Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

# Integration of Existing Cyber-Physical Manufacturing Systems into a Common Information Model

**Sebastian Schmied** * **Selvine G. Mathias** * **Daniel Großmann** *
**Ulrich Jumar** **

* *Technische Hochschule Ingolstadt, Zentrum für Angewandte Forschung, 85049 Ingolstadt, Germany (e-mail: [ Sebastian.Schmied, SelvineGeorge.Mathias, Daniel.Grossmann ]@thi.de ).*
** *Institut f. Automation und Kommunikation e.V., 39106 Magdeburg, Germany (e-mail: Ulrich.Jumar@ifak.eu)*

**Abstract:** In order to be able to serve constantly new customer requirements, manufacturing systems must be able to adapt to frequent changes. In addition, repeatedly objects are removed or added to the network. To control and monitor such a constantly changing system a mapping of existing manufacturing systems into a common information model is necessary. This model describes information that is produced and stored in different entities of the complete system. To create a common address space and expose the relations between the devices an aggregation of every element in the system is needed. This paper describes a methodology for the creation of an information model for a complete manufacturing environment, followed by an approach for the aggregation of the singular system entities. The concept of this paper is illustrated with a demonstrator. The results of this approach have been discussed in the following sections along with the proposal for further directions.

*Keywords:* Information Model, Aggregation, OPC UA, Cyber-physical manufacturing systems

## 1. INTRODUCTION

Manufacturing enterprises are encountering a variety of challenges today. Examples are ever-shorter product life cycles, changing market conditions and a increasing demand for individualized products. Therefore, there is an increasing demand for flexible manufacturing systems as stated by Grochowski et al. (2020) and Dotoli et al. (2018).

A fundamental feature of flexible manufacturing systems is that they can respond to changes in both the product and the production process. Especially in order-oriented manufacturing systems many different products are produced. In order to ensure the highest possible efficiency, a uniform language is necessary with which the information is communicated during the production process.

Most manufacturing systems have grown over many years and therefore consist of heterogeneous information sources. These provide information in a specific manner, depending on the single machine or database. If an exchange between several participants of the system is necessary, these interfaces are usually programmed for a specific application and can not be used by other participants in the production system.

In order to enable data exchange between all participants of a production system, an information model is needed which is a semantic description of all data available in the production process. For the technical integration of all elements of the system, a concept for aggregation is needed. Thus, vertical and horizontal integration between all levels and components of the production system can be achieved.

The creation of information models can either be done by industrial organizations e.g. associations of machine manufacturers or on user side (Schmied et al., 2019).

If a model is created by the manufacturer of a machine, all potentially necessary datasets are added and the information model is based on the detailed knowledge about the function of the machine. The collaboration of different machine vendors in the same field enables standardization of information models as it is for example done with OPC UA companion specifications.

In contrast, it is more relevant to the user side whether all information required for the specific production process is available in the information model. In addition, certain datasets such as the machine state must be standardized within manufacturing systems to provide comparability between different facilities.

If an information model is to be created for a production system, it makes sense to combine both approaches. On the one hand, a company has to determine which data it wants to retrieve from the information model and which data records are standardized throughout the company. On the other hand, preliminary work by industrial organizations can be used and thus the process of creating information models can be shortened. Due to the large number of components in use, an information model is not always

provided by the manufacturer. Therefore, it is necessary for the user side to create submodels by themselves.

This article presents a concept that identifies the minimal information exchange requirements of a manufacturing system and transfers them into an information model. This model is then aligned with existing standard information models. In order to enable horizontal and vertical integration it also shows a possible strategy to integrate all manufacturing entities via aggregation. To illustrate the concept it describes the implementation of the approach into a demonstration scenario.

After the introduction, an overview of the state of the art is given. Section three explains the concept, followed by a description of the demonstrator and a conclusion chapter.

## 2. STATE OF THE ART

In the state of the art section a brief overview about the Open Platform Communications Unified Architecture (OPC UA) and aggregation is given.

### *2.1 OPC UA*

The International Electrotechnical Commission (2016) describes OPC UA as the current state of the art information modelling and middleware technology. It implements a service oriented client-server and publish-subscribe communication architecture between different devices and systems independent of the used platform. Inheritance and type hierarchies are provided via an object-oriented architecture to enable information modelling (Faller and Höftmann, 2018). Mechanisms for a robust and reliable communication structure are parts of OPC UA. The recognition of communication interruptions, handling of lost messages and support for redundant systems can be named. Security features like user management and signing and encrypting messages are integral parts of the specification (Enste and Mahnke, 2011) (International Electrotechnical Commission, 2016).

OPC UA uses an object-oriented architecture, using inheritance and type hierarchies, to enable information modelling (Imtiaz and Jasperneite, 2013). The OPC UA address space is a network of object-orientend entities called nodes. Nodes are identified by a node-id, which is the combination of a namespace index and a node name. With hierarchical and non-hierarchical references semantic relations between the nodes are described and an information model is created (Derhamy et al., 2017). The OPC UA specification describes different node classes, such as objects, variables and methods. Predefined references and node types can be extended by subtypes, reference types or methods (International Electrotechnical Commission, 2015a). Instances of node types contain information about the data type and the unit of measurement of the attribute they describe. Information models are exposed by servers and can be read and explored by clients without the client knowing the information model themselves (International Electrotechnical Commission, 2015a) (Mahnke et al., 2009).

### *2.2 Aggregation*

The ongoing digitalization in the production environment leads to a scenario where every component of the production system is equipped with its own communication interface. For the introduction of a common information model it is necessary to access data from all devices. Multiple clients need to be able to read information of the integrated devices.

One solution is to create direct connections between clients and devices. This creates a situation where one client is connected to multiple servers. Every client needs to know the address of every server, to which a connection is established. If new devices are added the changes need to be reflected manually to the clients (Großmann et al., 2014). It is possible that a server is contacted simultaneously by various clients which can cause performance issues.

Another possibility is the use of an aggregation server. Such a server knows the connection details of all devices in the production system and adds them to a common address space. An aggregation server is able to expose a single point of entry to all clients, and handle security functions as for example user and load management (Großmann et al., 2014), (Wang et al., 2018), (Breunig and Schneider, 2019).

## 3. CONCEPT

This section presents the concept for the information model creation. The necessary steps are described subsequently. A demonstration scenario is explained during the process.

### *3.1 Identification of information objects*

At the beginning of the model creation process, a scope has to be identified. For this, one or more specific business processes can be chosen. As described by Irani et al. (2000) a business process produces a valuable output based on a defined input by executing a certain number of process steps. To create the valuable output an exchange of digital and physical elements needs to happen between process steps. Within every selected process, these elements have to be identified and described in information objects. An information object can generally be described as a collection of attributes similar to a class in object oriented approaches. To identify the information objects, a detailed definition of the inputs, outputs and used tools or machines of every process step is needed.

A possible documentation approach is the Supplier, Input, Process, Output and Customer Analysis (SIPOC). This analysis is performed for every process step. Every attribute or object that can be considered as an input or an output of the process step is documented in a predefined table. For all attributes the datatype, and if applicable the unit is documented. Supplier or customer can be machines, databases, other departments or computer programs. From the results of the SIPOC-analysis information objects are created. Often attributes are already grouped, for example if they are coming from the MES-System and are part of the same order. Examples are orders, work- or part-descriptions.

The analysis of the system and the creation of the information objects allows a detailed assessment of the manufacturing process. During this step it is possible to identify improvement opportunities as for example media breaks and duplicates of attributes or information objects. Strategies for system improvement are not part of this publication but should be considered during information object creation.

If the customer or supplier is a machine, the attributes that are sent or received to or from the machine are part of the machine information object.

An example of a SIPOC-sheet can bee seen in figure 1. From left to right, the input objects, the process and the output objects are described. For the input and output objects, the attributes and their suppliers are documented.

Milling Machine

| Supplier | Input | | | | Process | Output | | | | Customer |
|---|---|---|---|---|---|---|---|---|---|---|
| | Object | Attribute | Type | Unit | | Object | Attribute | Type | Unit | |
| Logistics | Part (Raw) | | | | Milling | Part (Milled) | | | | Logistic |
| Work Planning | Work description | program file | .h | | | | | | | |
| Work Planning | Order | due date | | | | | | | | |
| .. | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |

Fig. 1. Sipoc Analysis

The identified information objects are documented similar to a class diagram without relations following. Figure 2 displays the information objects of the demonstration scenario together with their corresponding attributes.

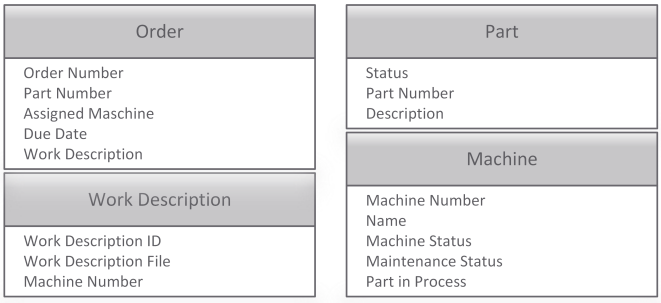


Fig. 2. Identified Information Objects

### *3.2 Classification of information objects*

To facilitate reference and object creation a classification of objects can be done. The paper proposes to create four groups of objects.

**- static objects**

**- dynamic objects**

**- manufacturing objects**

**- order objects**

The identified information objects will be assigned to one of the groups and specific actions for each group will be described in this chapter.

Static objects are objects that cannot be assigned to a specific order. This can be machines, tools, measuring devices or robots. The objects are used to change or measure something on the product and they perform a specific action. This action can be described with a method in the information model. A method returns a defined output based on a prior defined input. Static objects have parameters describing their own status, as oil temperature or which part is currently being processed.

The second group are dynamic objects. These are the objects to be changed during the process, for example milling parts, or a fluid in a process application. These objects are not capable to perform certain actions on their own, but their status is defined. Depending on the application this can be a single attribute describing the actions already performed on them, or a very detailed set of attributes that can also be used for simulation and further prediction of the manufacturing process. It might not be possible to store the information directly in the object, therefore the concepts of digital twins have to be used. Depending on the use case, a digital twin can be used for storing data or enabling complex simulation of the asset behaviour (Arm et al., 2018).

Manufacturing objects describe how a dynamic object is changed by a static object. These are for instance nc-programs or recipes. Often the information in these objects is saved in a specific file format.

The fourth group are objects containing order information. Typically this is information about quantities, due dates, the type of part to be manufactured and the allocations to specific static objects. Within this group there can be a certain hierarchy as order and suborder.

An overview about the object groups definition process can be seen in figure 3. Evaluating the information objects of the demonstration scenario as they can be seen in figure 2 the "Order" is part of the order objects group, "Part" is a dynamic object, "Machine" is a static object and "Work Description" is allocated in the manufacturing objects group.

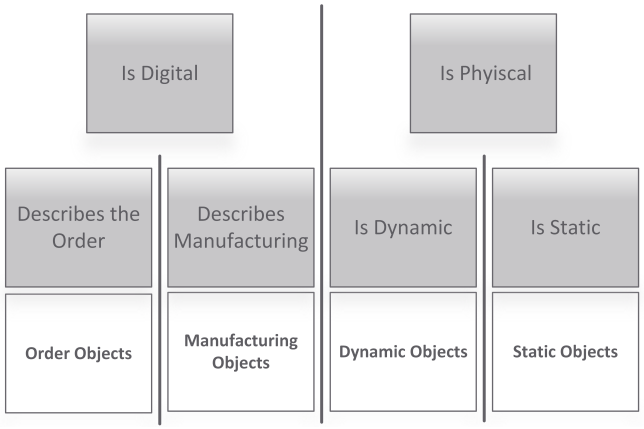


Fig. 3. Object Groups

### *3.3 Definition of References*

Following the definition of the information objects, references between the objects have to be documented. In the present approach these references are established based on so called linking attributes.

In general, a distinction is made between hierarchical and non-hierarchical references. Hierarchical references describe top-down relations such as order and suborder,

while for non-hierarchical references the order of precedence is irrelevant. In addition, references can be described by a string to increase semantic readability. For example, the "organizes" reference can be a subtype of a hierarchical reference.

Every information object is required to have at least one reference to another object. The references are created between instances of the information objects, based on one or more attributes of the object. The linking attributes can be compared to keys in relational database.

These attributes, as well as the type of reference, have to be defined, to enable the creation of the information model.

Figure 4 shows typical references between the objects identified in the previous section. The figure can be seen as a guideline, depending on the actual use case the references can differ. An example is the static object "Machine" that has a "processes" reference to the dynamic object "Part".

| | Static Objects | Dynamic Objects | Manufacturing Objects | Order Objects |
|---|---|---|---|---|
| Static Objects | Hierarchical Reference **„is part of“** | Non-Hierarchical Reference **„processes“** | Non-Hierarchical Reference **„uses program“** | Non-Hierarchical Reference „processes“ |
| Dynamic Objects | Non-Hierarchical Reference **„is processed on“** | Hierarchical Reference **„is part of“** | Non-Hierarchical Reference **„is described by“** | Non-Hierarchical Reference or Hierarchical Reference **„belongs to“** |
| Manufacturing Objects | Non-Hierarchical Reference **„is programmed for“** | Non-Hierarchical Reference **„describes“** | Hierarchical Reference **„is part of“** | Non-Hierarchical Reference **„is produced with“** |
| Order Objects | Non-Hierarchical Reference **„uses“** | Non-Hierarchical Reference or Hierarchical Reference **„organizes“** | Non-Hierarchical Reference **„uses“** | Hierarchical Reference **„has suborder“** |

Fig. 4. References Overview

An advisable starting point for reference creations are order related objects. These objects often have a logic structure based on their order type. Often complete assemblies are separated in different levels with varying granularity. The references between these levels are hierarchical references. The result is a treelike structure.

The second step is to evaluate every object based on its relation to an order. An order defines the main goal of the manufacturing process, therefore most objects are linked with an order in some way. Dynamic objects(e.g. "Parts") are "produced by" a certain order with the help of manufacturing objects (e.g. "NC-programs") and static objects (e.g. "Machines").

In a third step references between static, dynamic and manufacturing objects are added. These references describe the actual manufacturing process and are typically non-hierachical. A "Part" for instance "is produced on" a certain "Machine".

The last step is to add references based on the physical layout of the manufacturing area. Static and dynamic objects have a physical structure that can be mirrored in the information model. The physical dynamic object "Car" "contains" an "Engine". Both elements are manufactured separately and are connected in a later manufacturing step.

Within the given example the "Order" has a hierarchical reference to "Part", "Machine" and "Work Description". The non-hierarchical references between "Part" and "Machine" as well as between "Machine" and "Work Description" shows their dependencies during the production process. Figure 5 shows the references between the objects as well as the corresponding linking attributes.

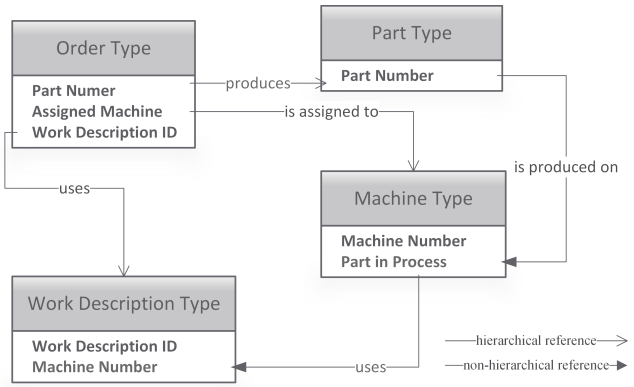


Fig. 5. Linking Attributes

### *3.4 Integration of existing standards*

Various industrial organizations have designed standard information models in order to enable standardization throughout different companies. Alternatively it is possible to design company specific standards, that can be used for the identification of system entities. A possible source for companion specifications is the OPC Foundation.

If a suitable companion specification is available, it should be compared with the information objects created in the previous steps. It is not always possible, that a companion specification covers the full extend needed for the relevant scenario, and in this case the missing attributes have to be added to the companion specification.

### *3.5 Definition of the information model*

Following the definition of relations, linking attributes and information objects, the actual information model can be defined. This publication has choosen the OPC UA information model notation as described in International Electrotechnical Commission (2015b) as a modelling standard. Every identified information object is represented as a OPC UA type. Subsequently, the references are added to the notation. In the class diagram linking attributes are present in both information objects that are linked to each other. Within the information model this is no longer necessary, as the references contain this information implicitly.

Figure 6 depicts the information model derived from the demonstration scenario. The four different types as well as their references can be seen.

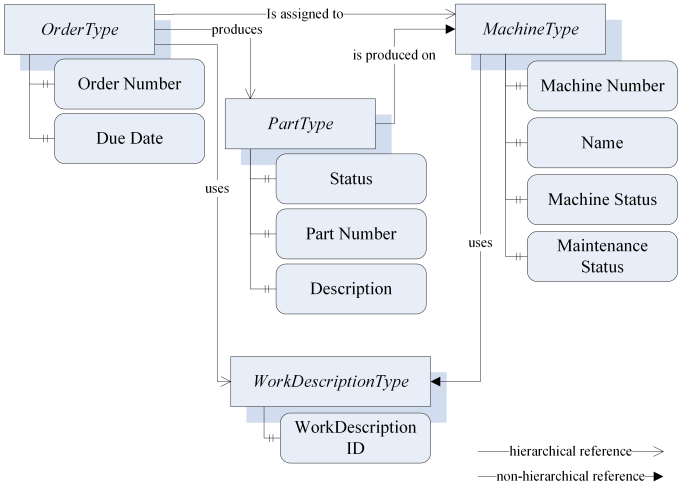


Fig. 6. Information Model Example

### *3.6 Aggregation of the system entities*

As a basis for the aggregation the work of Banerjee and Großmann (2017) was used. The aggregation process creates a common adress space and aggregates prior defined OPC UA servers into this address space. All servers are added on the same levels and the result is a flat address space. This approach does not reflect the relations between the single information objects in the address space. An example of a flat aggregated address space can be seen in figure 7.

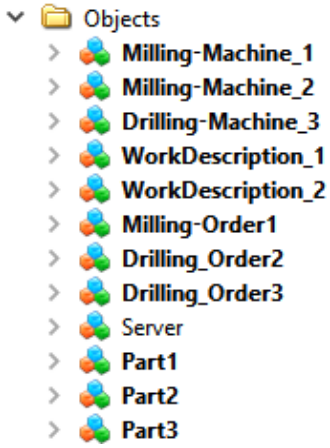


Fig. 7. Flat Aggregation

In order to create a structured address space based on the prior defined linking attributes and references, the concept of dynamic aggregation was developed. In addition to a flat aggregation, a server was created which is able to dynamically create references based on a prior defined overall information model. Figure 8 shows an example of a dynamically aggregated address space.

## 4. DEMONSTRATION

To show the viability of the concept, all process steps were executed in a demonstrator. The information model of the demonstrator is equal to figure 6. An integration of a simple milling process was established. The used components are the following:

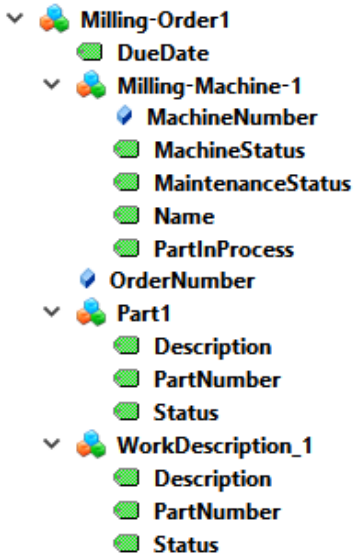


Fig. 8. Dynamic Aggregation

**- Order:** Order database in form of an SQL database

**- Machine:** Milling machine with Haidenhein ITNC530 controller

**- Work Description:** NC-programs in a network Storage

**- Part:** Digital twin of a milling part running of server

The SQL database and the NC-files on the network drive were exposed to the address space with the Prosys OPC UA SDK for Java (Prosys OPC Ltd, 2019). To display every row in the order database as a node in OPC UA a server was created directly connecting to the SQL database and performing a query at predefined intervals. For the NC-files the OPC UA server is monitoring a folder, as soon as a file is added it is a displayed as a node in the OPC UA adress space.

For connection of the Haidenhain ITNC530 controller the RemoTools SDK (HEIDENHAIN, 2017) was connected to a OPC UA server created with the Unified Automation .NET OPC UA SDK (Unified Automation, 2019). For testing, a real machine as well as a simulated machine was used. A OPC UA server was created for every machine.

As a basis for a simple digital twin of the "Part" object the Unified Automation .NET OPC UA SDK (Unified Automation, 2019) was used. An OPC UA server was created that is able to provide several instances of the "Part" type.

The Aggregation software is also based on the Unified Automation .NET OPC UA SDK (Unified Automation, 2019). In a first step the software individually connects to a predefined list of OPC UA servers. It reads every node and adds it to a common address space. If during this process a linking attribute is identified, it is saved to a list together with the corresponding node id of its parent node.

In a second step all linking attributes are compared and based on the predefined business logic, the references are added to the address space. This can be seen in figure 8.

During the testing of the demonstrator setup the correct function of the aggregation server was shown. But also limitations within the actual setup were identified. It was noticed that the aggregation of a large number of nodes (10000+) slows down the aggregation process to an unacceptable amount of time and increases the access time to the relevant information, in future work it has to be examined how this latency can be reduced.

If the address space changes due to certain events such as addition or removal of orders a re-browse within the aggregation software needs to happen. In the current implementation this is done by re-browsing in fixed intervals. It has still to be proven, that the re-browse period is frequent enough for the use case.

## 5. CONCLUSION

The paper shows how different entities of a production system can be integrated into a common address space. This includes the creation of an information model and thereby describing the relations between the different production entities. An approach for aggregation of the independent entities is shown.

As manufacturing areas and therefore information models undergo frequent changes a concept for information model life cycle management might be an interesting topic for future work. It is also planned to develop a concept for mapping of the identified objects into the Asset Administration Shell meta-model.

## ACKNOWLEDGEMENTS

The presented paper was elaborated within the research project InMoFlex. This project is founded by the Federal Ministry of Education and Research of Germany.

## REFERENCES

Arm, J., Zezulka, F., Bradac, Z., Marcon, P., Kaczmarczyk, V., Benesl, T., and Schroeder, T. (2018). Implementing industry 4.0 in discrete manufacturing: Options and drawbacks. *IFAC-PapersOnLine*, 51(6), 473–478. doi:10.1016/j.ifacol.2018.07.106.

Banerjee, S. and Großmann, D. (2017). Aggregation of information models — an opc ua based approach to a holistic model of models. In *2017 4th International Conference on Industrial Engineering and Applications - ICIEA 2017*, 296–299. IEEE, Piscataway, NJ. doi: 10.1109/IEA.2017.7939225.

Breunig, D.A. and Schneider, M. (2019). Multi-protocol data aggregation and acquisition for distributed control systems. *Procedia CIRP*, 81, 310–315. doi: 10.1016/j.procir.2019.03.054.

Derhamy, H., Ronnholm, J., Delsing, J., Eliasson, J., and van Deventer, J. (2017). Protocol interoperability of opc ua in service oriented architectures. In *2017 IEEE 15th International Conference on Industrial Informatics (INDIN)*, 44–50. IEEE, Piscataway, NJ. doi: 10.1109/INDIN.2017.8104744.

Dotoli, M., Fay, A., Miśkowicz, M., and Seatzu, C. (2018). An overview of current technologies and emerging trends in factory automation. *International Journal of Production Research*, 1–21. doi: 10.1080/00207543.2018.1510558.

Enste, U. and Mahnke, W. (2011). Opc unified architecture. *at - Automatisierungstechnik*, 59(7), 397–404. doi: 10.1524/auto.2011.0934.

Faller, C. and Höftmann, M. (2018). Service-oriented communication model for cyber-physical-production-systems. *Procedia CIRP*, 67, 156–161. doi: 10.1016/j.procir.2017.12.192.

Grochowski, M., Simon, H., Bohlender, D., Kowalewski, S., Löcklin, A., Müller, T., Jazdi, N., Zeller, A., and Weyrich, M. (2020). Formale methoden für rekonfigurierbare cyber-physische systeme in der produktion. *at - Automatisierungstechnik*, 68(1), 3–14. doi: 10.1515/auto-2019-0115.

Großmann, D., Bregulla, M., Banerjee, S., Schulz, D., and Braun, R. (2014). Opc ua server aggregation — the foundation for an internet of portals. In *IEEE [International Conference on] Emerging Technologies and Factory Automation (ETFA), 2014*, 1–6. IEEE, Piscataway, NJ. doi:10.1109/ETFA.2014.7005354.

HEIDENHAIN (2017). Remotools sdk. URL `https://www.heidenhain.de/`.

Imtiaz, J. and Jasperneite, J. (2013). Common automation protocol architecture and real-time interface (capri). In W.A. Halang (ed.), *Kommunikation unter Echtzeitbedingungen*, Informatik aktuell, 79–88. Springer, Berlin. doi:10.1007/978-3-642-33707-9_9.

International Electrotechnical Commission (2015a). *IEC 62541-3 OPC unified architecture - Part 3: Address Space Model*, volume IEC 62541-3 of *International standard Norme internationale*. International Electrotechnical Commission, Geneva, 2.0 edition.

International Electrotechnical Commission (2015b). *IEC 62541-5 OPC unified architecture - Part 5: Information Model*. International standard. International Electrotechnical Commission, Geneva, 2.0 edition.

International Electrotechnical Commission (2016). *IEC TR 62541-1 OPC UA Part 1: Overview and Concepts*. International standard. International Electrotechnical Commission, Geneva, 2.0 edition.

Irani, Z., Hlupic, V., Baldwin, L.P., and Love, P.E. (2000). Re–engineering manufacturing processes through simulation modelling. *Logistics Information Management*, 13(1), 7–13. doi:10.1108/09576050010306341.

Mahnke, W., Leitner, S.H., and Damm, M. (2009). *OPC Unified Architecture*. Springer, Berlin, 1st ed. edition.

Prosys OPC Ltd (2019). Prosys opc ua sdk for java. URL `https://www.prosysopc.com`.

Schmied, S., Grosmann, D., and Denk, B. (2019). A systematic top-down information modelling approach for workshop-type manufacturing systems. In IEEE Conference on Emerging Technologies & Factory Automation (ed.), *Proceedings, 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 1305–1308. IEEE, Piscataway, NJ. doi:10.1109/ETFA.2019.8869377.

Unified Automation (2019). .net based opc ua server sdk. URL `https://www.unified-automation.com/`.

Wang, H., Ma, Y., and Yu, F. (2018). An opc ua multi-server aggregator with cache management. In CAC (ed.), *Proceedings, 2018 Chinese Automation Congress (CAC)*, 68–73. IEEE, [Piscataway, New Jersey]. doi: 10.1109/CAC.2018.8623689.