KAON - Towards a large scale Semantic Web

Erol Bozsak, Marc Ehrig, Siegfried Handschuh, Andreas Hotho, Alexander Maedche, Boris Motik, Daniel Oberle, Christoph Schmitz, Steffen Staab, Ljiljana Stojanovic, Nenad Stojanovic, Rudi Studer, Gerd Stumme, York Sure, Julien Tane, Raphael Volz, Valentin Zacharias

Forschungszentrum Informatik FZI, 76131 Karlsruhe,
http://www.fzi.de/wim
Institute AIFB, University of Karlsruhe, 76128 Karlsruhe,
http://www.aifb.uni-karlsruhe.de/WBS

Abstract. The Semantic Web will bring structure to the content of Web pages, being an extension of the current Web, in which information is given a well-defined meaning. Especially within e-commerce applications, Semantic Web technologies in the form of ontologies and metadata are becoming increasingly prevalent and important. This paper introduce KAON - the Karlsruhe Ontology and Semantic Web Tool Suite. KAON is developed jointly within several EU-funded projects and specifically designed to provide the ontology and metadata infrastructure needed for building, using and accessing semantics-driven applications on the Web and on your desktop.

1 Introduction

The Web in its' current form is an impressive success with a growing number of users and information sources. Tim Berners-Lee, the inventor of the WWW, coined the vision of a Semantic Web in which background knowledge on the meaning of Web resources is stored through the use of machine-processable (meta-)data. The Semantic Web brings structure to the content of Web pages, being an extension of the current Web, in which information is given a well-defined meaning.

Thus, the Semantic Web will be able to support automated, electronic services using semantics-based descriptions. These descriptions are seen as a key factor to finding a way out of the growing problems of traversing an ever expanding Web. In this sense, ontologies and metadata are becoming increasingly prevalent and important in a wide range of e-commerce applications.

The technical foundation of the Semantic Web is RDF (Resource Description Framework) which provides a generic core data model. Several software components, such as parsers, schema and metadata editors, repositories, have already been developed. However, they generally fail to meet the requirements for sophisticated e-Commerce projects. To support advanced applications much more specialized, comprehensive and intregated tools are required.

The Karlsruhe Ontology and Semantic Web Tool Suite (KAON) builds on available resources and provides tools for the engineering, discovery, management, and presentation of ontologies and metadata. It establishes a platform needed to apply Semantic

Web technologies to e-commerce and B2B scenarios. Because of that, important design goals were robustness and scalability, since these are key quality factors for any enterprise application. In this paper the vision and the current status of KAON are presented. The official KAON community web site [1] also provides up-to-date information about the project and allows downloading the newest version of the software.

The paper is organized as follows: Section 2 introduces the layered architecture and technologies underlying the Semantic Web. Section 3 collects and summarizes requirements for an infrastructure for semantics-based Services and applications. Subsequently we present the formal ontology model behind KAON. This ontology model is implemented within the conceptual architecture section that is presented in section 5. The current status of the actual implementation effort is described briefly in section 6. Before we conclude, we give a short overview on related work and provide an overview on the next steps within KAON.

2 The Semantic Web

The term ”Semantic Web” encompasses efforts to build a new WWW architecture that enhances content with formal semantics. This will enable automated agents to reason about Web content, and carry out more intelligent tasks on behalf of the user. ”Expressing meaning” is the main task of the Semantic Web. Tim Berners-Lee has conceived a five layer architecture for the Semantic Web which is presented in the followin

XML - The syntax layer XML allows to markup arbitrary content by means of nested, attributed elements. The names of these elements don’t say anything about what the structure means, therefore further means are required for the Semantic Web and the role of XML is reduced to a syntax carrier.

RDF - The data layer RDF allows the encoding, exchange and reuse of structured metadata. Principally, information is represented by very generic means, i.e. directed partially labeled pseudographs. This graph may be serialized using XML. Contrary to XML, RDF allows to assign global identifiers to resources and allows to refer and extend statements made in other documents. This feature is the main motivation for its use as an data layer.

The ontology layer The third basic component of the Semantic Web are ontologies. Ontologies describe formal, shared conceptualizations of a particular domain of interest [4]. This description can be used to describe structurally heterogeneous and distributed information sources such as found on the Web.

By defining shared and common domain theories and vocabularies, ontologies help both people and machines to communicate concisely, supporting the exchange of semantics and not only syntax.

The basic building block for ontologies are concepts, which are typically hierarchically organized in a concept hierarchy. These concepts can have properties which establish named relations to other concepts. Several representation languages have been

[1] http://kaon.semanticweb.org

proposed for the specification of ontologies. Section 4 provides a concise description of the representation language used within KAON. Since RDF is very generic ontologies can be stored in RDF.

The logic layer The logic layer consists of rules that enable inferences, e.g. to choose courses of action and answer questions.

Current research is mainly focused on the first three layers - which is also the focus of this paper. Our formal ontology model (cf. section 4) includes means to extend the ontology by rules in an arbitrary logic language. This is a first step towards the transition to this fourth layer of the Semantic Web.

The proof layer A proof layer has been conceived to allow the explanation of given answers generated by automated agents. Naturally, you might want to check the results deduced by your agent, this will require the translation of its internal reasoning mechanisms into some unifying proof representation language.

3 Requirements

While building semantics-based applications within E-Commerce, Knowledge Management, Web Portals, etc. we have gained insight into application features that warrant a success. Based on that experience and in order to enabling reuse across projects, we have decided to build a framework addressing these issues. An extensive requirement gathering process was undertaken to come up with a set of requirements that such framework must fulfill. The following key requirements were identified:

- **Accessability**: A framework should enable loose coupling, allowing access through standard web protocols, as well as close coupling by embedding it into other applications. This should be done by offering sophisticated standard APIs.
- **Consistency:** Consistency of information is a critical requirement of any enterprise system. Each update of a consistent ontology must result in a ontology that is also consistent. In order to achieve that goal, precise rules must be defined for ontology evolution and an evolution service implementing these rules has to be provided. Also, all updates to the ontology must be within transactions assuring the usual properties of atomicity, consistency, isolation and durability (ACID).
- **Concurrency:** It must be possible to access and modify information concurrently. This may be achieved using transactional processing, where objects can be modified at most by one transaction at the time.
- **Durability**: An almost trivial requirement easily accomplished by reusing existing database technology. A sophisticated storage system must offer facilities for replication: for often used ontologies redundant copies must be maintained to address scalability and availability problems.
- **Security**: Guaranteeing information security means protecting information against unauthorized disclosure, transfer, modification, or destruction, whether accidental or intentional. To realize it, any operation should only be accessible to properly

authorized agents. Proper identity of the agent must be reliably established, by employing known authentication techniques. Sensitive data must be encrypted for network communication and persistent storage. Finally, means for auditing (logging) of sensitive operations should be present.
- **Reasoning**: Reasoning engines are central components of semantics-based applications. Our tools should have access to those engines which provide the reasoning services required to fulfill a certain task.
- **Mapping**: Often multiple ontologies have to be supported by an ontology system. This support is only complete if means for mapping and mediating between heterogeneous ontologies are provided.
- **Discovery:** We assume that data in the Semantic Web will be distributed. Therefore means for ontology-focused and intelligent discovery of metadata are required. Based on a semantic description of the search target, the system should be able to discover relevant information on the Web.
- **Internationalization:** The framework should allow users to create ontologies and their instances in different languages and should support non-Latin character sets.
- **Formal ontology** The formal semantics specified by an ontology must be unambiguous and clear.

4 Formal model for ontologies

Formal semantics of ontologies are an important requirement for us. The notion and formal semantics of ontologies currently supported by our tools is therefore presented in this section.

Definition 1. *A* core ontology *is a structure*

$$\mathcal{O} := (C, \leq_C, R, \sigma, \leq_R)$$

consisting of (i) two disjoint sets C and R whose elements are called concept identifiers *and* relation identifiers*, resp., (ii) a partial order $\leq_C$ on C, called* concept hierarchy *or* taxonomy*, (iii) a function $\sigma: R \rightarrow C^+$ called* signature*, and (iv) a partial order $\leq_R$ on R, called* relation hierarchy*, where $r_1 \leq_R r_2$ implies $|\sigma(r_1)| = |\sigma(r_2)|$ and $\pi_i(\sigma(r_1)) \leq_C \pi_i(\sigma(r_2))$, for each $1 \leq i \leq |\sigma(r_1)|$.*

Often we will call concept identifiers and relation identifiers just *concepts* and *relations*, resp., for sake of simplicity.

Definition 2. *For a relation $r \in R$ with $|\sigma(r)| = 2$, we define its* domain *and its range by* $\mathrm{dom}(r) := \pi_1(\sigma(r))$ *and* $\mathrm{range}(r) := \pi_2(\sigma(r))$.

If $c_1 \leq_C c_2$, for $c_1, c_2 \in C$, then c_1 is a subconcept of c_2*, and c_2 is a* superconcept of c_1*. If $r_1 \leq_R r_2$, for $r_1, r_2 \in R$, then r_1 is a* subrelation of r_2*, and r_2 is a* superrelation of r_1.

If $c_1 <_C c_2$ and there is no $c_3 \in C$ with $c_1 <_C c_3 <_C c_2$, then c_1 is a direct subconcept of c_2*, and c_2 is a* direct superconcept of c_1*. We note this by $c_1 \prec c_2$.* Direct superrelations *and* direct subrelations *are defined analogously.*

Definition 3. *Let $\mathcal{L}$ be a logical language. A* $\mathcal{L}$-axiom system *for an ontology $\mathcal{O} := (C, \leq_C, R, \sigma, \leq_R)$ is a pair $A := (AI, \alpha)$ where (i) AI is a set whose elements are called* axiom identifiers *and (ii) $\alpha: AI \to \mathcal{L}$ is a mapping. The elements of $A := \alpha(AI)$ are called* axioms*.*

An ontology with $\mathcal{L}$-axioms *is a pair $(\mathcal{O}, A)$ where $\mathcal{O}$ is an ontology and A is a $\mathcal{L}$-axiom system for $\mathcal{O}$.*

Definition 4. *An ontology with $\mathcal{L}$-axioms $(\mathcal{O}, A)$ is* consistent*, if $A \cup \{\forall x: x \in c_1 \to x \in c_2 \mid c_1 \leq c_2\} \cup \{\forall \boldsymbol{x}: \boldsymbol{x} \in r_1 \to \boldsymbol{x} \in r_2 \mid r_1 \leq r_2\}$ is consistent.*

In the sequel, *ontology* stands for either a core ontology or an ontology with $\mathcal{L}$-axioms.

Definition 5. *A* lexicon *for an ontology $\mathcal{O} := (C, \leq_C, R, \sigma, \leq_R)$ is a structure*

$$Lex := (S_C, S_R, Ref_C, Ref_R)$$

consisting of (i) two sets S_C and S_R whose elements are called signs for concepts *and* relations*, resp., (ii) a relation $Ref_C \subseteq S_C \times C$ called* lexical reference for concepts*, where $(c, c) \in Ref_C$ holds for all $c \in C \cap S_C$, (iii) a relation $Ref_R \subseteq S_R \times R$ called* lexical reference for relations*, where $(r, r) \in Ref_R$ holds for all $r \in R \cap S_R$.*

An ontology with lexicon *is a pair $(\mathcal{O}, Lex)$ where $\mathcal{O}$ is an ontology and Lex is a lexicon for $\mathcal{O}$.*

The requirement of support for internationalization is provided via such a lexicon.

Definition 6. *A* knowledge base *is a structure*

$$KB := (C_{KB}, R_{KB}, I, \iota_C, \iota_R)$$

consisting of (i) two sets C_{KB} and R_{KB}, (ii) a set I whose elements are called instance identifiers *(or* instances *or* objects *for short), (iii) a function $\iota_C: C_{KB} \to \mathfrak{P}(I)$ called* concept instantiation*, (iv) a function $\iota_R: R_{KB} \to \mathfrak{P}(I^+)$ called* relation instantiation*.*

Such instances are technically represented in RDF and may be physically located in several documents.

Definition 7. *An* instance lexicon *for a knowledge base $KB := (C_{KB}, R_{KB}, I, \iota_C, \iota_R)$ is a pair $IL := (S_I, R_I)$ consisting of (i) a set S_I whose elements are called* signs for instances*, (ii) a relation $R_I \subseteq S_I \times I$ called* lexical reference for instances*. A* knowledge base with lexicon *is a pair (KB, IL) where KB is a knowledge base and IL is an instance lexicon for KB.*

5 Conceptual Architecture

In this section we introduce the general architecture that is the basis of KAON. We mainly distinguish three layers within our conceptual architecture, namely the data and remote service layer, the middleware layer and the applications and services layer. Figure 1 depicts this layered architecture.

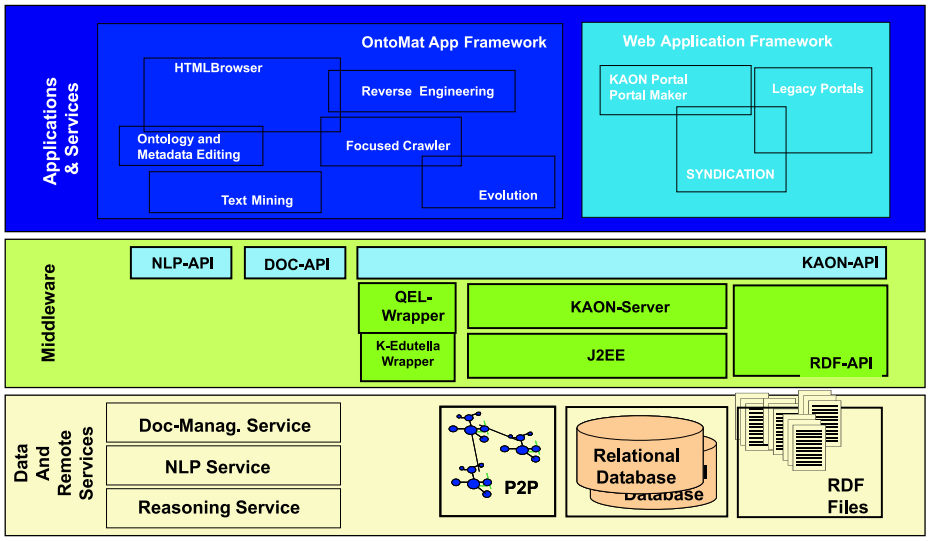

Fig. 1. KAON Architecture

Applications and Service Layer: Application and service clients can be either *(i)* the components of the Java-Application-based OntoMat application framework or *(ii)* applications extending the web-based KAON-PORTAL and web site management framework. All application clients connect with the middleware layer via KAON API, an application programming interface accessing ontology elements. The API realizes the application model by providing a set of object-oriented abstractions of ontology elements. Application clients provide views and controllers for model realized by KAON API.

Middleware Layer: The primary role of the middleware layer is to provide an abstraction for ontology access. Its second role is the dynamic instantiation and delegation of requests to the underlying external services layer. The first role is implemented by the KAON API, which isolates clients from different API implementations and provides a unified interface. For example, a transient ontology model is provided by implementing the KAON API on top of RDF files. This implementation may then be used for in-memory processing of ontologies stored in files and stand-alone deployment of tools. KAON RDF Server is a data source specialized in storing RDF data. It allows concurrent modification, supports transactions and persistence. Non-RDF data sources may be accessed using other implementations of the KAON API, thus creating an ontology-compatible view of data not in a format according to Semantic Web standards.

The dynamic instantiation and delegation of requests to services is out of scope for this paper. The implementation relies on the framework provided by the Java Management Extensions (JMX).

Data and Remote Service Layer: This layer has several roles. First it offers access to physical data stores such as databases or file systems. Second it groups external services such as reasoning engines, the aforementioned mapping engine etc. and announces availability to the middleware layer.

6 Implementation of the Conceptual Architecture – The KAON Tool Family

This section explains how the conceptual architecture has been implemented describing its current status and underlying technologies. We mainly distinguish between tools which are intended to be directly used by users (*frontend tools*) and tools which are intended to be used within other applications (*backend tools*).

6.1 FrontEnd Tools

KAON PORTAL KAON PORTAL is an ontology-based web portal generator. The underlying idea of the KAON PORTAL application is that based on a given ontology, a web application is automatically generated. It is important to mention that KAON Portal is capable to automatically provide metadata-driven services on the Web. Therefore all data is additionally published in RDF.

OntoMat Application Framework OntoMat is a component-based ontology and metadata application framework. Initially it was developed as an ontology-based annotation and HTML markup tool. On account of its flexible, component-based architecture it was chosen to be the platform to realize other functionalities, e.g.:

- It includes a multi-lingual Ontology Engineering and Evolution Environment (OntoMat-SOEP) that allows the manual development and maintenance of ontologies.
- It provides means for database reverse engineering via OntoMat-REVERSE, a tool and approach that allows mapping JDBC-compliant relational databases onto ontologies [6].
- The OntoMat-SILVA tool implements a comprehensive methodology for ontology mapping and mediation consisting of means for normalization heterogeneous ontologies, detecting similarities, providing graphical means for specifying associations and semantic bridges between two ontologies.
- The OntoMat-Catyrpel tool which provides user-support for the ontology-focused discovery of RDF-based metadata in the Web.

It is important to mention that due to the flexible and component-based approach each component is able to communicate with all other components realized using OntoMat.

6.2 Backend Tools — KAON API and Server

Backend tools provide the middleware layer and offer access to external services. Due to lack of space we will restrict our attention to the two focal components in this layer, namely the KAON API and KAON Server.

KAON API KAON API is the focal point of the middelware layer. It provides objects representing various pieces of an ontology, such as Concept, Relation, Attribute or Instance, objects for creating and applying changes to ontology entities as well as objects providing query facilities. KAON API itself doesn't realize persistence, concurrency or security. Rather, it relies on lower layers to provide these features.

The Observable design pattern is used for notifications about model changes, thus achieving low coupling between model and associated views. All changes to application model, whether local or remote, are propagated to registered listeners allowing them to display model updates immediately as they happen. Java Messaging Service (JMS) is used to propagate change notifications in distributed environment. The API is entirely based on interfaces, allowing users to choose the appropriate implementation, depending on the needs.

To provide ontology-compliant access to data stored in existing systems, such as relational or XML databases, special mapping implementations may be used. These implementations must align the respective data sources to ontologies. The conversion is dynamic - all modifications to the ontology and all queries are transformed and propagated to the underlying data source.

The KAON API is responsible for providing consistency of the underlying ontology. All access to the API is performed through a dedicated evolution strategy whose purpose is to define and implement a set of change rules. For example, when a concept is removed from an ontology, it must be decided what to do with its subconcepts - they may be deleted, attached to the parent of the deleted concept or attached to ontology's root concept. Several evolution strategies have been implemented for each of these policies, allowing the user to choose the appropriate one when the ontology is instantiated. Finally, in order to improve performance, KAON API allows using a pluggable caching scheme. In that way many costly requests to KAON SERVER may be avoided and the overall application performance increased.

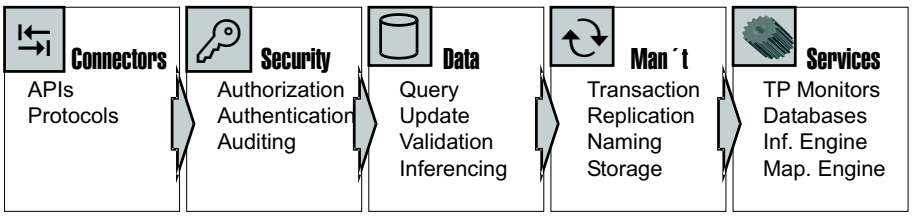

Fig. 2. KAON Server - Modules and Request Processing

KAON SERVER KAON SERVER is responsible for providing a persistent, transactional and secure RDF repository accessible by multiple users are the same time. It is realized within J2EE framework, technically it is therefore a component hosted by EJB application servers. The conceptual architecture of the system follows a Layers architectural pattern, as presented in Figure 2.

connectors layer Several APIs are provided to connect to KAON SERVER: An RDF API is used for accessing RDF data, a querying API is used for RDF querying and inferencing. Additionally a special remote implementation of the KAON API is provided. Since KAON SERVER is realized within J2EE framework, the SERVER is accessible to non-Java clients using the CORBA-IIOP protocol.

Security layer makes server operations available to the client only if the caller is properly authenticated and authorized to access them. Authentication and authorization are implemented using the Java Authentication and Authorization Service (JAAS) allowing easy integration to any existing (corporate) security services. In a nutshell, JAAS provides role-based autorization and authentication. Users are mapped into abstract roles, and a set of privileges is determined for each role.

Data access layer This layer allows management of RDF model elements, inferencing and querying. Queries are supported using RDF-QEL query language designed in the Edutella project [5]. KAON SERVER does not implement inference itself but interfaces to other systems. The integration with these systems is seamless - the users of KAON Server do not distinguish between inferred and ground facts.

Management layer encapsulates all ”basic” services such commonly found in information systems. The transaction management system is responsible for ensuring the commonly known ACID transaction properties. The replication service must ensure all external systems work with the same data sets (e.g. inference engine must be kept in synchrony with the persistent storage). By interplaying with the naming service, the replication service can also manage duplicate RDF models to enhance scalability and availability. The naming service maps model identifiers (as presented by URIs) to persistent identifiers (URNs) and keeps information about the location of the information. Finally, system configuration modules are realized in this layer.

External services are systems and services external to KAON Server. Databases are used for persisting RDF model data. Inference engines are reused to offer reasoning capabilities. Transaction Processing Monitors ensure transactional integrity if data is replicated to external systems.

7 Resume

With the upswing and proliferation of ontologies and metadata, the need for comprehensive managing infrastructure has been recognized recently. A comprehensive overview and state-of-the-art survey on ontology library systems with respect to the dimensions management, adaptation and standardization has been provided by [2]. Whereas these ontology library systems mainly focus on ontology storage and reuse, our approach provides a RDF-based framework including ontology management for semantics-driven applications. Comparing to available RDF data stores[2] our approach is the only available RDF data store dealing with replication. An approach that comes close to our

[2] See http://www.w3.org/2001/05/rdf-ds/DataStore lists existing RDF data stores.

open-source framework is the commercial system, ontology builder and server, proposed in [1]. Nevertheless, in contrast to this system, our approach is completely based on RDF and is therefore Semantic Web conform. Additionally we provide access to existing relational data sources via OntoMat-REVERSE.

In this paper we have introduced KAON, the Karlsruhe Ontology and Semantic Web Tool Suite. We have gathered requirements for large scale semantic web systems and presented a formal ontology model that can be mapped to several logic languages and may be extended with further axioms stated in these logic languages. Thereby we have established a first step towards the fourth layer of the Semantic Web. We have presented a conceptual architecture that allows to realize the established requirements and presented the first steps taken by us towards implementation of this architecture. The tools mentioned within the paper are freely available via the web at http://kaon.semanticweb.org/.

In the future we will focus on making existing corporate data sources available to ontology-based applications. We are also working on extensions of the formal ontology model to provide a more expressive core language. We are also working on a query language for ontologies which will offer view-support.

Additionally we will further extend the accessibility of the system towards peer-to-peer systems enabling a new level of knowledge interchange and ontology-based communication.

References

1. Aseem Das, Wei Wu, and Deborah McGuinness. Industrial strength ontology management. In *Proceedings of the First Semantic Semantic Web Working Symposium, SWWS-01, Stanford, USA, August 2001*, 2001.
2. Ying Ding and Dieter Fensel. Ontology library systems — they key to successful ontology re-use. In *Proceedings of the First Semantic Semantic Web Working Symposium, SWWS-01, Stanford, USA, August 2001*, 2001.
3. Erich Gamma, Richard Helm, Ralph Johnson, and John Vlisside. *Design Patterns*. Addison-Wesley, 1995.
4. T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 6(2):199–221, 1993.
5. Wolfgang Nejdl, Boris Wolf, Changtao Qu, Stefan Decker, Michael Sintek, Ambjoern Naeve, Mikael Nilsson, Matthias Palmer, and Tore Risch. Edutella: A p2p networking infrastructure based on rdf. In *In Proceedings of the 11th World Wide Web Conference — WWW-11, Hawaii, USA, 2002*, 2002.
6. L. Stojanovic, N. Stojanovic, and R. Volz. Migrating data-intensive Web Sites into the Semantic Web. In *to appear in: Proceedings of the ACM Symposium on Applied Computing SAC-02, Madrid, 2002*, 2002.