SUBMISSION OF ACCEPTED VERSION: 11.04.2008
REVISED VERSION 2: 15.09.2008, REVISED VERSION 3: 02.10.2008,
REVISED VERSION 4: 16.12.2008, REVISED VERSION 5: 15.01.2009

Conceptual Data Structures for Personal Knowledge Management

Max Völkel and Heiko Haller

Structured Abstract

Purpose – Designing a model and tools that are (a) capable of representing and handling personal knowledge in different degrees of structuredness and formalisation and (b) usable and extensible by end-users.

Design/methodology/approach – This paper presents the result of analysing literature and various data-models and formalism used to structure information on the desktop.

Findings – The unified data model (CDS) is capable of representing structures from various information tools, e. g. documents, file system, hypertext, tagging, and mind maps. The five knowledge axes of CDS are: identity, order, hierarchy, annotation and linking.

Research limitations / implications – The CDS model is based on text. Extensions for multimedia annotations have not been investigated.

Practical implications – Future PKM tools should take the mentioned shortcoming of existing PKM tools into account. Implementing the CDS model can be a way to make PKM tools interoperable.

Originality/value – This paper presents research combining cognitive psychology, personal knowledge management and semantic web technologies. The CDS model shows a way to let end-users work on different levels of granularity and different levels of formality in one environment.

Keywords personal knowledge management, semantic web, cognitive limits, semantic desktop, user interfaces, semantic authoring

Paper type Research paper

Introduction

“The most important contribution of management in the 20th century was to increase manual worker productivity fifty-fold. The most important contribution of management in the 21st century will be to increase knowledge worker productivity – hopefully by the same percentage. [...] The methods, however, are totally different from those that increased the productivity of manual workers.” *Drucker* (1999a, p. 79)

What might these methods be? The field of knowledge management investigates since 1995 (Stankosky, 2005) how people and knowledge work together. North (2007) defines *knowledge work* as work based on knowledge with an immaterial result; value creation is based on processing, generating and communicating knowledge. Polanyi (1966) makes a distinction between explicit knowledge encoded in artefacts such as books or web pages, and tacit knowledge which resides in the individual. The SECI-model of Nonaka (1994) describes knowledge transfers between humans and artefacts.

The field of knowledge management has focused on knowledge transfer between people, either via socialisation or via externalised artefacts. The high expectations of central enterprise knowledge repositories have often not been met (Braganza and Mollenkramer, 2002). The following wave of expert finders and corporate white pages focused mostly on connecting the right people and let them communicate.

Today, knowledge workers are flooded with information (Alvarado et al., 2003). The field of Personal Information Management (PIM) aims to help individuals to manage all artefacts in the *personal space of information* (PSI) which “includes all the information items that are, at least nominally, under that person’s control (but not necessarily exclusively so)” (Jones and Bruce, 2005, p. 9). Recently, more research is being focussed on the *individual* knowledge worker, establishing the field of *Personal Knowledge Management* (PKM):

- The knowledge-based organisation is no more effective than the sum of its knowledge workers (Davenport, 2005).
- One should focus on the individual and give individual users incentive and benefit before focusing on the social network (Oren, 2006).
- Schütt (2003) defines a knowledge worker based on the works of Drucker (1999b) and Taylor (1911): Simplified, workers (doing) are instructed by managers (thinking). These managers have to manage themselves. This self-managing is considered an

> important characteristic of a knowledge worker. Increasing the knowledge worker productivity has to be a company’s main goal, not storing documents in data bases. Knowledge workers have to manage themselves, because their tasks are constantly changing.

Seminal articles by Bush (1945) and Engelbart (1963) describe *tools* that allow an individual to work more efficiently and more effectively with external representations of knowledge.

In knowledge work, people are frequently confronted with two limitations of the human mind: long-term memory recall and short-term memory capacity. Limits of the long-term memory can be overcome partially with tools to help remembering or reconstructing knowledge. Human short-term memory can hold only around seven objects at a time (Miller, 1956). For user interfaces, Shneiderman (1998, p. 75) advises to “Do everything possible to free the user’s memory burden”. Interestingly, also the other limitation can be partly relieved by using external knowledge representations, e. g. by taking short notes, or drawing a diagram or mind-map that helps keep an overview over a somewhat larger set of items and quickly bring each single one into full conscience on demand. We conclude that both of these very prominent cognitive limits can be addressed by providing an adequate external knowledge representation tool.

Concerning *explicitness* of knowledge, Nonaka and Takeuchi (1995) distinguish two kinds of knowledge: explicit and tacit (internal). Later works (Despres and Chauvel, 2000; Nonaka and Konno, 1998) conclude that external and tacit knowledge are actually two extremes on a spectrum. Maurer (1999, page 12) states that knowledge resides in the heads of people and the computer can only store “computerized knowledge” which is to be understood as “shadow knowledge”, a “weakish image” of the real knowledge.

In PKM, we often deal with knowledge that is somewhere in the middle of these extremes. Note-taking e.g. is a core activity of PKM: An individual creates an external representation for internal concepts. Later, the external representation is internalised again to re-activate the knowledge in the individuals mind. If somebody writes a short informal note to himself it is often completely meaningless to others. The knowledge is thus not fully externalised – Yet this note is an external reminder about some knowledge that the author would otherwise forget.

Research Goals

The goal of this research is to find a general representation model for PKM tools. Different from other models, the model we are looking for is not meant to be hidden behind a yet to be defined user interface, but to be exposed as directly as possible to the end-user. The model should be easy to learn and it should be possible to import, represent and handle a number of existing knowledge organisation formalisms. In a more colloquial way, we look for a model that can do with concepts what spreadsheets can do with numbers.

Scenario

Today's knowledge workers are confronted with an overwhelming amount of information. Sometimes information is sent to the knowledge worker (e.g. by email or RSS feeds), found by chance (e.g. while meeting somebody in an airplane), or actively researched (e.g. in the library). A typical information professional could be a business analyst reviewing AJAX-frameworks in Web 2.0 start-ups; a biologist researching where sharks live and how and why their population changes, a lecturer in French history or a lawyer specialising in environmental law. The running example of this article will be a biologist called Linda writing a paper on white sharks for a conference in Italy.

The information typically encountered by knowledge workers is either of self-management nature, such as tasks and appointments or contact data, all of which is well covered by existing specialised PIM tools. Or the information belongs to the knowledge worker's domain of interest. Since the structure of this domain-specific information is typically rather unique and often even undergoing thorough change, there are often no specialised tools available that would support handling this information in a way appropriate for its structure. Popular generic tools are spreadsheets, text documents, slides, and the file system. However none of them would let Linda collect material about shark populations, reasons for their growth or decline, different shark species, shark hunting strategies, etc. in an integrated way. In a text file she would probably loose overview and in a spreadsheet she would not be able to represent relations between shark species. If she would use both tools, e.g. she could not easily refer to a specific cell in the spreadsheet from the document.

The two main tasks we intend to support are structured note taking and document creation. The core process from notes to a document can be described as steps in a knowledge maturing process (Maier and

Schmidt, 2007). Here, the question of granularity arises: Shneiderman (1989) found out that users are better able to answer questions when a text is modelled as more-fine grained (46 articles) hypertext instead of large chunks (5 articles). On the other hand, granularity is also an important cost-driver for PKM and a PKM system is only of value to a user if it provides more benefit in delivering relevant information than the cost of using it, i.e. externalisation, refinement, and search (Völkel and Abecker, 2008). We conclude that a PKM system should both be able to represent a whole range of granularity from short items (e.g. notes) to longer items (e.g. emerging documents).

Research Design/Methodology

In order to create a lean vocabulary for incremental recording and step-wise formalisation of personal knowledge, we conducted an extensive analysis of existing models and tools widely used to record, structuring and communicate knowledge. We identified a set of common knowledge structures found to be inherent in most knowledge artefacts – ranging from vague paper notes over books, hypertexts and folksonomies to highly structured documents and even taxonomies and most ontologies.

In order to allow gradual transitions between various degrees of formalisation, the types of these structural relations were modelled hierarchically as a lightweight top-level ontology of general relation types by subsuming the more specific ones under those they semantically imply. This resulted in the *conceptual data structures* (CDS) model (cf. Völkel et al., 2008).

A first open source CDS back-end is implemented on the *semantic web content repository* (SWCR, Völkel, 2007). On top of this CDS back-end, three user interface prototypes have been realised.
Additionally, we ran a small user survey to refine and extend requirements stated in state of the art literature. We first present briefly the CDS model and then in greater length the tools. After that, we explain how Linda could use these tools in her daily work.

CDS Model

The CDS model consists of two parts, a data model and a set of core relations found most often in existing models and tools used today for PKM.

CDS Data Model

The CDS data model consists of 1) items of unstructured text, structured text, images or other content, 2) names, 3) statements between items of all kind and 4) relation types.

Formally, we have a ***Model*** *M*, which is a set of *Items*. Each *Item* has a unique identifier (i.e. a URI) which makes it globally addressable. Each *Item* belongs to exactly one *Model*, has a creation date, a last modified date and an author. There are four kinds of *Items* a user can use:

A ***ContentItem*** represents a piece of addressable content. Content may be textual or binary. Binary content is defined as on the web (Jacobs, 2004), i. e. having an encoding, MIME-type and length measured in bytes. Textual content in CDS has by default UTF-8 encoding and may use some formatting using the *structured text interchange format* (STIF), as defined in (Völkel et al., 2008).

A ***NameItem*** models a term of the user's vocabulary. The name of the *NameItem* must be unique within a *Model*. There may be two or more *ContentItems* having the same content as a *NameItem* or as another *ContentItem*.

NameItems allow hiding URIs completely from user interfaces. In this respect, they are similar to e. g. titles of wiki pages. Note that *NameItems* represent only the name itself. E. g. a wiki page can be modelled as two *Items*: A *NameItem* to represent the wiki page title and a *ContentItem* to represent the wiki page content. The NameItems can be used as generic named containers, tags or formal types. *NameItems* allow jumping directly into certain nodes of the *Model*, similar to using known URLs to start browsing the WWW.

A ***Relation*** is a special kind of *NameItem*. Relations are used in *Statements*, which are explained in the next paragraph. Each *Relation* has a *mandatory inverse* Relation.

A ***Statement*** connects *Items*. A *Statement* is always of the form (*Item*, *Relation*, *Item*). As a *Statement* is itself an *Item*, the user can annotate statements as well – a handy feature e. g. for discussion systems.

It is possible that different statements with the same URI assert the same triple. But it is not possible that two different statements (differing in source, relation or target) have the same URI. For every *Statement* (*s*,*p*,*o*), the inverse *Statement* (*o*,–*p*,*s*) is inferred, where –*p* is the inverse of *p*. This is handy for user interfaces which allow browsing of items in both directions.

CDS Relation Type Hiearchy

The second part of the CDS model is a set of built-in *Relations*, which are repeatedly occurring across different knowledge organisation tools and models. The *Relations* are arranged in an inheritance hierarchy so that the relations with more specific semantics imply the relations with broader semantics.

The five core relation types deal with identity (similar to, same as, has alias), order, hierarchy, different forms of annotation (i.e. free-text annotations, tagging, and formal typing), and generic hyper-links. As the relation hierarchy is represented in the CDS data model, the user can (and is expected to) extend it.

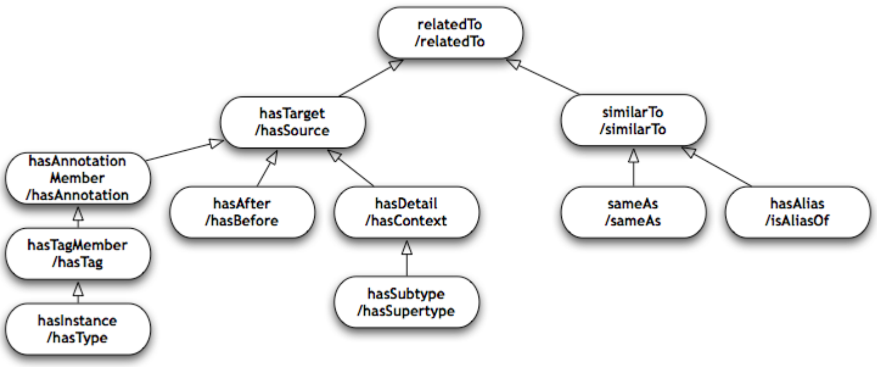


Figure 1: The CDS relation type hierarchy

The root type of the relationship hierarchy is *related*. Every *Item* is *related* to another *Item*, if any kind of *Relation* has been stated. This *Relation* allows to state very vague knowledge, i. e. “these items are related, but I can’t tell why or don’t want to spend the time to refine this now”. The next level in the hierarchy is either *similar*, to link items that describe the same real-world entity or *has target*, to interlink different items. *Has target* models a generic, directed hyper-link, as it is found in WWW, references in documents, or links in the file system. The CDS model contains three built-in refinements for *has target* which user interfaces should treat specially.

Has after and its inverse relation *has before* model any kind of ordering relation. It might be order in space, time or by other means.

Has detail and its inverse *has context* represent any kind of hierarchy and nesting. This relation represents hierarchies in a generic way, e.g. part-whole relations or type hierarchies are considered special cases of this relation. Types can be arranged in a type inheritance hierarchy, like classes in an ontology or programming language.

Both order and hierarchies are most often used among items that have the same type. To represent links between items of different modelling layers, CDS uses *has annotation* and its sub-relations.

Together with the hype around “Web 2.0”, tagging became popular for assigning easy-to-type keywords on items. In CDS, tagging is treated as a kind of annotation, hence *has tag* is a sub-relation of *has annotation*.

Assigning items a formal type is accomplished with the relation *has type*. In CDS, *has type* is a sub-relation of *has tag*, which leads to the desired effect that e.g. a species of shark that is typed as a *carnivore* implies it is also tagged as *carnivore*.

Tools based on CDS

In this section we present three prototypes of tools based on the CDS model, which have been developed within the NEPOMUK (2008) project.

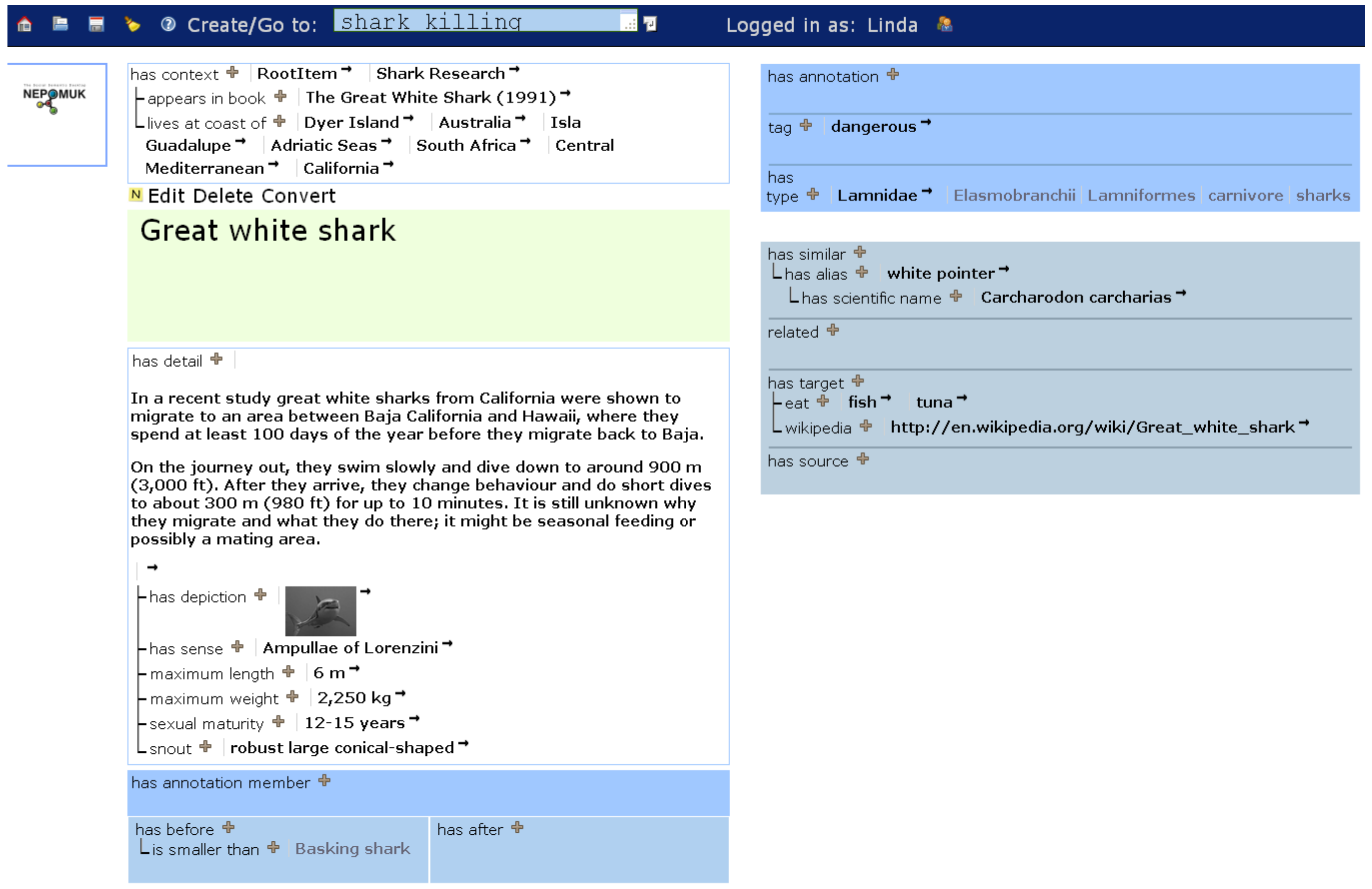
Hypertext Knowledge Workbench

The *Hypertext Knowledge Workbench* (HKW) resembles a semantic wiki, but without the tight coupling of one title to one page. HKW is different from semantic wikis: (a) it is backed by the more flexible CDS model, (b) allows to create and *change* formal statements easily, and (c) integrates authoring, structuring and formalisation into the retrieval.

Fig. 2 shows a screen-shot of the GUI[1] focusing on the *NameItem* “Great white shark”. The screen is divided into seven coloured areas, showing related *Items*. More formally, for a centred *Item i* the GUI shows a dynamic view for the query (*i*,*,*), including all inverse statements and inferred triples.

Below the “Great white shark” item, HKW shows the *Items* related via the relation *has detail*. E. g. the statement “Great white shark”-“maximum length”-“6 m” is rendered here. This tells the user also that ‘maximum length” is a *sub relation* of *has detail*. Behind the word “6 m” there are icons allowing the user to navigate to the *Statement* “Great white shark”-“maximum length”-“6 m”. In a *Statement* view, the *Statement* can be changed. E. g. the user can change the *Relation* or create a new source or target. Auto-linking is supported wherever possible.

[1]Try online or download from http://cds.xam.de


Create/Go to: shark killing

Logged in as: Linda

has context ✚ RootItem → Shark Research →
├appears in book ✚ The Great White Shark (1991) →
└lives at coast of ✚ Dyer Island → Australia → Isla Guadalupe → Adriatic Seas → South Africa → Central Mediterranean → California →

Edit Delete Convert

Great white shark

has detail ✚

In a recent study great white sharks from California were shown to migrate to an area between Baja California and Hawaii, where they spend at least 100 days of the year before they migrate back to Baja.

On the journey out, they swim slowly and dive down to around 900 m (3,000 ft). After they arrive, they change behaviour and do short dives to about 300 m (980 ft) for up to 10 minutes. It is still unknown why they migrate and what they do there; it might be seasonal feeding or possibly a mating area.

├has depiction ✚ →
├has sense ✚ Ampullae of Lorenzini →
├maximum length ✚ 6 m →
├maximum weight ✚ 2,250 kg →
├sexual maturity ✚ 12-15 years →
└snout ✚ robust large conical-shaped →

has annotation member ✚

has before ✚
└is smaller than ✚ Basking shark

has after ✚

has annotation ✚

tag ✚ dangerous →

has type ✚ Lamnidae → Elasmobranchii Lamniformes carnivore sharks

has similar ✚
└has alias ✚ white pointer →
 └has scientific name ✚ Carcharodon carcharias →

related ✚

has target ✚
├eat ✚ fish → tuna →
└wikipedia ✚ http://en.wikipedia.org/wiki/Great_white_shark →

has source ✚

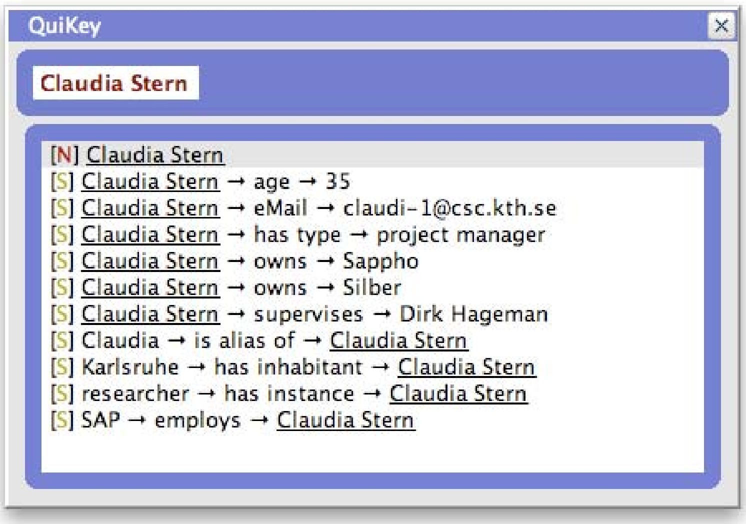
Figure 2: HKW prototype screen shot, focusing on *Great white shark*

Alternatively, she can delete this statement or create a new *Item* at the location (Great white shark, maximum length) by pressing the plus icon. This allows creating new semantically interlinked items easily. If the user enters a longer text or uses line breaks the system assumes the user creates a *ContentItem*. For short text, the system suggests existing *NameItems* or creates new ones.

Items related to the selected *Item* via *has context* (the inverse relation of *has* detail) are rendered above the "Great white shark" item. The other coloured boxes represent other CDS core relations. The GUI shows relations always in their most specific box. Items are only rendered in different boxes at the same time if the user assigned multiple super-relations to a relation.

QuiKey

QuiKey is a kind of smart semantic command-line that focuses on highest interaction-efficiency to browse, query and author CDS-based knowledge bases in a step-by-step manner. It combines ideas of simple interaction techniques like auto-completion, command interpreters and faceted browsing and integrates them to a new interaction concept. QuiKey forms a generic, extensible user interface for CDS models. Despite its versatility, QuiKey needs very little screen space, which also makes it a candidate for future mobile use. QuiKey is described in more detail in (Völkel et al., 2008). A screen-shot of its current implementation is depicted in Fig. 3.



[N] Claudia Stern
[S] Claudia Stern → age → 35
[S] Claudia Stern → eMail → claudi-1@csc.kth.se
[S] Claudia Stern → has type → project manager
[S] Claudia Stern → owns → Sappho
[S] Claudia Stern → owns → Silber
[S] Claudia Stern → supervises → Dirk Hageman
[S] Claudia → is alias of → Claudia Stern
[S] Karlsruhe → has inhabitant → Claudia Stern
[S] researcher → has instance → Claudia Stern
[S] SAP → employs → Claudia Stern

Figure 3: Screen shot of the current QuiKey implementation showing a list of statements about "Claudia Stern"

iMapping

iMapping is a technique for visually structuring information objects. It supports the full range from informal note taking over semi-structured personal information management to formal knowledge models. With iMaps, users can easily go from overview to fine-grained structures while browsing editing or refining the knowledge base in one comprehensive view. An iMap is comparable to a large white-board where information items can be positioned like post-its but also nested into each other. Spatial browsing and zooming as well as graphical editing facilities make it easy to structure content in an intuitive way. iMapping builds on a zooming user interface approach to facilitate navigation and to help users maintain an overview in the knowledge space.

The iMapping approach is described along with its motivations and foundations in more detail in (Haller, 2006). A small sample map mock-up is depicted in Fig 4.

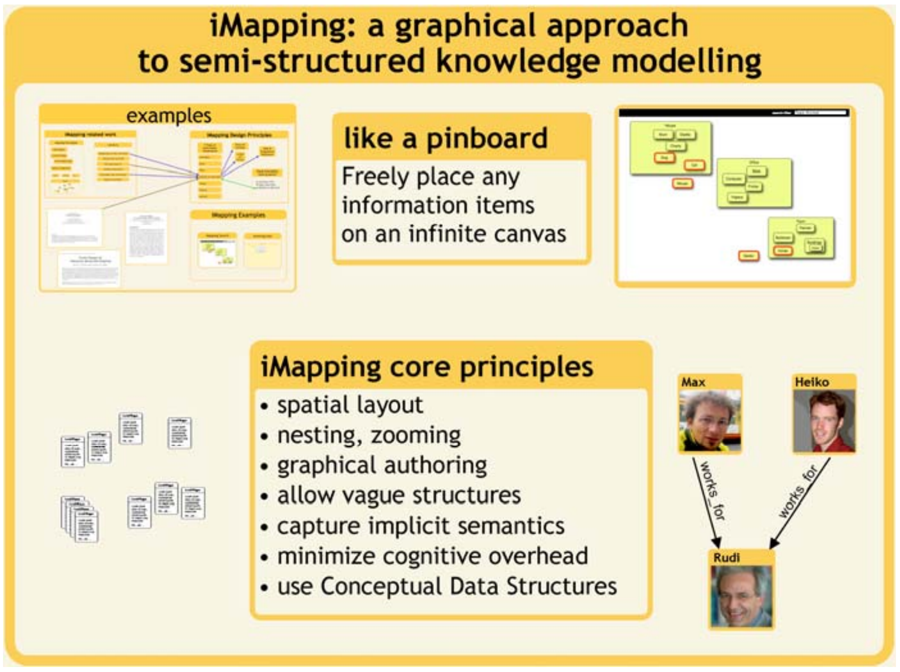


Figure 4: An iMap about iMapping

Analysis and Requirements

We asked 27 participants (of which 17 were researchers, largely in the field of computer science) in an online survey “*What should a Personal Knowledge Management (PKM) tool do for you?*".

Information Types

Users mentioned all kinds of information types to be tackled by a PKM tool. One user summarized the situation as “A broad amalgam of scientific papers, non-scientific articles, URLs and other documents, IM conversations, emails and personal notes comes in daily, forming sediments of data on my disk”. An exhaustive list of all information types mentioned by users includes:

- scientific papers (2x),
- non-scientific articles (1x),
- bookmarks (5x) ,
- instant messaging conversations (1x),
- e-mails (2x),
- personal notes (2x),
- social network (1x),
- scans (1x),
- pictures (1x),
- online documents (1x),
- account information (e.g. bank account number, 1x),
- topics (1x),
- how-tos (e.g. how to set a classpath in Java, 1x),
- mathematical knowledge (1x),
- contacts (2x),
- presentations (1x),
- projects (i.e. their notes, time plans, accounts, ideas, 1x),
- concept maps (1x),
- tools (1x),
- ideas (1x),
- events (1x),
- recipes (1x),
- favourite teas (1x),
- tax information (1x),
- and to-dos (1x).

Tasks

In the online survey, people mentioned a number of diverse tasks: Note taking, paper writing, birthday reminder, organizing to move to another country, strategic planning, scientific research, and consultation with friends and colleagues. Only the task "paper writing" (cf. Esselborn-Krumbiegel, 2002) was mentioned more than once (five times). One user summarized this as "[…] *locate the information by keyword, date, other metadata or by tracing a path of discovery, then attributing the source correctly, and communicating in a universally readable format* […]"

Functional Requirements

In the remainder of this section we discuss functionality requirements and their mapping to CDS and tools based on it. The basic processes in PIM have been identified (Jones and Bruce, 2005) as:

- *Keeping*, i.e. input of information into a PSI,
- *Finding or re-finding*, i.e. output of information from a PSI,
- and *Meta-activities*, e. g. mapping between information and need, maintenance and organisation.

For note-taking a user in our survey wrote: "*a PKM tool should help me aggregate, collect and view all the small bits of information, which are either needed for long term reference, or in the short term for completing a task.*"

Easily find things: At the heart of PKM is the requirement to easily find things that are stored in the PKM tool. Although the information stored in items and their relations is itself often not self-contained, it might suffice to remind the user of the knowledge that was present when the information was entered.

In HKW, Linda could e.g. retrieve the year in which the movie "Jaws" was shown if she remembered the director "Stephen Spielberg". She enters "Steph" and would get an auto-completion list containing "Stephen Spielberg". From there, she would probably look under the hyperlink "has directed" to see the title "Jaws". After clicking on it, she can see the details of the movie including the release data 1975. In general, if the item to be found is not a NameItem, HKW allows her browsing associatively from a known entity to the desired one, just like in a wiki. However, navigation in HKW is expected to be faster, because links are already grouped into different cognitive dimensions (ordering, hierarchical, typing or other links).

In the iMapping prototype, she would first zoom in the upper right corner for "private stuff". In there, she would go to her "Movies" item and browse in. As there are many movies, the first goes into the "Best Directors Ever" item inside the "Movies" item. In there, she selects "Stephen Spielberg" and can see all outgoing links, labelled with the type. Although there are quite some links she just looks at links pointing at movies and quickly identifies the "Jaws" movie. After zooming onto it she sees "1975" inside the "Jaws" item. On hovering over it, a relation "release date" is shown between the outer "Jaws" item and the inner "1975" item. IMapping allows finding an item based on spatial proximity simply by moving around in the infinite 2D space.

Fast entry of new items and **extension of existing items**: The survey revealed a desire for fast entry of new items (mentioned two times in the survey) as well as an easy way to extend existing items (2x). Oren (2006) advises to focus on simply capturing and representing the things that the user wants to store, before doing any reasoning with it.

QuiKey is the fastest tool by means of mouse clicks and keys typed for entering data. With one short-cut the tools is brought into focus. Now Linda can simply enter a new short note such as "reproduction is slow, with sexual maturity occurring at about 12-15 years of age“. Alternatively she can write "white shark" <tab>, "sexual maturity" <tab> "12-15 years" <return>. This will add a CDS statement to her PKM knowledge base without requiring here to navigate anywhere first and still extend the existing white shark items. New items and relations are created on the fly if needed. Re-use of existing items and relations is encouraged with auto-completion. After she wrote "white shark" <tab> QuiKey already presented here a ranked list of existing statements about the white shark. Thus QuiKey also includes browsing of the knowledge base.

Grouping of items: The next set of features required is centred on grouping of items. The tool should be able to let *me group seemingly unrelated content* (survey). Users need composition for navigation (Frank, 1988). This allows e.g. browsing and thereby narrowing down their view and allows discovering related, yet unexpected items. In the iMapping prototype, Linda could e.g. simply move the item "basking shark" next to "white shark" as both shark types have similar body shapes. This would not introduce any kind of statement in the underlying CDS model but help her to remember associations with minimal modelling effort.

Named containers: Users also demanded that it should be easy to place new items into a named container (survey). But on the other hand, Frank (1988) advises to not require a user to name all items. Consider e.g. several contacts in Linda's address book to link to the same postal address (e.g. all 20 people working for a non-governmental underwater-life-protection organisation). In this case it would be overhead to assign the address of the office a dedicated name. Yet it would also be cumbersome to have to change the entries of all these people in case the postal address of the NGO changes. Therefore it should be allowed but not required to give entities a name.

CDS accomplishes this with NameItems which are unique within a knowledge model and which can be used as contexts (i.e. like document folders), tags, types or anything. The three prototype tools support consistent re-use of NameItems by offering auto-completion features. Conceptually, the relationship types are also names from the same namespace, i.e. there cannot be a *relation* named ‘knows’ being something different from a *name item* ‘knows’. One NameItem represents only one thing, although multiple NameItems can represent the same thing (i.e. synonyms).

In iMapping, Linda can create an item “fish eaten by white shark” and put inside items named “rays”, “tuna”, and “smaller sharks”. She can simply click inside an existing item and thereby gets a cursor to enter the text of a newly created child item. She could also create first the three fish items and then add a new item and drag the fish items into it.

In HKW, the intended way would be to navigate to “white shark” and click on “add” in the “has detail” panel. She gets a pop-up window with two fields; the first one is pre-filled with the text “has detail”, the second has the user input focus. She could enter just some text “x” now and thereby add the statement (white shark, has detail, x). Instead she decides to create a new sub-relation of “has detail” by typing “eats” into the first field. In the second field she types “rays” and presses <submit>. This creates the two statements (has detail, has sub-relation, eats) and (white shark, eats, rays). HKW shows here now “eats” as a sub-relation of “has detail”. She clicks on the “add” icon of “eats” and enters “tuna” <submit> and repeats this for “smaller sharks”.

Categories: In the survey, users prefer categories over strict hierarchies (mentioned three times). All three CDS-based tools allow multiple parents, i.e. an item can have several tags, types, annotations, or contexts. A relation can have several super-relations. HKW presents the relationship inheritance graph as a flat tree; some relations appear simply as children of several other nodes.

Context: Users wish it should be clear *which data is from my personal information sphere and which is coming from outside* (survey). This is in line with Oren (2006): Understand the notion of context, capture it together with the information and use it to enhance recall and understanding. The CDS back-end records for each item the creation date and the author that entered it into the system. Items created by the system are marked with a different author.

Links: The next set of required features deals with explicit links between items. Oren (2006) summarizes: exploit the interlinked nature, do not rely only on search, and allow people to associate freely. Three users required links between items, e.g. a link between the tasks "buy food for dog" and "bring dog to veterinary".

The link is one of the four core CDS types and all three prototypes build on it. In iMapping, users can drag-drop typed links between items. In HKW the user can even annotate, tag and link the links themselves.

Order: Ordering a collection of ideas or text snippets into a coherent flow is one of the main tasks of authoring (Esselborn-Krumbiegel, 2002). A user should be able to create order gradually, e. g. by stating order between some sections, but not requiring a total ordering.

Partial or total order is one of the four core dimensions in CDS. It is supported by HKW which allows e.g. Linda to state explicitly that section A of her paper comes before B and before C but there is no relation yet between B and C. This allows Linda to gradually and consciously add order to her article outline without mentally keeping track what is at some place in the list because it was explicitly put there or because it is just currently there while it is being sorted.

Hierarchy: Hierarchies of all kind are commonly used in user interfaces to let the user narrow down his interests step-by-step. Users need ways to see multiple levels of detail at once (Frank, 1988). Shneiderman (1996, p. 336) emphasizes the need to get "Overview first, zoom and filter, then details-on-demand." Users in our survey required being able to hide level of details to get an overview of the content. Others wished a graphical overview that represents connections and interactions between notes.

CDS has a built-in relationship type to represent hierarchies, i.e. "has context" and "has detail". HKW allows seeing three level of a hierarchy, i.e. current item, context of that item, and details of that item. IMapping allows seeing infinitely many levels at once, only limited by screen resolution.

Transclusion: User often loose structure of knowledge cues when transforming from one tool to another. E. g. text snippets from a hypertext context loose their identity when pasted into a document. Instead of copying the value of an item it is more elegant to copy a reference to the item. If the content item is changed, the change is reflected in all parts where it is embedded. Embedding a reference and rendering the content is called transclusion. The need for transclusion is explained by Ludwig (2005) and Nelson (1995).

CDS makes it easy to reference all parts of a model, as each item has a globally unique URI. There is currently no tool support implemented for this in the prototypes.

The last set of requirements deals with adding and using more structure and semantics to the items.

Flexible schema: In the survey paper of Oren (2006) we find the requirement for flexible schemas: Leave users their freedom and do not constrain them into rigid schemas.

The CDS model can be used to simply capture the semantics on a level of node-and-arc diagrams. Items represent the nodes and labelled arcs can be represented with statement, where the content of the statement is the label of the arc. Arcs can be undirected and use the CDS-built-in relationship type “related”. Directed arcs are modelled with “has target”. More formal relationship types can optionally be created in lower levels of the relationship inheritance hierarchy.

Structuring: One of the most often requested features (five people) was support for (re-)structuring existing structures: A PKM tool should *help to structure and sort items*, *be easy to restructure*, *help to move from unstructured to more structured*, *organize pieces of larger text*, and *help to categorize items according to existing filing schemes such as taxonomies, tags, vocabularies and ontologies* (survey).

QuiKey is not well suited for re-structuring existing knowledge. HKW allows refining or changing existing statements, e.g. by navigating to a statement and changing the relationship type into something different, e.g. more specific. The auto-completion shows by default only refinements of the currently selected relation.

In iMapping, the user gets a graphical, zoom-able overview of all his items and can simply structure his items by drag-and-drop like on a physical pin board. After grouping related items together and moving them inside another item, a number of items can efficiently be manipulated at once. In this regard, iMapping has the same re-structuring capabilities like e. g. mind-mapping (Buzan, 1991) tools, but with the added value of spatial hypertext, i.e. the positions of items are chosen by the user which allows creating very lightweight “piles” of related items, just like on a physical desk.

Search and query: Besides browsing a user also needs the ability to search and query the data (Frank, 1988). The CDS back-end offers queries that let Linda exploit her modelling effort. She could e.g. ask in QuiKey ((white shark, has sense, ?x) AND (NOT(human, has sense,

?x))) to find out that white sharks have a sense for electrical fields that humans don’t. In systems that do not allow formalising content she would potentially have to read many articles and build up the query results in her mind. Semantic queries are especially useful for creating list of items fulfilling certain criteria. More details about the user interaction for creating such queries and the query language itself can be found in Haller (2008).

Formal knowledge: CDS allows to not only structure but also to formalise knowledge. This allows Linda retrieve e.g. “white shark” using an expressions like (?x, has type, Lamniformes) although she never told the system that this is true. She might just have entered (white shark, has type, Lamnidae) and also (Lamnidae, has supertype, Lamniformes). This allows the CDS back-end to deduct – using a standard RDFS reasoning engine – that white sharks also belong to the Lamniformes. It remains the responsibility of the user to decide which content should be formalised up to which degree.

Interoperability: To re-use the data in other systems (particularly other KM systems), Linda needs to export all items and structures into a common format. The Resource Description Format (RDF, Dan Brickley, 2004) defines an extensible, graph-based model for integrating distributed, heterogeneous information sources. The CDS back-end represents all data (besides binary content) natively as RDF data.

Related Work

Semantic wikis are designed and used not only for collaborative use but also for personal knowledge management (cf. Oren, 2005; Oren et al., 2006). Semantic wikis allow stepwise formalisation of content: First a page is created, then filled with text, spell-corrected, structured, re-structured, and linked to other pages. Then links are typed and pages linked to categories. Ironically, just like with paper-based approaches, *changing* things is not that easy in semantic wikis. Tasks such as renaming a relation require typically an administrator to run scripts over the database, as the wiki source text of many pages needs to be changed. Second, a common use-case of PKM tools is the need to import knowledge from external sources. In most semantic wikis, the import of semantic data needs to be represented by artificially generated wiki syntax inserted into pages, which does not integrate easily with existing content.

Ludwig (2005) sees redundancy within and among documents as a hurdle to efficient information usage. He questions if documents are the best container for knowledge representations and proposes to work more direct with redundancy-free semantic knowledge management systems. In such a system, the traditional notion of a document is replaced by virtual documents, which render parts of the knowledge base as an interactive tree.

Bernstein (2006) describes TinderBox, a personal content management assistant, which offers sophisticated HTML generation via templates.

Both systems (Bernstein, 2006; Ludwig, 2005) allow end-users to construct ontologies out of their linked information objects. The same direction can be observed in the larger fields of semantic desktop (Decker et al., 2005) and semantic wiki (Völkel and Schaffert, 2006).

Conclusions

In an attempt to create a lean vocabulary for incremental recording and step-wise formalisation of personal knowledge, we identified a set of common knowledge structures. These Conceptual Data Structures were found to be inherent to a variety of different knowledge artefacts ranging from vague paper notes to highly structured documents.

CDS allows to gradually represent knowledge in various degrees of formalisation in a uniform fashion. As a lightweight top-level ontology about relation types, CDS is designed to bridge the gap between unstructured content like informal notes and formal semantics like ontologies by allowing the use of vague semantics and by subsuming arbitrary relation types under more general ones.

It serves two purposes: First, as a guideline for future PKM tools, providing a set of crucial structural primitives. Second, the RDF-based representation of CDS can serve as a knowledge exchange format.

The three prototypical CDS front-end tools show the variety of visualisation and interaction paradigms that can benefit by using this common data model.

Acknowledgements

Part of this work has been funded by the European Commission in the context of the IST NEPOMUK IP - The Social Semantic Desktop, FP6-027705. Another part of this work has been done in WAVES (2008) funded by BMBF, Germany. The expressed content is the view of the author but not necessarily the view of any sponsor.

References

Alvarado, C., Teevan, J., Ackerman, M.S. and Karger, D. (2003), “Surviving the information explosion: how people find their electronic information”, Technical Report AIM-2003-006, MIT AI Lab.

Bernstein, M. (2006), “Shadows in the cave: hypertext transformations”, paper presented at Symposium on Interactive Visual Information Collections and Activity (IVICA), October 1–2, College Station, TX, USA, available at: www.csdl.tamu.edu/ivica/papers/L2-2-Bernstein.pdf (accessed 15 January 2009)

Braganza, A. and Mollenkramer, G.J. (2002), “Anatomy of a failed knowledge management initiative: lessons from PharmaCorp’s experiences”, *Knowledge and Process Management*, Vol. 9 No. 1, pp. 23-33.

Bush, V. (1945), “As we may think”, *Atlantic Monthly*, Vol. 176, pp. 101-8.

Buzan, T. (1991), *Use Both Sides of Your Brain: New Mind-Mapping Techniques*, 3rd ed., Plume, New York, NJ.

Brickley, D. and Guha, R. (2004), “RDF vocabulary description language 1.0: RDF schema”, [online], available at: www.w3.org/TR/rdf-schema/ (accessed 16 December 2008).

Davenport, T.H. (2005), *Thinking for a Living: How to Get Better Performances And Results from Knowledge Workers*, Harvard Business School Press, Boston, MA, USA.

Decker, S., Park, J., Quan, D. and Sauermann, L. (Eds) (2005), *The Semantic Desktop – Next Generation Information Management and Collaboration Infrastructure*, [publisher], Galway, Ireland.

Despres, C. and Chauvel, D. (2000), *Knowledge Horizons: The Present and Promise of Knowledge Management*, Butterworth-Heinemann, Boston, MA, USA.

Drucker, P.F. (1999a), “Knowledge-Worker Productivity: The Biggest Challenge”, *California Management Review*, Vol. 41 No. 6, pp. 79-94.

Drucker, P.F. (1999b), *Management Challenges for the 21st Century*, HarperBusiness, New York, NJ.

Engelbart, D. (1963), *A Conceptual Framework for the Augmentation of Man’s Intellect*, Spartan Books, Washington, DC, pp. 1-29.

Esselborn-Krumbiegel, H. (2002), *Von der Idee zum Text. Eine Anleitung zum wissenschaftlichen Schreiben*, 2nd ed, Schöningh-Utb, Paderborn,Germany.

Frank, G.H. (1988), “Reflections on notecards: seven issues for the next generation of hypermedia systems”, *Communications of the ACM*, Vol. 31 No. 7, pp. 836-52.

Haller, H. (2006), “iMapping – a graphical approach to semi-structured knowledge modelling”, paper presented at the 3rd International Semantic Web User Interaction Workshop (SWUI2006), November 6, Athens, GA, USA, available at: www.aifb.uni-karlsruhe.de/WBS/hha/papers/iMapping_SWUI2006_paper.pdf (accessed 15 January 2009)

Haller, H. (2008), “QuiKey” in *Proceedings of the Workshop on Semantic Search at the 5th European Semantic Web Conference*, Vol. 334, ISSN: 1613-0073, pp. 74-8.

Jacobs, I. and Walsh, N. (2004). “Architecture of the World Wide Web, Volume One” [online], W3C, available at: www.w3.org/TR/webarch/ (accessed 16 December 2008).

Jones, W. and Bruce, H. (2005), "A report on the NSF-sponsored workshop on personal information management", January 27-29, Seattle, WA, available at http://pim.ischool.washington.edu/final%20PIM%20report.pdf (accesed 15 January 2009).

Jones, W., Phuwanartnurak, A.J., Gill, R. and Bruce, H. (2005), "Don't take my folders away! : organizing personal information to get things done", in van der Veer, G.C. and Gale, C. (Eds), *CHI Extended Abstracts,* ACM, pp. 1505–8.

Klyne, G. and Carroll, J.J. (2004), "Resource description framework (RDF): concepts and abstract syntax" [online], available at: www.w3.org/TR/2004/REC-rdf-concepts-20040210/ (accessed 16 December 2008).

Ludwig, L. (2005), "Semantic personal knowledge management", Technical Report D11.01_v0.01, DERI Galway.

Maier, R. and Schmidt, A. (2007), "Characterizing knowledge maturing: a conceptual process model for integrating e-learning and knowledge management", in Gronau, N. (Ed.), *4th Conference Professional Knowledge Management – Experiences and Visions (WM '07), Potsdam*, Vol. 1, GITO, Berlin, pp. 325-34.

Maurer, H. (1999), "The heart of the problem: knowledge management and knowledge transfer", in *Proceedings of ENABLE'99*, Espoo-Vantaa Institute of Technology, pp. 8-17.

Miller, G. (1956), "The magical number seven, plus or minus two: some limits on our capacity for processing information", *Psychological Review*, Vol. 63, pp. 81-97.

Nelson, T.H. (1995), “The heart of connection: hypermedia unified by transclusion”, *Communications of the ACM*, Vol. 38 No. 8, pp. 31-3.

NEPOMUK (2008), “NEPOMUK – The Social Semantic Desktop” [online], available at: http://nepomuk.semanticdesktop.org/ (accessed 16 December 2008).

Nonaka, I. (1994), “A dynamic theory of organizational knowledge creation”, *Organization Science,* Vol. 5 No. 1, pp. 14-37.

Nonaka, I. and Konno, N. (1998), “The concept of “ba”: building a foundation for knowledge creation”, *California Management Review*, Vol. 40 No. 3, pp. 40-54.

Nonaka, I. and Takeuchi, H. (1995), *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*, Oxford University Press, New York, NJ.

North, K. (2007), “Produktive Wissensarbeit”, paper presented at the *5. Karlsruher Symposium für Wissensmanagement in Theorie und Praxis, 11 October, Karlsruhe, Germany..*

Oren, E. (2005), “SemperWiki: a semantic personal wiki”, in Decker *et al*. (Eds) (2005), *The Semantic Desktop – Next Generation Information Management and Collaboration Infrastructure,* [publisher], Galway, Ireland.

Oren, E. (2006), “An overview of information management and knowledge work studies: lessons for the semantic desktop”, in SemDesk (Ed.), *Semantic Desktop Workshop*, [full date and place, organisation].

Oren, E., Völkel, M., Breslin, J.G. and Decker, S. (2006), “Semantic wikis for personal knowledge management”, in *Database and*

Expert Systems Applications, Vol. 4080/2006, Springer, Berlin/ Heidelberg, pp. 509-18.

Polanyi, M. (1966), *Tacit Dimension,* Routledge and Kegan Paul Ltd, London.

Schütt, P. (2003), "The post-Nonaka knowledge management", *Journal of Universal Computer Science*, Vol. 9 No. 6, pp. 451-62.

Shneiderman, B. (1989), "Reflections on authoring, editing, and managing hypertext", in Barrett, E. (Ed.), *The Society of Text: Hypertext, Hypermedia, and the Social Construction of Information*, MIT Press, Cambridge, MA, pp. 115–31.

Shneiderman, B. (1996), "The eyes have it: a task by data type taxonomy for information visualizations", in *VL '96: Proceedings of the 1996 IEEE Symposium on Visual Languages*, IEEE Computer Society, Washington, DC.

Shneiderman, B. (1998), *Designing the User Interface*, Addison Wesley, Reading, MA.

Stankosky, M. (2005), *Creating the Discipline of Knowledge Management: The Latest in University Research*, Butterworth-Heinemann, New York, NJ.

Taylor, F.W. (1911), *The Principles of Scientific Management*, Harper and Brothers, New York, NJ.

Völkel, M. (2007). "A semantic web content model and repository", in *Proceedings of the 3rd International Conference on Semantic Technologies,* JUCS, Graz, Austria, pp. 254-61.

Völkel, M. and Abecker, A. (2008), "Cost-benefit analysis for the design of personal knowledge management systems", in Cordeiro J. and Filipe J. (Eds.), *ICEIS 2008 - Proceedings of the Tenth International Conference on Enterprise Information Systems,* *Volume AIDSS, Barcelona, Spain, June 12-16,* Springer, Berlin, pp. 95-105.

Völkel, M. and Schaffert, S. (Eds) (2006), *Proceedings of the First Workshop on Semantic Wikis – From Wiki To Semantics*, FZI Forschungszentrum Informatik Karlsruhe, CEUR.ws.

Völkel, M. *et al*. (2008), “Conceptual data structure tools”, *Deliverable 1.2,* Nepomuk Consortium.

WAVES (2008), “WAVES: Wissensaustausch bei der verteilten Entwicklung von Software” [online], available at: http://waves.fzi.de/bin/view/Public/WavesProjekt (accessed 16 December 2008).

Bio Heiko Haller

Heiko Haller works at Forschungszentrum für Informatik, a research institution specialising in technology transfer. While he mainly deals with using semantic technologies for knowledge management in general, his special interest lies in designing cognitively adequate techniques for personal knowledge management. Mr. Haller has studied cognitive psychology at the Free University of Berlin and is currently focussing on interaction design and visual knowledge representations. He wrote his diploma thesis on "Mapping Techniques for Knowledge Organisation" at Knowledge Media Research Center in Tübingen.

Bio Max Völkel

Max Völkel is working as a PhD student and Research Assistant at the Forschungzentrum für Informatik (FZI) at the Universität Karlsruhe(TH). His topics are Personal Knowledge Management, Semantic Web Infrastructure and Semantic Wikis. He has organised several workshops on semantic wikis (see http://semwiki.org). He works in the EU-project NEPOMUK to build a next-generation knowledge articulation tool. He is the author of a number of RDF-based tools such as RDF2Go (http://rdf2go.semweb4j.org), a triple store abstraction layer. Currently he works on a semantic web content repository (http://swecr.semweb4j.org), unifying RDF and Web 2.0 content management. He is also one of the founders of the Semantic MediaWiki project (http://ontoworld.org).