A Context Framework for Ambient Intelligence

Asuman DOGAC, Gokce B. LALECI, Yildiray KABAK
Software Research and Development Center, Middle East Technical University, Inonu Bulvari, METU(ODTU) Campus, 06531, Ankara, Turkey Tel:+90 312 2105598
asuman@srdc.metu.edu.tr, [1]

Abstract. ISTAG has developed the concept of Ambient Intelligence (AmI) to provide a vision on how the Information Society will develop. The aim is to achieve better integration of technology into our environment, so that people can freely and interactively use it. An indispensable component of AmI is user context. We claim that to realize the vision of AmI the following issues need to be addressed: the user context information need to be interoperable and machine processable since it will be exploited by a variety of devices. Therefore it is necessary to develop context ontologies. Furthermore, the user context should be globally accessible; this necessitates developing context servers. Considering that AmI devices accept input in different mark up languages; the context server needs to recognize the device and provide the information in the format that can be accepted by the device. More importantly, user context information should only be provided to the authorized entities and the user should be able to specify how much of that information should be available to whom. Finally, AmI is not restricted to local context obtained through sensors; integrating this information with the user profile and preferences opens up a way for real life applications where the user context can be exploited for Web service discovery and composition.

In this paper, we present a framework to address exactly these issues. We show how context ontologies can be developed, stored, queried. We then describe mechanisms for the privacy and security of user context. Finally we present the way to exploit user context for Web service discovery and composition through ebXML registries.

1. Introduction

Ambient Intelligence (AmI) stems from the convergence of three key technologies: Ubiquitous Computing, Ubiquitous Communication, and Intelligent User Friendly Interfaces [6]. The increased availability of commercial, off-the-shelf sensing technologies and the prevalence of powerful, networked computers and devices are bringing this vision much closer. However there are issues to be addressed before this vision can be realized. One of these issues is handling user context information.

We define user context to be any information that can be used to characterize the user and her situation. The user context can thus include a wide variety of information coming through sensors such as the current temporal and spatial location, and the stored information such as user preferences, and profile. The current context-aware applications

[1] This work is supported in part by the Scientific and Technical Research Council of Turkey, Project No: EEEAG 102E035

are usually build in an ad hoc manner and lack the generality and standards to be of use in AmI environments.

We claim that to realize the vision of AmI the following issues need to be addressed:

- *Developing Context Ontologies*: The context should be machine processable and interoperable since it will be accessed by a variety of devices. A solution is to develop ontologies for user context. In other words, to make user context information accessible by agents (software or human), several modular ontologies need to be developed addressing different parts of user context such as temporal aspects, spatial aspects, profiles, etc. Parts of already existing ontologies may be used for this purpose.
- *Developing mechanisms to respect user's right to privacy when revealing user context information:* User context information should only be provided to the authorized entities and the user should be able to specify how much of that information should be available to whom. For example the user may wish that certain aspects of her context should only be available to certain types of agents.
- *Developing globally accessible Context Servers:* The context information should be available to any authorized agent at any time, any where in a secure manner: This necessitates developing globally accessible, secure "context servers", that is, the user context information should be available any where, any time to a variety of devices from desktops to mobile devices. Since these devices accept input in different mark up languages; the context server needs to recognize the device and provide the information in the format that can be accepted by the device. Note that if the user permits, information on user activities should be collected to further improve user context.
- *Integrating context-awareness with Web service discovery*: AmI is not restricted to local context obtained through sensors; integrating this information with the user profile and preferences opens up the way for real life applications where the user context can be exploited for Web service discovery and composition.

To demonstrate our case, we present a part of one of the scenarios given in [6]:

"After a long haul flight Maria lands to an airport in a Far Eastern country. The immigration officer in this country has been replaced by a device. The immigration device, through its sensor, detects the identities of the people entering to the immigration area and performs visa and passport control. Maria carries a P-Com device that has her identity information and her personal software agent has arranged for her visa. Therefore she is able to stroll through the immigration. There is a rented car waiting for her at the exit and her hotel has been reserved thanks to her personal software agent."

We will use this scenario to describe the context information requirements needed by AmI environments: When Maria arrives at the immigration hall, the immigration device, through its sensor, can detect the identity of Maria from her P-Com. There are several issues to be considered here:

1. The identity information should be understandable by any authorized device any where which implies there has to be standards in this respect.
2. Through the identity of the person the necessary information should be accessible. In the scenario presented, this is the passport and visa information of Maria. This information can be obtained by the immigration device either by querying the context server or if the P-Com also has the information, it can transmit it. However in any case this information should be machine processable (the immigration device has to process it without human intervention) and interoperable (this data originating from Europe, should be processable in the Far East). In other words, ontologies need to be developed to make user context automatically accessible by agents.
3. Due to the personal and proprietary nature of the context information, access must be limited to authorized entities and the user should be able to state how much information

to disclose and to whom. In the above scenario, the immigration device should get only the location, visa and passport part of the user context. Indeed if context information can not be disclosed on a role or personal identity basis, if it can not be securely transmitted over untrusted networks and if it is not possible to authenticate communications with remote hosts, this will destroy many of the promised rewards of the Ambient Intelligence.

4. To provide security and global accessibility, the stored part of the context information (such as user profile and preferences) should be available through "trusted context servers". To provide for interoperability context servers to be developed must be based on open standards. Note that there could be more than one trusted context server on the Web to provide for scalability.
5. Furthermore, since the user context information should be available to any device from desktops to mobile devices and these devices accept input in different mark up languages; the context server needs to recognize the device type and provide the information in the format that can be accepted by the device.
6. Finally, it should be possible to exploit context to discover and compose Web services for a user. For example, in the above scenario, it is Maria's software agent which arranged the details for her trip. To realize this, the agent needs to discover services based on her context. For example, to obtain a visa for her, the agent might get her itinerary from her context; invoke a service at the related embassy by providing the information necessary to get a visa. This information may include her nationality, her previous trips to the country, etc. which can again be obtained from the context server.

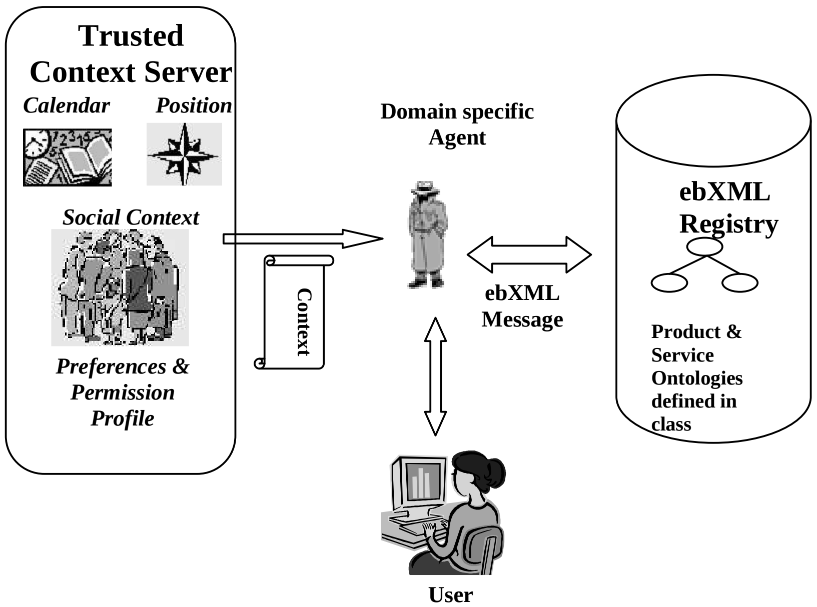

Figure 1. Overall Architecture of the System

In this paper we describe a framework to address these issues. Figure 1 shows the overall architecture of the system. User context is stored in an ontology server. A view based mechanism is used to provide for the privacy and security of user context information. Services are stored in a domain specific ebXML registry and a JADE agent is associated with the registry. When a user needs a Web service, the agent queries the service registry to find the explicit members of the given ontology class.

2. Describing User Context Through an Ontology Language

There is a need to develop context ontologies to realize the vision of AmI. A user context can involve, identity information, profile (e.g., the expertise area, company information), preferences (e.g., entertainment preferences, travel preferences), spatial information (e.g., location, orientation, speed and acceleration), temporal information (e.g., time of the day, date, and season of the year), environmental information (e.g., temperature, air quality, and light and noise level), social situation (e.g., the people nearby), resources that are nearby (e.g., accessible devices, sensors), availability of resources (e.g., battery, display, network, and bandwidth), physiological measurements (e.g., blood pressure, heart rate, respiration rate, muscle activity), activity (e.g., walking, talking, running), schedules and agenda setting. As mentioned previously, some of this information may be coming from the sensors; some of it are stored in context servers.

Ontologies provide machine-processable semantics of information sources that can be communicated between agents (software and human). The following properties of ontologies make them an indispensable tool for defining semantics:

- They have a formal specification. This property makes them machine processable. For example they can be queried through query languages to obtain the desired information.
- Ontologies define shared conceptualizations, that is, an ontology captures consensual knowledge meaning that it is not restricted to an individual but accepted by a group.

In practical terms, developing a context ontology includes determining the information relevant to context and then:

- Defining classes in the ontology
- Arranging the classes in a taxonomic (subclass, superclass) hierarchy,
- Defining properties (i.e. slots) and describing allowed values (facets) for these properties,
- Filling in the values for properties for instances,
- Defining the relations among the classes.

In order to enable as much reuse as possible, ontologies should be developed in small modules. If well-designed modular ontologies can be developed, new ontologies can be constructed by assembling these modules.

There are several ontology languages. Among these Web Ontology Language (OWL) [10] is a semantic markup language being developed by the World Wide Web Consortium for publishing and sharing ontologies. OWL is derived from DAML+OIL [4] by incorporating learnings from the design and application use of DAML+OIL. It builds upon the Resource Description Framework [11,12].

Developing domain specific ontologies are the responsibility of standard bodies and industrial consortiums. It should also be noted that there are libraries of reusable ontologies on the Web and in the literature such as the Ontolingua ontology library, or the DAML ontology library.

3. Storing Context Ontologies and Security and Privacy Issues

Ontologies are stored in knowledge-bases. Several tools have been developed for this purpose. However where and how to store the context ontologies depends also on the level of security and privacy the repository can provide.

There are two aspects of security and privacy in the context platform proposed:

- Security and privacy of data coming from the sensor devices. In our example, this corresponds to the signals coming from Maria's P-Com device and this should only be accessible to the authorized devices such as the immigration authority device. There are a number of systems developed for this purpose. For example, in Solar System [8], such an access-control mechanism is described where sensor data is represented as events flowing from sources through operators (which filter, transform, or aggregate events into new events) to applications. Each event is tagged with an access control list (ACL) thus restricting the receipents.

- Security and privacy of data obtained from the context server. Due to the personal and proprietary nature of the context information, access to the context server must be limited by authorized entities and the user should be able to state how much information to disclose and to whom. For this purpose, a mechanism similar to the well-established view mechanism for relational and object databases, is necessary. In [13], a view mechanism, implemented as a part of KAON [7] server, is described. As the query language of the view mechanism, RQL [1] is used. The main concerns of the authors in this work are as follows:
  - The view should again be based on an ontology
  - The approach should be functional, that is, it should allow to create views based on other views.

  To achieve these goals, they differentiate between views on classes and views on properties and view definitions are restricted to queries which return either views on classes or views on properties.

Such a view mechanism can be used for providing the security and privacy of context information in a context server. A user can define how much of her context information should be available to whom by defining views on her context and granting different access rights to different agents (human or software) on these views. Consider for example the context hierarchy given in Figure 2. It is possible to define views on the user context and grant access rights to different users as shown in the following example:

```
CREATE CLASS VIEW ScheduleOfMaria
        SUBCLASSOF Schedule
        SELECT X
        FROM ContextServer
        WHERE X.Name= “Maria”
GRANT SELECT ON ScheduleOfMaria TO PersonalAgentOfMaria
```

In this example, a view is created on the “schedule” class is created which contains the schedule information of the user “Maria” and this information is granted to her personal agent.

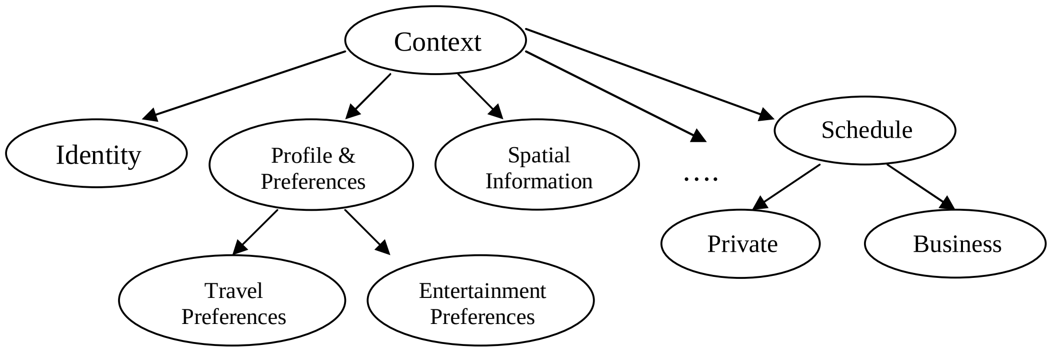

Figure 2. An example Context hierarchy

4. Exploiting User Context for Web Service Discovery

In order for agents to select services in a context–sensitive manner, they must be able to discover services based on their semantic description. A key issue in semantic discovery of services is to relate the service advertised in the registry with its semantic description. ebXML registry allows metadata to be stored in the registry through a "Classification" mechanism which helps to classify the objects (i.e. services) in the registry. In the following we describe how user context can be used to discover services.

Consider our running example. When Maria's agent discovers that she will be travelling to a country at the Far East, it starts to arrange her trip. The personal agent first obtains Maria's schedule by querying the context ontology given in Figure 2 as described in Section 3. If Maria does not have a valid passport for her trip, it is necessary to obtain a passport for her. Thanks to the e-government initiatives through out Europe, we can expect the passport services to be electronically available not in a distant future. Then the personal agent of Maria can invoke this Web service of her government to obtain her a passport. The information needed for this purpose can be obtained from her context information.

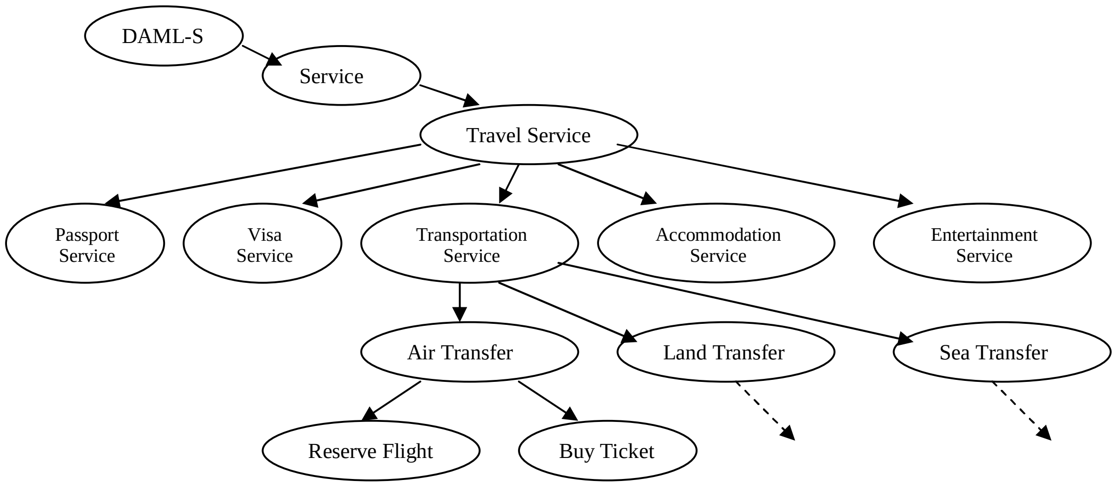

Figure 3. An example Travel Ontology

The personal agent then checks whether she needs a visa for the country she will be travelling and again it finds the service of the related Embassy on the Web to get a visa.

To further plan for the travel, the agent contacts the *domain specific registry agent* of a service registry for the "Travel" domain. Consider, for example, the "Travel" service ontology given in Figure 3 which is a classification schema that defines the semantics of a travel services. The *domain specific registry agent* queries the service ontology stored in the ebXML registry by predefined query templates to find the subclasses of the "Travel Service". This domain specific agent also has rules that are provided by a domain expert. These rules state the possible service ontology classes that might follow a given service ontology class. For example, a rule may state that a transportation service will follow a visa service. Using such rules the agent may infer that the next step is arranging a Transpostation Service. Again consulting to the ebXML classification Schema the agent discovers that there are several alternatives for transportation like, air, land and sea. Here the domain specific registry agent contacts with Maria's personal agent which in turn queries Maria's context to find out her "Travel Preferences".

In this way, the agent finds a generic service description (a node in the class hierarchy) from the ebXML classification hierarchy that is suitable for the user. For example, if Maria's travel preferences are air travel, her personal agent will convey this information to the *domain specific registry agent* so that the agent can access the instances of this specific generic service to find out the needed service. The details of this process are as follows: The services and the service ontology objects in the classification hierarchy are related with Classification objects. The agent retrieves the instances of the "Reserve A Flight" generic services by predefined query templates. The service instances in the registry have some slots for defining the properties of it. For example the instances of the “Reserve a Flight” generic services have a slot for relating them for specific Airline companies. The personal agent checks the profile of the user find out that Maria has a bonus card from a specific Airline company and chooses the corresponding airline service among the existing instances of the “Air Transfer” services registered to the ebXML registry.

Then a hotel may be reserved for Maria in the same way, by checking her preferences from the context server, and querying the ebXML registry.

After arranging the transportation and accomadation for Maria, the *domain specific registry agent* checks the Travel Ontology registered in the ebXML registry, and realizes that it can arrange an entertainment service for Maria. Then the personal agent querries the schedule of the user from the context server and finds out that she has a spare time between her meetings, and she likes to watch science fiction movies. The *domain specific registry agent* retrieves the insances of services selling cinema tickets, querries their timetables and offers Maria the possible choices.

With the help of her personal agent, Maria has a full travel plan before she arrives the Far East country. When she arrives at the airport, since her visa and passport have been arranged by her personal agent, she passes through the immigration office without stopping.

5. Related Work

The current context-aware applications are usually build in an ad hoc manner and lack the generality and standards to be of use in AmI environments.

Authentication services like Microsoft Passport [9] and AOL Screen Name [2], store and manage personal user data and provide single sign-in identity in different sites and pass personal information more easily. However the stored personal data is generally limited to user identification and user contact information that can be used in basic e-commerce sessions.

[5] describes the ePerson project developed at the HP Labs. An ePerson is a personal representative on the net that is trusted by a user to store and make available personal information under appropriate controls. Such personal information includes user profiles, shared content and shared meta-data (such as annotations, comments, ratings and categorisations). However the details of this project is not available in the literature.

Among several approaches for privacy management using service policies and privacy preferences, the most mature one is the Platform for Privacy Preferences Project (P3P) [3] developed by the World Wide Web Consortium (W3C). P3P enables Web sites to express their privacy practices in a standard format that can be retrieved automatically and interpreted easily by user agents like Web browsers. It should be noted that P3P is for Web sites and does not use Web semantics.

6. Conclusions

AmI environments combine ubiquity, context-awareness, intelligence and natural interaction. Context awareness refers to the ability of the system to locate and recognize objects and people, and their locations. Intelligence allows the system to analyze the context, adapt to people that live in it, learn from their behavior, and eventually to recognize. A consistently strong response to AmI is the need to build trust and confidence.

In this paper we describe a framework for handling context information in Ambient Intelligence environments. For this purpose, we describe the need to develop context ontologies for describing user contexts; the mechanisms necessary to provide for the security and privacy of context information. Finally, we explain how user context information can be exploited to find and compose semantically enriched services.

We conclude by noting that to be acceptable AmI needs to be driven by humanistic concerns, not technologically determined ones.

References

[1] S. Alexaki, V. Christophides, G. Karvounarakis, D. Plexousakis, and K. Tolle. The ICS-FORTH RDFSuite: Managing Voluminous RDF Description Bases, Proc. of the Second International Workshop on the Semantic Web (SemWeb2001), May 2001.
[2] AOL Screen Name, http://my.screenname.aol.com
[3] Cranor L., Langheinrich M., Marchiori M., Presler-Marshall M., Reagle J., "The Platform for Privacy Preferences 1.0 (P3P1.0) Specification", W3C Recommendation, http://www.w3.org/TR/P3P, April 16, 2002.
[4] DAML+OIL, http://www.w3.org/2001/10/daml+oil
[5] e-person: Personal Information Infrastructure, http://www.hpl.hp.com/semweb/e-person.htm
[6] ISTAG Scenarios for Ambient Intelligence in 2010, http://www.cordis.lu/ist/istag.htm
[7] Karlsruhe Ontology (KAON) tool suite, http://kaon.semanticweb.org/.
[8] Minami, K., Kotz, D., "Controlling Access to Pervasive Information in the "Solar System", Technical Report, TR2002-422, Dartmouth University, Feb. 2002, http://www.cs. dartmouth.edu/reports/abstracts /TR2002-422/
[9] Microsoft Passport, http://www.microsoft.com/myservices/passport.
[10] Web Ontology Language (OWL) Reference Version 1.0, http://www.daml.org/2002 /06/webont/owl-ref-proposed
[11] Resource Description Framework Schema Specification, W3C Proposed Recommendation, 1999, http://www.w3.org/TR/PR-rdf-schema.
[12] Resource Description Framework Model and Syntax Specification, W3C Recommendation, 1999, http://www.w3.org/TR/REC-rdf-syntax.
[13] Volz, R., Madche, A., Oberle, D., "Towards Views in Semantic Web", 2nd Intl. Workshop on Databases, Documents and Information Fusion (DBFUSION 02), July 2002, Karlsruhe, Germany.