PRODUCT-PROCESS ONTOLOGY FOR MANAGING ASSEMBLY SPECIFIC KNOWLEDGE BETWEEN PRODUCT DESIGN AND ASSEMBLY SYSTEM SIMULATION

Minna Lanz, Fernando Garcia, Timo Kallela, Reijo Tuokko

Institute of Production Engineering (IPE)
Tampere University of Technology (TUT)
P.O BOX 589
33101 Tampere, Finland
minna.lanz@tut.fi

Abstract The aim of the present paper is to introduce a feature-based *Product-Process-System model* and ontology in order to retrieve and share knowledge for simulation of manufacturing systems. Product knowledge is the combination of product specific information, such as functionality, colour and product variants, and the corresponding product model. The model provides understanding of the product structure, rules, constraints and assembly-specific information in relation to the product model. In the design and modeling of assembly processes, features form the foundation for analysis and knowledge acquisition of the product. This knowledge includes geometric and non-geometric information. In the present paper, an approach is proposed to share platform independent product-process knowledge between the assembly process and system design and even with the simulation environment.

Keywords product-process ontology, knowledge management, feature-based modeling and analysis

1 Introduction

Within the production framework, assembly plays an exceptional and specific role. In this phase of the value adding chain the different components lead to the function determining unit. The high significance of assembly to a company's success is given by its function and quality determining influence on the product at the end of the direct production chain. Rationalisation of assembly process is still technologically impeded by high product variety and the various influences resulting from the manufacturing tolerances of the parts to be joined. As a result, considerable distur-

bance rates are leading to a reduced availability of the assembly systems and a delay of the assembly operations [3].

Collaborative design systems exist, yet offer little or no help in the re-use and distribution of knowledge. Without a bilateral knowledge share the design of assembly systems is as much an educated decision as it is a gamble. The advent of social software for collaboration (i.e. wikipedia) and return of XML-formats with semantic web have provided means of transferring expert knowledge into information that can be reused by many, at any given point in time. However, information systems have to cope with the ever-faster innovation cycle, in which new processes, models and systems are being developed.

Ontologies represent a tremendous opportunity for the representation of knowledge to both human and "intelligent" information systems. Ontological classification of assembly systems offer a great approach to standardise the way people and machines communicate, embedding the meaning and the context within the message being sent.

A knowledge management system (KMS) for the design of re-configurable manufacturing systems has to provide tools for storing and retrieving tacit information and strict hierarchical classifications from different domains. The retrieval and share is not the only challenge but KMS must be able to parse and form the retrieved information into meaningful knowledge that aids the generation of manufacturing systems. Ontological classification of manufacturing systems with semantic web offers a great approach to improve the way people and machines communicate, embedding the meaning to the context.

The authors propose a method for integrating product and process specific knowledge via a platform independent *Knowledge Base* (KB), neutral XML-formats and *Generic Product-Process Ontology*. The following chapters go through the theoretical approach (currently under development in the IP-PISA project) to a feature-based product ontology, KB structures and interfaces for enhanced simulation environment.

2 Theoretical Background for the Use of Features in Assembly Process Definition

The geometric definition of the part relation is a result of the design process. The product designer intentionally defines the part relation to meet the functional specifications of the product, and, as a consequence, at the same time the majority of the assembly process is defined. [3, 1]

A feature is any geometric or functional element or property of an object useful in understanding its function, behaviour or performance. Features can be seen as the base elements, detailed description of product properties, analysed and categorised based on the product and/or process ontologies. Including the features in the product model enables the representation to be in a higher abstraction level than just a pure geometrical model. [5, 7]

In the design and modeling of assembly processes, understanding the relations between components is essential, but it is not enough for representation of the assembly process, since it requires also non-geometrical assembly-specific information. Van Holland in his research realised the importance of non-geometrical features for assembly process definition. He divided geometrical and non-geometrical features into their own categories. The assembly specific information is carried over with assembly features. [5, 7, 10].

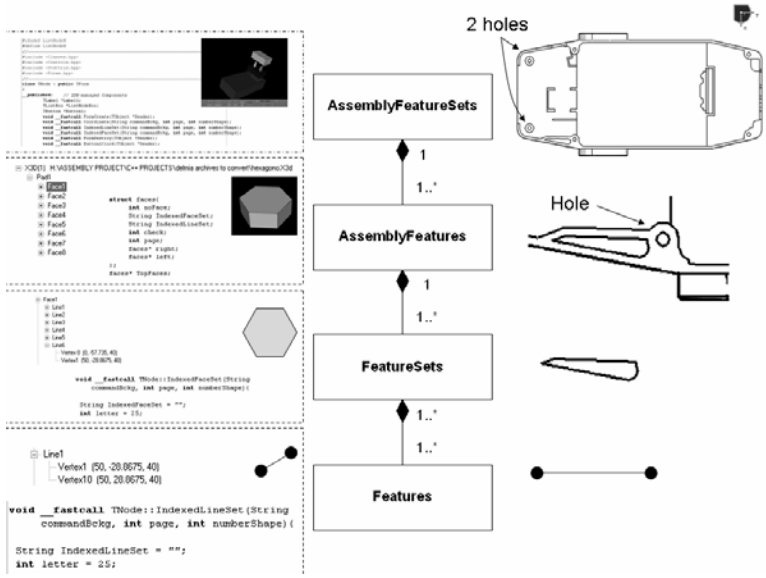

Fig. 1. Feature Layers

In the present model the assembly specific feature information is divided into four layers; *Features*, *FeatureSets*, *AssemblyFeatures* and *AssemblyFeatureSets*. Features in each layer can be either geometrical or non-geometrical features or combination of those. This hierarchical representation of feature classes will help in the definition of the structure of the knowledge base in relation to product knowledge.

3 Generic Product-Process-System Model and Ontologies

Figure 2 introduces how the product and process models are integrated. The product specific information is connected into the processes through two classes *Reasoning Machine* and *Reasoning Results*. Linking of the product model and process model is done by a reasoning machine which calculates similarities and processes both models by given mating rules. The assembly rules are used to define all the necessary assembly processes for the product to be assembled. This process information is

used to define the assembly system requirements to aid in the decision making before the manufacturing equipment is deployed. The actual reasoning is done inside the Knowledge Base, briefly introduced later in this paper.

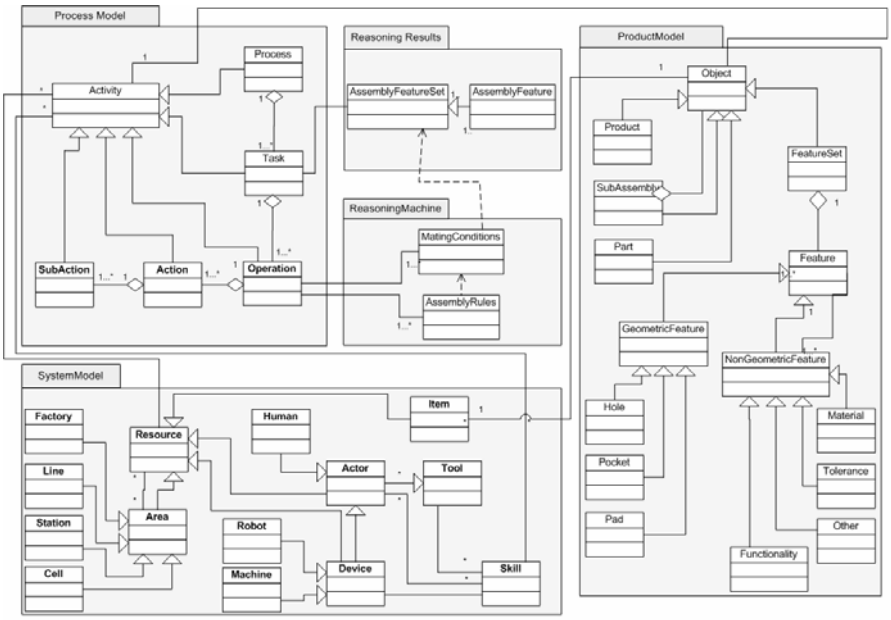

Fig. 2. Product-Process Model

The structure of the product model is a multi-layer structure, represented in the left side of Figure 2. The topmost layer is the product layer and it is for encapsulation of the product information. The layer consists of the information related to manufacturing of the current product. The second layer from the top is the subassembly layer, being a pseudo-class between single parts and the product. Each product can have several subassemblies, and a subassembly can belong to several products. This will reduce redundant information in the knowledge base and will improve reusability of the knowledge.

The third layer, *Parts*, is for individual parts of the product tree. A part cannot consist of any subassemblies or other parts, and it will only contain *PartFeatures* which are geometrical and/or non-geometrical *FeatureSets*. Geometrical *FeatureSets* are pads, pockets or holes. Geometrical *FeatureSets* can be multileveled and they simulate the parts structure by pads, pockets or holes.

The process model is visualised in the left side of Figure 2. Compared to the product model, the process model is similarly multi-layer architecture. The Process model is divided into five primary layers. The first and highest layer is called *Activity*. The second highest class is *Processes*. This class contains of the operations done in the manufacturing unit. Instances of manufacturing processes are such as assembly, part manufacture, test or packaging.

The third layer is for tasks related to manufacturing, such as fix, grasp, join, release and move. The third and forth layers are operations and actions. Operations for example in joining are snap-fit, screw, clue and weld. Each of the operations can contain a script or command sequence, which runs when needed. Actions are the most fundamental actions of movement; translation and rotation.

3.1. Product Ontology

The PISA Product Ontology, figure 3, has been defined to formalize further the product model structure and to enable general rules. The rules are defined as relations between different objects. The rules also define the type of relations and restrictions for the relations.

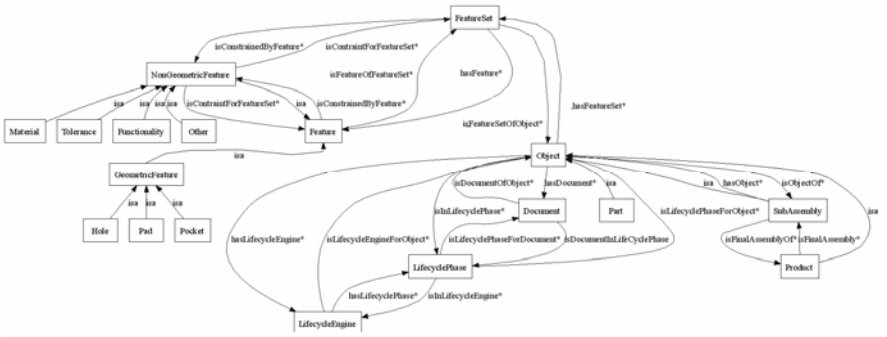

Fig. 3. Product Ontology

In the product ontology the class *Object* is defined to be in the top of the ontology. It serves as a super-class for *Product*, *SubAssembly* and *Part*. The *Object*-class enables the addition of the feature information into the product structure, when the parts of the product are not known. In the normal case the used features are linked in to the parts and not to the products. The *Part* is the lowest level class in the product structure and it has a defined geometry. A case example of the implemented product ontology can be seen in Figure 4.

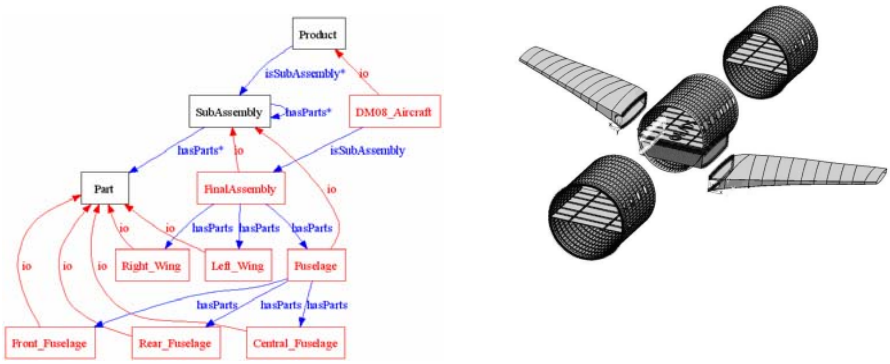

Fig. 4. Example of the product ontology of the DM08-aircraft [6]

3.2. Process Ontology

Manufacturing processes are described with process ontology. The process ontology is the key for combining the product and the system knowledge. The process ontology, in Figure 5, is designed based on the requirements given by *Product - Process model* in Figure 2.

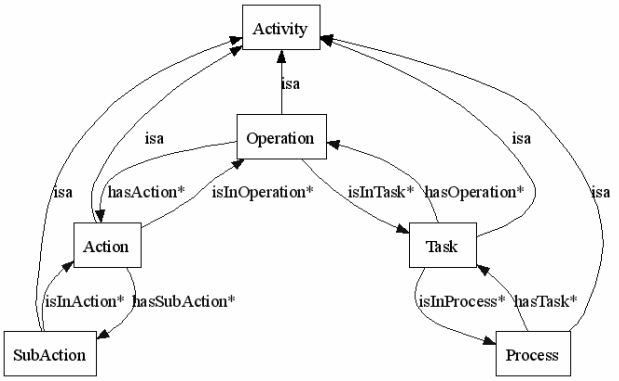

Fig. 5. Process Ontology

In the process ontology the highest class is *Activity*. The second class is named as Processes as the model in Figure 2 required. The process ontology defines classes for tasks and operations. The *Action* and *SubAction* classes are reserved for the most basic functions; rotate and translate and combinations thereof. There can be multiple sub-actions in the *Action* class.

3.3. System ontology

The system ontology, illustrated in Figure 6, is used to describe the manufacturing environment and its characteristics. The system ontology defines a structure for resources, areas and actors. The class *Item* was defined to be a connection between products and system. The class *Actor* is a super class for *Tool*, *Human* and *Device*. Actor has a class named *Skill* for defining further *Actor*'s characteristics. By separating tools, humans, devices and skills we can answer better for the re-tooling requirements. The *Resource* class has a connection to the *Activity* in the process ontology side.

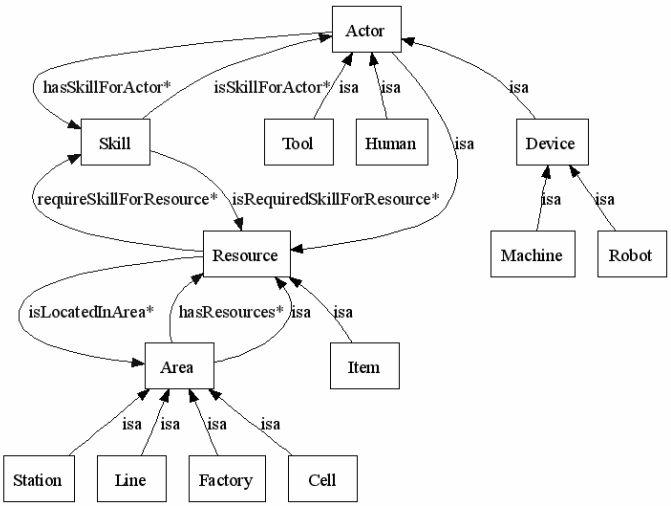

Fig. 6. System Ontology

3.4. X3D representation

The chosen knowledge visualisation language for the project is X3D, which is XML based language and a standard for 3D visualisation in web context. The OWL (Web Ontology Language) is used add the semantic meaning to raw XML and X3D information.

The PISA X3D structure is based on including and linking several files to form a product tree. The active product configuration is divided into several files. On the top is the product file, which has linking to a single sub-assembly (named as product-assembly). The product file consists of additional information about the product, its properties and lifecycle. Also, some information related to packaging or non-assembly processes can be included.

The lowest level file is the part file. Each part file contains a single part, its geometrical and non-geometrical features. At this level, the file content is based on lowest level information in the product model. In the middle of these two levels are the subassembly files. Subassemblies are like pseudo-classes and in X3D syntax they really don't have any other function than to include several lower level parts or subassemblies and to tell manufacturing instruction related. The subassembly files also provide a link to the system-specific process-files. The files are XML-based providing process requirements for the system. These files can also include a sequence or script which will be sent to an agent.

4 Knowledge Base as an Integration Solution between Clients

The knowledge of the product model needs to be stored in specific databases. These knowledge bases, serve multiple roles. It can be said that *KB*s are repositories of shared knowledge. However, overcoming the challenges in interoperability and reuse of components and declarative knowledge are crucial for the further development of the knowledge-based system. Unfortunately, it is hard to get components to interoperate and even harder to reuse other people's work. These difficulties are often a result of incompatibilities in the knowledge models (the precise definition of declarative knowledge structures) assumed by the various components. [4]

The main requirement for the *KB system* is the platform neutrality. This is indispensable in order to fulfill the objective of the product and process knowledge system. The system platform encapsulates all specific parts of a control system in order to provide a neutral interface to the application software. The system software, which provides the functionality needed to support openness for the application software, should be located on the top of the platform. Knowledge acquisition is also accessed by humans, as part of a job that doesn’t require previous experience of knowledge based systems [4, 9]

There are at least three stages to the construction and use of a knowledge base: ontology must be defined, reasonably static and long-lasting instances must be acquired, and run-time data must be entered (often when an application is running). For example, in the Assembly process knowledge domain, these steps might correspond to gathering of a design vocabulary, acquiring specific rules for joining parts together, and obtaining the details of assembly sequences. [4]

The knowledge system contains three knowledge domains based on the defined ontologies; product, process and system domain. The product-process ontology, constraints and geometrical information would be stored and accessed from the product domain. The process domain serves as a repository for process related information, such as assembly sequences, activities, tools, assembly paths etc. The system domain serves as information storage for equipment and workcell related information [9].

Figure 7 shows the *KB Architecture* which consist of four access layers, the detailed description of the tools used can be found from Figure 7b. These access lay-

ers have the intention to separate the model of the system and create groups according to its responsibilities.

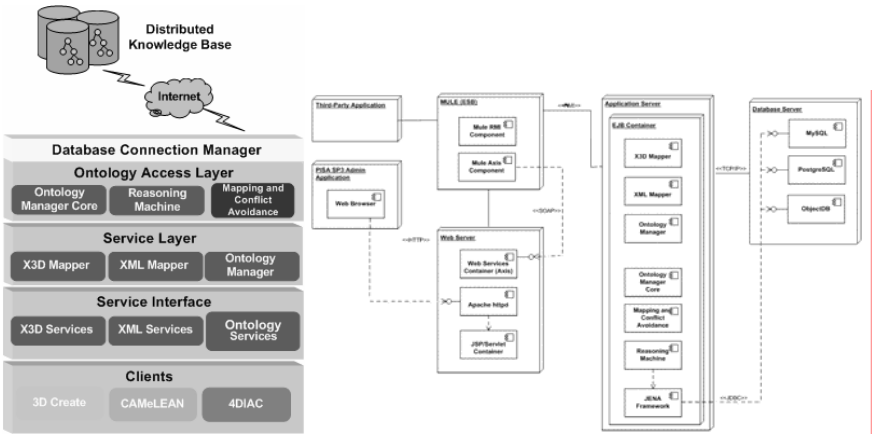

Fig. 7. a) Knowledge Base Architecture and ***b)*** Deployment Diagram of the KB

The three knowledge domains are implemented using semantic networks, with concept relations to be able to connect the knowledge from one domain to the next. This structure is flexible enough to accept the inclusion of other knowledge domains, such as logistics, supply chain management, order management, and others. The storage of information of each concepts or entity is implemented with X3D and XML-based files. This allows the knowledge base to be distributed in case of a need for collaborative design of the products and systems.

5 Discussion

The ontological approach and the structure of the KB system presented in the paper solve the problem of knowledge representation and sharing between the product-, process-, and system design domains. An effective way to acquire knowledge and share it internally and with outside strategic sources is a needed in today's rapidly changing markets. However, the system providers are not fully agreeable to sharing knowledge created in their proprietary systems with possible future competitors, even though customers are more and more often requiring knowledge capture and exchange between systems. Another major challenge is to provide a robust and acceptable solution for knowledge capture from different sources. Companies rightfully insist on reusing their information structures and familiar ways to use their categorisation system.

Acknowledgement

The development of the methods and actual tools discussed in this paper is done under the European commission funded IP project PISA – Flexible Assembly Systems through Workplace-Sharing and Time-Sharing Human-Machine Cooperation.

References

[1] H. Bley, C. Franke, *Integration of Product Design and Assembly Planning in the Digital Factory*, Institute for Production Engineering/CAM Saarland University, Germany, 2004 p. 6,

[2] D. Deneux, *"Introduction to assembly features: an illustrated synthesis methodology"*, Journal of Intelligent Manufacturing, 1999/10, pp. 29-39

[3] K. Feldman, H. Rottbauer, N. Roth, "Relevance of Assembly in Global Manufacturing", *Annals of the CIRP*, vol. 45/2/1996 pp. 545-552

[4] W.E., Grosso, J.H., Gennari, R.W., Fergerson, M.A., Musen *When Knowledge Models Collide (How it Happens and What to Do*), Technical Report, Stanford Medical Informatics, Stanford University Stanford, 1998

[5] W. van Holland, *Assembly Features in Modelling and Planning*, Ph.D. thesis, Delft University of Technology, 1997 p.160

[6] E. Järvenpää, Enhancement of Process Planning and Simulation of Final Assembly for Aerospace Industry, M.Sc thesis, Institute of Production Eng., Tampere University of Technology 2007, p. 104

[7] T. Laakko, Incremental Feature Modeling: Methodology for Integrating Features and Solid Models, *Acta Polytechnica Scandinavica*, 1993 p.85

[8] M. Lanz, *An Approach to Feature-based Modelling and Analysis for the Final Assembly*, , M.Sc. Thesis, Institute of Production Eng., Tampere University of Technology, 2005, p.100

[9]E. Lim, V. Cherkassky, "Semantic Networks and Associative Databases – Two Approaches to Knowledge Representation and Reasoning", *IEEE Expert: Intelligent Systems and Their Applications*, volume 7, issue 4, August 1992

[10] T. Tallinen, R. Velez Osuna, J. L. Martinez Lastra, R. Tuokko, "*Product Model Representation Concept for the Purpose of Assembly Process Modeling"*, in Proceeding of the International Symposium on Assembly and Task Planning ISATP 2003, Besançon, France, July 2003, p.6