Interoperability in Manufacturing by Semantic Integration

Christoph S. Thomalla
Monitoring and Control Systems
Fraunhofer Institute of Optronics, System Technologies and Image Exploitation IOSB
76131 Karlsruhe, Germany
christoph.thomalla@iosb.fraunhofer.de

***Abstract*— Data for automated manufacturing systems becomes more and more available in digital form. But still it may not be reused throughout all levels of a factory and during the complete life cycle of such systems because of various different machine readable formats and no formalized semantics of the data. Misinterpretations of the data result in inconsistencies, redundancies and a lot of inefficient work and evitable cost. This paper proposes a solution, how information may be read, interpreted and related to other information by using semantic techniques. The presented solution is evolutionary, respects existing heterogeneous semantic concepts and allows a stepwise enrichment of semantic content. Then this refined data may be used for the existing production information systems and as a basis for the digital factory, a concept of a data collection and a comprehensive network of digital models, methods and tools that enable the simulation of a factory in advance without producing physical prototypes.**

Keywords- Interoperability, Semantic Integration, Manufacturing

I. INTRODUCTION

The engineering and development of an automated plant comprises in most cases a heterogeneous system of subsystems and devices of various suppliers and is done by different actors from various disciplines.

During the design phase, once the processes have been planned one may decide about the resources needed for them. Depending on the hierarchy of devices and automated systems there are different functional, spatial, electrical aspects to be considered. Different tools are used for different aspects. Various providers have tools with different functionality and data models. There are various planning tools for mechanical or electrical engineering, different tool providers and differently skilled engineers using them.

During the operation phase systems on the different levels of a factory interact. Beginning with the shop floor, we have the process with the different devices, further up the manufacturing execution system (MES), planning and control systems (APS) and on plant or enterprise level the enterprise resource planning system (ERP).

The life cycle consists of several different phases or stages during the existence of a plant. Beginning with design, installation and ramp up, it follows the operation phase with monitoring and maintenance. Then various improvements may take place, bigger ones may result in a reconfiguration. The next ones are decommissioning and de-installation, disposal or recycling. During the whole life cycle of an automation or manufacturing system we encounter a lot of digital data that is needed by different devices and subsystems.

Life cycle integration then means passing data and information from one stage to another. Horizontal integration deals with different makes or brands of devices that pass data to overlying systems and different information systems at the same factory level. Vertical integration considers different levels of the factory automation and the interactions of the respective information systems. Additionally in different application areas the wording may differ. System gaps are therefore inevitable. Interoperability means integration along all of these paths.

Technical integration or interoperability deals with interfaces and data formats. It allows connecting different systems with heterogeneous infrastructures by open interfaces, data formats and protocols. The use of technical standards like XML and Web services is helpful for that.

Semantic integration or interoperability provides a common understanding of the meaning of the interchanged data, such that other applications may unambiguously interpret the information.

Organizational integration or interoperability produces synchronous and compatible business processes and is not dealt with here.

Interoperability includes data reuse and fusion of data from different sources, not merely data, but enriched information or knowledge. For the fusion process there exist a couple of tools. Some tools like Fusion Rule [1] or [2] are even capable to fusion contradictory information. In [3] the problem of exchanging product semantics along with e.g. shape is addressed. In the context of knowledge fusion [4] proposes a fusion model using a global ontology.

II. NORMS AND STANDARDS

A possible solution to achieve semantic interoperability is by norms and standards. One attempt to enable reuse of existing data proposed HTML (HyperText Markup Language) for viewing the product information on a browser, basically telling content how to display. To store information and maintain the relationships by use of XML (eXtensible Markup Language) is slightly better. It provides a highly defined way to automate the transfer of manufacturing information, and so on, and the use of XML is transparent to most designers. However, just using XML (and STEP) does not already guarantee interoperability and data reuse as claimed in [5].

XML is a well-established reliable data format. It is machine readable, but humans may read it as well. This technology is wide-spread, future proof and widely used for data exchange and as data exchange format.

There exist various description languages in manufacturing with predefined structure, most of them XML-based, and in most cases standardized.

The recommendation NE 100 by NAMUR, an international user association of automation technology in process industries, consists of lists of characteristics for the creation of process control device specifications.

FDT (Field Device Tool) is a component of FDT/DTM for field device operation. FDT/DTM is a cross-vendor concept that enables the parameterization of devices of different manufacturers with just one program.

The Device Description Language (DDL) is a formal language for the description of the handling and parameterization of devices of process and manufacturing automation. A single software tool may operate and parameterize many different device types by interpreting the device description (DD).

The fieldbus organizations have integrated and extended their respective description languages. The result is the Electronic Device Description Language (EDDL) and an IEC standard (IEC 61804). DDL describes data (e.g. parameters), communication (addressing), structure of the operation, processes (e.g. calibration).

The Field Device Configuration Markup Language (FDCML) provides a meta-language for the description of any device and its capabilities.

CAEX (Computer Aided Engineering Exchange) is a neutral data format for the storage of hierarchical object information. Due to its extreme scalability the format allows to describe simple and very complex plants as well. In this sense CAEX is more or less a powerful meta-language.

The Standard for the Exchange of Product Model Data (STEP) is an international standard for exchanging data between different CAD/CAM and Product Data Management (PDM) systems. STEP supports engineering, manufacturing, electrical/electronics, architecture and construction life cycle information. The unique feature of STEP is that it integrates product data. To use STEP one must have software capable of translating files from the proprietary formats to the neutral format created by STEP and back. STEP was initially not XML-based.

ANSI/ISA-95 is an international standard for developing an automated interface between enterprise and control systems.

Other exchange formats are specifications from the MES/ERP-area like BatchML or Business to Manufacturing Markup Language B2MML (XML-Specification of ISA-95).

The Web Services Description Language (WSDL) is an XML-based language that provides a model for describing web services. The WSDL defines services as collections of network endpoints, or ports.

WSDL is often used in combination with SOAP and an XML Schema to provide web services over the Internet. A client program connecting to a web service can read the WSDL to determine what operations are available on the server. Any special data types used are embedded in the WSDL file in the form of XML Schema.

AutomationML [6] has the potential to become a dominating standard because of the powerful consortium that is pushing it. But from the reasons above (other existing data, other tools), it is just an additional one.

These few mentioned examples show that there is no lack of standards; on the contrary, there are too many different ones for specific application areas, different tasks or phases of the lifecycle and all of them may have certain advantages.

III. Semantic Interoperability and Conflicts

Data for heterogeneous automation systems occurs in many different ways. If not paper based, this could be in some digital form or even in some XML-based description language. To put it all together becomes increasingly complex as there exist various machine readable formats. To transform information from one format to another, it is necessary to know each format, i.e. the formal description and the semantics of the data in both formats. The exchange is often accomplished through individually programmed electronic converters.

Sources providing knowledge in machine-readable form like non-structured text have to be annotated manually or semi-automatically, which means to create meaningful tags in an XML-document to interpret uniformly used terms. Technical and syntactical interoperability are a prerequisite for semantic interoperability, which we assume from now when using XML.

Even if data is available in digital form, it may occur in many different data models. A data model represents the abstract mapping of objects, their properties attributes and relations as well as their possible interactions, where the amount and the usage are mostly restricted to a certain domain or to specific tasks. Sometimes tools or tool sets enable a comfortable use of such models by common access to the same data records or creating appropriate interfaces, but in general there exist significant gaps between domains or different tasks. Such interfaces between heterogeneous systems just provide for technical interoperability.

Additionally the semantics of data – the meaning – is not described formally; therefore it is only interpretable with a high effort by experiential knowledge of the respective users.

Looking at e.g. a field device we find that there exists a lot of different planning data like the related control functions, supplier information, data from commissioning, from operation, maintenance, i.e. along the complete plant life cycle. We may expect a huge potential for savings and support for information users from explicit and formalized relations between the information of several information models. The added value of these connections described by an unambiguous semantics lies in the reuse, the detection of inconsistencies, the consistent change propagation and the creation of a formal semantic platform.

When exchanging data between systems, some similarities may occur. There are name similarity, similarity in structure, similar property name and range and instance similarity, where we have a common set of instances. These similarities give a hint that the things might be the same. But unfortunately this is not always true. If we have a symbol or term, the meaning or the concept of the respective elements, the objects, may differ. One uses mappings to resolve these semantic conflicts. They occur because of the heterogeneity of systems.

To make interoperability efficient, an intermediate or pivot language is used, to get a complexity reduction, especially if there are many description languages and tools. A huge amount of converters might then be necessary to enable the data exchange between all of the involved systems. From a computational point of view an intermediate language or format reduces the number of interfaces between different description languages from $O(n^2)$ to $O(n)$.

Without an intermediate format there are in the worst case interfaces needed between each pair of the existing standards and tools. Having n different formats, this means (n-1) interfaces to the other ones, (n-2) for the next remaining and so on. This sums up to n*(n-1) if one considers both directions as different, which is $O(n^2)$ instead of 2*n which is linear, when using an intermediate format.

The advantage with an intermediate language, which may be open and well documented is, we just have to know the own and the intermediate language. There is no need to have two experts of two different functional domains [7] that understand each other to align semantic mismatches. We may use the property of mappings to resolve semantic conflicts. With schema mappings we may preserve the structure or hierarchy and the semantics. These conflict resolving techniques aim for a semantics preserving information exchange.

As we already have seen, the heterogeneity of systems causes semantic conflicts during data exchange. A possible way to avoid conflicts is by a norm or standard, if we just stick to one of them. As this is not the case, the avoidance of conflicts is not possible due to the ambiguity of meaning; therefore we concentrate on resolving the conflicts.

Semantic interoperability is not easy, because there are a lot of reasons why mismatches may occur: We may have different contexts or representation languages, the same name for different concepts or different names for the same concept. There may be a different use of subclasses and attributes, different levels of granularity and different, but overlapping domains with own conventions.

An analysis of the types of conflicts, that may occur, shows that there are conflicts at data and at schema level.

Typical conflicts at data level are measure or unit of measurement conflicts, like meter per second and kilometer per hour. Other ones are data value conflicts, e.g. by using the same name for different things (homonyms) or by using different names for the same thing (synonyms). Then there are data representation conflicts, e.g. for date and time, which differ in a lot of countries, and, of course language conflicts, if there is no exact translation. Even worse, for example city names like München and Munich have to be translated, whereas for unknown cities there is no translation. Proper names e.g. do not change in general in different contexts.

Similar conflicts may occur at schema level. Again there are data value conflicts by using homonyms or synonyms for schema elements, generalization conflicts, if the schema name is a subset of the name on the other side, structural conflicts by alternatively using attributes and sub elements in XML, hierarchy conflicts, data type conflicts, cardinality conflicts e.g. of enumerations.

There may be a different structure that may appear in different order as well, as in device identity, which is a structure versus single elements identity and name, or address instead of street, zip code and city.

We have to guarantee the recognition of semantic conflicts and exchange information with conflict resolving techniques. The aim is a data exchange, where misinterpretations and misunderstandings of concepts, data structures, writing conventions etc. are eliminated and we have semantic interoperability. Whoever uses part of a description of an automated system (viewed as a product) must be sure to use a common understanding of the exchanged information by means of a “formalized semantics”, i.e. by making the meaning explicit.

How do we achieve semantic interoperability in practice? When defining an intermediate language, that claims to be interoperable, we have to specify some quality requirements concerning information width and depth, information quality and completeness, to be able to evaluate it. However there is a trade-off between completeness of concepts and size of the catalogue and the interface which affects its acceptance.

IV. Solution

Technologies from computer science offer a promising solution approach that might be applied to tasks in automation as well. These methods allow to formalize and unambiguously semantically interpreting the data, information and the planning knowledge. In this way it is possible to reuse them in different use cases, once the relations between the intersecting information are defined independently of the representation, i.e. between the data models.

The required integration should be possible with little effort, ideally with the help of just one or a few interfaces. Because of the heterogeneity and resulting complexity an intermediate format is needed.

Therefore the integration of the data models must be extensible and reusable. Only the knowledge of the own system and the open documentation of the goal format should be required to create the mapping rules. Local mappings of data models are preferred instead of a tedious development of cross-domain world models and incremental approaches instead of a complete system, such that the interface may be useful from the beginning.

The integration of data in a semantic knowledge network respectively in an ontology is called „semantic lifting“. Knowledge has to be modeled once by hand by experts. The semantic lifting is done by mapping the structuring elements of the respective data format onto the structuring elements of ontologies. Having e.g. an XML-based device description (GSDML, FDCML), the XML-elements may be mapped as ontology classes.

The different models then exist in some formal model. The expert in this area defines the mapping rules to an XML schema, verifies it and validates the requirements. The definition of conventions and rules to transform the model into XML schema is an essential step. The set of rules depends on the model and to which degree it is formalized. Design patterns are specified for typical constructs like UML models down to Excel sheets or data base tables.

To fusion ontologies, a simple exact coincidence of top level names and attributes of objects in the ontologies will not work because of the problems mentioned. It is done implicitly by the mapping rules that define relations among some variables of two ontologies. Automatic matching of ontologies involves the creation of mapping rules or alignments, i.e. finding sets of correspondences between the ontologies. Considering correspondence patterns, data fusion needs both correspondences between concepts and correspondences between properties. Again these approaches might be mistaken as explained above. The task of finding the correct mapping between the interesting parts of the ontologies is left to an expert in that topic. As the 'interesting part' refers to a non-predictable amount of common information in the ontologies, an automatic solution seems unlikely.

Instead of using a global ontology, a lot of ontologies for special areas and aspects are created. For a special application, i.e. the interface between two systems or the respective description languages only the involved ontologies will be needed. This simplifies the handling, extension and maintenance of them.

The semantic integration of the different knowledge builds the basis of various automated approaches, like the automatic generation of converters between different software tools, data bases and further knowledge sources. When describing the conversions by mapping rules this enables a uniform and central maintenance of automatically created converters.

The aim of this solution approach is to represent the implicit or partially formalized expert knowledge including the mapping rules explicitly, uniformly and formalized, which is needed for the creation of converters for data exchange and a seamless integration. The original formats may remain heterogeneous. The advantage of this approach is that semantic information, once formalized, does not get lost anymore, but may be reused for the integration of further descriptions and data models. At the same time this solution supports the reproducibility, transparency and traceability of the results and finally acceptance.

There are two projects we are currently working on. The first one is to construct a set of concepts with semantically enriched information and formalized meaning to create a norm for the exchange of data between the shop floor and MES systems.

The second one is more global and considers most of the XML-based description languages used in production. The aim is to find a common intermediate language and build a formalized semantics for the different application areas, phases in the life cycle and levels in the enterprise. AutomationML could be a candidate as intermediate language. It combines three standards, where CAEX defines the semantics and topology, Collada is for the geometry and kinematics and PLCopen XML describes the logic and behavior of the plant and its parts. A model in AutomationML then consists of XML schemas of the underlying standards, standard libraries to assist the user and textual definitions, constraints, rules and conditions. Thus we get a heavyweight ontology. We used Protégé for establishing the ontology and developed some tools for mapping support based on OWL.

V. Summary

Semantic integration is a crucial step to interoperability in manufacturing. It has a huge potential for savings and support for information users from explicit and formalized relations between the information of several information models. These connections described by an unambiguous semantics have a tremendous added value. They allow the reuse of knowledge, the detection of inconsistencies, the consistent change propagation and the creation of a formal semantic platform.

Such a semantic platform could be the basis for the Digital Factory, a technology to create a virtual factory, in which a product can be built and the logic may be validated prior to commissioning any of the equipment used to build the real factory or producing physical prototypes. As the engineering process is failure-prone and design errors are costly, it reduces time and cost till operation substantially.

Given such a formal semantic platform, if all devices in an automated production system carry their description of capabilities with them, they may be integrated almost automatically in the existing system. This type of application could be a plug and work functionality, similar to the known behavior of USB-devices in computers.

References

[1] A. Hunter and R. Summerton, "Fusion rules for context-dependent aggregation of structured news reports," Journal of Applied Non-classical Logic, vol. 14(3), 2004, pp. 329-366.

[2] F. Naumann , A. Bilke, J. Bleiholder and M. Weis, "Data Fusion in Three Steps: Resolving Inconsistencies at Schema-, Tuple-, and Value-level," Bulletin of the Technical Committee on Data Engineering, 2006, pp. 21-31.

[3] R. K. Gupta and B. Gurumoorthy, "A Feature-Based Framework for Semantic Interoperability of Product Models," In: Strojniski vestnik - Journal of Mechanical Engineering, vol. 54 (6), 2008, pp. 446-457.

[4] N. Xie, C. Cao and H. Y. Guo, "A Knowledge Fusion Model for Web Information," Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence, 2005, pp. 67-72.

[5] T. Kjellberg, A. von Euler-Chelpin, M. Hedlind, M. Lundgren, G. Sivard and D. Chen, "The machine tool model—A core part of the digital factory," CIRP Annals - Manufacturing Technology, vol. 58(1), 2009, pp. 425-428.

[6] R. Drath, Ed., "Datenaustausch in der Anlagenplanung mit AutomationML," Springer, Heidelberg, 2010.

[7] N. Chungoora and R.I.M. Young, "Semantic Interoperability Requirements for Manufacturing Knowledge Sharing", Enterprise Interoperability III - New Challenges and Industrial Approaches, K. Mertins, R. Ruggaber, K. Popplewell and X. Xu, Eds., Springer-Verlag London Limited, UK, 2008, pp. 411-422.

[8] H. Adametz and A. Billig, "Semantische Konflikte," White Paper: Semantic Interoperability, Fraunhofer Institute ISST, vol. 2, 2010.

[9] G. Chryssolouris, D. Mavrikios, N. Papakostas, D. Mourtzis, G. Michalos and K. Georgoulias, "Digital manufacturing: history, perspectives, and outlook," Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, vol. 223(5), 2009, pp. 451-462.