

Using Semantic Web Technologies for context-aware Information Providing to Mobile Devices

Fabian Abel
Institute Knowledge Based Systems
Appelstr. 4 - 30167 Hannover
Germany
Fabian.Abel@gmx.de

Jan Brase
L3S Research Center
Expo Plaza 1 - 30539 Hannover
Germany
Brase@kbs.uni-hannover.de

ABSTRACT

In this paper we discuss our ongoing implementation of a semantic web scenario. Visitors of the *L3S Research Center* should be enabled to make a self-guided tour by equipping them with a Pocket PC. On their tour they are provided with context-aware information about researcher projects and knowledge about the respective domain. The presented information is adapted to the specific information interests of the user. This scenario is based on the *Cooltown Project* by the HP labs.

1. BACKGROUND

1.1 The Semantic Web

By definition the Semantic Web is "an extension of the current web, in which information is given well-defined meaning, better enabling computers and people to work in cooperation" (see [1]). This extension has already proceeded. The W3C [17] defines the architecture of the Semantic Web as a layer model (see [2]) which is based on the Resource Description Framework (RDF) [14]. RDF takes advantage of URI, Namespaces, XML and XML Schema which are building the base of the RDF Specification. Standards like RDF and RDF Schema [13] and especially a high level standard like the Web Ontology Language (OWL) [16] allow developers to create a vocabulary of concepts. These vocabularies are called Ontologies. Ontologies provide a shared and common understanding of a domain that can be communicated between people and application systems like agents. They are developed to facilitate knowledge sharing and reuse [5]. In the simplest case, an ontology describes a hierarchy of concepts related by relationships [6]. In a more sophisticated ontology, suitable axioms are added in order to express other relationships between concepts and to constrain their intended interpretation.

1.2 Cooltown

The *Cooltown Program* [3], a program of HP Labs, acts in accordance to the definition of the Semantic Web as aforementioned. Given that an Ontology is a fundamental requirement to implement Semantic Web scenarios, a core aspect of the *Cooltown Idea* is that "everything - people, places and things - has a web presence". This aspect is of particular importance whenever web services or other applications do reasoning to humans. Other facets of the *Cooltown Vision* are that humans know the promise of mobile web services and that devices and services cooperate federated and context-aware.

In an implementation of a scenario, in which visitors of *Lasar Segall museum* in Sao Paulo, Brazil, are equipped with mobile devices, these core ideas of *Cooltown* were included (see [4]). We model on a similar scenario: Visitors of the *L3S Research Center* [9] should be enabled to make a self-guided tour by equipping them with a Pocket PC. On their tour they are provided with context-aware information about researcher projects and knowledge about the respective domain.

1.3 A Concrete Scenario:

Cooltown @ L3S Research Center

In order to point out the need for our architecture of a system that provides context-aware information by using Semantic Web Technology we introduce a concrete example: *Tom, who is working on his master thesis in the range of "Adaptive Hypermedia", visits the L3S Research Center in order to pick up literature. For that he equips himself with a Pocket PC and starts his tour. While he is walking around, websites are displayed on the Pocket PC giving him information about research projects, persons who work on these projects and suitable background knowledge. As these websites appear conformable to his actual position, Tom has the ability of having conversations with the corresponding researchers. When Tom passes a seminar room he is specially advised to a lecture entitled "Personalization on the Web" that initiates in 15 minutes, that perfectly fits to his research interests. Tom is looking forward to this lecture and clicks on a link to a corresponding paper and sends it to the next printer. Now a map is displayed on his mobile device that directs him to the printout. Tom has the feeling of being at the right place to do research for his thesis because most of the information that was displayed on the Pocket PC fits with his interests.*

As the provided information satisfies the users needs, the described scenario fits in very well in the *Cooltown Idea*. On some sections we will pick up this example to illustrate our approaches.

2. TECHNICAL REALISATION

2.1 Architecture

A general structure of the context-aware information providing system is shown in Figure 1. The system is distributed into a client and a server application. But different from regular tour guidance systems this is not a straight Push Technology as the client application and also the user can inherit an active part. We can differentiate two cases of how to cause a switch of website, that is displayed on the mobile device:

1. *change of position*: When Tom passes a seminar room (see section 1.3), the *position localizator* (see section 2.2) which is part of the client application indicates his new position and sends a direct request to the textit-Communication Servlet.
2. *user interaction*: By clicking on the link to the corresponding paper (see section 1.3) Tom takes an active part in requesting information.

Thus the tasks of the client application are outlined as: *position localization* (see section 2.2) and *providing browser functions*.

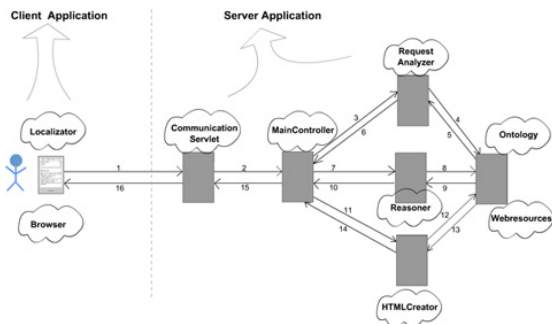


Figure 1: Architecture of the context-aware information providing system: It shows the different components of the system and indicates a general sequence of actions. The adjustment of the components also clarifies Types of components like communication components, controlling components, service components (RequestAnalyzer, Reasoner, HTMLCreator) and data components.

Although the server application does not take an active part in communication we also do not deal with a straight Pull Technology. If the server application is requested for information according the case *change of position*, this request does not reference to a specific website but only delivers the user's actual position. So the server application has to identify the information that suits to the current location and to the current user.

Thus main competences of the server application are: creating and publishing websites.

Before illustrating the sequence of action in creating websites we have to outline the character of data management: Information about a particular Researcher, Research Group or a Thing (i.e. a seminar room, see section 1.3) are called web resources and are stored in xml files, that are enriched with metadata. Interrelations between these web resources are defined in an ontology just like profiles, rules and other vocabulary (see section 3).

Before pointing out the interaction between requests, reasoning, data and the preparing of data we distinguish between four cases of websites:

1. *Type A Site*: Such websites are static; they are always created by the application in the same way. When Tom starts his tour, he first has to adjust his personal profile on a website in the Pocket PC.
2. *Type B Site*: These are sites that are generated according to a *personalized Style sheet*. As a result of Toms adjustments in his personal profile, a suitable Style sheet (*personalized Style sheet*) is assigned to him. This *personalized Style sheet* is used to create a lightly personalized website (according to i.e. language or background) out of a web resource.
3. *Type C Site*: We refer to *Type C Site* in the case of the preselection of web resources according to a personal profile. So the provided website is a *Type B Site* that was preselected.
4. *Type D Site*: In contrast to the other forms, these sites are not based on web resources in sense of xml/html files. *Type D Sites* appear as results of a search. For this purpose a new web resource is modelled only inside the application.

Depending on this cases the server application has to provide the information. As the server application must be able to differentiate these cases, it has to analyze the request. Thus the *Communication Servlet* passes the query to the *Request Analyzer* which is controlled by the *Main Controller*. The *Request Analyzer* delivers the results that are derived from the *Ontology* to the *Main Controller* (See Figure 1). Depending on the four pictured cases we have the following streams of action:

1. *Case "Type A Site"*: The *Request Analyzer* returns a specific URI that references to an already built website and is sent directly to the client application.
2. *Case "Type B Site"*: The *Request Analyzer* returns a specific URI that references to a web resource. Then the *Main Controller* calls the *Reasoner* for the *personalized Stylesheet* that is associated to the actual user and after that the *Main Controller* calls the *HTML Creator* which results in an URI for the generated site.
3. *Case "Type C Site"*: Getting the *personalized Style sheet* is equivalent to the preceding case. Additionally the *Reasoner* has to discover a suitable web resource by requesting the metadata of the resource. If successful,

a website is generated and the dedicated URI is sent to the client application.

4. *Case "Type D Site"*: The *Request Analyzer* returns a special task and corresponding parameters which are analyzed by the *Reasoner*. The *Reasoner* writes his results into a xml model. With that and with a standard style sheet the *Main Controller* calls the *HTML Creator* which results in an URI for the generated site.

In section 3 we address the issue of "Typ D Sites" in terms of *list of questions* which the system can handle.

2.2 Position Localization

To provide context-aware information not only in terms of *personalization* but also adapted to the facilities of the L3S we have to localize the position of the user.

In accordance with our architecture, outlined in section 2.1, the *Position Localizator* is part of the client application. So an implementation of the *Position Localizator* has to be realised with the components of the mobile device. As the Pocket PC of our scenario (see section 1.3) is equipped with a wireless Lan adapter pursuant to the standard IEEE 802.11b we use the following method:

The *Position Locator* is based on an analysis of the signal strength of the different access points that are set up in the environment. This environment, according to the scenario the L3S, is divided into sectors that correspond to research groups, special projects, persons etc. . This fragmentation is filed along with the ranges of signal strengths of each access point. These ranges have to result from several measurements performed under different conditions (like time of day). For detailed information we refer to the publication *On indoor position location with wireless Lans* [12].

At regular intervals the client application has to look up whether the actual signal strengths still match with the associated place. If a new place is detected a request is sent to the *Communication Servlet*.

3. USE OF SEMANTIC WEB TECHNOLOGY

huge variety of Semantic Web Technologies for providing websites containing context-aware information. We use OWL for defining a domain specific vocabulary so that content of web resources can be matched with users interests. Furthermore we need xml and xsl for creating websites containing the context-aware information.

3.1 Ontologies

A vocabulary of concepts for an information system described in the scenario (see section 1.3) requires definitions about the relationships between objects of discourse and their attributes. The W3C defines two standards that can be used to design an ontology:

RDF Schema [13]: RDF Schema (RDFS) allows the engineer of an ontology to create hierarchies of concepts (classes) and also hierarchies of attributes which specify a class.

Web Ontology Language (OWL) [16]: Because of the upward compatability of the Semantic Web Architecture all constructs of RDFS can also be used in an *OWL Ontology*.

Furthermore OWL allows to set *property restrictions* and to indicate whether a property is transitive, symmetric, functional or inverse to another property. Contrary to RDFS the value of a property can be allocated with an instance. This is a benefit in terms of defining properties, that describe relationships between classes. Supposed that in our scenario (see section 1.3) Tom wants to have a look at the members of a Working Group, the following properties *cooperatorOf* and *hasCooperator* can be used for determine them:

```
<owl:ObjectProperty rdf:ID="cooperatorOf">
  <rdfs:range>
    <owl:Class rdf:about="#WorkingGroup" />
  </rdfs:range>
  <owl:inverseOf>
    <owl:ObjectProperty rdf:about="#hasCooperator" />
  </owl:inverseOf>
  <rdfs:domain rdf:resource="#Person" />
</owl:ObjectProperty>
```

The W3C divides OWL into three syntax classes: OWL Lite, OWL DL und OWL Full. OWL DL is suited to be read by description logic reasoner.

Because of the larger complexity of constructs we choose OWL to implement the needed ontologies. According to our scenario (see section 1.3) there are several ontologies required:

1. *Researcher Ontology*: This ontology contains classes like Person, Organization and Publication. The *OWL Example* is an extract of it. The *Researcher Ontology* adopts to the *OntoWeb Ontology* [11].
2. *Field of Research Ontology*: Many classes of the *Researcher Ontology* have object properties like *Interests* and *Field of Research*. These properties allocate values of the *Field of Research Ontology*, making it some kind of glossary. A part of a conformable visualization is shown in Figure 2.
3. *Design Ontology*: In case of a *Type D Site* (see section 2.1) the *Reasoner* component has to add results of a search into a xml model. This xml model is build up the base of the website. The *Design Ontology* contains information about how to put the result elements into the xml tree (i.e. the headline of a website should not contain an explanation of a term but the term itself). Further this ontology includes assignments between Style sheets and Type of user.
4. *Request Ontology*: This ontology defines the vocabulary of requests. The *Request Ontology* is required by the *Request Analyzer* which has to detect what the user wants by analyzing the client's query. In addition this vocabulary is needed to create links that are embedded into the websites. Also the *list of questions*, which not define a static list of explicit questions, but rather types of questions the system can handle, are defined at this ontology.

It is obvious that the *Researcher Ontology* and the *Field of Research Ontology* describe primarily a vocabulary of concepts whereas the *Design Ontology* and the *Request Ontology* also include reasoning rules.

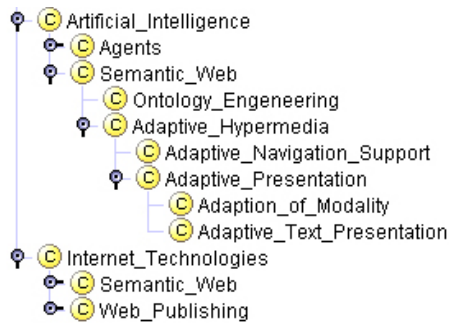


Figure 2: Extract of the *Field of Research Ontology*. It shows the hierarchical interrelations between classes. Note that the *subClassOf* relation can be applied to several classes (*multiple inheritance*): *Semantic Web* is a subclass of *Semantic Web* and of *Internet Technologies*

3.2 Reasoning

A major task of the *Reasoner* (see section 2.1) is discovering web resources that suit the user's interests. Each web resource contains metadata which describes the content of the web resource and which is noted in RDF. The *Reasoner* has to send queries to a web resource to detect whether the content is suitable to a given interest or not. These queries are modelled in RDQL [15]. **RDQL:** RDQL is an implementation of the SquishQL (SQL-like Query Language) [10]. It "regards RDF as triple data, without schema or ontology information unless explicitly included in the RDF source". Whereas RDF Statements are based on *resource*, *property* and *value* and can be interpreted as a graph with directed edges, RDQL provides a way of defining a graph pattern that is matched against a *RDF Statement Graph*. The metadata of a web resource quoted as a RDF model could for example be:

```

:publication1
  rs:title "Logic-Based Open Hypermedia for the Semantic Web" ;
  rs:author "Peter Dolog" ;
  rs:author "Nicola Henze" ;
  rs:author "Wolfgang Nejdl" ;
  rs:keyword "open hypermedia" ;
  rs:keyword "semantic web" ;
  rs:keyword "adaptive hypermedia" ;
  rs:keyword "ontologies" ;

```

To discover the keywords of this web resource the *Reasoner* queries:

```

SELECT ?resource ?value
WHERE (?resource <rs-namespace/keyword>
?value)

```

Resulting in:

resource	value
publication1	open hypermedia
publication1	semantic web
publication1	adaptive hypermedia
publication1	ontologies

3.3 Producing Websites

For client purposes, a website is composed of several pages according to the different *types of sites* (see section 2.1). Some are static, others are produced by applying different style sheets to the XML based web resources. If for example Tom browses a website of a working group (see section 1.3), the content of this website is described only in one xml file. For the required xml transformation we use the Extensible Style sheet Language Transformations (XSLT)[19]. Different template rules are defined in a style sheet. These template rules are matched against nodes of the source tree, corresponding to the xml web resource. XPath[18] is used to address these nodes. The template rules further contain a template which can be instantiated to form a HTML document as part of the result.

In view of the scenario (see section 1.3) we have to define template rules for static designs of pages (i. e. *welcome page*, *page for links*) on the one hand and specific designs adapted to the user types on the other hand. The second term leads to a *personalized Stylesheet* (see section 2.1).

4. CURRENT STATUS

The system discussed here is part of a bachelor thesis which will be finished in september. The major task of this thesis is an implementation of a context-aware information providing system according to the outlined scenario (see section 1.3). At present the client application (see section 2.1) is almost completed, without the *Position Locator*. We have developed a temporary solution that prompts the user of the Pocket PC to select his actual location.

The server application makes use of the *Jena Framework* [7] and the belonging RDF publishing server *Joseki* [8]. *Jena* provides Java APIs for RDF, OWL and RDQL. Currently the server application can handle a comparative high number of requests although not every ontology has a final status.

Towards the aspect of personalization we designed a few sample user profiles with corresponding stylesheets. Further we developed a method that enables the user to adjust his personal profile by using slider objects to define his interest towards a special field of research.

According to our schedule the estimated completion date is August 15th.

5. ACKNOWLEDGEMENTS

We would like to thank Prof. Nicola Henze who provides us with several advices in terms of Semantic Web Technologies. Further we thank Prof. Wolfgang Nejdl who worked out the idea to implement such a Semantic Web Scenario.

6. REFERENCES

- [1] Berners-Lee, T. and Hendler, J. and Lassila, O. , *The semantic web*
Scientific American, may, 2001
- [2] Berners-Lee, T., *Semantic Web Architecture*
<http://www.w3.org/2000/Talks/1206-xml2k-tbl/>
- [3] Cooltown Program, HP Labs Mobile Systems and Solutions
<http://www.cooltown.com>
- [4] Duan, M. *An Introduction to Art, the Wireless Way*
mpulse, october, 2002
- [5] Fensel, D. *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*
Springer, 2001
- [6] Guarino, N. *Formal Ontology in Information Systems*
Proceedings of FOIS'98, Trento, Italy.
- [7] Jena A Semantic Web Framework for Java
<http://jena.sourceforge.net/>
- [8] Joseki - Jena RDF Server
<http://www.joseki.org/>
- [9] Learning Lab Lower Saxony (L3S)
<http://www.l3s.de>
- [10] Miller, L. and Seaborne, A. and Reggiori, A. , *Three Implementations of SquishQL, a Simple RDF Query Language*
april, 2002
- [11] OntoWeb - Ontology-based information exchange for knowledge management and electronic commerce
<http://www.ontoweb.org>
- [12] Prasithsangaree, P. and Krishnamurthy, P. and Chrysanthi, P. K. *On indoor position location with wireless lans*
The 13th IEEE International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC 2002), Lisbon, Portugal, september 2002.
- [13] RDF Schema, W3C Specification 1.0
<http://www.w3.org/TR/rdf-schema>
- [14] Resource Description Framework (RDF), W3C Recommendation
<http://www.w3.org/RDF/>
- [15] Seaborne, A. , *A Programmer's Introduction to RDQL*
<http://jena.sourceforge.net/>, february, 2004
- [16] Web Ontology Language (OWL), W3C Recommendation
<http://www.w3.org/TR/owl-ref/>
- [17] World Wide Web Consortium:
<http://www.w3.org/>
- [18] XML Path Language (XPath) Version 1.0, W3C Recommendation
<http://www.w3.org/TR/xpath>
- [19] XSL Transformations Version 1.0, W3C Recommendation
<http://www.w3.org/TR/xslt>