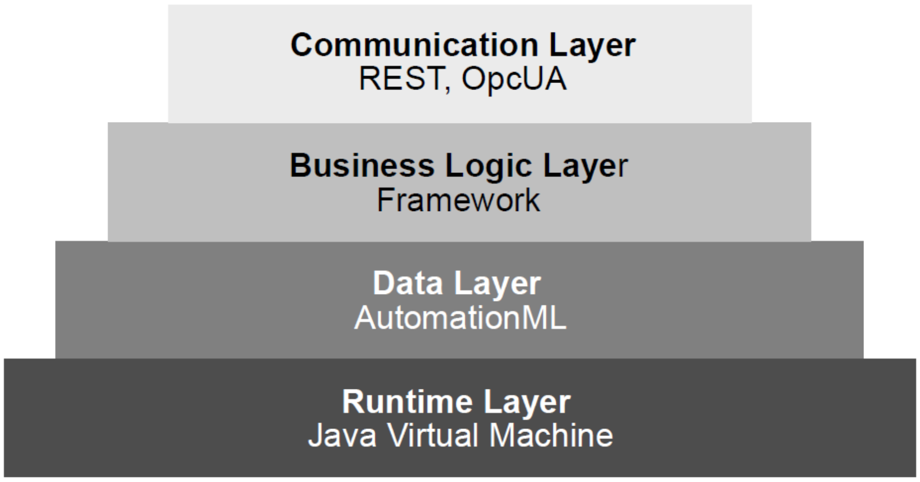

Figure 4. Implementation Layer Structure.

2.1. 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 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. Therefore, 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.

2.2. Java AutomationML Framework

The framework provided by AutomationML e.V. is currently only available on the basis of the .NET Framework programming language C#. A use in Java is therefore not possible. For this reason, 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., an in 2006 founded industrial consortium, 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 p0 by a given quaternion qn 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.

2.3. 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

The first 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.

2.3.1. Machine to Machine Communication

One 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. Therefore, the Asset Administration Shell has to be able to communicate in this standardized way. In the recent past the OPC UA standard proves itself as valid competitor for future standardized machine to machine communication. Thus, this standard was implemented in the Smart Asset Administration Shell Framework.

2.3.2. 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. Therefore, the Human-Machine-Communication is realized comparably to a service orientated architecture, to enable a user to easily interact with arbitrary control units or Asset Administration Shells.

3. 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. These models can automatically be generated through a production pipeline based on graph-based design languages as described by Kiesel et al. [5] and Beisheim et al. [6], which allows a higher number of simulations for functional validation. In order to confirm this assumption, however, further research is necessary in the future.

Acknowledgement

The German research project ´Digitaler Produktlebenszyklus (DiP)´ (information: http://dip.reutlingen-university.de) is supported by a grant from the European Regional Development Fund and the Ministry of Science, Research and the Arts of Baden-Württemberg, Germany (information: http://www.rwb-efre.baden-wuerttemberg.de).

References

[1] DIN SPEC 91345, *Referenzarchitekturmodell Industrie 4.0 (RAMI4.0)*, Beuth Verlag, 2016.

[2] AutomationML e.V., *Standardized data exchange in the engineering process of production systems*, Accessed: 15.01.2020. [Online]. Available: https://www.automationml.org/o.red/uploads/dateien/1544706233-automationml.pdf

[3] R. Draht, A. Lüder, J. Peschke and L. Hundt, *AutomationML - the glue for seamless automation engineering*, IEEE International Conference on Emerging Technologies and Factory Automation, Hamburg, September 15-18, 2008, IEEE Xplore, 2008

[4] A. Lüder, L. Hundt and A. Keibel, *Description of manufacturing processes using AutomationML*, IEEE 15th Conference on Emerging Technologies & Factory Automation (ETFA 2010), Bilbao, September 13-16, 2010, IEEE Xplore, 2010

[5] M. Kiesel, P. Klimant, N. Beisheim, S. Rudolph and M. Putz, *Using Graph-based Design Languages to Enhance the Creation of Virtual Commissioning Models*, Procedia CIRP, Vol. 60, 2017, pp. 279-283.

[6] N. Beisheim, M. Kiesel and S. Rudolph, *Digital manufacturing and virtual commissioning of Intelligent Factories and Industry 4.0 systems using graph-based design languages*, Proceedings of the 25th ISPE Inc. International Conference on Transdisciplinary Engineering, Modena, July 3–6, 2018. Vol. 7, IOS Press, 2018, pp. 93-102