Context-Aware Processing of Ontologies in Mobile Environments

Günther Specht and Timo Weithöner
University of Ulm
89069 Ulm, Germany

Abstract

This paper describes the ideas, open issues and the aims of a new research project, called MobileOntoDB. The goal is to develop a context-aware, database based ontology reasoner for mobile devices.

Users would benefit a lot from ontology based information systems on mobile devices, but since mobile resources are restricted, new techniques for main memory saving resolution, for partial evaluation, replication and synchronization of ontologies are needed. We propose a database supported approach, based on our "meta mapping" approach, successfully developed in a former project for huge ontologies on servers. It has the capability to be efficiently scaled down to mobile devices. In addition dynamic context-awareness will be added to the system, in order to get better and more appropriate resolution results. Context is not only time and location but also the situation and user preferences, which broadens today's location based services by far. In this paper we discuss the fundamental ideas, the architecture, applications and a number of open issues on the way to an integrated context-aware ontology-supported information system for mobile devices.

1. Motivation

Mobile computing becomes more and more important. Since mobile access to external information or services is one of the main beneficial application in mobile computing, machine readable description of content, access methods, and functionality is essential. Ontologies, as part of the semantic web, are proposed and widely used in static environments for this task. Adding ontologies to mobile environments requires an efficient storage and processing of ontologies on mobile devices with their restricted resources. Thus we need architectures where ontologies can be resolved locally, where relevant parts of huge ontologies can be downloaded incrementally, and where queries can be processed partly on mobile devices, and partly on backend servers, depending on available resources, capabilities and workload. In addition techniques for replication and, after local updates, synchronization of ontologies become important.

Mobile applications, from location based services to mobile gaming, will profit enormously from context-aware ontology systems. Today the processing of ontologies considers neither the location of the user nor the question, which facts are appropriate and relevant at the current location. In our new project a context-aware, database supported (Description Logics) DL reasoning system for the processing of ontologies on mobile devices will be developed. Therefore it is necessary on the one hand to extend ontologies and their evaluation by a dynamically changing context (spatial, temporal, situation-referred) and on the other hand to develop a reasoning mechanism on mobile clients with their restricted resources (CPU, main storage, etc.). Since today's reasoner work main memory based, and since these are too small in mobile devices, we propose a database based approach, in which the ontology reasoner is realized in the form of a mapping from DL-queries into queries on a mobile relational databases. A prototype of our database based reasoner is already working on servers and PCs. The aim of our project is now twofold: First, how can we scale down the database reasoner to work on mobile devices too, including new features like disconnected mode, fragmentation, replication and synchronization of ontologies and second, how can we represent, include, and process dynamic context information in the ontology reasoner.

Already now mobile context-aware ontology systems are urgently needed in the areas of mobile city information system, mobile medical information system, mobile web service search and querying as well as in resource sharing in mobile Peer to Peer systems and ad-hoc-networks. Since mobile gaming is always an interesting and ambitious application scenario, we choose this field for evaluating our results.

The rest of the paper is organized as follows: Section 2 presents a brief introduction to context-aware reasoning. Section 3 introduces our database based reasoning approach and Section 4 presents the overall architecture. Since we are in a very early stage of the project the main focus is more on addressing open research issues than on presenting final

results.

2 Context-Aware Reasoning

Today's mobile devices know quite a bit of the context they are used in: For example they know the location (via GPS), they know the situation the user is in (from the schedule they manage) and they know the persons which are around (via Bluetooth connection to their mobile devices). But as of today only very few applications use this context information.

In the following we understand context as the information about:

- the absolute and the relative location of the user (where the absolute location is given in coordinates while the relative position might be something like "across the river from the restaurant", which would require qualitative spatial reasoning techniques [3] to determine the impact on a given query (e.g. is there a bridge nearby to cross the river)),
- time and date as well as the schedule of the user (again qualitative reasoning techniques [1] will be required when a system has to understand queries like "Give me all dates scheduled for Monday morning after I meet Bob"),
- the situation and current activity of the user (in a meeting, in the gym),
- availability of networks and network services,
- availability of persons (possibly detected via Bluetooth connections to nearby devices) and resources,
- further sensor data like weather condition, health monitor data, etc.

There are several possibilities for modeling and representation of context. The easiest way is to integrate the knowledge about context into a given ontology using existing language elements. Assertions about the context are then limited to the expressivity of the ontology language. Another approach is to extend the ontology language with special elements to declare context. This would hopefully provide full expressivity. The drawback of this approach is that extended reasoners would have to be developed and the required elements to express all context information can hardly be foreseen as context is not a closed domain. Additionally calculability is questionable. For this reason we prefer to integrate context into the ontologies using (separate) existing language elements.

Querying a knowledge base using context raises two question: First, how can a context be overwritten? This is necessary if a query should use a different context than the users current context. Imagine a situation where a user queries a restaurant database for dinner locations while still in a train. We thus need an optional ability to specify query context manually. Second question is what to do if the query context is too specific and therefore no results are returned. A "relaxation" of the query concepts might be desirable to a certain extend. If one is looking for a restaurant for dinner in a certain quarter and there are no free tables, it might be a good idea to relax the location to the neighboring quarters, but not to relax the time (by including morning and afternoon). Thus we need an additional reasoning which context condition can be relaxed in which context.

Also research has to be conducted how a context-aware ontology editor should look like. There are lots of situations where some context could be integrated automatically, when new facts are entered. E.g. location is relevant when adding the fact "great restaurant". Time might or might not be relevant. Thus the ontology editor must prompt the user which context information should be integrated into the ontology and which precision is to be used. Of course certain situations allow the use of presets for these settings.

3 Database supported Reasoning Technologies

Reasoning on logical theories requires logic based inference systems which have been well studied within the field of knowledge representation in the AI community in the past decades. Description logics (DL) as a decidable fragment of first order logic (FOL) turned out to be an adequate formalism for representing and reasoning about expressive ontologies. As an alternative to tableaux-based DL theorem provers, a mapping of a DL subset into Logic Programs (LP) suitable for evaluation with Prolog has been suggested in [4]. This intersection of DL with LP called DLP completely covers RDF Schema [2] and a fraction of OWL [8] (notably most of OWL Lite extended with general concept inclusion). In a previous work [10, 11] we presented different mappings of such DLPs into logic programs suitable for deductive databases. For other ongoing projects we picked up these ideas and incorporated them into a working reasoner based on relational databases.

The motivation for this was twofold. First we expect ontologies to grow beyond the size which can be represented in a computers main memory. This is especially the case if we think about a network of linked ontologies forming one huge virtual ontology like suggested in the semantic web. This growth of ontologies requires reasoning techniques which work in secondary storage as well. Using database technology we overcame these limitations. Secondly we envisioned a mobile scenario in which client devices with very limited resources would work with parts

of the ontology. Thus replication and synchronization will take place. Procedures which are well understood in the database community and are available for small footprint mobile database management systems. Ideally a reasoner which works inside a database management system allows to use the same reasoning techniques on the server as well as the clients and would solve most of the replication and synchronization issues.

As already mentioned the relational reasoner is based on a mapping mechanism which in a first step converts an ontology into a logic program with a fixed rule set and a varying set of base facts. In a second step the program is converted into a program suitable for an off-the-shelf relational database. This program recursively calculates (and materializes) new facts from the existing facts in the relations. Similar approaches have been chosen for the implementation of reasoners like OWLIM [6] which is working in main memory only or the Oracle RDF Framework [7] which only supports RDFS entailments [5].

4 Overall Architecture

The overall architecture includes backend servers and mobile devices. Typically both hold part of the ontology. Thus ontology reasoning has to be done on server side and on client side [9] . If the client is too small, part of the resolution may be split and delegated to the server side. This is the reason why it is a great advantage if a reasoner, working on the same internal representation of the ontology resides on both sides. We can more easily download, split, share, update, replicate and synchronize parts of the ontology - and part of the ontology execution - if we use a homogeneous storage and reasoning environment on server and mobile client side. Since we use a database approach and map DL into SQL-statements, we can benefit from distributed databases, if we use the same relational representation on both sides. Of course, the size of the part of the ontology that is replicated to the mobile device might differ depending on the capabilities of the device and the reasoner. Figure 1 shows the architecture of such a mobile setting.

It is obvious that one should try to perform as much reasoning as possible on the mobile device. Like this querying the knowledge base is independent of the availability of a wireless network and no personal data has to be revealed, which is especially interesting when the query contains the context of a user.

Even though mobile databases already offer replication and synchronization capabilities, research has to be conducted if there is a suitable way to delegate parts of the reasoning dynamically from a mobile device to a central reasoning server if the query exceeds the limitations of the mobile device.For this queries must be split into subqueries that can be computed independently. Another influencing factor when deciding whether to delegate (parts of a) query is whether the complete subset of the ontology, which is necessary to compute the query, already is available on the mobile device and how expensive the replication of the missing parts would be in contrast to the expenses for the transmission of the query and the query results.

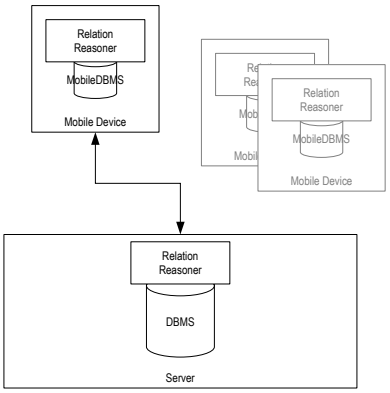

Figure 1. Architecture: Mobile setting with a homogeneous storage and reasoning environment on server and mobile clients.

5 Conclusion

In this paper we discussed the ideas, the architecture, applications and a number of open issues on the way to an integrated context-aware ontology-supported information system for mobile devices. We proposed a database driven approach in order to avoid main memory restrictions on mobile devices. This leads to a very scalable system. Our specific enhancement is the context-awareness of the reasoning system.

References

[1] J. F. Allen. Maintaining Knowledge about Temporal Intervals. *Commun. ACM*, 26(11):832–843, 1983.

[2] D. Brickley and R. V. Guha. RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation, February 2004. http://www.w3.org/TR/2004/REC-rdf-schema-20040210/.

[3] A. G. Cohn and S. M. Hazarika. Qualitative Spatial Representation and Reasoning: An Overview. *Fundam. Inform.*, 46(1-2):1–29, 2001.

[4] B. N. Grosof, I. Horrocks, R. Volz, and S. Decker. Description Logic Programms: Combining Logic Programms with Description Logic. In *Proceedings of the* 12^{th} *International World Wide Web Conference*, Budapest, Hungary, May 2003.

[5] P. Hayes. Rdf semantics. W3C Recommendation, Februrary 2004.

[6] A. Kiryakov, D. Ognyanov, and D. Manov. OWLIM - A Pragmatic Semantic Repository for OWL. In M. Dean, Y. Guo, W. Jun, R. Kaschek, S. Krishnaswamy, Z. Pan, and Q. Z. Sheng, editors, *WISE Workshops*, volume 3807 of *Lecture Notes in Computer Science*, pages 182–192. Springer, 2005.

[7] C. Murray. Oracle Spatial, Resource Description Framework (RDF), 10g Release 2 (10.2). Technical report, Oracle, 2005.

[8] P. F. Patel-Schneider, P. Hayes, and I. Horrocks. OWL Web Ontology Language Semantics and Abstract Syntax. W3C Recommendation, February 2004.

[9] T. Weithöer and G. Specht. Verarbeitung von Ontologien in mobilen Umgebungen. In *Proc. Informatik 2004, 34. Jahrestagung der GI, 20.-24. Sept. 2004, Ulm*, volume P-50 of *GI Lecture Notes in Informatics*, pages 303–307. GI, 2004.

[10] T. Weithöner, T. Liebig, and G. Specht. Storing and Querying Ontologies in Logic Databases. In *Proceedings of the Workshop "Semantic Web and Databases" at the VLDB 2003*, Berlin, Germany, September 2003. Online Proceedings.

[11] T. Weithöner, T. Liebig, and G. Specht. Efficient Processing of Huge Ontologies in Logic and Relational Databases. In R. Meersman, Z. Tari, and A. Corsaro, editors, *OTM Workshops*, volume 3292 of *Lecture Notes in Computer Science*, pages 28–29. Springer, 2004.