7th International DAAAM Baltic Conference
INDUSTRIAL ENGINEERING
22-24 April 2010, Tallinn, Estonia

DYNAMIC REPRESENTATION OF PRODUCT DEVELOPMENT DATA USING SEMANTIC WEB TECHNOLOGIES

Ulbrich E.; Zoier, M. & Rosenberger, M.

Abstract: *The product lifecycle can be seen as a sequence of phases, where specialized applications are used throughout the development cycle. One data standard to realize a common data model connecting different applications in an Integration Platform are Semantic Web Technologies.*
It is important to represent this information to the user in a filtered, task- and domain oriented way e.g. to show cross domain dependencies.
This paper shows a concept for a new interface that has the ability to visualize information in an optimized arrangement and level of detail. It has the capability to represent data in a way that supports user specific views by only showing relevant data but also in terms of usability.
The concept is realized in a demonstrator using a Semantic Web data model.

Key words: product development data, development process, data visualization, Semantic Web Technologies, user interface, information management, views

1. INTRODUCTION

The motivation for this work has its roots in a project initiated at the Virtual Vehicle Competence Center. One of the goals of this project has been to enable a quick, reliable and traceable data exchange during the development of a product.
Around 75 percent of all products within the automobile industry have a predecessor. The engineers of those earlier products gained much experience during the development. Not only because of timely manners but also because of cost reasons the reuse of experience and knowledge in product development is mandatory [1]. Engineers in various development phases often work in different departments or even in different companies along a supply chain. Therefore, ways to represent the product data can vary. Even if the development of a product is properly documented, it is not always understandable.
The Virtual Vehicle strived to find a technological solution for this problem. One way to achieve this goal is the use of semantic technologies to structure the product data and represent the relationships. The developed solution has an interface that requires expert knowledge to the ontology-based data model as well as the query language SPARQL. Therefore, there was a need for a suitable interface to improve the usability.
For this a new method to query and represent data has been developed to enable engineers to access the semantically organized data.

1.1 Overview of current problems

Globalization, outsourcing, customization of products, support of products with long lifetimes, and other regulations lead to a more and more complex process of development, production and aftersales. Product Lifecycle Management (PLM) enables companies to have a better overview on their products. Lifecycle management can be especially of use during phases of development where a product only exists virtually. Furthermore, support of products with long life spans gets easier [2].
A product lifecycle consists of different phases wherein a product exists in various stages. Throughout these stages, different

software tools support engineers developing a new product.
Departments can be geographically separated, several tasks can also be outsourced to suppliers leading to networks of engineers working on the same product. Those networks require a time-consuming communication and sophisticated management of the complex data.
Software applications are commonly developed to support a specific group of tasks and do not provide an overview of the complexity of the product and the overall situation [1]. A significant number of tools is needed to provide a complete data set of the product.
Relying of different groups of engineers working geographically separated, this data is often in their own wording and not fully understandable to developers from other departments or companies. Different competences, abilities, focus and needs on the information can be additional reasons for semantic variance. Searching for specific information within this mass of data in unfamiliar structures can be time-consuming.
One way to represent and connect the inhomogeneous product data originating from different software solutions is to use semantic nets as data structure. They can be seen as nodes connected by unidirectional links with attributes and rules and are also called ontology.
To model the complex net structure of data within an ontology, new methods with views according to user's roles are needed.

1.2 Aim of this work

This paper is based on the assumption that searching and gathering context relevant information can be supported by the use of semantic technologies. It is also assumed that not all users have time or knowledge to understand the structure of data within an ontology. It is therefore required to find new representation methods enabling a majority of product developers to access this data in a fast and efficient manner.
No specific knowledge about the structure of the data or any query language should be needed.
In this paper a method and a prototype is shown which can be seen as part of a semantic integration platform. Through this prototype the following goals are to be achieved:
1. A new method has to be developed to enable users to search through semantically structured data without knowledge of query languages by dynamically adapted queries triggered by user interaction.
2. Realisation of the concept in a prototypic software application by design of a new interface to represent data according to users' needs regarding the gathering and understanding of information.
These goals will be verified by the development of a prototype populated with reference data from industry partners.

2. REPRESENTING AND SEARCHING INFORMATION

2.1 Representing information

90 percent of all currently used visualisations are hierarchically structured [3]. Most of them are two-dimensional; three-dimensional visualization is often not sufficiently supported by interface technology.
To fully grasp information, Shneiderman [5] explains a certain way for representation. First, an overview of all information has to be given to enable users the possibility to orientate themselves. Second, users should be provided a way to zoom into areas of interest. Less interesting areas should be filtered. After finding interesting objects, users should be given more detail on these objects.
The prototype has been developed according to this suggestion.

2.2 Searching for information

Bates (1989) developed a model for the search for information called "evolving search". Users with a certain need for information start their search with a small part of a broad topic, for example a reference. Starting from this point, several strategies are used to gain an overview about the topic. With every new bit of information the need for information alters

and the search is adapted. Not only one well structured query leads to a satisfaction, but many small queries in a row produce small bits of information that slowly satisfy the also changing need for information. [4].
The search for nodes within the semantic net will be implemented by selecting one of them at an overview list at the prototype. Each interaction with the interface will provide additional information.

2.3 Tree representation

Some structures consist of objects connected to parent- or child objects. These connections may have various attributes. Such structures can be represented as so called trees. Two special cases of trees are those with a flat hierarchy and many objects in few levels and such with a particularly deep hierarchy and only few objects per level [3].

2.4 Net representation

Sometimes, objects do not resemble a hierarchical structure and cannot be displayed as trees building up complex structures. Representing such structures is often tough [5]. Complex net structures containing all relations often lead to a confusing picture where some objects are overlapping with others, hiding relevant information. Furthermore, users are usually not interested in the whole amount of data but search for information particularly important for their role and often use only a certain part of available information [3].
The developed prototype displays data according to a user's role to prevent mental overloading and confusion.

3. TOOLS AND FRAMEWORKS

The net data structure has been designed in Protege, a popular open source ontology editor, using OWL (Web Ontology Language) data format. A part of a net data structure is shown in Figure 1. By the help of a small server application, this ontology can be queried. This server is basing on Jena, an Open Source Semantic Web Framework, emerging from the „HP Labs Semantic Web Programme". It is written in Java, features an integrated Reasoner and supports Ontologies in RDF, RDFS and OWL data standard. Jena manages the structure of the ontology.

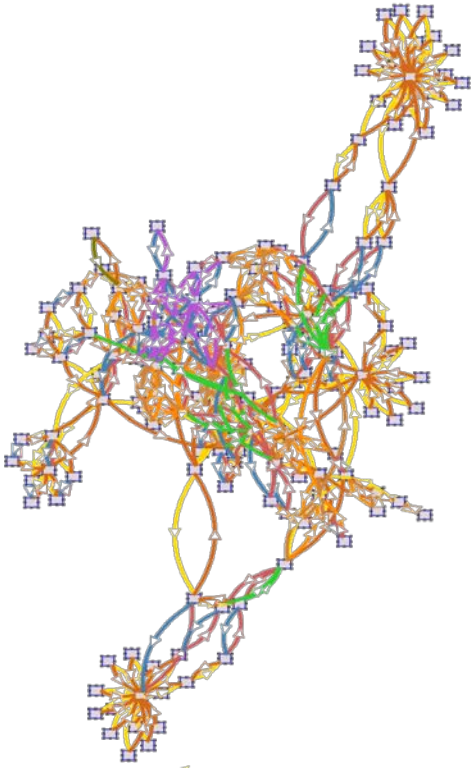

Figure 1: Visualisation of an OWL Ontology representing part of a net data structure used as common data model of an integration platform

The RDF-framework is connected to a web service called Joseki providing a basic web interface. Joseki is optimized to search through the RDF-framework via queries formed in the SPARQL, an SQL-like language optimized for ontologies.
To provide an adequate user-interface, a Rich Internet Application (RIA, a browser based application looking similar to a native application) basing on Adobe Flex is used. Adobe Flex is often used for applications at enterprise front-ends and supports HTML, CSS, JavaScript and Ajax as well as Flash.

4. INTERACTION CONCEPT

The interaction is structured in two steps. The first step is the selection of a certain role connected to a template filtering future

queries after a certain role specific scheme. A static query is launched leading to a defined initial state of the displayed data. The second step allows interaction with the displayed data leading to self generating and altering queries searching for further data. The interaction scheme runs as follows:

- A search for information within the semantic net is launched by a user interaction
- A SPARQL query is defined and gets automatically altered according to the user interaction containing questions about the required node, selected parts of their environment, and connected nodes.
- The ontology is queried and returns an XML file.
- Irrelevant returned data is filtered according to the user's role.
- This XML is transformed into an Adobe Flex structure that is optimized for being displayed to the user.
- Flex represents the resulting information in context to the user interaction. Additional queries are launched for further details and displayed in combination.

5. QUERY CONCEPT AND PROTOTYPE

For a search through the ontology an URL is formed by Joseki containing all information necessary for a query pointing to a document with the queries result. This document can either be structured as an XML or a JSON document. The prototype requests this document as an XML.
After loading the external XML the prototype filters and transforms the document into output XML structures or arrays easier to understand for Adobe Flex and ready to display for its interface components.

5.1 Start screen

After starting the prototype called "OntoNavi", it provides the possibility to choose between three generic roles of product developers to represent the necessity of satisfying typical need for information of groups working in different departments (see Figure 2).
Selecting one of them launches a static query displaying a group of objects according to the selected role.

Figure 2: OntoNavi start-screen

5.2 The search screen

The screen to search through the ontology contains three main components (see Figure 3). An area displaying a tree to provide an overview of a specific group of objects within a structure is positioned left. On top of the screen, a list of objects referred (by the ontology) to a selected object in the tree can be found. Below this list, a text area provides additional information of the selected nodes relations to nodes in other structures.
Navigation through selected nodes is displayed and documented by the expanding tree. This navigation element is listing all elements of a certain structure representing a particular view on the underlying data network (e.g. components

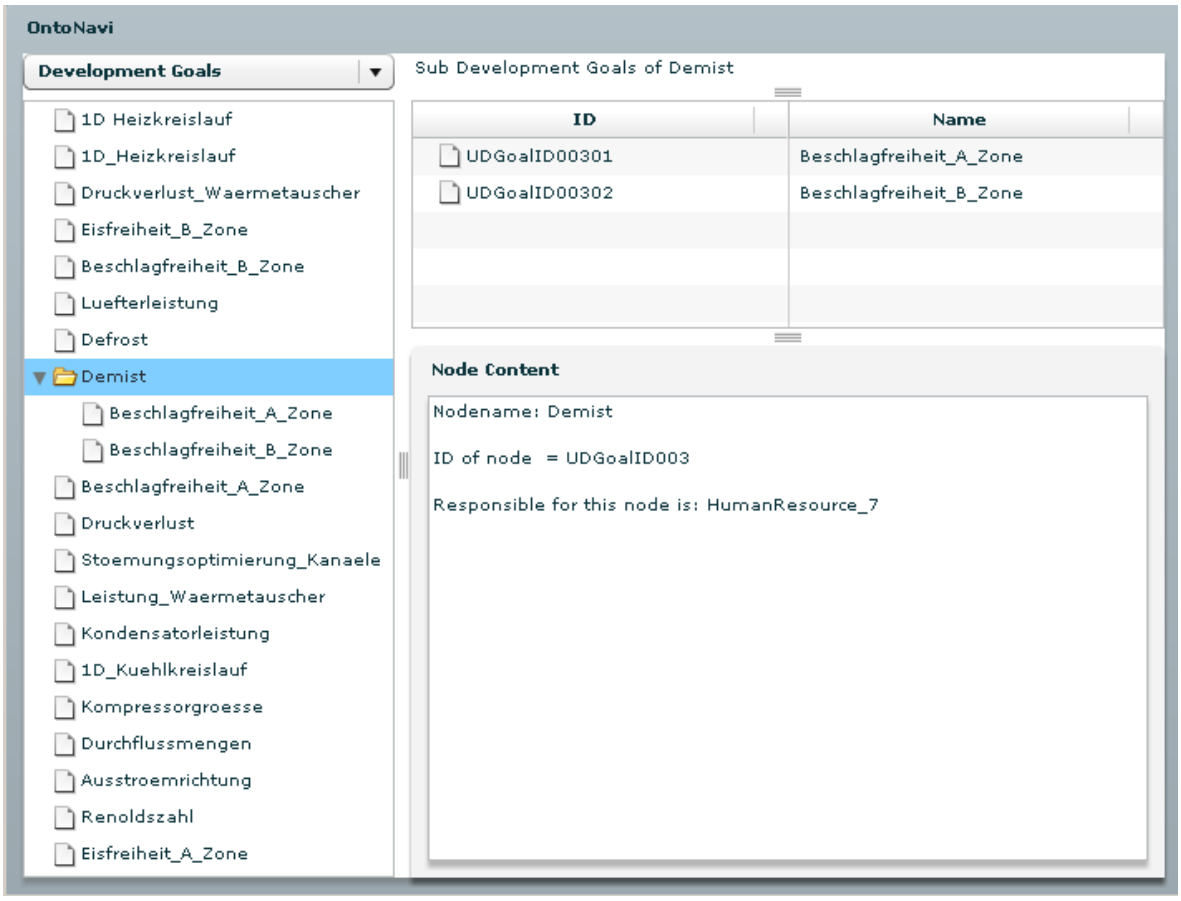

Figure 3: OntoNavi Main Screen

of a bill of material, development goals, or individuals responsible for certain tasks). Those views can be changed at any time during runtime via a dropdown element.

6. VERIFICATION

Questions representing "typical questions by industrial product developers to the system" case were used to verify the prototype allowing identifying the pros and cons. The ontology data structure has been enriched by reference data representing part of the product development process with focus on the relationships of components to data from other domains.

6.1 Questions to the Use-Case

Following questions amongst others typically meet the need for information of developers of products concerning the components forming the whole product:

- Out of which project components does a parent component consist?
- Who is responsible for that component?
- Which project goals are connected to a component?
- Which components have changed, are new or have been deleted?
- Which are the goals and components an organizational unit is responsible for?

6.2 Answers by using the prototype

Subparts of parent nodes are directly represented by the networked data model. The responsible person for the selected component is displayed at the human resources structure. The tree structure can be changed by the drop-down element. Selecting an element of the human resources structure displays its responsibilities.

Answers about goals can be given by selecting one at the development goal structure (as seen at figure 3 where human resource 7 is responsible for the goal demist having two sub goals).

Questions about changed, deleted or new objects cannot be answered by the prototype. But small changes of the assembling

of queries could easily allow searching for such nodes.
Which human resource is member of which structure, having which role and is responsible for which component can be queried at the human resource structure.
However, the prototype does not provide a possibility to insert SPARQL queries manually. In case of a very specific need for information, searching for one specific node with the required data takes more time than simply forming a query and asking for a known node.

7. DISCUSSION

The advantage of the shown concept and prototype is the composition of visual elements leading from an overview to more detailed information. The prototype automatically provides a representation of result of queries with a variable level of detail. Users do not have to understand the underlying data model or the query language SPARQL. Therefore, semantically organized data is accessible in an advanced and optimized way to different user groups in the product development lifecycle.
They are able to use common user interaction like mouse clicks to gain information.
Disadvantages of the prototype are the partly slow response time at complex interactions that cause cascades of SPARQL queries with respect to one query replying all desired information.

8. REFERENCES

1. G. Zrim, M. Maletz, R. Lossack: *Experience Based Cost Management in the Early Stages of Product De-velopment*, Proceeding der International Design Conference-Design 2006 Dubrovnik – Croatia (15. – 18. Mai 2006), 2006
2. J. Stark, *Product Lifecycle Management – 21st Cen-tury Paradigm for Product Realization*, Springer, London, ISBN 1852338105
3. R. Däßler: *Informationsvisualisierung - Stand, Kritik und Perspektiven*, In Methoden/Strategien der Visuali-sierung in Medien, Wissenschaft und Kunst, Wissenschaftli-cher Verlag Trier, 1999.
4. M.J. Bates: *The design of browsing and ber-rypicking techniques for the online search interface*, Online Review, Vol. 13, No. 5., p 407-424, October 1989, http://www.gseis.ucla.edu/faculty/bates/berrypicking.html
5. Shneiderman B. 1996: *The eyes have it: A task by data type taxonomy for information visualizations*, Proceedings des IEEE Symposium on Visual Languages, p 336-343, 1996
6. J. Dmitrieva, F. J. Verbeek: *Multiview Ontology Visualization*, Proceedings of the 11th Protégé Conference, 2009,
7. R. Burkhard, M. Eppler: *Knowledge Visualization*, Encyclopedia of Knowledge Management, Idea Group Inc., 2005 http://www.knowledgemedia.org/modules/pub/view.php/knowledgemedia-67
8. T.R. Gruber, *Towards Principles for the Design of Ontologies Used for Knowledge Sharing*, Iternational Journal Human-Computer Studies 43, p 907-928, 1993
9. S.J Fenvez, R.D. Sriram, F.Wang: *A product information modeling framework for product lifecycle management*, Computer-Aided Design Volume 37, Issue 13, p 1399-1411, November 2005

9. ABOUT THE AUTHOR

Eva Ulbrich
Virtual Vehicle Competence Center
Inffeldgasse 21A, 8010 Graz, Austria
eva.ulbrich@v2c2.at
Phone +43 316 873 9006
www.v2c2.at