

Artificial Memory Prototype for Personal Semantic Subdocument Knowledge Management (PS-KM)

Digital Enterprise Research Institute (DERI) www.deri.org
DERI Galway, National University of Ireland, Galway, Ireland

Lars Ludwig

lars.ludwig@deri.org

David O'Sullivan

dos@nuigalway.ie

Xuan Zhou

xuan.zhou@deri.org

Abstract

This document, a demonstration proposal for ISWC 2004, introduces the Artificial Memory prototype system which enables Personal Semantic Subdocument Knowledge Management (PS-KM).

Keywords

personal knowledge management, distributed web knowledge management, ontology visualization

1. Introduction

The semantic web has gained widespread popularity. A large number of important research fronts are now being explored by researchers from around the world. Besides Semantic Web Services, one important area is how knowledge is managed and in particular how documents can be enriched through techniques such as semantic annotation. Annotations of documents and pages provide machine-readability of information. Software agents and other programmes can then be configured to automatically extract important knowledge from documents. Semantic annotations are typically enabled by ontologies. Ontologies in turn are enabled by ontology languages and ontology meta-data models. Semantics can be added to pages and documents manually, semi-automatically, and automatically [1]. This can be done dynamically when writing the document or after the document has been written [2]. Annotations represent content-independent meta-data which describes information in the document or page in some meaningful way. Typical documents, pages or reports represent serializations of information. Information is expressed by the author in 'chunks' and these chunks are serially joined together to form paragraphs, pages, sections, etc. Documents which have

been annotated therefore represent serializations of instances or concepts that can be extracted by software agents. However annotations and ontologies have one major setback. An ontology cannot be defined in annotations alone because the occurrence of identical concepts and instances cannot be prevented in semantic annotations of documents. An ontology defined solely in semantic annotations would soon result in a de-normalized data base that is difficult, if not impossible to maintain. In addition, semantic annotations and by association the semantic web, needs to have separate definitions of ontology concepts and instances. For semi-automatic annotation during the writing process of a document both the document and a separate data base (ontology) have to be edited in parallel. It therefore seems unlikely that annotation of documents will be widely provided because of the additional and considerable work load necessary by the writer: "Experience has shown that it takes roughly one hour to annotate five pages" [3, page 132]. Automatic annotation on the other hand is restricted to providing references to given entities of an ontology which most likely will result in 'semantic islands' within documents. In both annotation scenarios (semi-automatic or automatic), parts of the document will be annotated while other parts will not. More importantly, the relationship between an ontology entity (e.g., concept, instance, relation) and a document will not be explicitly stated in annotations. One cannot for example annotate annotations. Not only does annotation create semantic 'dark spots', the process doesn't provide the inner meta-relationship between ontology entity reference and the corresponding document. Semantic annotations therefore become an insufficient means for semantically rich documents and pages. This paper and demonstration presents the concept of Personal Semantic Subdocument Knowledge Management (PS-KM) to solve the before-mentioned problems. PS-KM does this by storing and presenting information in a novel yet vaguely familiar way. PS-KM is a knowledge management process that encourages the user to store information as information chunks, which can later be serialized into document-like formats as required. In this way PS-KM avoids annotation dark spots between explicitly serialized semantic information chunks (ontology entities). PS-KM thus allows for an explicit definition of the relationship between information chunks and their container (document). PS-KM reverses the process of generating documents by first writing and semantically networking information chunks and only afterwards serializing those into documents. PS-KM thus combines content-independent metadata and content-dependent metadata in a more efficient way that allows easier access to important knowledge by users and software agents. Moreover, it helps to avoid de-normalization of texts in documents.

2. Demonstration Overview

Artificial Memory (AM) is a multipurpose Knowledge Management system prototype currently designed and tested for single users and small groups. In order to realize PS-KM, AM explores the possibility of managing knowledge using information chunks in much the same way as we use our own human memory (hence the working title 'Artificial Memory'). It comprises an (i) Ontology Editor, (ii) Online Semantic Processing (OLSP) engine, (iii) Ontology Search Engine, and (iv), Ontology Browser. The latter makes use of the OLSP engine by offering to the user semantic processing functionality depending on the semantic context. AM's primary goal is to allow for thought-accompanying storing, association, and retrieval of information chunks. Information chunks represent instances of concepts comprising instance attributes. Attributes of instances may contain or point to structured as well as unstructured information. All attributes stored in AM are themselves relatable instances. 'Thought-accompanying' means that AM supports its users in storing information precisely at the moment when it is created or edited, and later in retrieving that same information when it is to be reused. In its current phase of development AM assists in PS-KM. It strives to provide a means to create a proper and current reflection of personal knowledge. Indirectly, it also helps to structure perception, thoughts, and memory by the user. By allowing for information chunks to be serialized and communicated AM can be extended into becoming a semantically rich collaboration platform for (virtual) teams. Personal Semantic Knowledge Management through the current version of AM is seen as a primary 'building block' to enabling the semantic web vision of the future for collaborative knowledge management. Expressing knowledge and information in semantically explicit information chunks mimics the natural process of our own memories. When

these chunks are related to each other semantically, they provide rich information presentation, enhanced communication, and enhanced human-machine-interaction - effective 'knowledge management'.

3. System Overview

3.1 Artificial Memory System Architecture

General remarks: The AM prototype is a Web-application. It has an HTML Web-user-interface. It does not require plug-ins of any kind. However, for its rich user interface to work properly the user's browser has to be JavaScript-enabled. AM uses a relational database for high-performance data extraction. AM is available as open-source software under the GNU General Public Licence from the author. A showcase can be found online: www.artificialmemory.net.

System Architecture: In general, AM is a typical multi-layered Web application. It comprises a relational data base, a data access layer, a functional/logic layer, and a user interface layer. The four functional main system components are the Ontology Editor, the Ontology Browser, the Ontology Search Engine and the OLSP Engine. The proprietary Entity-Relationship data model used can be seen as a performance-optimized metadata-model to store triple-based information. The AM prototype does not depend on a specific ontology language. However, it currently supports RDF export in N3 notation and will provide RDF import.

Ontology Building Blocks: AM is ontology-based. Its building blocks are akin to those of RDF(S). AM does not restrict itself to a specific ontology language. It strives to provide a flexible data model that represents an intermediate ontology language able to integrate features of different ontology languages. AM had to integrate a couple of new ontology features due to special practical needs of applying ontologies in Personal Knowledge Management. The following main ontology building blocks are being provided:

- Concept / class
- Instance
- Binary (bi-directional) relation
- Attribute
- Several basic data types
- MIN-, MAX-value constraint
- Default instance attribute value
- Default instance relation
- MAX-cardinality relation constraint
- Domain and range restrictions to relations
- Classification of relations (e.g., transitive, hierarchical, associative)
- Bags / sets
- Sequence numbers (for predicate-object)

Ontology Maintenance / Ontology Editor: Besides adding new concepts, global attributes and relations to the ontology schema at any time, it is possible to add new attributes to concepts or delete given attributes from concepts at any time without instances becoming invalid. Every instance can be optionally up-dated to integrate new attributes of the concept it is originally based on.

Presentation Layer / Ontology Browser Features: AM has been developed as a universal Knowledge Management tool. Therefore, it has a general user interface, not dependent on a specific domain of knowledge. There are only four different, dynamically built basic forms / views: a tree view, an editing-form, a print / read view, and a search-form. The tree view includes a semantically expressive, browsable tree. Instances can be serialized into a document view using sequence

numbers. The user interface supports English, German, and Chinese.

Knowledge Retrieval / Ontology Search Engine / OLSP Engine: AM supports 4 different ways of knowledge retrieval

1. Key word search (full-text search, wildcards support, result highlighting)
2. Ontology browsing (includes click stream measurement and click stream display)
3. Ontology search (basic functionality; dynamically built attribute search forms)
4. Online Semantic Processing (OLSP), e.g., hierarchy building

Multi/Parallel-User Support: AM supports multi-/parallel-user mode. User rights related to basic operations such as administer, delete, add, relate, and change entities can be granted. The Web user interface allows for accessing different knowledge bases / data bases.

3.2 Outlook

In the future, AM shall be enhanced by adding more Online Semantic Processing (OLSP) features. OLSP will provide for ontology-based knowledge management what OLAP currently provides for Data Ware-housing: an intuitive way of implicitly and interactively specifying complex queries and retrieving information in real-time. Furthermore, AM shall be a starting point for implementing semantic collaboration in order to replace less structured means of communication such as email- and document-exchange in general. Regarding organizational memories, AM shall provide means to efficiently exchange information between personal knowledge bases managed in AM, and means to consolidate personal knowledge bases. Thus, the AM system will be developed into a peer-to-peer system for distributed knowledge and innovation management scenarios. It would be the first of its kind enabling federated PS-KM.

4. References

- [1] M. Erdmann, A. Maedche, H. Schnurr, and S. Staab. From manual to semi-automatic semantic annotation: About ontology-based text annotation tools. In K. Hasida P. Buitelaar, editor, Proceedings of the COLING 2000 Workshop on Semantic Annotation and Intelligent Content, Luxembourg, August 2000.
- [2] S. Handschuh and S. Staab. Annotation of the shallow and the deep web. In S. Handschuh and S. Staab, editors, Annotation for the Semantic Web, volume 96 of Frontiers in Artificial Intelligence and Applications, pages 25-45. IOS Press, Amsterdam, 2003.
- [3] Decker, S.: Semantic Web Methods for Knowledge Management. Ph.D. thesis, University of Karlsruhe, 2002.

5. Screenshots

EDIT KNOWLEDGE OBJECT

Navigation Item Options

name:
Dublin Core RDF

hyperlink:
<http://dublincore.org/2003/03/24/dces#title>

content:

language:
EnglishUK

date_for_deletion:

validity_date:

submission_date:

Figure 1: Editing view

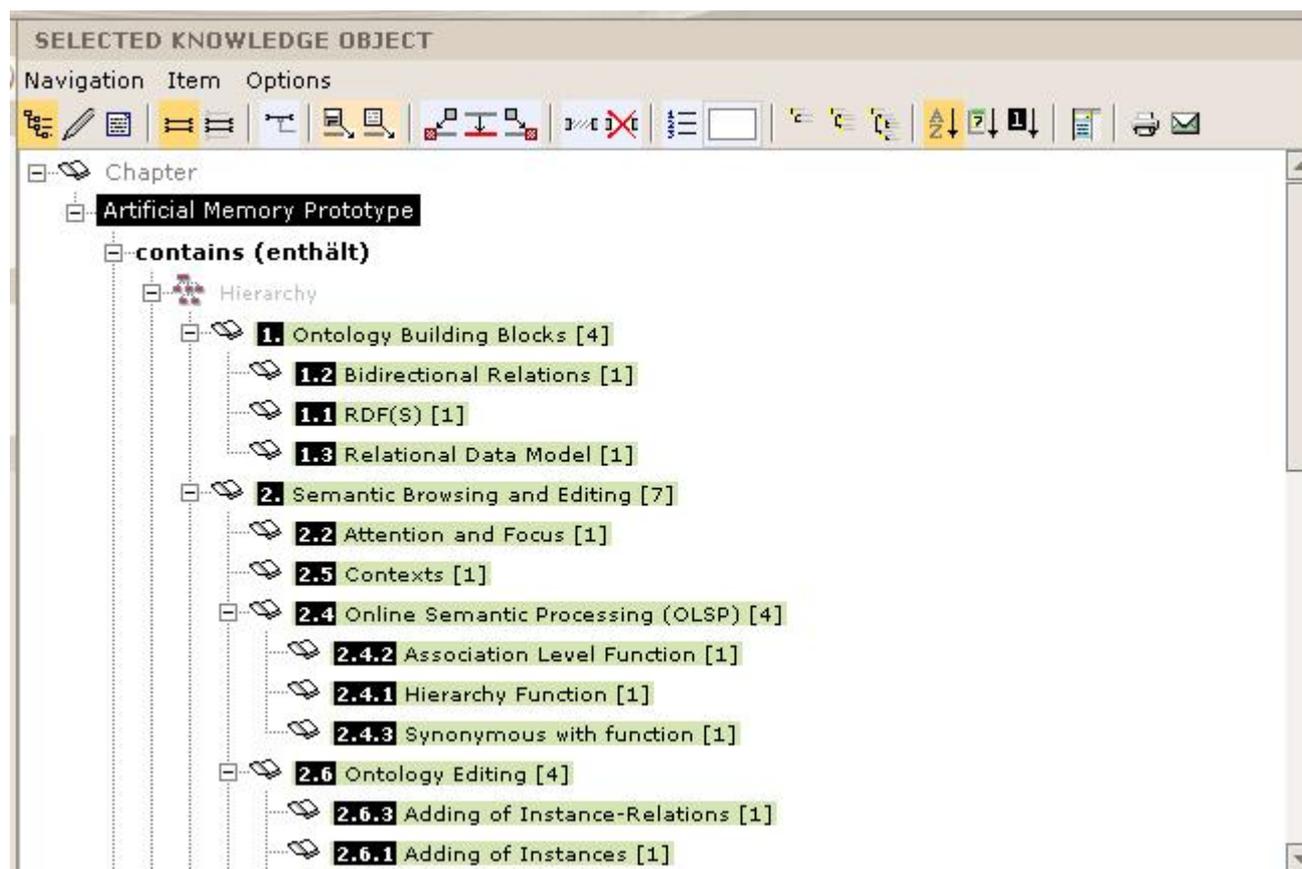
READ KNOWLEDGE OBJECT

Navigation Item Options

name:
LARS LUDWIG

hyperlink:


document_type:
JPG JPEG [IMAGE]

Figure 2: Print/Read view**Figure 3: Tree view (with relation-specific OLSP result view)**

Acknowledgements

This paper is based upon works supported by the Science Foundation Ireland under Grant No. 02/CE1/I131.