Corporate Memory: A framework for supporting tools for acquisition, organization and maintenance of information and knowledge

Marek Ciglan[*], Marián Babik[*], Michal Laclavík[*], Ivana Budinská[*], Ladislav Hluchý[*]
marek.ciglan@savba.sk

Abstract: In this paper we describe corporate memory which can support multiple knowledge acquisition, organization and maintenance tools. Memory holds and manages documents and related information and knowledge processed and created by such tools. Tools can work with several types of data such as documents, data in relational database and semantic data. Such diversity of information is needed due to different tools use which require and produce data in such form. Corporate Memory is being developed as part of Slovak national project NAZOU SPVV 1025/04.

Key Words: Knowledge Memory, Semantics

1 Introduction

Recently the importance of knowledge management relates to the huge amount of knowledge that is available for wide society of information systems' users. The influence of information and communication technology has been changing the society. Bringing new ideas and technologies for any users requires adjusting of presenting knowledge to the special users' requirements. Each system that aims to provide wide society of user customized information has to deal with developing kind of organizational memory. Corporate memory (CM) plays crucial role in the process of knowledge system development. CM provides two basic tasks: storing of data, information and knowledge and their innovation and maintenance. Data stored in CM are either in the form of their source form (e.g. HTML, pdf, etc.) or in the form of ontologically organized data, data in database system, etc.

Corporate memory that is developed within the NAZOU project serves as a tool for exchange data, information and knowledge among many components of the system. The project NAZOU (Research and Development of Tools for Knowledge Discovery, Maintenance and Presentation, a Slovak national project SPVV 1025/04) has started in September 2004 and it is focused on discovery, maintenance and presentation of knowledge. The NAZOU is applicable in any information domain dealing with offers. It can support gathering offers from internet, offer knowledge processing, searching navigating and presentation.

The Pilot application is Job Offer search application, where tools are used to find, download, categorize, annotate, search and display job offers to job seekers.

2 State of the Art

2.1 Semantic Knowledge Memories

One of the well known Semantic Web frameworks is the Jena toolkit. It is a Java framework for building Semantic Web applications [1] available as open source software under a BSD license. Jena implements APIs for dealing with Semantic Web building blocks such as RDF and OWL. Jena's fundamental class for users is the Model, an API for dealing with a set of

[*] Institute of Informatics, Slovak Academy of Sciences, Dubravska cesta 9, Bratislava, 845 07, Slovakia

RDF triples. A Model can be created from the filesystem or from a remote file. Using JDBC, it can also be tied to an existing RDBMS such as MySQL or PostgreSQL.

There are three main models in Jena, namely JenaLocalModel, RDQLLocalModel and RDQLHybridModel. JenaLocalModel behaves as if the entire set of triples were locally in-memory. The RDQLLocalModel naively queries a Model using the RDQL query language instead of Jena's Model API. Very little is cached, so all user interaction through the web application ends up relying on accessing the store through non-optimal queries. Jena implements the Fastpath algorithm to take advantage of the native database engine for RDQL queries using table joins, which may result in some performance gains when using a Jena RDBMS store. The RDQLHybridModel uses RDQL queries to initialize a local in-memory cache from a store. All user interaction after initialization uses Jena's Model API on the cached data and does not require any communication with the store. Jena contains a rich set of features for dealing with RDF including an inference engine, the ability to reason using custom rules and OWL-based ontology processing.

The Semantic Web group at HP Labs also maintains Joseki, which is a web application for publishing RDF models [2]. It is built on Jena and, via its flexible configuration, allows a Model, represented as a set of files or within a database, to be made available on a specified URL and queried using a number of languages. Joseki's other query mechanisms include fetching the entire model, fetching only the direct relations to a particular node, and a subject-predicate-object query language similar to Jena's API.

Tucana Technologies [3] maintains an open source version of its commercial Tucana Knowledge Server called Kowari [4]. Kowari is an entirely Java based transactional, permanent triple store available under the Mozilla Public License. It has its own transactional database and supports several different methods of communication including RDQL via its implementation of the Jena Model and a custom language named iTQL (Interactive Tucana Query Language). Kowari allows connections through SOAP and RMI as well as implementations of interfaces from Jena and JRDF.

3store [5] is developed by The University of Southampton and used within the Advanced Knowledge Technologies (AKT) project [21]. 3store is a core C library that uses MySQL to store its raw RDF data and caches [6] and relies on Redland [7], an RDF interface project grounded in its C library, developed by Dave Beckett of Bristol University's Institute for Learning and Research Technology.

Sesame is an open source RDF database with support for RDF Schema inferencing and querying [8]. Written in Java, Sesame is intended to be run as a web application. Much like Jena, Sesame can use open source RDBMS's MySQL and PostgreSQL in addition to its in-memory database. Sesame also supports custom inferencing and can perform RDFS entailment. Despite its excellent querying performance, its native API called SAIL API is superseded by the Jena API.

RDF Schema Specific Database (RSSDB) [22] is a persistent RDF Store for loading resource descriptions in an object-relational DBMS (ORDBMS) by exploiting the available RDF schema knowledge. Querying of stored RDF descriptions is accomplished by RQL. RQL fully supports XML Schema data types (for filtering literal values), powerful grouping primitives (for constructing complex XML results) and aggregate functions (for extracting statistics).

2.2 File & RDB repositories

File repositories and relational databases are the most frequently used systems, in today's IT world, for storing information in electronic form. HTTP, FTP and WebDav are examples of widely used application level protocols, standards for accessing file repositories in the distributed information infrastructure. HTTP (hypertext transfer protocol) [9] is a protocol for distributed hypertext information access. Since 1990, it is used by World-Wide Web initiative as a primary communication protocol. Although HTTP was primary designed for accessing hypertext information, its capabilities reach far beyond hypertext use, allowing it's use for distributed object management systems and much more.

FTP is a component of the TCP/IP protocol stack's application layer [10]. FTP is a protocol for transferring files over TCP/IP network, where two parties are involved – the FTP server (data provider) and FTP client (data consumer).

The basic data access and transfer protocols are used by higher-level solutions for data repositories. Distributed File Systems (DTF) are examples of such data repositories solutions. To illustrate DTF functionality, we briefly describe Red Hat Global File System (GFS). GFS is open source cluster software [11], that allows manage a cluster of servers, as if it were one server. GFS allows a cluster of Linux servers to share data in a common pool of storage, allowing simplifying data infrastructure; scaling clusters seamlessly, adding storage or servers on the fly and manage storage capacity as a whole, not by partitions.

OGSA-DAI [12] is the middleware product allowing access and integration of data from various separate sources via the distributed computing infrastructures. It allows exposure of data resources, such as relational or XML databases and file storages, providing also components for querying, transforming and delivering data in different ways. With clearly defined API and Web Services-based implementation, OGSA-DAI is becoming de-facto standard for data grids systems.

Internet search engines developers also deals with the problem of data storage. The architecture of data storage that is used by up-to-date internet search engines is very similar to the corporate memory architecture. Well known search engine Google [13] stores each searched HTML document in a sequence storage (in the form of comprised packets), that are described by ISAM index [14]. Another component of the Google storage is dictionary with about 14 millions words, list of word occurrences within documents together with indexes and inverted indexes. Dictionary and indexes are maintained by relational databases. Very similar architecture has also search engine Inktomi [15], with storages and index databases that are stored in distributed system connected by HTTP protocol. Inktomi serves for search engines of Microsoft, Yahoo! Companies, etc.

3 Architecture of Corporate Memory

Suggested architecture of corporate memory (CM) respects requirements that are given by the distributed character of knowledge system.

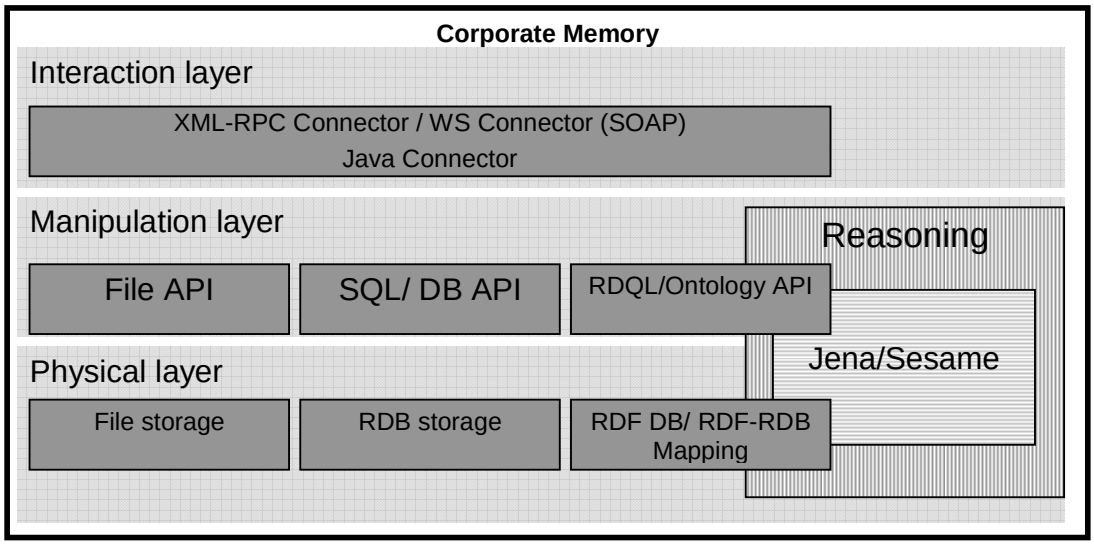

Figure 1: Corporate Memory architecture

Corporate Memory is accessible for other components using relevant client. Each part of CM has client implementation (see figure 2). Core of the CM is running as XML-RPC server and other components can call relevant client method via XML-RPC.

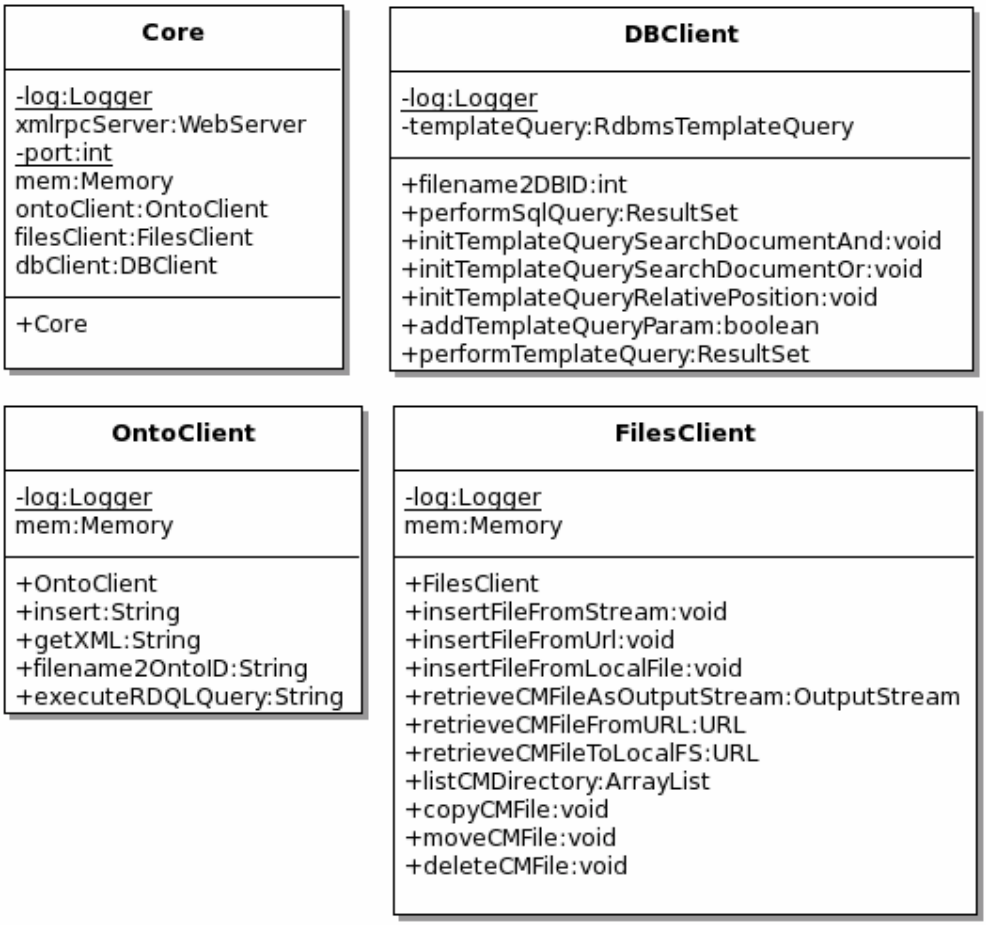

Figure 2: UML Class diagram of main CM classes of Interaction Layer

Corporate memory is organized into three layers (see figure 1):

- **Physical layer** that contains a standard file system, database system, and ontological models,
- **Manipulation layer** that provides acces to the stored data and information for other components of knowldge system. That means storing knowledge and information into physical layer, indexing, anotationand organization of stored data and their maintenance before presenting.
- **Interaction layer** that is responsible for interaction of CM with other system's componets.

3.1 Semantic part of Corporate Memory

Semantic part of the corporate memory is responsible for providing user interfaces for querying and manipulating the semantic content of the memory as well as providing the physical backend for persistent and transient storage of the semantics. The semantic model of the Web content is represented in the form of ontologies. The common language that we have used for capturing the ontologies is the Web Ontology Language (OWL) [16]. OWL facilitates machine interpretability of the Web content by providing additional vocabulary along with formal semantics. In order to support maximum expressiveness while retaining computational completeness and decidability we have used two light versions of the full OWL, namely, OWL-DL and OWL-Lite [16]. The semantic part of the corporate memory is also responsible for maintaining the knowledge expressed in the form of RDF and RDFS language.

The architecture has two main parts; the first one is the core interface, which provides transparent access to the underlying knowledge repositories and reasoners. The second one is the OntoClient interface, which defines the possible interactions between the components of the system and the semantic part of the corporate memory. This way it is possible to support multiple backends, while maintaining the common user interface used by the components.

The components can manipulate semantic part of memory using OntoClient, which is shown in figure 2. Current implementation of the corporate memory uses Jena API and MySQL storage backend; however since the OntoClient interfaces are independent of the backend and API, we intent to extend the client to support also the Sesame API and query interfaces.

OntoClient consist of 4 main methods:

- *Insert()* – inserts OWL models in XML/RDF format into Onto CM
- *getXML()* – returns plain XML of resource from Onto CM
- *filename2OntoID()* – converts filename to RDF ID of resource in CM
- *executeRDQLQuery()*- returns RDF ID result list for given RDQL query

3.2 File & RDB repository

The part of corporate memory dedicated to file management provides a way for manipulating the file storage using unified application interface, making actual physical file storage transparent to the user or application. This virtualization allows the information and knowledge management applications to access file storage in a uniform way if the file storage resides on the same computing device as the application is running on, as well as in the case that the file storage service is hosted at another computing resource.

In the current implementation the corporate memory's file storage is realized as a directory subtree of a file system directory tree. Support for distribution of the storage among multiple storage resources is not presently available.

Operations provided by CM for file management are: list, copy, move, delete files and directories in CM; files can be inserted to CM from file input streams or by specifying local file path, or by specifying URL of a document. Files can be retrieved from CM in the form of the input stream, can be saved to defined location in local filesystem or application can request an URL of a file stored in CM.

File management part of CM consist of core operations implementation and the client toolkit. Client toolkit can be configured to access the CM's file storage through local java API, XML-RPC call or through Web Service interface. Web Service interface to CM file storage is realized by OGSA-DAI framework.

The relational database management part of the CM was also designed with virtualization concept in mind, making actual database system and database connection object transparent to the client applications. Advantage of this approach is to make possible to access distributed resources in the same way as the local database system, from the application point of view. This allows us to use CM implementation on a single computer as well as on a set of servers with clearly separated functionalities (e.g. database server, file storage server, ontology management server, set of application servers).

RDB part of CM provides interface to execute basic SQL queries, as well as interfaces to execute predefined queries over common database structures. In the pilot implementation, the common database structures of CM store data about documents that serve as the input data for information and knowledge management applications. Indexed content of those input documents is also available in the RDB part of CM for enabling fast, content-based, identification of required documents from the large collection.

Application developers can provide their own predefined queries for application specific relational data. New predefined queries can be plugged-in the CM, provided they satisfy CM RDB query interface.

As the file storage part, the RDB part of CM is accessible through local java API, XML-RPC call or through Web Service interface. Web Service interface to CM file storage is realized by OGSA-DAI framework.

As the file and RDB management services of CM are still in prototype state of implementation, no security mechanism is provided. Standard security mechanism - username/password as the authentication tokens will be implemented in the future work.

4 Example of Use

In this chapter we describe example how NAZOU components work with information and knowledge stored in CM. On Figure 3 data transformation chain is presented. On each stage information or knowledge about files and offers is accumulated.

Following tools or components transform, generate and manipulate data, information and knowledge in CM in listed order:

- RIDAR (Relevant Internet Data Resource Identification) connects to existing search engines and identify relevant web resources
- WebCrawler and ERID (Estimate Relevance of Internet Documents) [17] recursively explore web resources and store
- DocConverter transforms documents to TXT format.

- OSID (Offer Separation for Internet Documents) extract offers (e.g. job offers) from document. If there is more offers on one document, or if there is only one it select offer without page header, footer, menu, banners and other offer not related stuff.
- DaiDocIndexing ,DaiDocSearch and JDBSearch [18] index text documents and offers; this allow other tools (searching, clustering) to use indexes for further processing.
- Ontea (**On**tology based **te**xt **a**nnotation) [19] annotates text version of offers by ontology individuals which are detected via regular expressions as relevant semantic properties of the offer. Ontea thus create ontology form of offers from file offer version according to defined domain ontology.
- Tools Prescott and faceted browser [20] support presentation, which transforms ontological data to XML and XML is further transformed to HTML via XSL. Indexes, found clusters or tool JobClusterNavigator are also used by presentation to search, categorize and navigate in offers accumulated in CM.

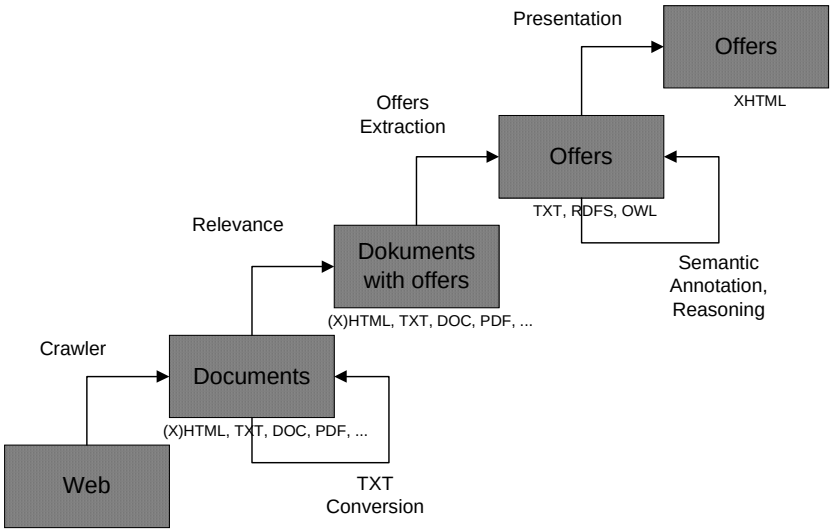

Figure 3: Data Transformation Chain

In given example you can see chain of almost independent tools, which are integrated around proposed corporate memory. The memory works with 3 types of data – files, relational data and semantic data, but fundamental conversion between the data types and formats is supported by chain of independent tools.

5 Conclusion

In this paper architecture and first implementation of corporate memory was presented. Such architecture supports chain of almost independent tools, which are integrated around proposed corporate memory. As shown in previous chapter Corporate Memory is integrated with independent tools. Such tools and Corporate Memory presents useful results to potential user of system for given application. The Memory was designed and built based on previous experience from R&D projects and it is further developed and used in several ongoing national and international projects.

This work has been supported by the Slovak national project NAZOU SPVV 1025/04, K Wf-Grid EU RTD IST FP6-511385, RAPORT APVT-51-024604, VEGA No. 2/6103/6.

References

1. HP & Sourceforge: *Jena Semantic Web Library*. http://jena.sourceforge.net/, 2006
2. Sourceforge: *Joseki - A SPARQL Server for Jena*. http://www.joseki.org/, 2006
3. Tucana Technologies, Inc.: *Tucana Knowledge Server*. http://www.tucanatech.com/, 2006
4. Open Source: *Kowari Metastore*. http://www.kowari.org/, 2006
5. Sourceforge: *3store*. http://sourceforge.net/projects/threestore/, 2006
6. MySQL: *MySQL - The world's most popular open source database*. http://www.mysql.com/, 2006
7. Dave Beckett: *Redland RDF Application Framework.* http://www.redland.opensource.ac.uk/, 2006
8. Open Source: *Sesame.* http://www.openrdf.org/, 2006
9. The Internet Society: *HTTP Protocol*. http://www.w3.org/Protocols/rfc2616/rfc2616.html, 1999
10. D. J. Bernstein: *FTP - File Transfer Protocol - Reference manual*. http://cr.yp.to/ftp.html, 2006
11. RedHat*: Red Hat Global File System.* http://www.redhat.com/en_us/USA/home/solutions/gfs/, 2005
12. The University of Edinburgh: *OGSA-DAI project*. http://www.ogsadai.org.uk/, 2005
13. Gonsalves, A.: *Google Gains Search-Engine Market Share.* In InformationWeek, http://www.informationweek.com/story/showArticle.jhtml?articleID=171202724, 3rd October, 2005
14. Brin, S., Page, L.: *The Anatomy of a Large-Scale Hypertextual Web Search Engine.* Stanford University, USA.
15. Brewer, E.A.: *Inktomi Architecture.* UC Berkley. http://www.acm.org/sigs/sigmod/disc/disc99/disc/nsf_acad_ind/brewer/index.htm, 2005
16. W3C: *The Web Ontology Language*, http://www.w3.org/TR/2004/REC-owl-features-20040210/, 2004
17. GATIAL E., BALOGH Z., LACLAVIK M., CIGLAN M., HLUCHY L.: *Focused Web Crawling Mechanism based on Page Relevance.* In: Proceedings of ITAT 2005 Information Technologies - Applications and Theory, Peter Vojtas (Ed.), Prirodovedecka fakulta Univerzity Pavla Jozefa Safarika v Kosiciach, 2005, pp.41-46. Slovakia, September 2005. ISBN 80-7097-609-8
18. Lencses R.: *Indexing for the Information Retrieval System supported with Relational Database*. In Sofsem 2005 Communications (editors: Vojtas et al), Slovenska informaticka spolocnost, Bratislava, 2005
19. LACLAVIK M., GATIAL E., BALOGH Z., HABALA O., NGUYEN G., HLUCHY L.: *Semantic Annotation based on Regular Expressions.* In: Proceedings of ITAT 2005 Information Technologies - Applications and Theory, Peter Vojtas (Ed.), Prirodovedecka fakulta Univerzity Pavla Jozefa Safarika v Kosiciach, 2005, pp.305-306. Slovakia, September 2005. ISBN 80-7097-609-8.
20. NAVRAT P., BIELIKOVA M., ROZINAJOVA V.: *Methods and Tools for Acquiring and Presenting Information and Knowledge in the Web.* CompSysTech'2005 http://ecet.ecs.ru.acad.bg/cst05/Docs/cp/SIII/IIIB.7.pdf, 2005
21. AKT Technologies: *3store from the University of Southampton.* http://www.aktors.org/technologies/3store/, 2004
22. FORTH Institute of Computer Science: *The RDF Schema Specific DataBase (RSSDB).* http://139.91.183.30:9090/RDF/RSSDB/, 2006