Use Case 1 in Technology Provider – Research Challenges

Product Lifecycle Management

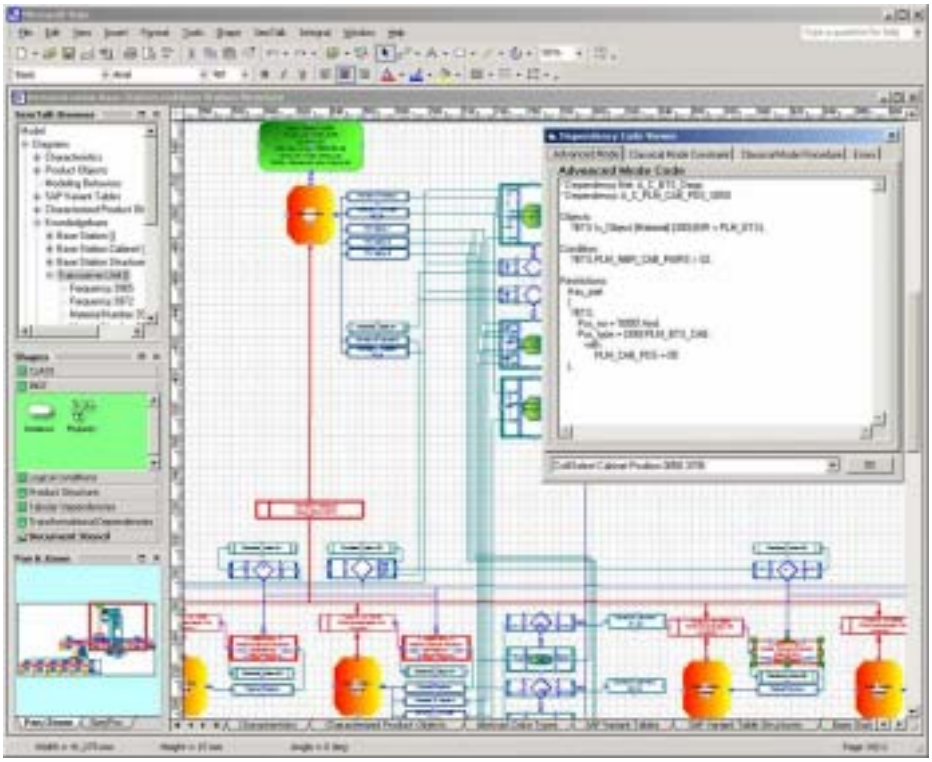
KW Partner: FU Berlin
IB Member: Semtation GmbH

1. General Description of Business Use Case

There is a high cost associated with the development and maintenance of product catalogues throughout the product lifecycle. Expensive and complex tools need to be used and the structure of product catalogue data, in which characteristics and relationships are not expressed through some standardized terminology, causes difficulties in development and maintenance of that data. For example, the data must be reinterpreted for a different context of use, may be worked with in a collaborative environment, or shared between departments who may have a different understanding of its purpose and meaning.
A product portfolio contains a lot of knowledge and rules about the products and their relationship to one another, but this is hidden in the data without it being easily possible to extract and use in tools. Ontologies are seen as a viable approach to improving the development of product catalogues and their maintenance over the entire product lifecycle.

2. Proposed Semantic Web-based Solution

A tool called Integral supports the representation of product configurations as Visio diagrams, in which the diagram components are also tied to descriptions using a common terminology. The current tool is SAP-based but it could be extended to support Semantic Web standards such as OWL in the future.



3. Identified Research Challenges

3.1 Ontology Development/Management

3.1.1 Problem Statement

Product lifecycle management currently are based around the manual development of a product catalogue in which the product knowledge and relationships are defined by the developers at the implementation stage. This is a very specialised approach: it requires a consultant to ensure a properly customised system and approximately 3 months to learn how to model products in that system. While ontologies offer clear value in formalizing and making explicit product knowledge, the complexity of this task is moved from the tool user to the ontology developer, who must ensure he or she can appropriately and adequately model all the knowledge the user wishes to express.

3.1.2 Knowledge Processing Task and Component

The **ontology manager** is the component that shall enable the development of the ontology for the product catalogue. We can identify that there are different levels of ontology that will need to be created, possibly by different management tasks. The tool developer will need an ontology that models the common aspects of product catalogues. The tool user will need an ontology (or ontologies) that can express knowledge about his or her type of product. Finally, users of the product catalogue data (not of the tool) may have their own requirements that will have to be met.

3.1.3 Requirements Analysis

An ontology development tool should be able to aid the creation of the ontology by being able to communicate the correct understanding of the ontology to the developer, give feedback based on best practises and guidelines, and provide an ontology visualization.
Product ontologies need to be available or developed, and best practises and guidelines should be available for their development or re-use.
As we foresee that existing ontologies may be extended or altered to meet individual user needs, it is preferable that the user can do this without requiring specialized ontology experts. Rather, an user tool should be hiding the complexity of the ontology, permitting the user to express his or her needs in an intuitive way that it can map internally to ontology changes.
Ontology changes should also be trackable, e.g. tied to a specific user, so that the user can produce the knowledge he or she needs, but maintaining compatibility for data exchange with users who have not made or have made other changes.
As ontologies will necessarily evolve, both as product catalogues change and knowledge needs of users change, there is a need for ontology engineering and maintenance practises and tools.

3.2 Personalization

3.2.1 Problem Statement

A large company will not only have a large product catalogue, but the catalogue data will be used by different departments of the company for different purposes. Furthermore, in each department, there may be a different understanding of what the data means. Product knowledge needs to be used in different contexts, yet the data defining this knowledge is inflexible, lacks a clearly understood meaning and hence could easily lead to conflicts in a collaborative situation.

3.2.2 Knowledge Processing Task and Component

A **profiler** is the component tasked with personalization. It is in charge of tailoring services available from the system to the specifics of each user. It should offer each user a view of the data which respects their understanding, needs and skill level.

3.2.3 Requirements Analysis

An ontology inherently supports reasoning so that different views on the same product could be offered to different users. This requires a suitably expressive ontology as well as the possibility of different users having access to different aspects of the knowledge. Users could be tied, through profiles, to their personal knowledge about the product (i.e. that which is relevant to them, and not to other users). Access rights may also need to be clearly defined, i.e. which users can alter which parts of the product knowledge.

Collaborative work tools that support ontologies are required to allow simultaneous different-view work on a product catalogue, handling conflicts and maintaining a common internal view.

3.3 Support for logic/rules

3.3.1 Problem Statement

Product catalogue data is very complex, and may need to represent dependencies between products or could benefit from automatically inferring some knowledge on the basis of other knowledge that has been explicitly provided. While ontology languages such as OWL support this at different levels of expressiveness, it is not clear if all that a user may wish to express can in fact be expressed in the chosen flavour of OWL.

3.3.2 Knowledge Processing Task and Component

The **reasoner** component supports an appropriate logical model to enable the tool to work with a sufficiently expressive set of knowledge to fulfil the user requirements. However the implementer may need to make a trade-off between expressiveness and computability, so that the tool also demonstrates satisfactory performance.

3.3.3 Requirements Analysis

OWL is of course available in different flavours, which offer a trade-off between expressiveness and computability. Best practises and guidelines should be able to help the developer in making a decision, i.e. is it better to support everything a user may want to express (and maybe end up with OWL-Full?) or to restrict users (e.g. to Lite or DL) in order to ensure efficiency and computability.

Given the needs of product catalogue data, the implementer should also be able to assess if OWL is in fact the most appropriate logic to choose, against e.g. F-Logic or Datalog. OWL-based approaches should be extendible, e.g. to support rules or measurements. Being able to define compatible logic subsets may facilitate both expressiveness and knowledge sharing.

Finally, performance issues are important for an industrial application. Suitable expressiveness needs to be combined with satisfactory performance if the tool is to be industrially useful.