Using Topic Maps for Visually Exploring Various Data Sources in a Web-based Environment

Eicke Godehardt[1,2] and Nadeem Bhatti[2]

[1] SAP AG, Germany
eicke.godehardt@sap.com
[2] Fraunhofer IGD, Germany
{eicke.godehardt, nadeem.bhatti}@igd.fraunhofer.de

Abstract. Today every company and every single person owns a lot of data. However, easy ways for exploration and navigation are not very widespread. Especially, if complex relations between these data are defined and the amount of data becomes bigger, an easy to use interface is necessary.
This paper describes a topic map viewer, which is easy embeddable into websites to offer none-experts an innovative way to find new and relevant information. To display even huge information spaces in a web environment, this paper describes two concepts supporting scalability are explained, namely data exchange and visualization. For the data exchange between client and server pre-fetching mechanism is introduced to load data in small chunks. Garbage collection complements this to prevent overloading of the viewer by unloading not needed topics. Furthermore the clustering concept for the visualization is used to handle the huge amount of information.

1 Introduction

Most of the data sources today are not accessible at all or only understandable by experts. Especially if complex relations between these data are defined and the number of data becomes bigger, an easy to use interface is necessary to allow also non-experts to browse through this data and gain new information.

In this paper, we describe how we use XML Topic Maps to visualize data in a web-based environment. Topic maps provides a model and grammar for representing the structure of information resources [1]. The topic map viewer displays the topic map and allows the users to navigate and explore through the map.

The goals and requirements for the topic map viewer are the following:

- web-based and easy integration in existing websites (Web2.0)
- visualizing already existing data sources
- large amount of data
- easy navigating/exploring through the data

Based on the requirements and goals above, the following key points will be explained. How we use existing databases via a XML Topic Map interface? In addition we focus especially on using different data sources and how we solve the issue of handling very large data sets.

The paper is organized as follows. Section 2 gives a brief overview on related work, which are using topic maps as a basis for some kind of visualization. Section 3 describes our approach, in particular how we retrieve the data as XML Topic Maps and handle scalability issues through loading on demand and pre-fetching. Subsequent in section 4 we show a use cases for this technology. We summarize our findings in section 5 and give an outlook on what to solve in future increments of the topic map viewer.

2 Related Work

There are several approaches for using topic maps for visualization in a web based environment. Below, some of these works will be mentioned and described in brief.

TopiMaker[2] is a stand alone application, which has the main goal to provide a novel authoring environment. It positions the topic maps into several layers placed in a 3D environment. The visualization of TopiMaker is more suitable for authoring and exploration is not widely supported. The second main difference is that TopiMaker is designed to be a stand alone application in contrast to a web application as our solution is designed for. Another examples which focus more on editing are [3] and [4], but the described solutions also offers only a very primitive view, which is not sufficient for exploring huge and complex data sources.

There is also some research going on about visualization and navigation related to our work. For example a simple topic map viewer [5] or visualizing auto-generated topic maps [6]. In addition there is also some work in visualizing search results or metadata [7]. The systems are mostly Java based and desktop applications.

While there are several approaches and some of them consider related goals none of them focus on scalability or handling large topic maps in an embeddable web-based environment.

3 Approach

In this section the main way of acquiring the data is described. Figure 1 below gives an overview of the data flows on how our topic map viewer accesses different data sources. The response is encoded as an XTM – XML Topic Map (XTM [1]) file.

3.1 Scalability

The goal of our visualization tool was to allow an easy integration into existing websites to visualize existing data. Options for the visualization technology in web are Flash and using Java Applets. Graphical visualization is not possible with Javascript alone and SVG is not yet widespread. But for Applets it can become very complex to get them running in every single environment. To reach the main objective of this paper, the Flash technology and XML Topic Map format as the data exchange format are used to achieve a web-based solution. The Flash Player is a client application and available in most common web browsers (over 98% [8]) and supports vector and raster graphics and an object-oriented language called Actionscript (AS).

On the other hand, this decision, which satisfies the required flexibility, has an important drawback when it comes to scalability. A straight forward realization may handle a few hundred topics. But in real-word scenarios this is not sufficient. In large companies and even small enterprises the number of topics easily becomes several thousand. But as it is not the goal to visualize all topics at once. Instead the topic map viewer will support kind of exploration behavior by loading the data on demand. The user chooses one starting point in the topic map. Then the topic map viewer displays a small amount of information around this starting point (chosen topic). Afterwards s/he can navigate through the map by selecting different topics according to their own interest. By selecting the viewer requests for small packets or chunks from the server (described above), which allows an interactive and responsive user interface. Every response from the server is still a complete XTM file (see figure 1) and is merged with current state of topic map inside the viewer.

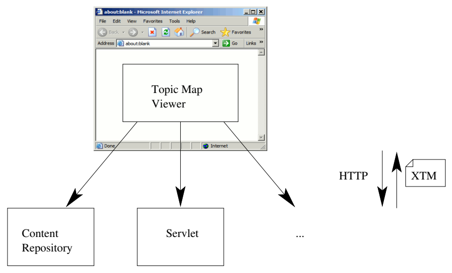

Fig. 1. General access of data sources

The user may not even know that not all data is loaded at the beginning of his/her survey. The figure 2 below points out this behavior. It is very important to notice that the figure 2 shows the status of the loaded topic map and *not* the visualization of the topic map itself.

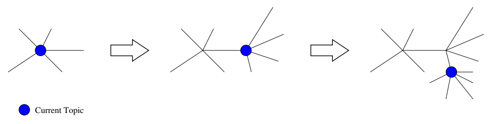

Fig. 2. Loading parts of the topic map on demand

In the image three steps of changing the current topic is shown and how the internal topic map is changed accordingly. The topic maps are loaded on demand as the user clicks on a different topic and merged with the already load topic map inside the viewer. This is only done once fore every new topic to reduce traffic, as we do not expect the topic map to change at the server side.

To merge all responses from the server and integrate more and newer information, the topic map viewer technically uses the original definition of `mergeMap`. `mergeMap` is defined by the TopicMaps.Org Authoring Group for XTM (see [9]) and normally allows the subdivision of a topic map over more than one file.

3.2 Pre-fetching

The above described technique can be improved as there is still some latency when the user selects a new topic, as the new data chunk has to be loaded. We achieve this by pre-fetching or pre-loading additional parts of the topic map in advance. For example starting from the current topic, the topic map viewer first loads all direct connected topics together with the current one. This is exactly the same behavior as described in the section above. After all direct connected topics are loaded and displayed, the viewer starts loading all topics that are "two steps away" from the current topic, i.e., direct connect topics from all loaded topics in the first place. This can be repeated several times to pre-load the context of the user in advance, e.g., three level context of the current topic as shown in figure 3. When the user starts navigating around in his/her current context, the topics can be displayed immediately as they are already loaded. Additionally pre-fetching starts again in respect to the new current topic.

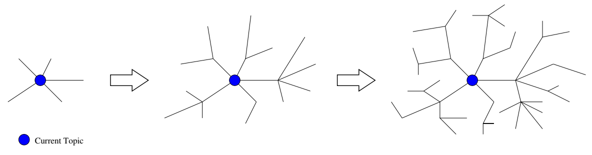

Fig. 3. Pre-loading topics of the current context of the user

As in figure 2 the displayed topic map represents the loaded topics and not the visualization of the topic map, which is subject of section 4 below. Unlike the earlier figure 2, the current topic does not change in figure 3 although the internal topic map inside the viewer is enhanced over time.

The main advantage of pre-fetching the user's context is a smoother user experience. Especially over a slow Internet connection, the general response time while browsing the topic map is much shorter than using the elementary on-demand solution alone described in section 3.1 above. The combination of loading on demand and pre-fetching is still scalable, which means the viewer can handle topic maps of huge sizes while still providing a responsive user interface.

3.3 Garbage collection

A drawback of pre-fetching and loading on demand is the growth of the topic map inside the viewer. The number of already loaded topics can become really big while using the system. This growth can lead to a slow down of the topic map viewer response time, as more topics have to be handled. To circumvent such a slow down, we suggest kind of a garbage collector to reduce the maximum number of loaded topics. In contrast to the usual definition of a garbage collector, we just focus on the task in freeing memory of our application. While the deleted topics are not garbage in the usual sense (not referenced/accessible), they may not be used in the future any more. But even in this case, they can be loaded once again without problems at any time in the future.

Every topic, which may not be interesting to the user, is dropped. To figure out if a topic is still of interest to the user, the viewer uses some heuristics. The most practical and obvious indicators for such an action could be how long ago is the last time of display are or how far away is a topic from current topic? One possibility could be to delete every topic from the current topic map which was last displayed at least 15 minutes ago or the distance from the currently selected topic is greater than 10 steps. All this may indicate topics, which may not be used in the near future and can thus be deleted to optimize the runtime behavior of the topic map viewer.

4 Use case/Case study

The objective of the project "SAP-TM-Viewer" is to develop an SAP applications integrated information system in order to support the knowledge engineers within the enterprise processes. When the knowledge engineers need help during the enterprise process, this information system should provide the context specific help.

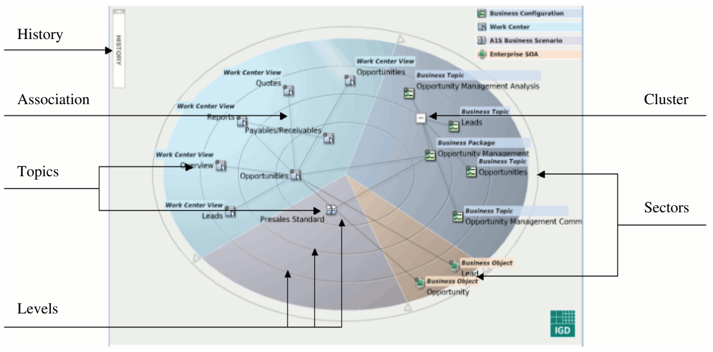

Fig. 4. TM-Viewer

In order to achieve the goals of this project, the key approach was to develop a Knowledge map so called SAP-TM-Viewer on the base of the TM-Viewer technology (see above) to visualize the knowledge items, the relationship and hierarchical structure between the knowledge items. The SAP-TM-Viewer consists of fields which are abstracted from the SAP topic map. Knowledge concepts in each field are represented with specific icons, lines between the knowledge concepts represent the associations and the levels represent the abstraction level of the knowledge concept (inner level show generic knowledge concepts) as shown in figure 4.

The pre-fetching, garbage collection mechanisms of TM-Viewer techniques and cluster concept make it possible to handle a huge amount of knowledge items (>4000) while still being responsive. But both concepts do not solve the problem to visualize the huge amount of data. The visualization of huge number of knowledge items, e.g., more than 100 topics can lead to cognitive overload. That is why the cluster concept was added to keep the visualization manageable for the knowledge workers. According to the cluster concept all the topics, which have same sibling, will be clustered as it is shown in the figure below.

5 Conclusion and future work

In this paper we have shown, how we used Topic Maps as a generic kind of data exchange used to view and explore huge data sources in a web-based environment. In particular it was shown, how issues like scalability and responsiveness are handled using loading on demand, intelligent pre-fetching and garbage collection.

Future research directions on the data retrieval part may be performance evaluation by user tests and awareness. Awareness (first addressed in section 3.1) concept will enable the user to get an instant visual feedback of dynamical changes in the topic map. For example, if the enterprise applications change the semantic information the topic map viewer shows these changes in semantic visualization. This approach will keep the visualization up-to-date. Another important focus for future research could be to expand the ability of topic map viewer to visualize other different types of data structures as intuitive as the tree-like structure described above (e.g. process visualization).

References

1. TopicMaps. ISO/IEC 13250.
2. David De Weerdt, Rani Pinchuk, Richard Aked, Juan-Jose de Orus, and Bernard Fontaine. Topimaker - an implementation of a novel topic maps visualization. In *Proceedings of International Conferences on Topic Maps Research and Applications, Leipzig*, 2006.
3. Boriana Ditcheva and Darina Dicheva. Visual Browsing and Editing of Topic Map-based Learning Repositories. In *TMRA 2006*, 2006.
4. Hornung, J., Hornung, C., Oliveira, A., and Fernandes Marcos, A. Knowledge Visualization based on Topic Maps for Computer-Based Learning Applications. In *World Conference on Open Learning and Distance Education*, 2001.
5. Ontologies for Education Group. Simple topic map viewer: very general and java applet based o4e (ontologies for education) portal. http://iiscs.wssu.edu/o4e/.
6. Nadine Amende and Stefan Groschupf. Visualizing an Auto-Generated Topic Map.
7. Lynne C. Howarth and Thea Miller. Visualizing search results from metadata-enabled repositories in cultural domains. In *TMRA*, pages 263–270, 2005.
8. Adobe. Flash player penetration. http://www.adobe.com/products/player_census/flashplayer/.
9. TopicMaps.Org Authoring Group. Xml topic map - mergemap specification. http://www.topicmaps.org.
10. TM4L. Darina dicheva, towards reusable and shareable courseware: Topic maps-based digital libraries. http://compsci.wssu.edu/iis/nsdl/.