DeepaMehta—A Semantic Desktop

Jörg Richter[1] and Max Völkel[2] and Heiko Haller[2]

[1] Co-Founder and Lead Architect of DeepaMehta, Berlin, Germany
jri@freenet.de
http://www.deepamehta.de

[2] AIFB, University of Karlsruhe, Germany
{mvo,hha}@aifb.uni-karlsruhe.de,
http://www.aifb.uni-karlsruhe.de/WBS

Abstract. DeepaMehta is an open source semantic desktop application based on the topic maps standard. It's conceptualisation and especially the innovative graph-based user interface have been guided by findings in cognitive psychology in order to provide a cognitively adequate working environment for knowledge workers of all kind. DeepaMehta aims to evolve nowadays' separated desktop applications into an integrated workspace enabling the user to organize, describe and relate information objects like text notes, external documents and media, browse the web, search databases and create semantic networks—all this in one seamless, semantics-enabled desktop environment.

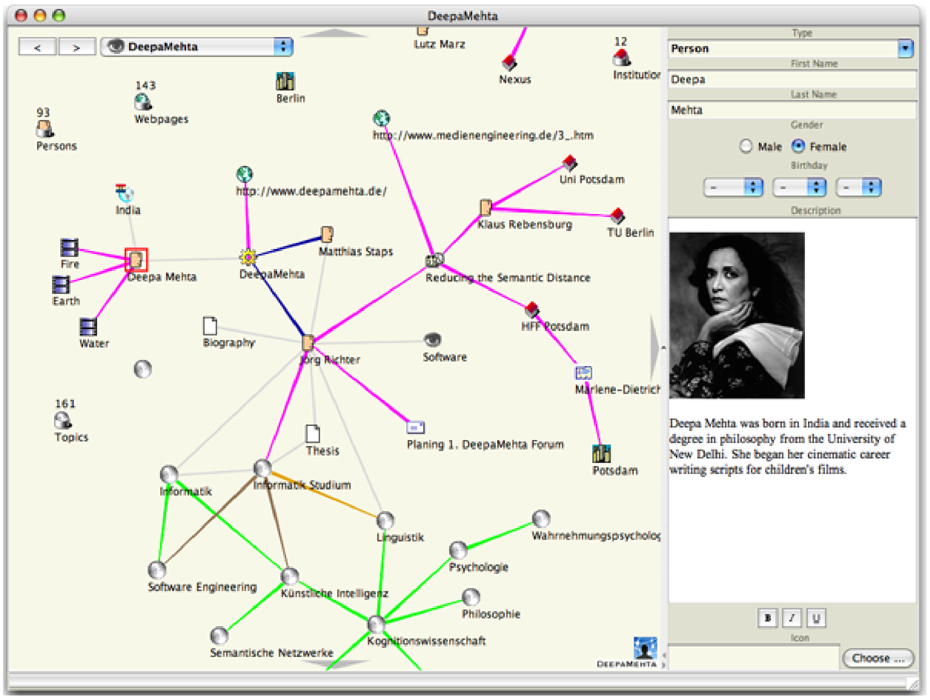
1 Introduction

In this paper we present the *networked semantic desktop*[1] *DeepaMehta*, which is based on the topic map paradigm.

DeepaMehta is a personal knowledge management (PKM) tool, that integrates information objects of all kinds (external application files, aswell as personal notes, e-mail, web pages et c.) into a coherent and intuitive user environment. Many insights of modern cognitive psychology have been taken into account for the conceptualisation and design of the DeepaMehta user interface (Fig. 1). Today, DeepaMehta is already successfully running in several commercial projects.

It is DeepaMehta's goal, to have machine-interpretability together with cognitive adequacy or, in other words, to combine semantic knowledge management with ease of use.

Outline of this paper: First, we explain the psychological background of PKM (Section 2). Then the conceptual design is described (Section 3), which enables cognitively adequate semantic authoring. Also the service oriented architecture and the extensible type system are sketched there. The implementation is briefly described (Section 4) before goning into a detailed evaluation with respect to psychological criteria (Section 5). At the end you will find some related work (Section 6) and our conclusions (Section 7).



Type: Person
First Name: Deepa
Last Name: Mehta
Gender: Male [✓ Female]
Birthday: - - -
Description: Deepa Mehta was born in India and received a degree in philosophy from the University of New Delhi. She began her cinematic career writing scripts for children's films.
Icon

Fig. 1. A Topic Map (typical DeepaMehta working screen)

2 Basics

2.1 Psychological Background to Personal Knowledge Management

Knowledge Representation According to some of the most common definitions of knowledge, the constituting attribute of knowledge as opposed to mere information is, that it's meaning is being understood—something that (currently) only occurs in human minds. Whether one shares this definition of knowledge or not, it should be easy to agree that information or knowledge can only be of use, when it is either formalized in a machine interpretable format (like in ontologies or the vision of the Semantic Web [2]) or accessible and understandable by human beings. Speaking in terms of the above definition, machines can not store knowledge itself. What they actually can store and manage are so-called *cues*: items (pictures, text, media, ...) that help the user to reconstruct the original knowledge in his mind. It should thus be the main goal of any knowledge management software, to facilitate the creation, externalisation, and (re)construction of knowledge [3].

In a human mind, knowledge is not simply stored away and recalled like a database dump. Instead it is abstracted and different aspects of knowledge (like declarable facts, procedures, spatial context, visual impressions) are stored in different places in the brain [4], and this storage is never perfect. Knowledge retrieval from memory is actually a process of reconstruction [5].

Since there is evidence, that conceptual human knowledge is actually stored in an *associative* way comparable to semantic networks [6], it stands to reason to provide the knowledge worker with a UI, where the contents are displayed, managed, created, and refined in such an associative manner (i. e. items together with their relations to other items) that enables the construction of semantic networks—like *concept maps* [7] do, as well as their more formal derivatives called *knowledge maps*[8].

Complex Problem Solving One goal of personal knowledge management is the facilitation of problem solving tasks. One of the main problem in complex problem solving is understanding the way things are interconnected [9]. Thus it is very helpful to have a way to adequately represent the complex relational structure of a problem.

To solve complex problems, three things are most relevant: having access to problem-related knowledge, reducing complexity [10] and avoiding cognitive overhead [11]. Accessing problem-related knowledge(-cues) can be done by searching or browsing the web or local resources. The main method to reduce a problem's complexity to a manageable level is the abstraction of the problem to clusters [10] thus reducing the number of items the mind has do deal with simultaneously. This is crucial, because the number of items we can consciously deal with at a time is limited to "seven plus or minus two" [12]. Because of this limitation, it can be crucial to minimize all *Cognitive overhead*, which is "the additional effort and concentration necessary to maintain several tasks or trails at one time" [11]. Because our working memory and thus capacity for conscious processing are so limited, we should avoid wasting it to secondary tasks like worrying about saving files, dealing with layout and formatting or regaining orientation in the information environment while researching or writing the actual content.

However the good news is: While every task that needs conscious processing interferes with other such tasks, automated activities don't [13].

Mapping Techniques[3] One of these highly automated activities is our natural sense of spatial orientation. This sense has been optimized by evolution neither for texts nor hypertexts, but it easily distinguishes spatial positions and layouts—also in a plane. Managing knowledge resources with the help of visualisation techniques has the advantage that we can use our natural sense of orientation to gain orientation in our *knowledge space.*

Furthermore, research on *dual coding* effects has shown that graphical images leave richer traces in human memory than mere text in general [14,15].

[3] In this article the Term "Mapping" is used as coined in the domain of instructional psychology, i. e. in the sense of *creating and using visual knowledge representationes* called "maps" like mind-maps, concept maps etc.

Apart from that especially structural depictions like concept maps facilitate the generation of mental models because they do not only already contain the bare propositions (S-P-O triples), but also the meta-structure: The map's spatial layout represents the structure of it's contents [16]. Especially in situations where we are trying to gain overview on a topic, grasping the content's inherent structure or trying to outline a subject, visual mapping techniques can be of great help.

For an overview of research in cognitive and instructional psychology on various visual mapping techniques their versatile applications and manifold positive effects on learning, comprehension and problem solving (knowledge generation, acquisition, retention and use, see [8,17,18,19].

2.2 Topic Maps

Topic Maps are a human-oriented approach to encode knowledge. The "TAO" [4] of topic maps consists of *topics*, *associations* and *occurrences*. In the semantic web, this relates to *resources*, *relations* and *instances*. For readers unfamiliar with either RDF or topic maps, here is a short introduction:

- A *topic map* is a semantic network, consisting of topics, associations between them and occurrences of topics.
- A *topic* denotes just about anything that people can talk about. A topic has an ID and can have one ore more names.
- A topic can have one ore more *topic types*; topic types are topics.
- An *association* has an ID and relates two or more topics. Each end of an association has a type. An association can also have a type.
- An *occurrence* denotes an instance of a topic. In a topic map occurrences are external to a topic map and are addressed through identifiers (e. g. URLs).

Most types are topics themselves, thus a topic map is it's own meta-model. Additionally topic maps have well-thought out concepts for topic map merging, contexts of topics and associations and so on.

There now exists an ISO standard [20] and a documentation of the formal data model [21] building on a more concise ISWC paper [22]. The relation of RDF and topic maps is described in detail in [23], of which a good summary can be found in [24]. Topics maps stem from a bibliographic domain and have grown to all kinds applications, even groupware, as explained in [25].

3 Design

DeepaMehta is a service oriented application framework with a data model based on topic maps and a UI that renders them as a graph, similar to concept maps [7] (see Fig. 1). Information of any kind as well as the relations between information

[4] Term coined by the introduction at `http://www.ontopia.net/topicmaps/materials/tao.html`

items can be displayed and edited in the same space. The user is no longer confronted with files and programs. There are no overlapping windows, no menu bars and no dialog boxes. Topic Maps are individual views on interconnected content and they may evolve on their own, along as the user works with the system.

3.1 User Interface

The design of the DeepaMehta user interface has been guided by findings in cognitive psychology (c. f. Sec. 2). One of the most obvious problems in current desktop user interfaces is that of context switching. Users currently have to switch applications for every sub-task. Every switch presents a different interface to the user.

Stable Views in DeepaMehta let the user focus on the task itself, without leaving the work-context: In one and the same view the user can read an e-mail, link it to an existing topic, attach a note to it, search for related media, save the search results, make semantic statements and spatially arrange all these items on the screen—simply scattered or in a graph. And he will later always find his workspace exactly as he left it. Today's desktop UIs are application-oriented, not data- or task-oriented.

Constructive Browsing: Browsing the web is easy. Figuring out later where one has been is not that easy—because 1) a browser history is merely time-based and 2) offers no means to attach any kind of additional information. Even worse, 3) after a fixed time interval, the history is usually erased automatically. If not, 4) it grows so large that it becomes virtually impossible to handle. Additionally 5) the browser history contains no information about other resources accessed than web pages. Bookmarks sometimes do offer an annotation feature, but 6) never a semantic one and problems (4) and (5) still remain. Maintaining bookmarks in a way they remain usable requires quite some additional effort compared to mere web browsing. DeepaMehta offers *constructive browsing* as a solution: Each resource visited (be it a web page or something else) is automatically represented as a topic in the current workspace. Each new topic is placed right next to the preceding one. The user can conveniently move these newly created topics around to other places in his workspace, which is always visible. Now, surfing the web or accessing other resources automatically creates a map of viewed objects. Even searches and search results are represented in the same consistent fashion. This spatially arranged map visualises a work process better then a list of named URLs and it is persistent, automatically saved and fully navigable. Furthermore a snapshot of every page viewed is taken and automatically stored in the repository, so it can be referred to and annotated, even when it is offline or no longer on the web.

3.2 The DeepaMehta Type System

Traditional applications have a fixed set of objects they deal with: documents, addresses, tasks, images et c.. The DeepaMehta type system is extensible and closely corresponds with user interface objects.

Meta Modelling (brought over from the topic map concepts) is a method DeepaMehta offers for it's type system where topic and relation types are themselves topics. DeepaMehta exposes the full power of meta modelling to the user: He can *construct* new topic and relation types on the fly in the same interface where they are *used*. New topic types can be used instantly, even in collaborative settings. DeepaMehta does not use topic maps exactly as they were defined.

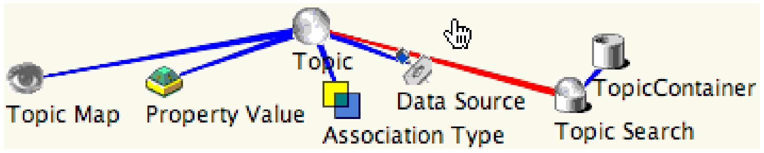


Fig. 2. DeepaMetha showing its building blocks of the type system

It uses only a subset. DeepaMehta currently makes no use of "scope", "theme", "identity attribute", "public subject indicator" or "facet". Other types ("topic characteristic", "occurrence", "occurrence role type" and "association role type") are not defined as in topic maps but using DeepaMehta's type system.

There is a set of core topics, which form the basis of DeepaMehta's type system:

topic The most basic and universal type is a *topic*. A topic can be thought of as a symbol, referring to anything the user has in mind.

search Each type has a corresponding search type. This allows searches to have custom behavior.

association There are four predefined association types: A *relation* is a directed association between types. It assigns cardinality constraints (one or many) to the associated instances. A *derivation* of a topic inherits the assigned property types and the behavior. A *composition* assigns properties to topics and values to properties. *Aggregation* associates a specific search type to every type. An association corresponds to an owl:ObjectProperty.

property A property is a named attribute of a topic. Each property defines the property values it can take. Properties can also have default values. It's roughly comparable to an owl:DataTypeProperty.

property value A property value is either just a text string or a more specialised type like gender or date. For each property type DeepaMehta has built-in views for data display and data entry.

data source Data cannot only be entered by a user, instead it may come from sources of all kinds. DeepaMehta has built-in adapters for databases and LDAP.

search The special topic search is the mother of all searches. Each topic has a corresponding search type. *Topic container* is the superclass for searches.

map A *map* is a container for topics. It's the space in which a topic has a location. Topics can occur in multiple maps. A map is—you guessed it—a topic itself.

workspace A workspace contains a set of maps. Each user is member of at least one workspace. Other users in the same workspace see the same topic types and content. More about collaboration can be found in Sec. 3.4.

As the users (or a defined administrator for a given workspace) can create or customize the topic and association types,

3.3 Service Oriented Architecture

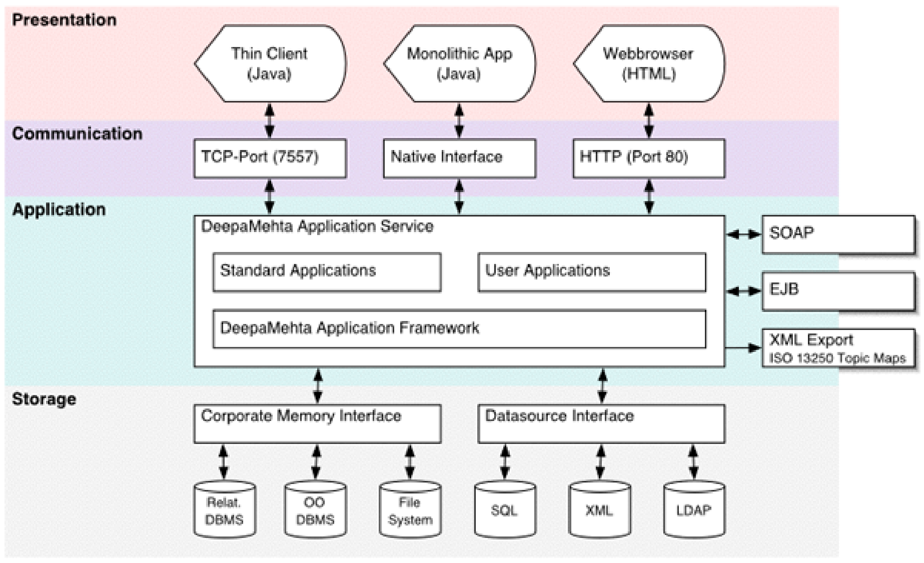


Fig. 3. DeepaMehta Architecture

DeepaMehta has a layered, service oriented architecture, typical for modern application servers (c. f. Fig. 3). The main layer is the *application layer*, which sits in the middle. It offers various ways for the *presentation layer* to communicate with it via the *communication layer* (API, XTM export, SOAP, EJB). The built-in web server offers an out-of-the box user interface which runs in almost any browser. The *storage layer* manages the corporate memory, which holds all topics and their data either in a relational database or simply in the file system.

The most notable part of DeepaMehta's architecture is probably it's topic-centric thin-client API. Information objects (topics) in DeepaMehta are not only graphical objects, but active data objects: Each topic type can be provided with its own java class, to give it unique functionality. How they relate to and inherit from each other can be configured in the seamless DeepaMehta user interface. These Java classes can opt to override the default behavior to react to all kinds of events[5] from topic creation, spatial movement and context menu display up to permission control and topic publishing in a workspace. To cut a long story short: The behavior of topics can be altered in almost any imaginable way.

The DeepaMehta thin client acts as a generic topic map user interface. It retrieves topics, their icons, position, relations and commands from the server and handles only user interaction. As all business logic resides on the server, completely different user interfaces can be implemented easily, operating on the same data—even simultaneously. Notably the thin client can also receive push commands from the server e. g. with instructions to show, hide or move topics.

The **integrated web interface** makes it possible to use DeepaMehta with a standard browser. Upon each page-reload all new types and topics are available.

3.4 Collaboration

All Topics and associations are stored in a "corporate memory" on the server and can be used by several users collaboratively. A shared workspace stores it's users as topics using the association type "membership". Like this, user administration can be done by admin-users within the same environment as everything else.

4 Implementation

The DeepaMehta thin client faces some interaction challenges: It must always be up-to-date and thus talk to the server. And it should provide high reactivity, which disallows time-consuming server-calls. To deal with this, the thin client uses four socket connection threads. One socket stays open to receive server push commands; another socket is used for normal user interaction such as topic movement on the screen. A third socket handles file transfers, in order not to block the other sockets. A fourth socket thread, running with high priority, is the type definition synchronisation. It fetches topic behavior definition and even the icon from the server. This has to run extremely smooth in order to provide a good user interface experience.

Internally, DeepaMehta uses *Jakarta Tomcat* as it's JSP and Servlet engine. To the outside world, it can export the full content in XTM, SVG and even PDF format.

[5] Exhaustive list at `http://www.deepamehta.de/docs/apidocs/de/deepamehta/topics/LiveTopic.html`

5 Evaluation

5.1 Technical Evaluation

The DeepaMehta architecture defines a new application model and gives developers a framework to design DeepaMehta-applications. Such applications are easy to maintain and update as the business logic resides on the server. Also a range of interaction front-ends is offered: a thin client, web front-end and even a PDA interface[6].

For dynamic web sites, a flexible framework with full access to all DeepaMehta services is available. The logic resides in a controller and the web pages are generated using JSP pages or XSL style sheets.

The thin client framework provides a solid base for many kinds of interaction clients.

5.2 User Interface

DeepaMehta's UI is still in a prototype state and a more up-to-date one is planned. However because it takes a very promising approach, we checked it against a set of criteria set up to evaluate visual mapping tools for personal knowledge management from a cognitive psychological point of view [19].

Freely Placing information item anywhere on the canvas is possible in DeepaMehta (unlike in some wide-spread tools like *TheBrain*[7] or some of the MindMapping-like tools).

Free Relations Stating relations between items in DeepaMehta is possible in all degrees of formality: *unlinked nodes*,*unlabeled links*,*labelled links* and semantically *typed links*, even with heritage of properties.

Annotations, whether plain text or semantic, can be added in various ways: either by using the content-pane every topic has, or right in the maps e.g. by using a topic type 'annotation'.

Facility of Inspection / Macro-Structure As soon as contents become more complex, it becomes important to help recognition of it's macro structure, so the user can grasp the overall structure first, before drilling down into the details [26] . The most basic and useful way, to deal with complexity and clarify the macro structure of a domain, is to use clustering. [5,12] It should thus be supported by the visualisation approach, to *group* objects together. Apart from leaving the user free to simply spatially gather items together to form visual clusters, DeepaMehta does not currently support grouping. Extending it's visual mapping paradigm to support collections of items, could maybe increase the readability

[6] Download at `http://www.deepamehta.de/docs/deepamobil.html`
[7] `http://www.thebrain.com/`

of more complex maps. DeepaMehta however does allow chunking in a way by supporting *sub-maps*: Since a topic in a map can contain a topic-map by itself, parts or details of a complex scenario can be modelled separately in an own topic map, that can be referenced as a whole through the containing topic.

Hyperlinks to URLs or local files enable using DeepaMehta for management of external knowledge resources.

Easy Editing of maps should be possible with minimum cognitive overhead (see 2). Standard tasks like fast note taking (brainstorming) and refining existing content should thus be possible with the least possible disturbance. To enable *easy brainstorming*, it should be possible, do add new items with one single user action (mouse click, double click or key stroke), and without having to worry about the layout or anything else than the mere content. In DeepaMehta, currently, this is more cumbersome and creating new items without leaving the keyboard, is not possible at all. However both of these UI implementation details can be easily improved. *Insertion* of new topics into an existing map is possible, but, as with most concept map like tools, the problem is, that the user often has to move around existing items first, in oder to create space for the new ones. This can make a tool rather awkward for quick note-taking. A visual mapping UI with nice spatial layout algorithms would increase DeepaMehta's usability.

Integration of Detail and Context: Navigating complex information environments often leads to a loss of orientation, because when surfing hypermedia or drilling down into detailed views of a matter, in most environments context information gets lost [11]. An information environment built to manage complex or large amounts of information should therefore offer some orientation-aids of facilitate the mental integration of details and context. This can happen in many different ways [27].

Currently neither zooming or any other overview modes or levels-of-detail techniques are implemented in DeepaMehta so the use of large maps requires panning and may become somehow complicated. DeepaMehta's approach is different: Because every item in a map also exists in the background repository, existing maps do not have to be reused, even if they contain relevant items. For a new task, a new, empty map is started, and by search and browsing only those items are fetched into the new map, that are needed.

In an other field of detail-and-context integration however, DeepaMehta is very interesting: When using DeepaMehta to browse the web or local content, the user has both the actual content and the context information in one view. Like this he can navigate classically by following hyperlinks inside the web pages or read and write content but at the same time have all related information items at his fingertips from the meta perspective, that combines the PKM perspective with a structured browsing history.

DeepaMehta's UI is very lean and only shows control elements when they are needed. Although compared to production status mapping tools like *Mind-*

Manager[8] DeepaMehta's UI is not really fluent to use, but it is still easier to use than comparably flexible and feature-rich tools or research prototypes.

Conclusion DeepaMehta's UI and interaction paradigm takes a consequent approach of minimalist design where only relevant controls are shown. As this differs from common interfaces, it requires some initial time to get acquainted. Although still somewhat clumsy to use, it provides a rich and powerful basis of innovative UI concepts well-thought-through without a bloated interface. Providing zooming capability and a grouping feature would surely increase it's utility especially for the use of larger and more complex maps.

User studies could be valuable to further identify strengths and weaknesses of this innovative approach and distinguish key issues on the way to make the UI more easy and intuitive to use.

5.3 Usage in Real-Life Projects

DeepaMehta has been successfully deployed in several commercial sites in a variety of domains. Among these are two eLearning projects, a geographic information system about city quarters "Kiezatlas[9]" (4) which has won a Best Practice Award "we make IT – Berlin.Brandenburg". For consultants, a competence analysis tool was implemented (5). Another DeepaMehta application acts as an information management system for modern and contemporary artwork[10].

6 Related Work

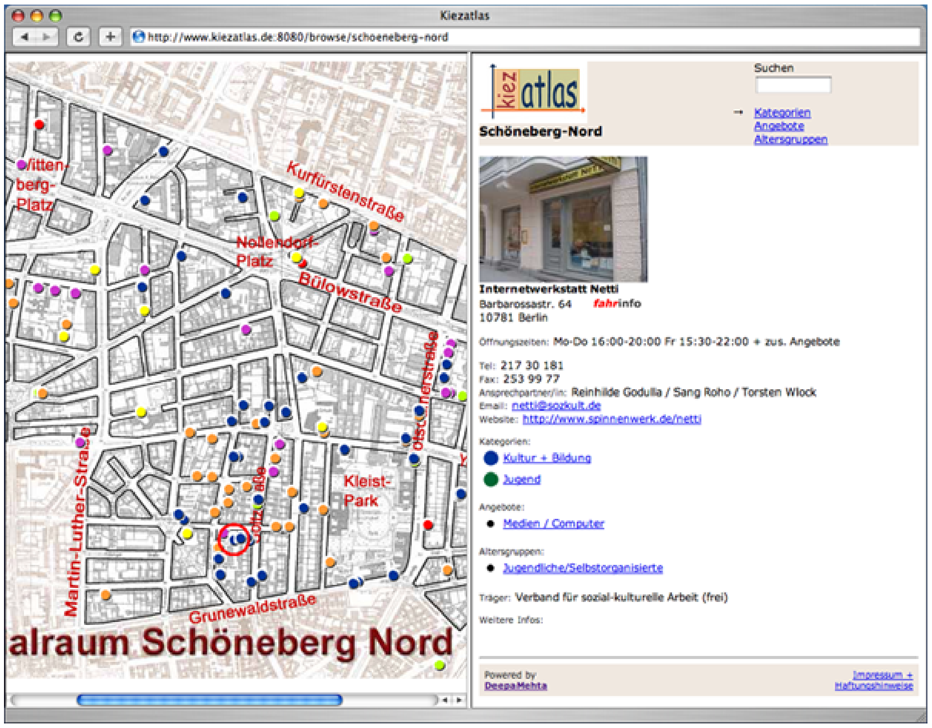
DeepaMehta's vision and implementation are broad in scope and thus cross quite a number of domains.

The node-and-link type of DeepaMehta's visualisation is inspired by the *concept mapping* approach [7], that has proven successful in improving learning in many different scenarios (see [17] for an overview of studies). There has also been a lot of research on a semi-formalized derivative of concept mapping, dubbed *knowledge mapping*, that uses fixed sets of typed relations [8].

Existing concept mapping tools like *cMap Tools* [28] do not support typed nodes and relations nor do they integrate or plug into web browsers or offer anything comparable to DeepaMehta's background repository.

DeepaMehta as a user interface framework for semantic content can be compared to *Haystack* [29]. Haystack offers excellent consistent drag'n'drop and context menu support throughout the application. For visualisation, it uses box-shaped proxy objects which are mainly composed of forms. In DeepaMehta, we distinguish between the content of an object, which is displayed in the property pane on the right, and the context of an object, consisting of the object's relations to other objects, which is graphically rendered in the left part.

[8] `http://mindjet.com`
[9] `http://www.kiezatlas.de`
[10] `http://artfacts.net`



Schöneberg-Nord

Internetwerkstatt Netti
Barbarossastr. 64
10781 Berlin

Öffnungszeiten: Mo-Do 16:00-20:00 Fr 15:30-22:00 + zus. Angebote

Tel: 217 30 181
Fax: 253 99 77
Ansprechpartner/in: Reinhilde Godulla / Sang Roho / Torsten Wlock
Email: netti@sozkult.de
Website: http://www.spinnenwerk.de/netti

Kategorien:
- Kultur + Bildung
- Jugend

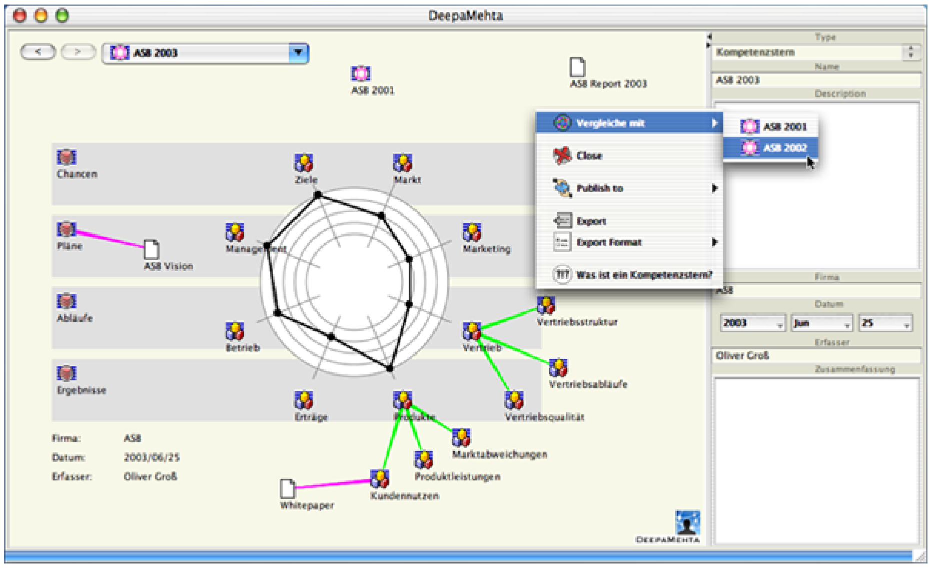
Angebote:
- Medien / Computer

Altersgruppen:
- Jugendliche/Selbstorganisierte

Träger: Verband für sozial-kulturelle Arbeit (frei)

Weitere Infos:

Fig. 4. "Kiez Atlas", a geographic information system built with DeepaMehta web interface



Firma: ASB
Datum: 2003/06/25
Erfasser: Oliver Groß

Type: Kompetenzstern
Name: ASB 2003
Description:
Firma: ASB
Datum: 2003 Jun 25
Erfasser: Oliver Groß
Zusammenfassung:

Fig. 5. Competence analysis tool, built with DeepaMehta thin client

The open source PIM system *Chandler* offers similar rich support for semantics and interaction as Haystack and also lacks the emphasis on context and relation visualisation.

Gnowsis [30], another semantic desktop project, focuses on desktop application integration. It enables linking on the desktop object level (individual e-mails or contacts), which has not been possible before. It uses RDF adaptors to achive unification of knowledge objects. The Gnowsis approach lives in parallel to existing desktop applications while DeepaMehta tries to replace them—to some extend. Both approaches meet somewhere in the middle and future will have to tell us where the sweet spot between integration and specialisation lies.

7 Conclusion

Wether DeepaMehta will succeed in replacing nowadays standard applications or not, in any case it introduces and combines several quite promising innovative approaches to personal knowledge management and user interaction design. And it has already proven it's utility in several production-status commercial projects.

As is common for prototypes, usability is still improvable, however it becomes clear that DeepaMehta bears a high potential, combining the advantages of visual mapping techniques and semantically specified topic maps. Furthermore it offers a solid and web service enabled back-end for collaborative creation and use of knowledge bases ranging from informal collections of notes to fully fledged semantic networks.

Acknowledgments: Research reported in this paper has been partially financed by the EU in the IST-2003-507482 project Knowledge Web [11] and is supported by the German Federal Ministry of Education and Research (BMBF) under the SmartWeb project. We would like to thank our colleagues for fruitful discussions.

References

1. Decker, S., Frank, M.R.: The networked semantic desktop. In: WWW Workshop on Application Design, Development and Implementation Issues in the Semantic Web. (2004)
2. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American (2001)
3. Nonaka, I., Takeuchi, H.: The knowledge-creating company. Harvard Business Review (1991) 97–130
4. Kolb, B., Wishaw, I.Q.: Fundamentals of Human Neuropsychology. W H Freeman (2003)
5. Anderson, J.R.: Cognitive Psychology and Its Implications. 6th edn. Worth Publishers (2005)

[11] see `http://knowledgeweb.semanticweb.org`

6. Quilian, M.R.: Semantic Memory. In: M. Minski (ed.). Semantic Information Processing. MIT Press, Cambridge, MA (1968)
7. Novak, J.D., Gowin, D.B.: Learning how to learn. Cambridge University Press, New York (1984) For a crisp overview on "The Theory Underlying Concept Maps and How To Construct Them" by J.D. Novak, see http://cmap.coginst.uwf.edu/info/.
8. O'Donnell, A.M., Dansereau, D.F., Hall, R.: Knowledge Maps as Scaffolds for Cognitive Processing. Educational Psychology Review **14** (2002)
9. Vester, F.: Die Kunst vernetzt zu denken. Dtv (2002) Report to the Club of Rome.
10. Dörner, D.: Die Logik des Mißlingens. Strategisches Denken in komplexen Situationen. Rowohlt Tb. (2003)
11. Conklin, J.: Hypertext: an introduction and survey. Computer **20** (1987) 17–41
12. Miller, G.A.: The magical number seven, plus or minus two: Some limits on our capacity for processing information. Psychological Review **63** (1956) 81–97
13. Shiffrin, R.M., Schneider, W.: Controlled and automatic human information processing: Ii – perceptual learning, automatic attending, and a general theory. Psychological Review (1977) 127–190
14. Paivio, A.: Imagery and verbal processes. Holt, Rinehart and Winston (1971)
15. Paivio, A.: Mental Representations: A Dual Coding Approach (Oxford Psychology Series, 9). Oxford University Press (1990)
16. Schnotz, W. In: Wissenserwerb mit Texten, Bildern und Diagrammen. third, completely revised edn. Belz, PVU, Weinheim (2002)
17. Jonassen, D.H., Beissner, K., Yacci, M.: Structural Knowledge: Techniques for Representing, Conveying and Acquiring Structural Knowledge. Lawrence Erlbaum Associates, Inc (1993)
18. Fischer, F., Gräsel, C., Kittel, A., Mandl, H.: Entwicklung und untersuchung eines computerbasierten mappingverfahrens zur strukturierung komplexer information. research report ("Forschungsbericht") 57, Ludwig-Maximilians-Universität, Lehrstuhl für Empirische Pädagogik und Pädagogosche Psychologie, München (1995)
19. Haller, H.: Mappingverfahren zur Wissensorganisation (2003) Knowledge Board Europe. Availlable online at http://heikohaller.de/literatur/diplomarbeit/.
20. Michel Biezunski, Martin Bryan, S.R.N.: ISO/IEC 13250:2000 Topic Maps. Technical report (1999)
21. Lars Marius Garshol, Graham Moore, J..S.: Iso/iec jtc1/sc34 topic maps (iso 13250) data model—final committee draft. Technical report (2005)
22. Auillans, P., de Mendez, P.O., Rosenstiehl, P., Vatant, B.: A formal model for topic maps. In: ISWC '02: Proceedings of the First International Semantic Web Conference on The Semantic Web, London, UK, Springer-Verlag (2002) 69–83
23. Garshol, L.M.: Living with topic maps and rdf. In: XML Europe 2003. (2005) Availlable online: http://www.ontopia.net/topicmaps/materials/tmrdf.html.
24. Pepper, S.: Ten theses on Topic Maps and RDF. Technical report (2002) Available online: http://www.ontopia.net/topicmaps/materials/rdf.html.
25. Smolnik, S., Nastansky, L.: K-discovery using topic maps to identify distributed knowledge structures in groupware-based organizational memories. In: HICSS '02: Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)-Volume 4, Washington, DC, USA, IEEE Computer Society (2002) 106.2
26. Reigeluth, C.M., Stein, F.S. In: The Elaboration Theory of Instruction. Erlbaum (1983) 335–381

27. Furnas, G.W.: Generalized fisheye views. In: Human Factors in Computing Systems CHI '86. (1986) 16–23
28. Tergan, S.O., Keller, T., eds. In: Concept Maps: Integrating Knowledge and Information Visualization. Volume 3426 of Lecture Notes in Computer Science. Springer (2005) 205+
29. Adar, E., Kargar, D., Stein, L.A.: Haystack: per-user information environments. In: CIKM '99: Proceedings of the eighth international conference on Information and knowledge management, New York, NY, USA, ACM Press (1999) 413–422
30. Sauermann, L.: The gnowsis semantic desktop for information integration. In: Wissensmanagement. (2005) 39–42