Proceedings of the TMCE 2004, April 12-16, 2004, Lausanne, Switzerland, Edited by Horváth and Xirouchakis
© 2004 Millpress, Rotterdam, ISBN

CONSIDERATIONS ON IT SYSTEMS INTEOPERABILITY IN PRODUCT DEVELOPMENT

Mario Štorga
Chair of Design
Faculty of Mechanical Engineering and Naval Architecture
University of Zagreb
Croatia
mario.storga@fsb.hr

Dorian Marjanović
Nenad Bojčetić
Chair of Design
Faculty of Mechanical Engineering and Naval Architecture
University of Zagreb
Croatia
{dorian; nenad.bojcetic}@fsb.hr

ABSTRACT

This paper firstly presents the background of the research project aimed to: (i) building the formal models for representation of the product development context (entire body of data, information and engineering knowledge that evolves throughout the product development process); (ii) developing tools and methods for support of the product development context interoperability. After research introduction, review and systematization of the IT systems interoperability evolution are provided. Accordingly to the different computer and informational scientists' and experts' research results, different interoperability approaches are presented: custom coding, single standard, data warehouses, single systems, integration, and semantic-based information interoperability. In order to achieve the research goal, product development ontology harmonization is recognised as a main prerequisite for successful implementation of the newest computer technology and tools. The short overview on ontology definitions is provided. Different methodologies for building ontologies are summarized around tree major stages of the ontology life cycle i.e., building-manipulating-maintaining. Finally, directions of the next research steps are proposed.

KEYWORDS

Information systems interoperability, product development context, product development ontology

1. INTRODUCTION

Information system technology has long promised to integrate business processes by providing communication channels that seamlessly enable members of a business teams to exchange data and information across physical and temporal boundaries. The reality, however, has been very different. Data is stored and duplicated in so many different places that one often does not know “good” data from “bad”, even if one can find the data that one is interested in. Also, when communicating with each other, participants in distributed processes may not speak the same “language” or understand particular solutions adopted by others. And these are not the biggest problems that integration efforts must overcome. A bigger obstacle is the fact that data is tightly bound to the applications that create/use the data, which means that moving data between applications requires a conversion or translation process.

Despite the hype around the newest generation of Enterprise Application Integration (EAI) and process automation products, many IT managers remain cautious about adopting them. There is also growing press coverage about a Standish Group report that 88 percent of integration projects fail (Pollock, J.T., 2001). Sure, there are many project related causes for these failures (implementation process, poorly managed risks and requirements, etc.), but something fundamental is missing from most available technology solutions that can improve the odds for integration success - interoperability.

The words integration and interoperability seem to be used interchangeably, but they are different concepts. Interoperability-based approaches focus on the exchange of meaningful, context-driven data between autonomous systems. Integration approaches, in contrast, typically attempt to build a monolithic view of the enterprise. They integrate processes and applications at the event and message levels so multiple systems become one logical unit. The two approaches can be complementary, but an interoperability solution would usually focus on how to exchange the minimal amount of information (not just data) to make two or more systems interoperable. Interoperability is also about maintaining the autonomy of the participating members of an exchange community, allowing them to share and use information intelligently while maintaining their own vocabularies, computing environments, and general perspective on the data (Pollock, J.T., 2001).

To achieve interoperability it is necessary to realise the both application integration and information integration. Application integration is the technology solution, where most of the today's product development effort has focused. Information integration is the linguistic, social, and philosophical solution, where technology is only beginning to catch up with academia. Faced with these problems today, a retrospective viewpoint leads to the observation that if interoperability issues had been successfully addressed earlier on, industry might have avoided some of these wasted costs entirely, instead of working to reduce waste after the fact. It is this observation, along with a projection of how product development in industry is changing, that motivates the need to address these issues in next-generation product development software systems (Szykman, S., 2001.).

While existing efforts focus on enabling interoperability among tools that address a specific product development activity (such as geometric CAD), the more significant demand in next-generation tools will be definition of human-to-human, human-to-application and application-to-application communication that should allow data, information and knowledge used or generated in various product development activities to feed forward and backward into others by way of direct electronic interchange.

The overall goal of research that we are going to present in this paper, is to satisfy a need for models and tools to effectively support the formal representation, capture and exchange of product development related data, information and knowledge that are unavailable in traditional CAE tools (mostly non-geometric knowledge as information about development process, physical, behavioural or functional decompositions of product, specifications, and various kinds of relationships among these entities…). Under the context of the ongoing research project at Chair of Design (Bojčetić, N., 2002.), the solutions that will be proposed during this research should serve as a communication "backbone" for building future integrated engineering environments.

This paper is organized as follows: Section 2 presents a research project in details; Section 3 provides historically look at the evolution of the information system interoperability solutions; Section 4 discussed ontology building approach to achieve interoperability; and Section 5 provides a conclusion and directions for the future research.

2. OBJECTIVES OF THE RESEARCH PROJECT

2.1. Purpose of research

According to the definition given by Blessing [Blessing, L., 2002.], engineering design research should integrate two different aims because they are closely linked and therefore have to be considered together. The engineering design research involves:

- the formulation and validation of models about phenomenon of design, and;
- the development and validation of knowledge, methods and tools – founded on these models.

According to previous statement, the first aim of research presented in this paper is defined as: ***building of generic models as a framework for formal representation of the "product development context"*** *(entire body of data, information and engineering knowledge that evolves throughout the product development process, Figure 1.).*

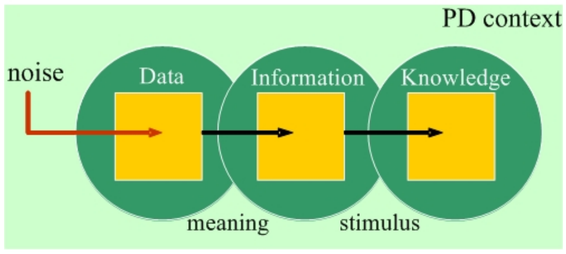

Figure 1 Progression from data to knowledge (classic information hierarchy)

The generic models of *product development context* are intended to be an abstraction of *concepts* (objects, relationships, attributes, etc.) that are common across many product development activities, and *rules* that guide what kind of assertation about domain of discourse may be made based on concepts, and what kind of *conclusions* may be drawn from these assertations. The proposed formal framework for representation of *product development context* should be:

- Open, extensible and flexible;
- Generic and not depend on any one product development process;
- Capable of capturing the portion of the product development context that is most commonly shared in product development activities;
- Conformed to way of thinking during the engineering design synthesis;
- Not tied to a single vendor software solution.

The second aim of this research, based on previously described framework, is to ***develop the tools and methods for support of the product development context interoperability.*** Such developed tools and methods and the newest computer technology should support both, enterprise application- and information-integration. The crucial issues that must be addressed in this part of research can be summarized as follows: (Štorga, M., 2003.)

- Interoperability in sense of ubiquitous information exchange between different applications within the same organization. This is domain of research, which will propose mechanism for connecting and integration disparate applications. These applications are ones for the creation of information (CAE systems) and management tools for controlling the flow of information (ERP/PDM systems, knowledge management systems, human resource). In this scope, the emphasis is on interoperability between these functionally differently applications.
- Interoperability between organizations, applications, and content between different organizations in the same virtual enterprise. Mergers and acquisitions have been quite common and frequent especially in the past 10 years. In most cases the acquired or merged organizations have different CAE systems, engineering databases, processes, and applications. The interoperability should involve consolidating the various applications and content from different organizations. The main difference with respect to the previous issue is that this category of interoperability should involve different processes between many collaborative partners over extended enterprise networks.

2.2. Research methodology

The topic under investigation in this research belongs to the large area of study that is generally called Product Development. By nature, research in any aspect of product development necessarily involves a multidisciplinary approach, with a number of different disciplines being brought together. The research has been divided into four phases briefly discussed below:

Phase 1- Literature review and the initial establishment of research foundation

In order to clarify the aims that the research is expected to fulfil, the first phase of the research is characterized by analyze of the product development features based on review of nowadays research results. The purpose of multidisciplinary sources and discussions with colleagues is to extend cognitions in the research area and understanding of the real nature of product development process from the information/knowledge flow point of view. This phase should help the researchers in clearly identification of research requirements and research borders. In addition, the literature review of different interoperability approaches and visions should be done, with a purpose to help researchers in defining the research strategy and research evaluation criteria.

Phase 2 – Developing the reference generic models of product development context

The framework for representing *product development context* that will be comprehended in this research, in the first place will be based on experiences from existing research approaches in the research area of design languages and design modelling. The future study in this area should serve in sense of determination of the main concepts, relations and rules found in product domain theoretical foundations. For the reason of extending the generic models to cover design information/knowledge within product development, research on product development activities should be done as a next step of study in phase 2. This approach will lead to the finding and describing the different concepts, relations and rules from the domains of the product development cycle (from needs, through discovering of product principle, product design, product preparation, to the realization) in a continuous way.

Phase 3 – Implementation of the reference model by using existing computer technology

Leveraging existing computer technologies in the implementation of tools and methods for support of the product context interoperability requires firstly the identification of the desirable features of the existing technology. From the today's viewpoint, two technologies in particular can be central to support enterprise application and information integration: Web Services and Semantic Web. The main steps in this part of the research, based on existing technologies, will be identification and description of use case scenarios and proposing the implementation architecture. This phase of research is aimed to the verification of research concept.

Phase 4 – Success evaluation

There is a need for different types of evaluation to assess more aspects of the proposed product development support than only functionality. For that reason it is necessary to use the proposed product development support in the situations for which it is intended and answers questions about usefulness, implications and side effects of real implementations. The real evaluation should be seriously carried out only as a real industrial application with respect to the formulated criteria.

2.3. Research status

This paper reports main comprehensions and conclusions related to interoperability issue, drawn after the first phase of research was done. We used it as a base for defining the research strategy that is presented in a discussion section at the end of this paper. In the next section of this article, we continue with the review of the research efforts in area of IT systems interoperability that were done during the last 50 years by computer scientist and information experts.

3. SYSTEMATIZATION OF IT SYSTEMS INTEROPERABILITY EVOLUTION

(from Pollock, J.T., 2001., 2001.,2002.)

When we look historically at how IT systems have evolved, they generally follow two high-level categories: software applications that solve problems and software applications that link to other software applications. Statistics show that typically, the single largest portion of IT investment is in making different computer applications talk with each other. This statistic makes sense because the more you can share information and link systems; the more work can become automated "often offering both faster and higher-quality productivity.

Information system interoperability is not a new goal. Simply stated, interoperability is the ability to seamlessly share information among autonomous computer systems without tightly coupling them. Technology has followed behind and solved portions of these problems with a variety of different approaches.

Historically, it goes something like this:

- 1954 – 1975: the age of custom coding
- 1960 – ONGOING: myth of the single standard
- 1985 – 1995: the rise of the data warehouse
- 1990 – 1999: the illusion of the single system
- 1993 – ONGOING: the promise of integration
- 1995 – ONGOING: semantics-based information interoperability

3.1. Custom Coding

Computers were first installed in a business by 1954. General Electric's Appliance Division installed a UNIVAC system to do some number crunching. The need for interoperability software emerged the minute the very next computer was installed at GE. Since there was no software company providing application interoperability services the de-facto integration approach was a custom-coded solution. The custom-coding approach worked well because companies could tailor solutions to fit their specific needs "integrating two or more systems with custom software bridges linking their systems together". Another benefit of this approach was that they were not limited, or locked into, a particular vendor's way of doing things. Avoiding vendor lock-in meant that they were largely self-sufficient and did not require high-priced or narrowly focused consultants.

However, the downsides of this approach were soon apparent especially after businesses started to scale the number of integrated systems over several years. Once these systems were implemented, the ongoing maintenance challenges would continue to generate costs the organization. Changes required because of changing processes or business rules were difficult implement, and the custom bridges frequently broke because they simply could not adapt to changes. Today, custom-coded application bridges are still used, but only in a smaller number of legacy situations and specialized cases where technology has not caught up yet.

3.2. Single Standard

As early as 1960, groups of companies began to collaborate on an ad-hoc basis to formalize information exchange formats. By 1978, as a response to the high costs of data entry (time and money) and the need to interchange information between companies, both the ANSI X12 and EDIFACT standards began to proliferate. More recently we have seen many of these existing standards, as well as brand new ones, shift into the more modern technology of XML.

In addition to the typical, and highly publicized, interchange standards, much work has been done in the realm of standard composite data exchange technologies. STEP (ISO 10303) is the principal product data exchange standard in the world. STEP pioneered several significant and revolutionary innovations in the use and exchange of data, notably the interpretation and use of generic data structures in different application domains.

The benefits of a standards based approach to solving interoperability problems include the fact that each standard has input from a wide community and typically has cross-organization collaboration and support. But standards-based exchanges have not grown to dominate inter application communication for several key reasons. For one, the more effective a standard is, the more compromise is required. Historically, the standards based formats have either proliferated almost out of control (XML standards) or have been very narrow in focus (X.25, EDIFACT), thereby creating a problem for IT managers when attempting to decide on one or only a few. Standards certainly have their place in the IT toolkit, but they can never be the primary arbiters of information among all systems that need a flexible and robust method to communicate with each other.

3.3. Data Warehousing

For a while around 1988, the data warehouse was viewed as a candidate framework for integrating computer systems. The idea was that a centralized database could be used as the primary owner of enterprise data "and that client applications would use the warehouse as their main source". The data warehouse was seen as a way to centralize the critical information that was distributed as consequence of the client-server paradigm shift in the IT organization. As an approach for enabling different applications to share information, the data warehouse did a good job because people were asked to model the information resources of an entire enterprise for the first time. A unified view of an entire organization was the primary goal of many of the early data warehousing efforts that were engaged in making systems talk with each other.

Once IT departments attempted using warehouses as central data stores, the poor results and spiralling projects turned managers away from the idea. First off, the amount of time that it took for analysts to create a unified data model was grossly underestimated. Frequently, by the end of an analysis effort, the source applications, line of business, or project scope had changed anyway - requiring analysts to go back to square one. In addition, the problems with persisting the data in a central repository created a host of other issues, including uncovering hidden problems in source systems (bugs, rules, etc.) and data cleanliness problems (bad data in extract and load routines).

Another major lesson learned from using warehousing to achieve interoperability was that homogenized data is frequently not as valuable because context and relationships are typically lost. Combined with problems caused by the need for speed and real-time access these issues ensured that the data warehouse would increasingly be seen as a valid place to archive historical data for mining, but not as a central repository of timely business information.

3.4. Single System

Starting in the early 1990's vendors began to consolidate and extend MRP (manufacturing resource planning) applications to include a wider range of the production process. By 1995, the term Enterprise Resource Planning (ERP) was introduced, and a whole host of software companies thrived on the billions of dollars spent on these systems.

As their businesses began to expand, each ERP software company wooed clients to let them become their single dominant provider of business information systems. They rapidly expanded to include business functions that were not central to their original manufacturing focus and lobbied that the problems of integrating systems and sharing information were irrelevant if a business selected them as the sole provider of their systems. The advantages of ERP systems only sometimes outweighed the costs of them, but they included: increased consistency within the enterprise, improved functionality of line-of business applications and the adoption of best-practice processes that the software enforced.

A few of the key disadvantages of an ERP platform as the end-all of interoperability issues are: vendor lock-in, overwhelming costs, virtually no interoperability with non-ERP systems, single systems can not account for the diversity of business needs. ERP solutions companies will need to adjust their models in order to play nice with other systems that will invariably exist within the modern enterprise.

3.5. Integration

Even as ERP vendors were fighting to convince IT managers that they had a complete solution, another market was forming to specifically address the issues of incompatibility between applications within the enterprise. Message Oriented Middleware (MOM) arose from the telecommunications industry with the inception of the Message Oriented Middleware Association (MOMA) in 1993. Early MOM technologies eventually adopted basic Extract Transform and Load (ETL) capabilities and spawned in to a new service offering that, in 1997, was renamed Enterprise Application Integration (EAI) software. EAI companies began to offer integration platforms that offered adapters, transport, and transformation capabilities to companies who needed to extend the reach of their existing systems. More recently, EAI companies are starting to layer even more management tools on top of their transport systems in an attempt to re-brand themselves as B2B vendors.

Modern EAI approaches are successful because they have adapted to the needs of exchanging data within its appropriate context (process integration) and created a centralized management paradigm for controlling information flow. But EAI-type approaches are struggling in their own right because IT managers are wary of several key disadvantages that are impediments to widespread adoption. From the technology standpoint a typical EAI solution creates a very tightly coupled integration environment that severely restricts the flexibility, agility, and adaptability of a given integration framework.

The continued focus of EAI vendors on process-based solutions will contribute to the further rise and then decline of the utility of the EAI solution - the management overhead. A new method has gained strength in the past few years and is now poised to improve the interoperability options available to IT managers.

3.6. Semantic-based information interoperability

Beginning in the late 1990's companies, primarily specialized niche consultancies, started to develop in-house expertise at building a new breed of software. This next generation interoperability software focuses on making information among heterogeneous systems compatible via the use of semantic mapping and context management. Semantics-based enterprise information interoperability solutions are a category of tools focused on solving the problems of disjointed vocabularies, data definitions, terminology, and world-views of enterprise IT systems, and seams to be a missing link in integration. Semantic-based information interoperability is concerned with the transference of meaning and intent – the necessary ingredients for truly rich collaboration. It focuses on the logical information infrastructure of an enterprise rather than the physical connectivity and routing infrastructure (Figure 2).

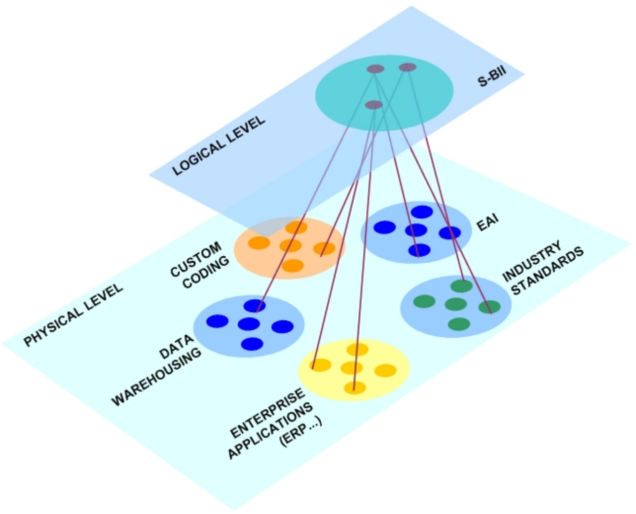

Figure 2 Different levels of focusing for semantic-based information interoperability and traditional approaches

The roots of semantics-based interoperability technologies derive from parts of the work done in the Artificial Intelligence (AI) community in the early 1970s. At that time, computer scientists were grappling with issues of knowledge representation in digital systems, attempting to solve the problem of how to digitally represent human knowledge. Today's typical semantic interoperability solutions have three key characteristics reminiscent of old school AI technologies:

- Semantic mediation - Information interoperability solutions use ontology (a model that makes concepts explicit) as a mediation layer to abstract particular data terms, vocabularies, and information into a shareable, distributable model.
- Semantic mapping - Ontology is only as good as the quality of the map that associates the enterprise data to it. Mapping to ontology preserves the native semantics of the data and eliminates the need for custom code. In information interoperability solutions, mapping accounts for much more than simple many-to-many data formatting rules or data syntax arbitrations – it is how the semantics are captured, aligned, and structured in relation to the data itself.
- Context sensitivity - The meaning of any data is always bound to a particular perspective or context. Thus, any information interoperability solution set must accommodate the fact that the same data can mean many different things from different viewpoints. Typically, the business rules, context definitions, and environmental metadata are captured and stored during the mapping process, making them reusable in any run-time process.

Technology leaders have reached the same conclusion: ontology-based semantic mediation of disparate information resources is the most viable alternative to today's inefficient, inflexible solutions. According to the expertises (Pollock, J.T., 2001.), full-featured semantic interoperability solution should include:

- Transport service - a mechanism or protocol to move messages from one place to another.
- Message container - a codified wrapper and structure for the data and metadata during the exchange.
- Integration interface (transport or message) - A way to get your data in the container and send it over your protocol.
- Process controller - a conductor that defines and orchestrates multistep, batch, publish/subscribe, and request/reply transportation of messages within the scope of analyst defined business processes.
- Data encapsulation - the encapsulation of the raw elements of information exchange.
- Ontological information representation - A specification of a conceptual model that your data sources are represented by. This representation may be accomplished by a direct mapping or a third conceptual specification of a higher order than source and target specification.
- Context and metadata - roughly speaking, the data about the data from the application's perspective is the metadata, while the data about the data from a business perspective is the context.
- Information transformation - a capability to manipulate entire sets of data, relationships and all, based on information contained in metadata and user-supplied context information. Typically, this requires a conceptual model using ontological mapping techniques.
- Data translation - a set of algorithms and user-modifiable scripts that enable your integration tool to translate based on pre-defined rules (e.g., string manipulation, mathematical functions, conversion routines, and filtering.

4. DISCUSSION

Based on the previous review of IT systems interoperability evolution, we can summarize some observations:

- An initial step to solving interoperability problems is to capture a description of all types of information/knowledge managed by the various existing modelling methods and their inter-relationships. The specification of domain ontology provides formal and informal definitions of the basic semantic categories and the logical connections between those categories.
- To avoid an interoperability problem of having numerous, independent enterprise-modelling tools we should provide a neutral computational medium (ontology) in which the information/knowledge contained in various models and their inter-relationships can be represented and maintained.
- The problem of different contexts can be addressed by defining precisely the terms used in the various contexts, that is, by capturing the ontology of the domains relevant to the contexts.

Congruently to the research goal and after the previously described observations, we have resolved that the computer science and software companies offer us technology and tools to achieve our goal, but the main step is still on us – product development research community. If we want to use these solutions and implementing them for support of *product development context* interoperability, we should make the efforts aimed to the harmonization of the *product development ontology*.

Based on this reasoning, the research strategy that had been chosen for fulfilling the first aim of presented research (Section 2.1) is defined as: building product development ontology (PDO) for dynamic and distributed product development environments.

4.1. What is ontology?

The word ontology was taken from philosophy, where it means a systematic explanation of being, or the kinds of existence. In the last decade, the word ontology became a relevant word for the knowledge engineering community that has borrowed it from philosophy and has given its meaning a twist. One of the first definitions in new sense was given by Neches and colleagues (Neches, R, 1991.) who defined ontology as follows: "ontology defines the basic terms and relations comprising the vocabulary of a topic area as well as the rules for combining terms and relations to define extensions to the vocabulary". This descriptive definition tells what to do in order to build ontology, and gives us some vague guidelines: the definition identifies basic terms and relations between terms, identifies rules to combine terms, and provides the definitions of such terms and relations.

Studer and colleagues (Studler, R., 1998,) explained ontology by as follows: "Ontologies are defined as a formal specification of a shared conceptualization. Conceptualization refers to an abstract model of some phenomenon in the world by having identified the relevant concepts of that phenomenon. Explicit means that the type of concepts used, and the constraints on their use are explicitly defined. Formal refers to the fact that the ontology should be machine readable". In one of the newest definition, Guarino and colleagues (Guarino, N., 1995.) proposed to consider ontology as "a logical theory that gives an explicit, partial account of a conceptualization", where conceptualization is basically an idea of the world that a person or a group of people can have.

There also exists another group of definitions based on the process followed to build the ontology. These definitions also include some highlights about the relationship between ontologies and knowledge bases. Where a knowledge representation system specifies how to represent concepts, the ontology specifies what concepts to represent and how they are interrelated (Heflin, J., 2003.). Most researchers agree that ontology must include a vocabulary and corresponding definitions, but it is difficult to achieve consensus on a more detailed characterization. Typically, the vocabulary includes terms for classes and relations, while the definitions of these terms may be informal text, or may be specified using a formal language like predicate logic. The advantage of formal definitions is that they allow a machine to perform much deeper reasoning; the disadvantage is that these definitions are much more difficult to construct.

Today, the ontology community distinguishes lightweight and heavyweight ontologies. On the one hand, lightweight ontologies include concepts, concept taxonomies, relationships between concepts and properties that describe concepts. On the other hand, heavyweight ontologies add axioms and constraints to lightweight ontologies.

Since ontologies are widely used for different purposes (natural language processing, knowledge management, e-commerce, intelligent integration information, the semantic web, etc.) in different communities (i.e., knowledge engineering, databases and software engineering), Uschold and Jasper (Uschold, M., 1999.) provided a new definition of the word ontology to popularize it in other disciplines. "Ontology may take a variety of forms, but it will necessarily include a vocabulary of terms and some specification of their meaning. This includes definitions and an indication of how concepts are inter-related which collectively impose a structure on the domain and constrain the possible interpretations of terms." As a main conclusion to this section, we can say that ontologies aim to capture consensual data, information and knowledge in a generic and formal way, and that they may be reused and shared across applications (software) and by groups of people.

4.2. Building ontologies

The literature reports about numerous methodologies, tools, and languages for building ontologies without real correspondence between them, and each one following different ontology definition and research approaches. There is also lot of proposals for the other ontology related tasks, such as ontology reengineering, ontology learning, ontology evaluation, ontology evolution, etc (Figure 3, Corcho, O., 2003.).

Methodologies	Tools	Languages
Uschold & King	OILed	OWL
Gruninger & fox	OntoEdit	DAML+OIL
Bernaras at all	Semantic Broker	RDF(s)
SENSUS	Ontolingua Server	XML
On-To-Knowledge	Ontosaurus	Ontolingua
METHONTOLGY	WebOnto	LOOM
		FLogic

Figure 3 Ontology methodologies, tools and languages

Different methodologies for building ontologies, and supporting tools, can be summarized around tree major stages of the ontology life cycle i.e., *building-manipulating-maintaining* (Kayed, A., 2002).

(i) Building. There are many attempts to define a methodology for ontology construction. In building stage four steps are needed:

- Specification – a plan of main ontology tasks should be defined. Many questions should be answered, for example: What are your scopes and purposes? Why is the ontology being built? What are the types of uses? Who are the end-users? What will look like using scenarios?
- Conceptualization – include extracting terms and categorizing them in a conceptual model. It is possible to use different resources for extracting and collecting terms, for example: experts, books, handbooks, tables, other ontologies, text analysis, interviews, etc. Seed terms should be refined after extraction and conceptualized into groups.
- Formalization – means explicit representation of the conceptualization captured in the previous stage in some formal language. Such a formal representation must have a syntax and semantic. The syntax describes the elements of the ontology, and how these elements may be combined to form assertions. The semantic formally relates assertions and ontology elements to object and relations of the conceptualization.
- Implementation – means determination of the technology that will be used to implement the ontology and integrate the new ontology with existing one. Ontology implementation should provide systematic tools that meet ontology purpose.

(ii) Manipulating. In this stage, an ontology query language and mechanisms should be provided for browsing and searching; efficient lattice operation; and domain specific operations.

(iii) Maintaining. In this stage, ontology developers should be able to syntactically and lexically analyze the ontology, adding, removing, modifying definition, and translate from first ontology language to another.

According to the previous analysis of ontology methodologies and the literature reviews we can conclude that:

- None of the methodological approaches is fully mature if we compare them with software engineering and knowledge engineering methodologies. Many key activities are not proposed by most of them (for example: project management, concept exploration, requirements analysis, training, ontology configuration management, maintenance, retirement procedures)
- Current methodology proposals are not unified: each research group applies its own approach. Consequently, great effort is required for creating a consensuated methodology for ontology construction. Collaboration between different groups to unify their approaches seems the most reasonable way to achieve it.
- General ontologies can be used to organize and classify term in lower ontologies; defining the relation between general and domain ontologies is an important step in ontology building process; the existing resources are the main inputs for domain ontologies.
- Ontology markup languages are still in development phases, and they are continuously evolving, which makes it difficult to have up-to-date technology for managing them.

5. CONCLUSION

During the first phase of our research, it has been recognised that the newest computer technology provides efficient and revolutionary tools to help enterprises change their way of managing and integrating data, information, and knowledge. The importance of explicit specification of conceptualization (i.e. ontology), in presented research is defined as a main factor to successful introduction of the newest technology for product development domain interoperability. Ontological statements (commitment) founded in work of the knowledge representation research community, can be used as simplifying assumptions to improve the robustness, complexity, and computability of *product development context* representation, and to avoid misinterpretation of it. Mixed approach of existing methodologies for building ontologies in the next step of the research, together with review of the current and past research of product development related topics, will be aimed to successful abstraction and formalization of entities (objects, relationships, rules, attributes, etc.) that are common across the grater part of the product development activities. As a theoretical background for the PDO conceptualization phase, the future study of Genetic Design Model System (Mortnesen, N.H., 1999.) will be performed. Accordingly to research results (Mortnesen, N.H., 1999.), GDMS seems to be able to capture the totality of results created in product development project, and it is a more comprehensive than other design model systems that can be found in literature. That is the main reason why we decided to follow this approach in the next step of our research.

ACKNOWLEDGMENTS

This project is part of funded project "Models and methods of improving the computer aided product development" supported by the Ministry of Science and Technology of the Republic of Croatia. The next research phase will be done in cooperation with Section of Engineering Design and Product Development, Technical University of Denmark.

REFERENCES

Blessing, L.: "What is thing called Design Research"; Annals of 202 Int'I CIRP Design Seminar; Hong Kong; 2002.

Borst, W.N.: "Construction of engineering ontologies for knowledge sharing and reuse"; CTIT PhD thesis series No. 97-14, University of Twente, Netherland, 1997.

Burkett, W.C.: "Product data markup language: a new paradigm for product data exchange and integration"; Computer-Aided Design 33; 489-500; 2001.

Corcho, O., Fernandez-Lopez, M., Gomez-Perez, A.: " Methodologies, tools and languages for building ontologies. Where is their meeting point?"; Data & Knowledge Engineering 46; Elsevier Science; pp.41-64.; 2003.

Guarino, N., Cararara, M., Giaretta, P.: "Ontologies and knowledge bases: towards a terminological clarification"; in: N. Mars (ed.) Towards Very Large Knowledge Bases; IOS Press, Amsterdam; pp. 25-32; 1995.

Heflin, J., Hendler, J., and Luke, S.: "SHOE: A Blueprint for the Semantic Web"; In Fensel, D., Hendler, J., Lieberman, H., and Wahlster, W. (Eds.); *Spinning the Semantic Web*; MIT Press; Cambridge; MA; 2003.

Kayed, A., Colomb, R.M.: "Extracting ontological concepts for tendering conceptual structures"; Data & Knowledge Engineering 40; Elsevier Science; pp. 71-89; 2002.

Mortensen, N.H.: "Design modelling in a Designer's Workbench – Contribution to a Design Language"; PhD thesis; IKS 00.02.A; DTU; 1999.

Neches, R., Fikes, R.E., Finin, T., Gruber, T.R., Senator, T., Swartout, W.R.: "Enabling technology for knowledge sharing"; AI Magazine 12 (3); pp. 36-56; 1991.

Pollock, J.T. "The Big Issue: Interoperability vs. Integration"; eAI Journal October 2001.; http://www.modulant.com/ResourceLibrary/Library_WhtPapers.htm; 2001.

Pollock, J.T.: "IT Systems Interoperability and the Revolution in Semantic Computing: Same Problem, Better Solutions"; A Modulant White Paper; http://www.modulant.com/ResourceLibrary/Library_WhtPapers.htm; 2001.

Pollock, J.T.: "Integration's Dirty Little Secret: It's a Matter of Semantics"; A Modulant White Paper; http://www.modulant.com/ResourceLibrary/Library_WhtPapers.htm; 2002.

Studler, R., Benjamins, V.R., Fensel, D.: "Knowledge engineering: principle and methods"; Data and Knowledge Engineering 25; pp. 161-197; 1998.

Sugumaran, V.; Storey, V.: "Ontologies for conceptual modelling: their creation, use, and management"; Data & Knowledge Engineering 42; Elsevier Science; pp.251-271.; 2002.

Szykman, S.; Fenves, S.J.; Keirouz, W. Shooter, S.B.: "A foundation for interoperability in next-generation product development systems"; Computer-Aided Design 33; 545-559; 2001.

Bojčetić Nenad; Štorga Mario; Marjanović Dorian: "Integrated Engineering Environment", Proceedings of the DESIGN 2002 7th International DESIGN conference, Marjanović Dorian (editor); Zagreb: FSB, The Design Society Glasgow; 2002., 297-300.

Štorga, M., Bojčetić, N., Marjanović, D.: "Web Services as a Virtual Product Development Environment"; Proceedings of 14th International Conference on Engineering Design - ICED03, Research for Practice; Stockholm; The Design Society, pp. 619-620; 2003.

Uschold, M., Jasper, R.: "A framework for understanding and classifying Ontology Application"; Proc. IJCAI99 Workshop on Ontologies and Problem-Solving Methods; Stockholm; 1999.