A Context-Aware Meeting Alert Using Semantic Web and Rule Technology - Preliminary Report

Grigoris Antoniou and **Antonis Bikakis**[1] and **Anna Karamolegou** and **Nikos Papachristodoulou** [2]

Abstract. This paper describes work in progress developing a context-aware meeting alert. This application integrates semantic web technology in RDF (for representing calendars), semantic web rules (for making a context-dependent decision about the precise timing of the alert), and mobile technology for location sensing and message delivery. The outlined work is an early experiment seeking to demonstrate the feasibility of applying efficient, semantically sound semantic web reasoning to mobile applications.

1 MOTIVATION

Computing is moving towards pervasive, ubiquitous environments in which devices, software agents and services are all expected to seamlessly integrate and cooperate in support of human objectives, anticipating needs and delivering services in an anywhere, any-time and for-all fashion [23]. *Pervasive Computing* and *Ambient Intelligence* are considered to be key issues in the further development and use of Information and Communication technologies, as evidenced, for example, by the IST Advisory Group [16].

Pervasive applications aim at providing the right information to the right users, at the right time, in the right place, and on the right device. In order to achieve this, a system must have a thorough knowledge and, as one may say, "understanding" of its environment, the people and devices that exist in it, their interests and capabilities, and the tasks and activities that are being undertaken. Such a system must be able to recognize that a person works on different projects, collaborates with other colleagues, and has specific tasks to fulfill and a unique set of skills and concerns, and where this person can be found. All this information falls under the notions of *context*.

More recent efforts sought to develop models of context(s) that integrate information from a variety of sources, support interoperation of context-aware applications and context management systems, and allow reasoning about context. The most modern approaches make use of the increasing popularity of ontologies and the emergence of the Semantic Web. These approaches use ontology-based models of context and aim to better support interoperation by formally defining common concepts and the relations and mapping between them. In fact, most such works distinguish between upper-level context ontologies, aiming at capturing types of context information at a high level of abstraction, and concrete, application-dependent context ontologies that extend and customize the upper-level ontologies; a typical high-level context modeling language is SOUPA [9]. Representative projects and prototypes reported in the most recent literature include [8], [12], [17], [20] and [21]. These works make use of the semantic web standards of RDF Schema [7] and OWL [11] as foundation for the context ontologies, and of associated tools to process the ontologies.

Once context information has been modeled, it has to be processed. Some limited forms of reasoning are provided by tools associated with Semantic Web languages such as RDF [7], RDF Schema and OWL, such as RACER [13] and FaCT [14] e.g. for deducing subsumption relations and for detecting inconsistencies. However, more forms of reasoning are required for making decisions as to which information to deliver to which users, at what time, at what location, and on what device. Such reasoning goes well beyond the capabilities of reasoning methods associated with RDF Schema and OWL. In fact, many of the works mentioned above perform reasoning, but in ad hoc ways.

The underlying basic assumption of the authors' work is that efficient and semantically sound methods for reasoning about contextual information must be developed and studied. Context reasoning can be expected to enhance existing work on context modeling in the same way as research on semantic web reasoning currently extends the mature layers of the semantic web initiative, which are concerned with semantic annotations (RDF) and ontologies (OWL).

This paper reports work in progress developing a context-aware meeting alert. This application integrates semantic web technology in RDF (for representing calendars), semantic web rules (for making a context-dependent decision about the precise timing of the alert), various types of context (location, time, environment, calendar) and mobile technology for location sensing and message delivery. The outlined work is an early experiment seeking to demonstrate the feasibility of applying efficient, semantically sound semantic web reasoning to mobile applications.

2 THE SPECIFIC APPLICATION

2.1 Application Description

The application scenario concerns staff of the University of Crete, who is also working at the Institute of Computer Science at FORTH. The University and FORTH are located at opposite ends of the city of Heraklion. The meeting alert system is meant to send an SMS to the staff's cell phone, alerting in time of an upcoming scheduled meeting, based on the user's calendar.

Determining the appropriate time for sending the SMS is a key challenge of this work. This decision depends on various parameters:

- If the meeting is scheduled at the location (FORTH or UoC) where the user currently is located, 5 minutes is sufficient.
- If it's in a different office location, 40 minutes.

[1] Institute of Computer Science, FO.R.T.H. Heraklion, Greece, email: {antoniou, bikakis}@ics.forth.gr
[2] Department of Computer Science, University of Crete, Greece, email: {karamol,nikosp}@csd.uoc.gr

- But in rush add another 10 minutes.
- If it rains add another 10 minutes.
- If one has a class alert (the meeting is actually a teaching class) must come earlier to allow for preparation.

2.2 TYPES OF CONTEXT

The scenario described in 2.1 combines different types of context.

Obviously, *location* is a key type of context. In a first prototype, location will fall into one of the areas: FORTH, UoC, city centre. Refinements will be considered in later stages. The user's location is determined by the phone cell closest to his current position.

Time is another key type of context.

Environmental information (e.g. rain) can be collected either by web services (e.g. weather services) or using sensors. Finally, calendars, information about locations, types of meetings etc. are described in RDF(S) documents.

3 SEMANTIC WEB TECHNOLOGY

3.1 Representing Calendar Information in RDF

The calendar events are described using the RDF Calendar vocabulary [24], which is based on the iCalendar standard [10].

3.1.1 iCalendar Specification

The Internet Calendaring and Scheduling Core Object Specification (iCalendar) has been defined to provide the definition of a common format for openly exchanging calendaring and scheduling information across the Internet. The top-level object in iCalendar is the Calendaring and Scheduling Core Object which is a collection of calendaring and scheduling information. The components contained in the body of the Core Object include events, to-do items, journal entries, free/busy time items, time zone information and alarms.

3.1.2 Translating iCalendar into RDF

iCalendar has been developed only to model calendar events. It does not try to encompass all the different aspects of context e.g. people, locations, environmental information etc. However, we often need to combine the calendaring information with other types of information, described in different formats and originating from various sources, e.g. the Web. The solution that we chose for the needs of the application described in this paper is RDF; and the first step was to translate the iCalendar events into RDF descriptions, using the iCalendar RDF Schema (RDFiCal). The schema defines classes and properties that simulate the iCalendar compononents.

In Figure 1a, we present a simple example of an iCalendar object that defines a scheduled meeting event ("Meeting with Grigoris") occurring from April 7, 2006 09:00 (UTC) through April 7, 2006 010:00 (UTC). The object was created at April 3, 2006 10:41 (UTC). The RDF description of the same event is depicted in Figure 1b.

(a)

```
BEGIN:VEVENT
UID:20060403T104117Z-6644145-4384@ics.forth.gr
SUMMARY:Meeting with Grigoris
DESCRIPTION:Meeting with Grigoris
CLASS:PUBLIC
ATTENDEE;ROLE=OWNER;STATUS=CONFIRMED:Default
Administrator
DTSTART:20060407T090000Z
DTSTAMP:20060403T104117Z
DTEND:20060407T0100000Z
END:VEVENT
```

(b)

```
<Vevent>
  <uid>
     20060403T104117Z-6644145-4384@ics.forth.gr
  </uid>
  <summary>Meeting with Grigoris</summary>
  <description> Meeting with Grigoris</ description>
  <class
rdf:resource='http://www.w3.org/2002/12/cal/ical#public'/>
 <attendee rdf:parseType='Resource'>
   <calAddress rdf:resource=\default administrator'/>
   <role>OWNER</role>
   <status>CONFIRMED</status>
 </attendee>
 <dtstart rdf:parseType='Resource'>
   <dateTime>2006-04-07T09:00:00Z</dateTime>
 </dtstart>
 <dtstamp rdf:parseType='Resource'>
   <dateTime>2006-04-03T10:41:17Z</dateTime>
 </dtstamp>
 <dtend rdf:parseType='Resource'>
   <dateTime>2006-04-07T10:00:00Z</dateTime>
 </dtend>
</Vevent>
```

Figure 1. (a) iCalendar object defining a meeting event, (b) RDF description of the meeting event

3.2 Representing Declarative Decision Making Using Web-Based Defeasible Reasoning

The basic reasoning task of the application is deciding on the time of sending an SMS. Informal rules were given in section 2.1. Obviously, they can be implemented in many ways, both in traditional programming languages and in simple rules system. In the reported work, we wish to implement a solution with the following characteristics:

1. The solution should use some kind of *logical reasoning*, to allow for integration with semantic web technologies.
2. The solution should be in line with current research on *semantic web reasoning*. Given the type of knowledge to be modelled, a semantic web rules language was adopted.
3. The semantic web rules language should represent and reason about conflicting rules in a formal, declarative way. By conflicting rules we mean the situation where different rules leading to conflicting conclusions may fire. Thus a *nonmonotonic rules language* is adopted.

Defeasible Logics ([1], [18]) combine all the desired features. The basic characteristics that make DLs an appropriate solution are:

- Being nonmonotonic, defeasible logics can deal with potential conflicts (inconsistencies) among knowledge items. Thus they contain classical negation, contrary to usual logic programming systems. Although the object language does not contain Negation As Failure, it can easily be simulated when necessary, as shown in [3].
- Conflicts among rules are indicated by a conflict between their conclusions. These conflicts are of local nature. The simpler case is that one conclusion is the negation of the other. The more complex case arises when the conclusions have been declared to be mutually exclusive, a very useful representation feature in practical applications.
- Defeasible logics are skeptical in the sense that conflicting rules do not fire. Thus consistency of drawn conclusions is preserved.
- Priorities on rules may be used to resolve some conflicts among rules. Priority information is often found in practice, and constitutes another representational feature of DLs.
- The logics take a pragmatic view and have low computational complexity. This is, among others, achieved through the absence

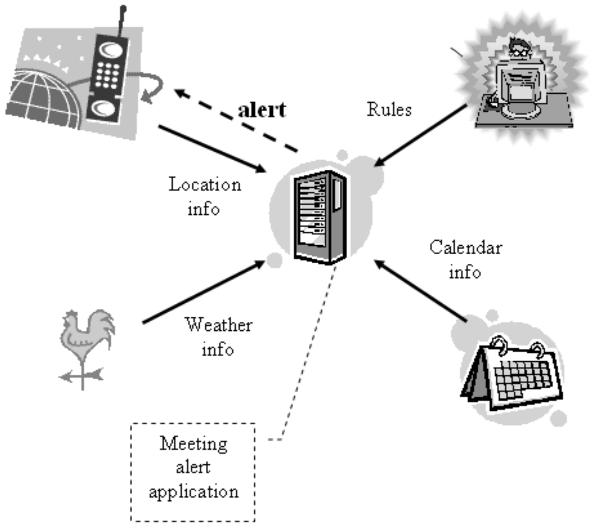

Figure 2. Information flow between the components of the system.

of disjunction and the local nature of priorities.

Two systems that combine defeasible logics with Semantic Web are DR-Prolog [6] and DR-DEVICE [4]. Using such a system, the user can express his / her preferences as defeasible rules and reason with ontology data expressed in RDF. The details about how DR-Prolog is used as part of the meeting-alert application are presented in the next section of this paper.

4 TECHNICAL SOLUTION

The technologies combined to implement the meeting alert system are:

- A calendar application (WebCalendar).
- A location sensing system for cell phones (CellPos, CellGPS).
- A web weather service (accuweather.com).
- A reasoning engine for making decisions about when to send the alerts based on the rules imported by the user and on the available calendar and context information.
- A service for sending text messages to mobile phones (groupsms).

The information flow between the different components of the system is depicted in Figure 2. The meeting alert application is fed with (a) the user's location information from the location sensing application; (b) information about the local weather conditions from a web weather service; (c) calendar information from the user's personal calendar application; and (d) rules concerning the user's "alerting policy". Based on the rules and the available information, it decides when to alert the user about an upcoming calendaring event and schedules the dispatch of the alerting message to the user's mobile phone. In the next paragraphs, the basic components of the system are described in more detail.

4.1 WebCalendar

WebCalendar [22] is a PHP-based calendar application that can be configured as a single-user calendar, a multi-user calendar for groups of users, or as an event calendar viewable by visitors. One of its basic features is that it allows importing or exporting calendar events as iCalendar event descriptions. In the context of the meeting alert application, WebCalendar has been setup as a calendar server that can be viewed with iCal-compliant calendar applications. Using the rdf2iCal application, we have configured WebCalendar to export the calendar events in RDF descriptions, as presented in section 3.1 of the paper.

4.2 Location sensing

In order the meeting-alert application to work, the user must allow some location-sensing mechanism to localize him / her during the working hours of the day. For this reason, CellPos[3], a location sensing service for mobile phones has been employed. CellPos, a Symbian software, records the position of gsm cells automatically using a Bluetooth-capable GPS receiver. The application, which works automatically, tries to determine the center of each cell by storing the GPS position of the places with the highest signal strength value in that gsm cell.

For sending the information acquired by CellPos to the server hosting the user's calendar and the meeting-alert application, the mobile phone is equipped with an additional program, CellGPS. This Symbian compatible software sends the mobile phone location information to a web server. It connects automatically to the server using a predefined GPRS connection at predefined time periods.

4.3 Reasoning

All the context information (user's location, local weather conditions, calendar entries) is gathered at the server hosting the user's calendar. To reason with the user's rules and the available calendar and context information, we use the DR-Prolog defeasible reasoning engine. DR-Prolog can reason with defeasible rules and ontological knowledge written in RDF(S) or OWL. Its basic characteristics are:

- It is compatible with RuleML [19], the main standardization effort for rules on the Semantic Web, allowing its users to use a XML syntax for their rule theories.
- It supports strict as well as defeasible rules plus priorities between the rules, and implements various variants of defeasible logic.
- It is based on Prolog. The core of the system consists of a well-studied translation [2] of defeasible knowledge into logic programs under Well-Founded Semantics [15].
- It can reason with rules and ontological knowledge written in RDF/RDFS or OWL.

In order DR-Prolog to reason with the available context data, the data must be in a suitable format (RDF descriptions or first-order predicates). The calendar application exports the calendar entries as RDF descriptions, so they are ready to import. We also format the location and the weather information into RDF and import it into the system. The system translates the RDF descriptions into triples of the form: *predicate(object,subject)*, and then applies the user's rules on the knowledge base and gives some conclusions, which are used for deciding the time of dispatch and the content of the alerting messages. As an example of the rules, we present below the defeasible theory that simulates the application scenario, described in section 2.1.

[3] Detailed information about CellPos and CellGPS are available at: www.vikinggames.hu

- If the meeting is scheduled at the location (FORTH or UoC) where the user currently is located, 5 minutes is sufficient.
  r_1*: location(Evt,X), location(user,Y), X=Y, description(Evt,D), dtstart(Evt,S)* $\Rightarrow$ *alert(D,S,5).*

- If it's in a different office location, 40 minutes.
  r_2*: location(Evt,X), location(user,Y), X* $\neq$*Y, description(Evt,D), dtstart(Evt,S)* $\Rightarrow$ *alert(D,S,40).*

- But in rush add another 10 minutes.
  r_3*: description(Evt,D), dtstart(Evt,S), dayperiod(S,rush)* $\Rightarrow$ *alert(D,S,10).*

- If it rains add another 10 minutes.
  r_4*: description(Evt,D), dtstart(Evt,S), weather('Heraklion',rainy)* $\Rightarrow$ *alert(D,S,10).*

- If one has a class alert (the meeting is actually a teaching class) must come earlier to allow for preparation.
  r_5*: description(Evt,'Class'), dtstart(Evt,S),* $\Rightarrow$ *alert(D,S,30).*

- If the meeting is scheduled at the same place with the user's location, do not need consider the weather or traffic conditions.
  $r_1 > r_3$, $r_1 > r_4$

By adding the numbers of minutes contained in the conclusions that derive from the theory, the application computes the time that it will send the alert message to the user.

4.4 Messaging service

The meeting alerts, created at the server running the user's calendar, are sent as text messages to the user's mobile phone. We assume that the user has already inserted his/her phone number in the knowledge base of the application. The service employed to dispatch the messages is the ForthNET groupsms service[4]. An alerting message contains the description, the scheduled start time and the venue of a calendar event.

5 SUMMARY AND FUTURE WORK

In this paper, we described current efforts on implementing a semantic meeting alert application. This application combines location based, calendar and weather services to gather the information needed in order to decide about when to alert the user about an upcoming scheduled event. In order to integrate the available heterogeneous information, and perform reasoning based on the user's rules, we adopted the RDF model for the representation of the context data, and employed the DR-Prolog reasoning system, which supports the RDF model. The next steps of this work will focus on:

- Extending the location model, by finding the correspondences between the cell ids and the real location names of the city of Heraklion.
- Using more context information, for instance traffic information for the roads of Heraklion or the user's profile information.
- Extending the functionality of the system by employing multi-user calendars for the needs of a workgroup.
- Employing alternative technologies, especially for the localization of the user.

[4] www.groupsms.gr, 2006

The semantic meeting alert application is a case study within our more general goal to investigate the use of Semantic Web technology and semantic reasoning in mobile and ubiquitous applications and in ambient intelligence environments.

REFERENCES

[1] G. Antoniou, D. Billington, G. Governatori, and M. J. Maher, 'Representation results for defeasible logic', *ACM Transactions on Computational Logic*, **2(2)**, 255–287, (2001).
[2] G. Antoniou, D. Billington, G. Governatori, and M. J. Maher, 'Embedding Defeasible Logic into Logic Programming', *Theory and Practice of Logic Programming*, to appear
[3] G. Antoniou, M. J. Maher, and D. Billington, 'Defeasible Logic versus Logic Programming without Negation as Failure', *Journal of Logic Programming*, **41(1)**, 45–57, (2000).
[4] N. Bassiliades, G. Antoniou, and I. Vlahavas, *DR-DEVICE: A defeasible logic system for the Semantic Web*. In 2nd Workshop on Principles and Practice of Semantic Web Reasoning, H. J. Ohlbach and S. Schaffert (Eds.). 134-148, Lecture Notes in Computer Science, Volume 3208, Springer, Berlin, 2004.
[5] D. Beckett, RDF/XML Syntax Specification. W3C Recommendation, 10 February 2004. http://www.w3.org/TR/rdf-syntax-grammar/
[6] A. Bikakis and G. Antoniou, *DR-Prolog: A System for Reasoning with Rules and Ontologies on the Semantic Web*. In Proc. 25th American National Conference on Artificial Intelligence (AAAI-2005), 2005.
[7] D. Brickley and R. V. Guha, RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation, 10 February 2004. http://www.w3.org/TR/rdf- schema/
[8] H. Chen, T. Finin, and A. Joshi, *Semantic Web in the Context Broker Architecture*. In Proc. IEEE Conference on Pervasive Computing and Communications (PerCom), Orlando, 2004.
[9] H. Chen, F. Perish, T. Finin, and A. Joshi, *SOUPA: Standard Ontology for Ubiquitous and Pervasive Applications*. In Proc. International Conference on Mobile and Ubiquitous Systems: Networking and Services, Boston, August, 2004.
[10] F. Dawson and D. Stenerson, "Internet Calendaring and Scheduling Core Object Specification" (iCalendar) Network Working Group RFC: 2445, November, 1998. http://www.faqs.org/rfcs/rfc2445.html
[11] D. McGuiness and F. van Harmelen. OWL Web Ontology Language Overview. W3C Recommendation, 10 February 2004. http://www.w3.org/TR/owl-features/
[12] M. Hatala, R. Wakkary, and L. Kalantari, 'Ontologies and Rules in Support of Real- time Ubiquitous Application', *AJournal of Web Semantics*,(to appear).
[13] V. Haarslev and R. Moller, *RACER System Description*. In Proc. International Joint Conference on Automated Reasoning, (IJCAR'2001), 2001
[14] The FaCT system. http://www.cs.man.ac.uk/ horrocks/FaCT/
[15] A. van Gelder, K. Ross, and J. Schlipf, 'The well-founded semantics for general logic programs', *Journal of the ACM* , **38**, 620-650, (1991).
[16] IST Advisory Group. Ambient Intelligence: from vision to reality. September 2003. http://www.cordis.lu/ist/istag-reports.htm
[17] I. C. Millard, D. De Roure, and N. Shadbolt, *The use of ontologies in contextually aware environments*. Proceedings of First International Workshop on Advanced Context Modelling, Reasoning And Management, J. Indulska and D. De Roure (Eds.). 42–47, Nottingham, 2004.
[18] D. Nute, *Defeasible Logic*, *Handbook of logic in artificial intelligence and logic programming (vol. 3): nonmonotonic reasoning and uncertain reasoning*, Oxford University Press, 1994.
[19] RuleML. The Rule Markup Language Initiative, 2006. www.ruleml.org
[20] S. Toivonen, J. Kolari, and T. Laakko,*Facilitaing Mobile Users with Contextualized Content*. In Proc. Artificial Intelligence in Mobile Systems (AIMS) 2003 Workshop held in conjunction with Ubicomp, Seattle, October 2003.
[21] X. Wang, J. S. Dong, C. Y. Chin, S. R. Hettiarachchi, and D. Zhang, 'Semantic Space: An Infrastructure for Smart Spaces', *IEEE Pervasive Computing*, **3(3)**, 32–39, (2004).
[22] WebCalendar, 2006. http://www.k5n.us/webcalendar.php
[23] M. Weiser, 'The computer for the 21st century', *Scientific American*, **265(30)**, 94–1004, (1991).
[24] W3C. RDF calendar workspace. http://www.w3.org/2002/12/cal/