Multidimensional Topic Maps

A brief introduction to the theoretical foundation of the Topic Grid

Thomas Schwotzer

HTW Berlin
thomas.schwotzer@htw-berlin.de

Abstract. Topic Maps are means to formalize an ontology. Ontologies are used to describe a domain of discourse. In same cases, a domain comprises facets from different domains. This short paper briefly describes such domains and give an idea how to define multidimensional Topic Maps. The concept of more dimensional Topic Maps are basis of the Topic Grid system.

1 Topic Grid – General Concepts

There are two general approaches to understand distributed nature of knowledge based systems: *world knowledge* and *micro theories*[1]. In short: The *world knowledge* concept assumes that (theoretically, not in practise) any knowledge can be merged into a single knowledge base. This idea has several implications. It assumes the possibility of a single vocabulary for any knowledge. Moreover, it assumes that a general agreement about relations between concepts of this world wide knowledge base can (at least theoretically) be achieved.

I claim, that (at least the origins of) Topic Maps as well as RDF/OWL follow this idea. An indication is the presence of the concept of ontology *merging* in both standards. A concept of fragmentation, separation isn't defined. Note, queries languages like TMQL[2] or SPARQL[3] are means to search in an existing ontology but not to split them. It can also be shown that creation of a world wide vocabulary is be possible. A general accepted set of *relations* between concepts cannot be achieved, though.

The opposite model is the model of *micro theories*. It states that the overall knowledge must be seen as a collection of independent local knowledge bases (micro theories). It isn't necessarily possible to merge two micro theories. Moreover, it isn't necessarily possible to find a common vocabulary between two micro theories at all. Thus, there are cases in which local knowledge bases aren't able to exchange something due to the lack of a shared vocabulary. But even with such a vocabulary, merging of knowledge can fail due to logical contradictions between the sources.

I claim that the micro theory models the nature of knowledge in the real world. Every moment, knowledge is created in somebodies mind. Of course, it is created with a huge amount of background knowledge. But this background

Maicher, L.; Garshol, L. M. (Eds.): Linked Topic Maps. Fifth International Conference on Topic Maps Research and Applications, TMRA 2009 Leipzig, Germany, November 12–13, 2009 Revised Selected Papers. Leipziger Beiträge zur Informatik. ISBN 978-3-941608-06-1

knowledge isn't *any* knowledge in the world. Any new knowledge is created based on individual experiences and from knowledge that is borrowed, received or whatever from (*a limited number* of) sources. Each knowledge is created – so to say – on an island. Islands can communicate. They can cooperate. But they remain islands – local theories. They can evolve independently without restrictions even if they cooperate.

Topic Grid is based on micro theories, Topic Maps and the P2P paradigm. The Topic Grid (TG) has a single entity: The TG peer. A TG peer is software with a single owner. The owner can be a person or a group of person, e.g. developer team, company etc. A group can recursively contain other groups. A TG peer has a local knowledge base. Peers can communicate directly.

The TG system is an application created on top of the Shark framework (e.g. [4]). A TG peer *is a* Shark peer[1]. Each TG peer hosts it's own local knowledge base and can communicate via the Shark Knowledge Exchange Protocol (KEP). There are TG implementations on J2SE and Android.

Topic Grid is a semantic P2P system for knowledge exchange. It uses the concept of Distributed Context Spaces (DCS) which is based on Topic Maps and the basis of the Shark data model. In the next sections, the concept of DCS will be motivated and introduced. Knowledge exchange is the key feature of Topic Grid. Strategies of learning in distributed micro theories based on DCS are introduced. A summary and outlook will be at the end of this paper.

2 Knowledge

Knowledge is something that helps to solve a problem. This understanding of knowledge is common in knowledge management (KM) and differs from definitions in artificial intelligence. AI knowledge is often defined by its *structure* and not its content or usage. Each AI ontology is knowledge – per definition. I follow the understanding of knowledge management. Knowledge can be represented as ontology but an ontology isn't necessarily knowledge.

Knowledge is something that helps to solve a problem. What does it mean in practise? What is the *thing* that helps to solve a problem? It is a e.g. text with pictures, a video explaining something or maybe a audio file with explanations about an interesting topic. The help is usually stored in a file of an appropriate format (audio, text, video, picture or multimedia). We call it information resource (IR) in Topic Maps. An IR is attached to a topic in TM. The topic describes a thing of the real world (a subject) to which this IR fits.

There might be e.g. the topic of the *Ontopia software system.* An appropriate information is e.g. how to get the source and to run the system. From another point of view we could say that the topic is the **context** in which this information is valid or useful. This interpretation fits to the knowledge definition of KM.

Context is just another *name* for the technical concept topic in a specific use case: It underlines the special usage of a topic as holder of information in a KM

[1] in the of object-oriented sense: a TG peer is special Shark peer

system. A topic map can e.g. also be used as basis for an AI that plays chess. In this case a topic would *not* be a context. It would be (maybe) part of a decision tree or something else.

Thus, a network of topics form a net of context to which information resources are attached. Finding knowledge is process with two steps: a) find the context and b) find information.

3 Context and Interest

Let's have a deeper look into context. This shall be done by an example. Let's assume we are about creating a knowledge base about tourist information. This is a KM system because it helps to travellers.

Assume, we have a topic *Oslo* which represents the city of Oslo, capital of Norway. We can attach different information resource e.g. about history, restaurants etc. Furthermore we won't forget to use at least two basenames, (Oslo and Christiania) in different scopes.

Finding information about Oslo is pretty simple: We look for the appropriate context (topic Oslo) and get information. Moreover, we could define a couple of occurrence types the distinguish different types of information, e.g. restaurant information and historical data etc. This approach works very well in the good old Web 1.0. It does in Web 2.0 applications.

Web 2.0 sums up WWW based applications that allows users to change the content. Imagine our tourist information application goes Web 2.0. Now, any (registered) user can add information.

Information will be different. Without any doubt it makes a huge difference if a well-educated historian writes an article about the Oslo harbour or if a young foreign school boy makes notes about the restaurant in the same area. I don't talk quality. For other youngsters, information from their comrades might be more interesting than this serious historical stuff from our historian. And vice verse of course.

Web 2.0 applications offer information from a different authors. Authors have different background knowledge, different interests and different meanings about the world. What does an information seeker need to find appropriate information (knowledge) in a Web 2.0 application? She/he needs to know a topic (e.g. Oslo). But she/he must also declare in some way her/his background knowledge and interests. Our young information seeker isn't interested in historical documents. He probably wants to get a good beer.

4 Modelling the Context Space

How can these facts be put into a Topic Map. There are different possible approaches:

- We define a topic (Oslo) and two topics (youngster and historian). The later two topics are used as occurrence types. Information of e.g. our youngster will referenced with the occurrence type youngster.

Unfortunately, this approach does not scale. Imagine, the concept of time shall be added later. We would need occurrence types representing information of an historian which are valid in the 17th century, in the 18th century etc. pp. This leads to a fast growing number of types.
– We could define a topic Oslo and related topics representing Oslo in different times. There can be e.g. a topic *Oslo in the 17th century*. This wouldn't solve any problem. There would be the same number of concepts than in the first approach.

This problem is obviously. E.g. Authorship and topics[2] are independent features. They can be seen as a two dimensional matrix. In Shark, we say that they form a two dimensional *context space*. Each dimension is an ontology e.g. represented by a Topic Map. A concept inside the map is called *context point*.

The following figure illustrates two context points in a two dimensional context space. Both represent the topic Oslo. One is from our youngster, the second one from our historian. Information can be added to each context point.

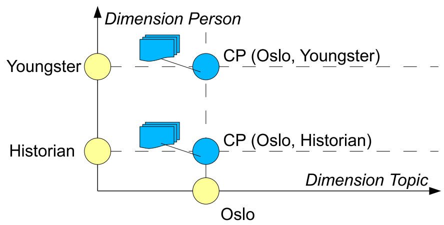

Fig. 1. Two two-dimensional context points

The context space model scales. A new independent facet of knowledge can be modelled by a new dimension. The context space itself can be modelled by a Topic Map. A context point can be represented by a topic. The identity of the context point is defined by n parameters (n is the number of dimensions of the actual context space). A context point can have an arbitrary number of attached information (occurrences if Topic Maps are used).

More specific, each context point has n sets of subject identifier (SI). Each set describes a subject on a dimension. One set describes e.g. Oslo in our example. The other set describes the author.

Actually, there are *three ontologies* in our example: One ontology for each dimensions and the third for the context points. All ontologies together represent the context space.

Topic Grid follows the concept of micro theories. Each TG peer has its own context space. Context points hold the actual *content* of the peer. The dimensions are the vocabulary which are used for knowledge exchange.

[2] in the sense of our sample application and not as Topic Map topic

Thus, the context point are necessarily locally stored on each peer. It is the *local knowledge, the local theory* of the peer. The dimension ontologies can also be stored locally on the peers. They can even be stored in a single Topic Map as already described in [5].

Dimension ontologies can also be external ontologies. A context point can reference e.g. a social network webpage like LinkedIn to identify an author, It can use a Website like Subj3ct to define its topic. In this case, the external ontologies are used as vocabulary. There is at least one vocabulary for each dimension.

Moreover, a peer can use multiple vocabularies for each dimensions. A context point can use an arbitrary number of SI to define a subject on a dimension. The topic Oslo can be identified by a reference to Wikipedia but also to a page on Subj3ct. Both, Wikipedia and Subj3ct play the role of a vocabulary. Now, our context point became bilingual on the topic dimension.

5 Topic Grid

Topic Grid (TG) is a P2P system based on the Shark framework. A TG user can define his/her interests. An interest comprises following information: A set of concepts on each dimension (called *anchor*, list of potential communication partners, two flags declaring if whether knowledge is to be sent or retrieved (can be both), fragmentation parameter, time which describes how long a interest is valid (can be for ever), list of addresses over which the peer is getting data (e.g. URL for a webserver, Bluetooth service, TCP address).

Fragmentation parameter are non-negative value *depth*, a set of *allowed association types* and a set of *forbidden association types.*

Based on these parameters, a *context map* is calculated which is a n-dimensional context space but without content (context points). This process starts be finding the anchor in the local n dimensional knowledge base. It copies these anchor to the context map. It copies also any concepts which are reachable via allowed and not forbidden association types over a distance not longer than depth. We call this process *fragmentation.* The result is the *local context map.*

If two TG peers meets they exchange their context maps with the other information of the interest. An intersection of the received and the local context map is created. If the intersection isn't empty and the other interest-parameter fit – knowledge is extracted from the local knowledge base. This knowledge is send from the peer with a sending interest to a peer with a receiving interest.

Figure 2 illustrates parts of this process. The TG owner on the left side declares anchor which are concepts in his/her local knowledge base. It also declares a fragmentation depth of 1. A context map can now be fragmented which contains few concepts than the original context space. The same process take place in another peer.

TG peers can now exchange their context maps and calculate a kind of intersection. this process is similar of the fragmentation and uses the local fragmentation parameter. Thus, the result can differ on both sides. A possible result

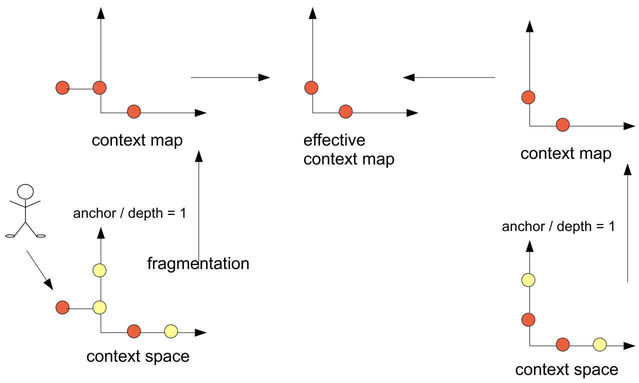

Fig. 2. Generating Context Maps

is illustrated in the middle of the figure. We call it the *effective* context map because it can be used to extract context points from the local knowledge base and send it to the remote peer. This final step isn't illustrated in figure 2.

6 Status and Outlook

Major parts of this theory are already implemented in the Shark framework. Based on the framework there is a general system called Shark Grid. Currently, it runs on Android and J2SE. It is written in Java 1.3 and thus compatible with J2ME. It should run on Symbian devices.

Topic Grid is a special Shark Grid edition which uses Topic Maps as persistent storage for the context spaces. We already used tinyTIM in the framework but Topic Maps and not fully integrated. We will have a booth at CeBIT 2010 and show several mobile spontaneous but also Internet based semantic P2P applications.

References

1. Sowa, J.F.: Knowledge Representation: logical, philosophical and computational foundations. Brooks and Cole (Thomson Learning) (2000)
2. Garshol, L.M., Barta, R., Entry, R.: Topic maps query language. working draft, ISO/IEC JTC 1/SC34 (02 2005)
3. W3C: Sparql query language for rdf (2008)
4. Schwotzer, T.: Building context aware p2p systems with the shark framework. In: Proceedings of the International Conference on Topic Map Research and Applications (TMRA'08), Leipzig, Springer (2008)
5. Schwotzer, T.: Modelling distributed knowledge management systems with topic maps. In: Proceedings of the 4th International Conference on Knowledge Management (I-KNOW '04), Graz. (July 2004) 52–59