Chapter 2: Ambient Intelligence: The MyCampus Experience

Norman Sadeh, Fabien Gandon, Oh Buyng Kwon

Chapter 2

Ambient Intelligence: The *MyCampus* Experience

Norman M. Sadeh, Fabien L. Gandon and Oh Byung Kwon

School of Computer Science
Carnegie Mellon Univerity
Pittsburgh, PA 15213
{sadeh; gandon; obkwon}@cs.cmu.edu

Abstract. Over the past five years, the MyCampus group at Carnegie Mellon University has been developing and experimenting with *Ambient Intelligence* technologies aimed at enhancing everyday life. The project has drawn on multiple areas of expertise, combining the development of an open Semantic Web infrastructure for context-aware service provisioning with an emphasis on issues of privacy and usability. In this paper, we review key motivations behind the project, discuss the MyCampus Semantic Web infrastructure and report on our experience tailoring the architecture for different environments (e.g. everyday campus life applications, office applications, museum tour guide). This includes a discussion of Semantic e-Wallets aimed at reconciling user demands for context awareness and privacy as well as a description of different context-aware applications developed and evaluated during the course of the project. We also discuss our experience using Case-Based Reasoning (CBR) functionality developed to overcome usability issues associated with capturing complex, context-sensitive user preferences. The paper concludes with a summary of lessons learned so far and of challenges still to be addressed.

Keywords: Ambient Intelligence, Privacy, Interoperability, Usability, Web Services, Context Awareness, Semantic Web, Intelligent Agents, Mobile Access, Case-Based Reasoning.

Introduction

Increasingly, application developers are looking for ways to provide users with added levels of convenience and ease of use through functionality that is capable of capturing the context within which they operate. This may involve knowing where the user is located, the task she is currently engaged in, her eating preferences, who her colleagues are as well as a variety of other contextual attributes. While there are many sources of contextual information, they tend to vary from one user to another and also over time. Different users may rely on different location tracking functionality provided by different cell phone operators; they may use different calendar systems, etc. Traditionally, context-aware applications and services have been hardwired to predefined sources of contextual information (e.g. relying on a particular set of sensors and protocols to track a user's locations). As a result, they remain prohibitively expensive to build and maintain and are few and between. We argue that what is needed is a more open environment, where context-aware applications can automatically discover and access a user's personal resources such as her calendar or location tracking functionality. This can be done by viewing each source of contextual information (or personal resource) as a Web service. Unfortunately, current Web Services standards such as UDDI [1] or WSDL [2] are not sufficient when it comes to describing a user's personal resources and to enabling automated access to them by context-aware applications. Another challenge, as we move towards more open platforms for access to a user's personal information, revolves around privacy issues. Users should be able to retain control over who has access to their personal information under different conditions. For instance, I may be willing to let my colleagues see where I am or access my calendar activities between 8am and 5pm on weekdays but not over the weekend. In addition, I may want to fine tune the granularity of the answer provided to a given query, depending on the context of that query. For instance, I may be willing to disclose the room that I am in to some people but only the city where I am to others In fact, I may even want to give different answers to different people, telling my secretary I am off to see my dentist, while telling my customers I am busy in a meeting.

Over the past five years, the MyCampus group at Carnegie Mellon University has been developing and experimenting with *Ambient Intelligence* technologies aimed at enhancing everyday life. The project has drawn on multiple areas of expertise, combining the development of an open Semantic Web infrastructure for re-usable, context-aware service provisioning with an emphasis on issues of privacy and usability. In this paper, we provide an overview of our Semantic Web infrastructure for Ambient Intelligence. Within this infrastructure, each source of contextual information (e.g. a calendar, location tracking functionality, collections of relevant user preferences, organizational databases) is represented as a Semantic Web service. A central element of our infrastructure is its *semantic e-Wallets*. Each Semantic e-Wallet acts as a directory

of contextual resources for a given user, while enforcing her *privacy preferences*. Privacy preferences enable users to specify what information can be provided to whom in different contexts (*access control*). They also allow users to specify what we call *obfuscation rules*, namely rules that control the accuracy or inaccuracy of the information provided in response to different queries under different conditions.

We have validated our infrastructure in the context of several Ambient Intelligence environments: an environment aimed at enhancing everyday campus life at Carnegie Mellon University (CMU), a prototype context-aware museum tour guide developed for a museum in Taiwan and several context-aware office applications. Thanks to our Semantic Web infrastructure, each of these environments allows for the development of a growing collection of context-aware applications (often implemented as agents) capable of dynamically leveraging contextual information. This generally includes accessing location information, calendar activities as well as a variety of other contextual attributes and preferences relevant to each environment. At CMU, students access the MyCampus environment from PDAs over the campus's 802.11 wireless LAN. Empirical results obtained with a group of students over a period of several days are summarized at the end of this article along with a brief discussion of Case-Based Reasoning (CBR) functionality developed to learn context-sensitive user preferences.

The remainder of this article is organized as follows. Section 2 provides a brief overview of the state of the art in context awareness, privacy, Web Services and the Semantic Web, emphasizing limitations of the work reported so far in the literature. In Section 3, we provide an overview of our open Semantic Web infrastructure for context-aware service provisioning and privacy. Section 4 focuses more specifically on the Semantic e-Wallet and includes a high-level scenario outlining its operation in response to a query about the current location of a user. Section 5 discusses issues relating to capturing user preferences. Section 6 discusses our experience tailoring and deploying our infrastructure in support of different Ambient Intelligence environments. This includes examples of context-aware applications (or agents) our team has developed. Section 7 reports on an evaluation study conducted with a number of users on Carnegie Mellon's campus along with a discussion of CBR functionality to learn context-sensitive user preferences. Section 8 summarizes what we view as some of the main contributions of our work so far and briefly discusses our ongoing research.

1 Prior Work

Prior efforts to develop context aware applications are many. Early work in context awareness includes the Active Badge System developed at Olivetti Research Lab to redirect phone calls based on people's locations [3]. The ParcTab system developed at the Xerox Palo Alto Research Center in the early nineties relied on PDAs to support a variety of context-aware office applications (e.g. locating nearby resources such as printers, posting electronic notes in a room, *etc*.) [4, 5]. Other relevant applications that have emerged over the years range from location-aware tour guides to context-aware memory aids. More recent research efforts in context awareness include MIT's Oxygen [6], CMU's Aura [7] and several projects at Berkeley's GUIR (e.g. [8, 9]) to name just a few.

While early context-aware applications relied on *ad hoc* architectures and representations, it was quickly recognized that separating the process of acquiring contextual information from actual context-aware applications was key to facilitating application development and maintenance. Georgia Tech's Context Toolkit represents the most significant effort in this direction [10, 11]. In the Context Toolkit, widgets act as wrappers that provide access to different sets of contextual information (e.g. user location, identity, time and activity), while insulating applications from context acquisition concerns. Each user (as well as other relevant entities such as physical objects or locations) has a context server that contains all the widgets relevant to it. This is similar to our notion of e-Wallet, which serves as a directory of all personal resources relevant to a given user (e.g. relevant location tracking functionality, relevant collections of preferences, access to one or more calendar systems, *etc*.). Our Semantic e-Wallet however goes one step beyond Dey's Context Toolkit. It makes it possible to leverage much richer models of personal resources - what personal information they give access to, when to access one rather than the other, how to go about accessing these resources. In addition, it includes access control and obfuscation functionality to enforce user privacy preferences. This richer model is key to supporting automated discovery and access of a user's personal resources by agents. In other words, while the Context Toolkit focuses mainly on facilitating the development of context-aware applications through off-line, re-use and integration of context-aware components (i.e. widgets), our architecture emphasizes real-time, on-the-fly queries of personal resources by context-aware agents. These queries are processed through several layers of functionality that support automated discovery and access of relevant personal resources subject to user-specified privacy preferences.

The notion of e-Wallet as introduced in systems such as Microsoft's .NET Passport is not new. However current implementations have been limited to storing a very small amount of information and offer very restricted control to the user when it comes to specifying what information can be made available to different services. For instance, in Passport, users can specify whether or not they are willing to share parts of their profiles with all participating sites but cannot distinguish between different participating sites. Our notion of Semantic e-Wallet lifts these restrictions and allows users to control access to any of their personal resources. It also allows for multiple sources of similar information (e.g. multiple calendars or multiple location tracking functionality) and for functionality that can dynamically select which of these resources to tap based on the context and the nature of the query at hand (e.g. using your car's GPS system when you are driving and your cell phone operator's location tracking functionality when you are not).

Our notion of semantic e-Wallet extends recent efforts to develop rich languages for capturing user privacy preferences such as P3P's APPEL language [12]. It does so by making it possible to leverage any number of domain ontologies and by allowing for preferences that relate to any

number of contextual attributes. In addition, it allows users to specify obfuscation rules through which they control the level of accuracy (or inaccuracy) at which their contextual information is disclosed to different parties under different conditions. This includes telling some people which room you are in, while simply telling others whether you are at work or not, or whether you are in town or not. It also includes scenarios where you might want to pretend you are in one place, while you are really elsewhere.

Last but not least, while the security community has developed powerful languages to capture access control privileges such as the Security Assertion Markup Language (SAML) [13], the XML Access Control Markup Language (XACML) [14] and the Enterprise Privacy Authorization Language (EPAL) [15], these languages, like P3P, do not take advantage of Semantic Web concepts. Our work builds directly on recent efforts aimed at moving the Web from an environment where information is primarily made available for human consumption to one where it is annotated with semantic markup that makes it understandable to software applications. These efforts are part of a long-term vision generally referred to as the *Semantic Web* [16, 17]. They have already resulted in a succession of semantic markup languages [18, 19] as well as early efforts to define Web Service ontologies and markup in the context of languages such as DAML-S [20], OWL-S [21] and WSMO [22]. In our work, we have relied on the use of DAML+OIL [DAML01] and more recently OWL [19] to represent contextual information (e.g. location, calendar activities, social and organizational relationships, *etc*.) and privacy preferences and on Semantic Web service concepts to support the automated discovery and access of personal and public resources. While a number of recent efforts concurrent to our work have looked at different aspects of privacy in Ambient Intelligence environments (e.g. [9,23,24,25,26,27]), to the best of our knowledge, the MyCampus project was the first to introduce a Semantic Web architecture for context-aware service provisioning and privacy. In addition, the project distinguishes itself by its unique blend of a strong technology-push approach to Ambient Intelligence (e.g. Semantic e-Wallets, Case-Based Reasoning, Agent technologies) with a strong emphasis on usability.

2 Overall System Architecture

We consider an environment where users rely on an open set of task-specific applications (or agents) to assist them in the context of different activities (e.g. scheduling meetings with colleagues, reminding them of purchases they need to make, arranging trips or filtering incoming messages) [28,29]. Some of the agents may be public or semi-public resources (e.g. an agent to help locate nearby printers in a building); others may be applications the user is subscribing to or applications she has downloaded on her PDA or her onboard computer. To function, each agent needs to access some information about its user as well as possibly other users. Access to a user's personal (or contextual) information is controlled by that user's e-Wallet subject to privacy (enforcing) rules. The e-Wallet Manager (or simply e-Wallet) serves as a repository of static knowledge about the user – just like .NET Passport, except that here knowledge is represented using OWL. In addition, the e-Wallet contains knowledge about how to access more information about the user by invoking a variety of resources, each represented as a Web Service. This knowledge is stored in the form of rules that map different contextual attributes onto one or more possible service invocations, enabling the e-Wallet to automatically identify and activate the most relevant resources in response to queries about the user's context (e.g. accessing the user's calendar to find out about her availability, or consulting one or more location tracking applications in an attempt to find out about her current location). User-specified privacy rules, also stored in the e-Wallet, ensure that information about the user is only disclosed to authorized parties, taking into account the context of the query. They further adjust the accuracy or inaccuracy of the information provided in accordance with so-called "obfuscation" preferences.

Figure 1 provides an overview of our Semantic Web environment. It illustrates a situation where access is from a PDA over a wireless network. However, our architecture extends to fixed Internet scenarios and more generally to environments where users can connect to the infrastructure through a number of access channels and devices – information about the particular access device and channel can actually be treated as part of the user's context and be made available through her e-Wallet [30]. As can be seen in Figure 1, other key elements of our architecture include:

- One or more *Platform Managers* that build on top of Directory Facilitators and Agent Management Systems, as defined in FIPA [31]. They manage the agents running at their sites, and maintain white and yellow page directories of these agents and the services they provide.
- *User Interaction Managers* that are responsible for interactions with the user. This includes managing login sessions as well as interactions with the user's agents and her e-Wallet. Because different users interact with different sets of agents, this also includes the dynamic generation of interfaces for interacting with these agents and the customization of these interfaces to the current interaction context (e.g. particular access device). Communication with the User Interaction Manager typically takes place through a number of APIs, e.g. an Instant Messaging API, an HTTP/HTML API, etc.

Clearly, agents are not limited to accessing information about users in the environment. Instead, they also typically access public Web Services, Semantic Web annotations, public ontologies and other public resources. On CMU's campus, where we have deployed *my*Campus, this includes access to a variety of services such as 23 restaurant web services or a pubic weather forecasting web service.

In the following sections, we focus on the e-Wallet functionality. Additional details on *my*Campus and some of the agents we have deployed can be found in [28,29,32,33].

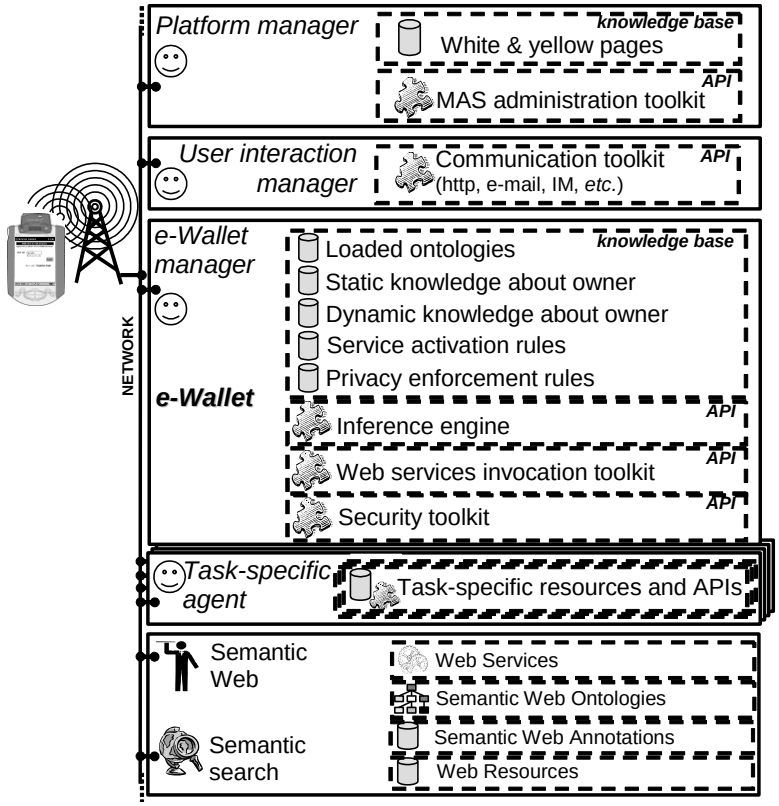


Figure 1 *my*Campus architecture: a user's perspective - the smiley faces represent agents.

3 A semantic e-Wallet

The e-Wallet is a central element of our Semantic Web architecture for context-awareness and privacy. It provides a unified and secure semantic interface to all the user's personal resources [34], enabling agents in the system, whether working for the owner of the e-Wallet or for other users, to access and, when appropriate, modify information about the user subject to that user's privacy preferences (e.g. not just determining whether the user is available between 3 and 4pm but also, possibly, scheduling a meeting at that time). The e-Wallet is *not* a static information repository. While it does contain some static information about the user, it is an agent acting as clearinghouse and gatekeeper for a user's personal resources. Its knowledge about the user, her personal resources and preferences falls into four categories:

1. **Static knowledge**. This context-independent knowledge typically includes the user's name, her email address, employer, home address as well as context-independent preferences (e.g. "I like spicy vegetarian cuisine"). This knowledge, like all other in the e-Wallet, can be edited by the user via the User Interaction Manager.
2. **Dynamic knowledge**. This is context-sensitive knowledge about the user, often involving a variety of preferences such as "When driving, I don't want to receive instant messages".
3. **Service invocation rules**. These rules help leverage information resources external to the e-Wallet – both personal and public. They effectively turn the e-Wallet into a semantic directory of personal resources that can be automatically discovered and accessed to process incoming queries. Specifically, service invocation rules provide a mapping between contextual attributes and personal resources available to access these attributes, viewing each personal resource as a Semantic Web service. An example of one such mapping is a rule indicating that a query about the user's current activity can be answered by accessing her Microsoft Outlook calendar. We have developed Web Service wrappers for a variety of personal resources such as Microsoft Outlook Calendar or location tracking functionality. Service invocation rules are not limited to providing a one-to-one mapping between contextual attributes and personal resources. Instead, they can leverage rich ontologies of personal resources, enabling the e-Wallet to select among a number of possible personal resources based on availability, accuracy and other relevant considerations. For instance, in response to a query about the user's location, the rules can specify that, when the user is driving, the best method available is the GPS in her car. If she is at work and her wireless-enabled PDA is on, her location can be obtained using location tracking functionality running over the enterprise's wireless LAN. If everything else fails, her calendar might have some information about her location. Finally, it should be noted that to answer queries about the user, additional mapping rules that support automated discovery and access of public services may also be needed. For instance, a query like "Tell me whether Fabien is in a sunny place right now" will typically require accessing Fabien's location as well as a public weather service.
4. **Privacy preferences**. These preferences encapsulate knowledge about what information about herself the user is willing to disclose to others under different conditions. These preferences themselves fall into two categories:
   - *Access control rules*. These rules simply express who has the right to see what information under different conditions e.g. "My location should only be visible to members of my team during week days between 8am and 5pm".
   - *Obfuscation rules.* Often user privacy preferences are not black-and-white but rather involve different levels of accuracy or inaccuracy: *Obfuscation by abstraction* is about abstracting away some details about the user's current context such as telling people whether or not you are in town without giving your exact location. *Obfuscation by falsification* is about scenarios where the user may not want to appear as if she is

withholding information but would rather provide false information. For instance, a user may not want to reveal her true email address to a web service for fear of getting spammed.

All the above knowledge (including rules [35]) is represented in OWL, referring to a number of relevant ontologies (e.g. ontologies about contextual attributes, personal resources, as well as more specific knowledge such as cuisine types and food preferences or message types and message filtering preferences).

A scenario will help illustrate the key steps an e-Wallet goes through when processing an incoming query (Figure 2). For the sake of argument, we consider a query submitted by a user (Norman) to the e-Wallet of a second user (Fabien) to find out about that second user's current location. We assume that Fabien has specified that only his colleagues can access his location and only when he is on campus. In addition, when on campus, Fabien is only willing to disclose the building he is in but not the actual room.

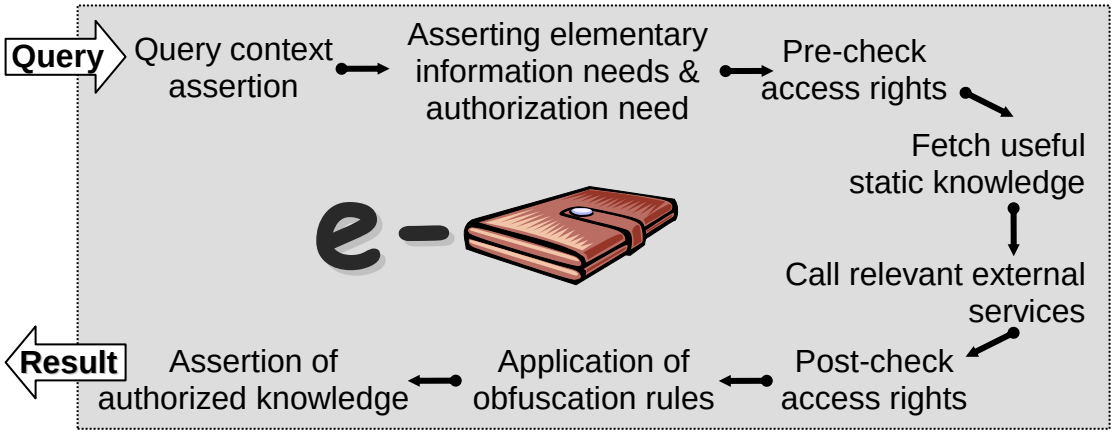


Figure 2 Main steps involved in processing a query submitted to an e-Wallet.

The main steps involved in processing this query are:

1. **Asserting the query's context**: As a first step, facts about the context of the query are asserted – namely they are loaded into the e-Wallet's inference engine for possible use as part of inferences to be made in processing the query. In our example, one such assertion is that "the sender of the query is Norman".
2. **Asserting elementary information needs and the need to go through an authorization process**: Here the query is translated into an aggregate goal that includes (a) a combination of elementary information needs – in our example the need to find "Fabien's location", along with (b) a requirement to go through an authorization process. The authorization process, which is distributed across some of the following steps, results in the request being either denied or cleared, the latter possibly following the application of obfuscation rules. In our example, the authorization goal requires checking that Norman is entitled to having access to Fabien's location and that the level of resolution at which the query is answered is compatible with Fabien's privacy preferences.
3. **Pre-checking whether the query is allowable**: A first check is performed to see whether the query is allowable based on access rights considerations. In our example, Fabien has specified that only his colleagues can access his location and only when he is on campus. In this example, for the sake of simplicity, we will assume that Norman is indeed a colleague of Fabien's and this fact is pre-stored in Fabien's e-Wallet. The other access right condition specified by Fabien, namely that he has to be on campus, still needs to be checked. This information however is not available locally. So the e-Wallet postpones the verification of this condition – see below.
4. **Checking the e-Wallet's local knowledge base**: Some queries can be answered in whole or in part, using facts in the e-Wallet's local knowledge base, which, as we have seen in Section 3,

contains both static (namely, context-independent) and dynamic (namely, context-sensitive) knowledge about the user. In our particular example, Fabien's location, which changes all the time, is not known locally by his e-Wallet. Accordingly, the e-Wallet needs to turn to outside sources of personal information to answer the query (see next step).

5. **Invoking personal resources as Web services**: When local knowledge is not sufficient to answer a query, the e-Wallet turns to its service invocation rules to identify external resources that might help answer the query. This may involve accessing one or more of the user's personal resources such as his calendar and/or one or more trusted public services. In our example, the campus where Fabien works has a wireless LAN that supports location tracking. This functionality can be invoked by the e-Wallet to obtain Fabien's location. The actual invocation takes place through the web service invocation toolkit already introduced in Figure 1. In this particular case, the campus's location tracking service reports back that Fabien is in "room Smith 234", which is on Carnegie Mellon's campus.
6. **Post-checking whether the query is allowable**: Now armed with knowledge of Fabien's location, the e-Wallet can check the second access right condition specified by Fabien, namely that he be on campus. In our example, this condition is met since Smith Hall is on Carnegie Mellon's campus. For the sake of simplicity, we assume that Fabien's e-Wallet possesses facts about buildings on Carnegie Mellon's campus. Otherwise, another external service would have to be identified and invoked to obtain this information. Fabien's e-Wallet has now established that Norman's request is allowable. This does not mean however that the authorization process required as part of the goals set in step 2 has been fully completed. Obfuscation rules may still need to be applied.
7. **Application of Obfuscation Rules**: Returning Fabien's room information obtained from the location tracking functionality would violate his obfuscation policy of just disclosing the building he is in. Accordingly, using its knowledge of rooms and buildings on Carnegie Mellon's campus, the e-Wallet looks for the building that "room Smith 234" is in, namely "Smith Hall".
8. **Generating an answer**: The query has now been fully processed and an acceptable answer generated. This answer (e.g. "Fabien is in Smith Hall") can be returned to Norman.

As shown in Figure 3, we developed a three-layer implementation of our e-Wallet:

- *Core Layer*: At the most basic level, the e-Wallet's knowledge includes an OWL meta-model – required to interpret OWL statements. In addition, it maintains both static (context-independent) and dynamic (context-dependent) knowledge about the user. This knowledge is obtained by loading available annotations about the user along with relevant ontologies and is currently completed using forward-chaining reasoning – to avoid having to infer the same facts over and over again.
- *T*he *Service Layer* completes the e-Wallet's core knowledge with invocation rules that map information retrieval goals about contextual attributes onto external service invocations. These are modeled as backward-chaining rules. Given an information retrieval goal such as "Give me Fabien's location", they help identify and invoke one or more relevant information resources, each modeled as a Web service.
- The outer layer is referred to as the *Privacy Layer*, as this is where privacy (enforcing) rules are applied. *Only authorized knowledge can be sent in response to queries*. Authorized knowledge is generated by applying privacy enforcing rules to service knowledge and core knowledge, thereby ensuring that information about the user is only disclosed to authorized parties and in accordance with relevant obfuscation rules.

  Privacy enforcing rules are encoded as backward-chaining rules. These rules map needs for authorized knowledge onto needs for service knowledge from the service layer to be post-processed subject to the privacy enforcing rules. Upon receiving an incoming query, the e-

Wallet generates a need for authorized knowledge required to answer the query. This need in turn typically triggers needs for service knowledge and core knowledge, eventually resulting either (a) in the generation of authorized knowledge that can be returned in response to the query or (b) in an exception, if the query is found unallowable (e.g. an unauthorized party requesting your location or trying to schedule a meeting in your calendar). Security is directly enforced through the typing of knowledge representation structures.

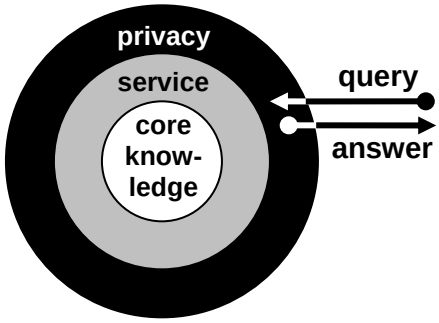


Figure 3 Three-layer architecture

The current implementation of our e-Wallet is based on JESS [36], a high-performance Java-based rule engine that supports both forward and backward chaining – the latter by reifying "needs for facts" as facts themselves, which in turn trigger forward-chaining rules. The e-Wallet's knowledge base is initialized with: (a) a model of RDF [37] triples as a template for unordered facts, (b) a model of specialized triples used in our three layers (core triples, service triples and authorized triples) along with associated migration rules between the layers, and (c) an OWL meta-model.

Additional knowledge is loaded into the e-Wallet by translating OWL input files into JESS assertions and rules, using a set of XSLT stylesheets [38] (Figure 4). The OWL input files include ontologies and annotations that are transformed into (core) triple assertions, forward-chaining rules [35] (used to complete knowledge at the core layer) as well as service invocation rules and privacy enforcing rules – both represented as backward-chaining rules. The XSLT templates act as meta-rules that generate the body, the head and typing used by the JESS rules.

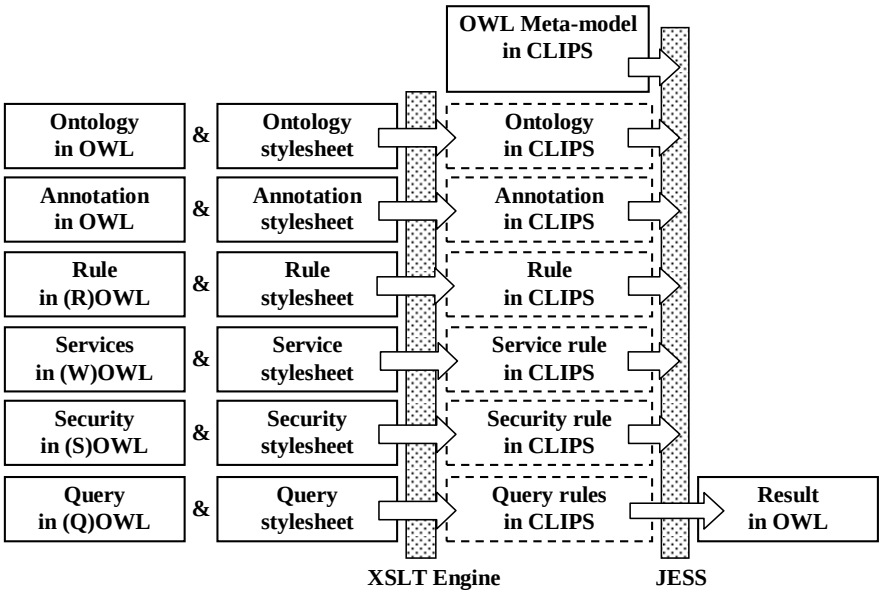


Figure 4 High-level flows and processes in the e-Wallet

As shown in Figure 5, *privacy enforcing rules* are defined using three tags: the content of the *target tag* describes the piece of knowledge to which this rule applies; the content of the *check tag* describes the conditions under which read access is granted; the content of the *revision tag* describes the obfuscation to be applied before migrating triples to the authorized layer. Note that, at the time of writing, our e-Wallet also supports limited write access rules.

```
<sowl:ReadAccessRule>
  <rdfs:label>people can only know whether or not I am on campus</rdfs:label>
  <sowl:target>
    <mc:Person rdf:about="&variable;#owner">
      <mc:location rdf:resource="&variable;#location"/>
    </mc:Person>
  </sowl:target>
  <sowl:check>
   <rowl:And>
     <rowl:condition>
       <mc:E-Wallet rdf:about="&variable;#e-Wallet">
         <mc:owner>
          <mc:Person rdf:about="&variable;#owner"/>
         </mc:owner>
       </mc:E-Wallet>
     </rowl:condition>
     <rowl:condition>
       <mc:Place rdf:about="http://www.cmu.edu">
          <mc:include rdf:resource="&variable;#location" />
       </mc:Place>
     </rowl:condition>
     <rowl:not-condition>
       <qowl:Query rdf:about="&variable;#query">
         <qowl:sender rdf:resource="&variable;#owner" />
       </qowl:Query>
     </rowl:not-condition>
   </rowl:And>
  </sowl:check>
  <sowl:revision>
    <mc:Person rdf:about="&variable;#owner">
      <mc:location rdf:resource="http://www.cmu.edu"/>
```

```
    </mc:Person>
  </sowl:revision>
</sowl:ReadAccessRule>
```

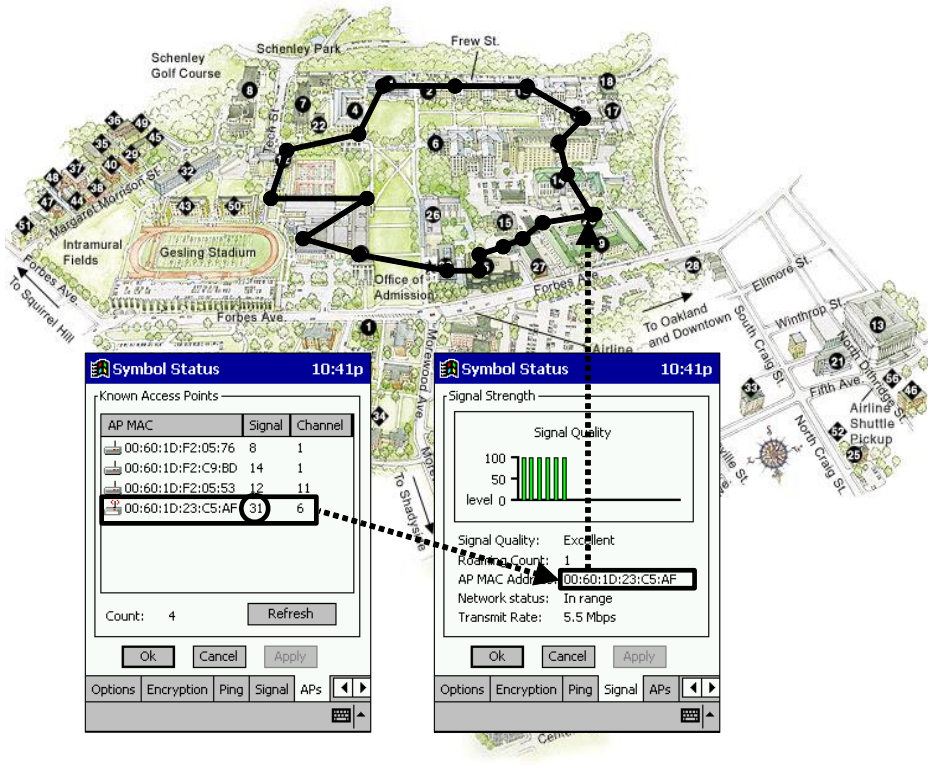
Figure 5 Privacy rule obfuscating the location of the owner

```
<wowl:ServiceRule wowl:salience="50">
  <rdfs:label>provide activity status for a person</rdfs:label>
  <wowl:output>
    <mc:Person rdf:ID="&variable;#person">
      <mc:has_activity rdf:resource="&variable;#activity" />
    </mc:Person>
  </wowl:output>
  <wowl:precondition>
    <mc:Person rdf:ID="&variable;#owner">
      <mc:PDA_endpoint>&variable;#endpoint</mc:PDA_endpoint>
    </mc:Person>
  </wowl:precondition>
  <wowl:call>
   <wowl:Service wowl:name="call-web-service">
     <wowl:qname>http://mycampus/PDAService#</wowl:qname>
     <wowl:endpoint>&variable;#endpoint</wowl:endpoint>
     <wowl:method>GetCurrentWeekAppointments</wowl:method>
     <wowl:user_id>&variable;#owner</wowl:user_id>
   </wowl:Service>
  </wowl:call>
</wowl:ServiceRule>
```

Figure 6 Service rule for activity-tracking invocation in WOWL

As shown in Figure 6 the *service rules* have three child tags: the content of the *output tag* describes the piece of knowledge that this rule can produce; the content of the *precondition tag* describes the knowledge needed for calling the service; the content of the *call tag* describes the function to trigger and its parameters.

The body of each service rule requires a need for a particular piece of information (e.g. Fabien's location) along with the availability of a specific set of arguments (e.g. knowledge of the IP address of Fabien's PDA). When these conditions are matched, the rule fires and calls the service (Figure 7 depicts the semantic web service used to support location-tracking over CMU's wireless LAN).



Annotation of access points on campus:

```
<mc:AccessPoint>
  <mc:MAC>00:60:1D:23:C5:AF</mc:MAC>
  <mc:location rdf:resource="http://www.cmu.edu/SmithHall" />
</mc:Entity>
```

Result of invoking the location-tracking web service:

```
<mc:Entity rdf:ID="http://cs.cmu.edu/~fgandon">
  <mc:location rdf:resource="http://www.cmu.edu/SmithHall" />
</mc:Entity>
```

Figure 7 Semantic web service for location-tracking over CMU's wireless LAN.

4 Capturing User Preferences

As should be clear by now, our framework based on semantic web technologies is capable of capturing a broad range of user preferences that may refer to any relevant set of OWL ontologies. This is true for message filtering preferences, food preferences, privacy preferences, scheduling preferences, *etc.* One approach to capturing these preferences is to develop a variety of special-purpose editing tools that enable users to specify their preferences with regard to predefined sets of ontologies. For instance, each time a user subscribes to (or acquires) a new task-specific agent, she might be prompted by a special-purpose editor to select from a predefined set of preference options. The same could be done to capture predefined sets of privacy preferences. However, a key objective in our architecture has been to provide for an open environment, where new sources of contextual information, new contextual ontologies and new agents can be introduced over time. Supporting the capture of user privacy preferences in this broader context ideally requires a general-purpose privacy preference editor that can refer to any relevant source of contextual information and any relevant contextual ontology. Figure 8 shows screenshots of such a general-

purpose privacy preference editor – to capture both access control preferences and obfuscation preferences. While the editor is clearly too complex to be placed in the hands of lay users, it can be made available to system administrators who can use it to capture a user's individual preferences (as well as relevant company policies in the case of employees). The editor uses XSLT stylesheets and allows users (typically system administrators) to browse (Figure 8-a) and edit privacy rules/policies (Figure 8- b and c). It allows users to create new rules as well as edit and delete existing ones. The editor enables users to express privacy/confidentiality policies that relate to concepts and properties defined in currently available ontologies, namely ontologies loaded in the e-Wallet. The editor takes into account the OWL meta-model as the user edits rules. For instance, it will restrict the instantiation of a given concept to be within the range of a given property, as specified using the OWL "ObjectProperty" construct [19].

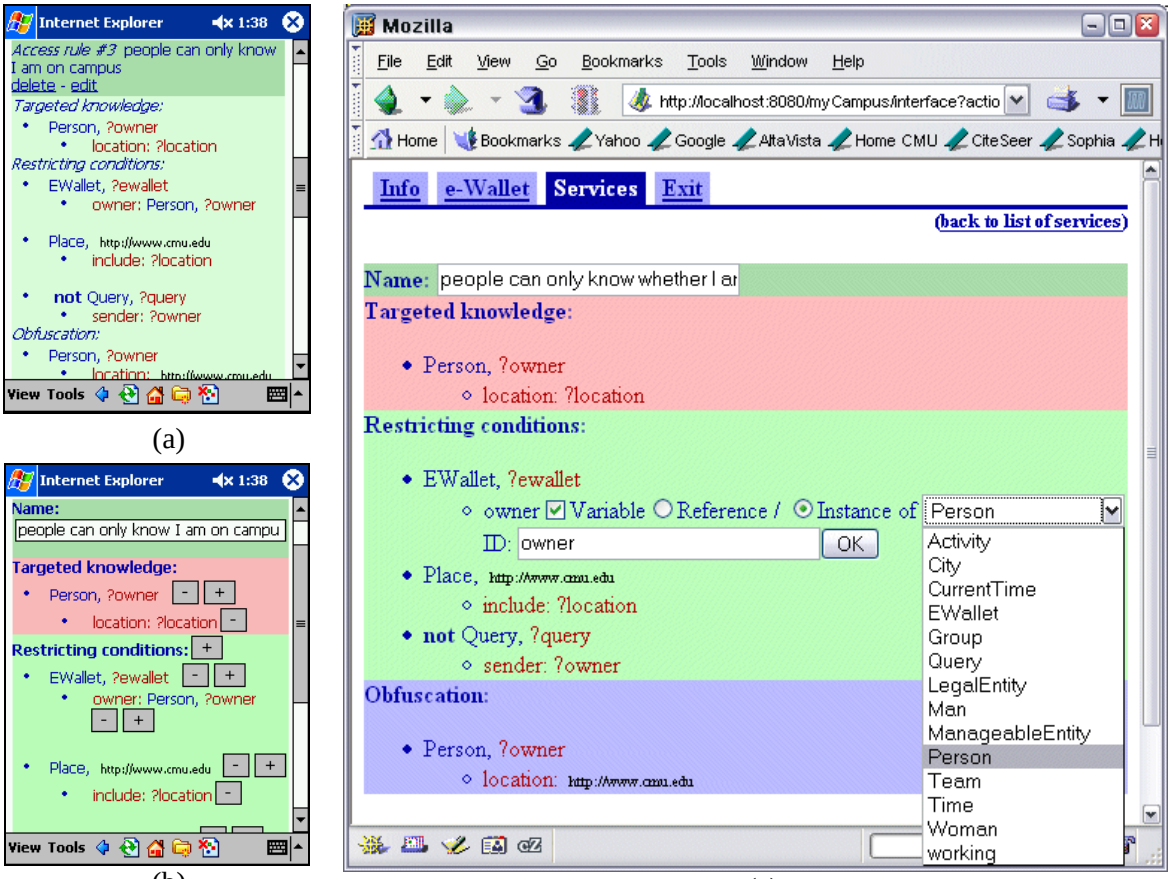


Figure 8 Generic rule editor that enables administrators to (a) browse and (b) (c) edit OWL-based privacy/confidentiality preferences.

Our experience developing and evaluating context-aware applications, as discussed below in Section 6 and 7, has shown that users often have complex and nuanced privacy preferences. Putting general purpose interfaces such as the one shown in Figure 8 in their hands and expecting them to specify their preferences is simply unrealistic. General purpose preference editors can at best be given to well trained system administrators. In addition, users often do not even know their privacy preferences until actually confronted with a situation. Accordingly, we believe that a more pragmatic approach to capturing user preferences will often require the introduction of learning technology capable of progressively developing preference models based on user

feedback. In Section 7, we report on our experience using Case-Based Reasoning (CBR) to learn a user's context-sensitive message filtering preferences. While preliminary, empirical results obtain so far are rather encouraging and we are starting work aimed at generalizing this functionality in the context of privacy preferences.

5 Instantiating the *MyCampus* Infrastructure

The MyCampus infrastructure has been instantiated in the context of several prototype Ambient Intelligence environments, including:

- An environment aimed at enhancing everyday campus life at Carnegie Mellon University
- A museum tour guide environment developed for the National Museum of Natural Science in Taiwan
- A smart office environment

Below, we briefly review the instantiation of MyCampus developed for Carnegie Mellon's campus. Details of our museum tour guide instantiation can be found in [33] and details on our work on smart office applications are provided in [30].

The MyCampus infrastructure was originally conceived to enhance everyday campus life at Carnegie Mellon University through the incremental development of an open collection of context-aware applications. Campuses can be viewed as everyday life microcosms, where users (e.g. faculty, staff and students) engage in a broad range of activities, from attending lectures and studying to socializing, having meals, making purchases, attending movies, etc. As such a campus is representative of many of the challenges involved in successfully developing and deploying Ambient Intelligence applications. Over the years, the MyCampus team has worked with different groups of users to identify, design and refine applications that could help them in the context of their daily activities. Most of these applications are implemented as context-aware agents in JADE [39]. They can automatically discover the e-Wallets of relevant users and access their contextual information subject to privacy preferences specified by these users in their e-Wallets. Sources of contextual information are wrapped as web services. Shared ontologies ensure that the information provided by these services is understood by the agents. Some of these applications are briefly described below.

- **Context-aware recommender services:** Several such applications have been developed and experimented with over the past few years. They include recommender services for places where to eat, nearby movies and public transportation recommendation services. Many of these services extend beyond the geographical area covered by the campus.

Figure 9 Screenshots of the restaurant concierge

Screenshots of one such service, namely a context-aware "Restaurant Concierge", are shown in Figure 9. The "Restaurant Concierge" makes suggestions on where to eat based on a number of preferences (e.g. types of cuisine, budget) as well as contextual considerations such as where the user is, how far she is from a given restaurant, how much time she has until her next meeting, the weather, etc. The concierge operates as a public service and obtains the user's preferences and contextual information by querying her e-Wallet. For instance, in response to a query about its user's location, the e-Wallet checks the user's privacy preferences and contacts location tracking functionality wrapped as a web service. A total of 23 web services are used to provide information (e.g. cuisine, location, menu, etc.) about different places where users can eat on or around campus. The "eat here" button helps collect feedback from users - to determine the extent to which current settings properly capture the user's preferences as well as to help assess the overall usefulness of the application.

- **Context-aware message filtering services**

The idea behind this application is that users do not necessarily want to see right away messages sent to them. In it simplest form, the service allows a user to specify preferences as to when she wants to see different types of messages based on the nature of the message (i.e. subject and sender) as well as based on her activities as specified in her calendar (see Figure 10). For instance, a user can specify that messages sent to her while she is in class should only be shown when her class is over. Here again the user's calendar is wrapped as a web service. Calendar information is obtained via the user's e-Wallet, which in turn invokes the user's calendar web service. The application also allows users to select among different delivery channels depending on their contexts. In addition, users can provide feedback to help the system refine the preferences they originally entered. This latter functionality, which relies on case-based reasoning, has proved particularly effective, as reported in Section 7.

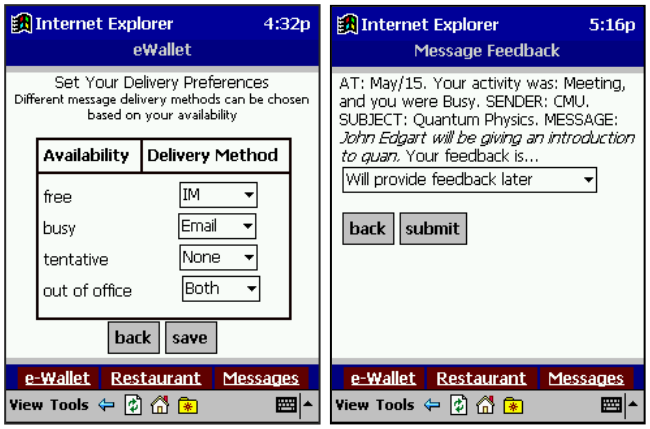


Figure 10 Screenshots of an early version of the context-aware message filtering service.

- **Context-aware reminder applications**: Several variations of this application have been developed over time, each helping remind users about tasks they have to perform in relation to their location and possibly other contextual attributes (e.g. time of day, other calendar activities). Examples include a shopping list reminder application to tell users about items they have to purchase when they get within the vicinity of a store where the item can be found and an application that reminds students to pick up assignments when they have time or are near the place where the assignment is available. Some of these applications have proved more useful than others, in part due to the limited number of places where items can be purchased on campus.
- **Context-sensitive "crime" alert** application. Different areas of campus and its vicinity are more prone to incidents than others, depending on the time of day. This application was an

experiment aimed at trying to warn users as they entered areas where an incident might have been recently reported. While such an application would a priori seem useful to people who venture in areas they are not familiar with, it did not prove very popular with campus users it was tested with, in part, we believe, due to the fairly low number of incidents on and around campus.

- **Collaboration applications.** One such application enables people to selectively share powerpoint presentations subject to access control policies specified in their e-Wallets. The slides can be viewed by users on nearby projectors, using their PDAs as a remote to flip from one slide to the next (see Figure 11).

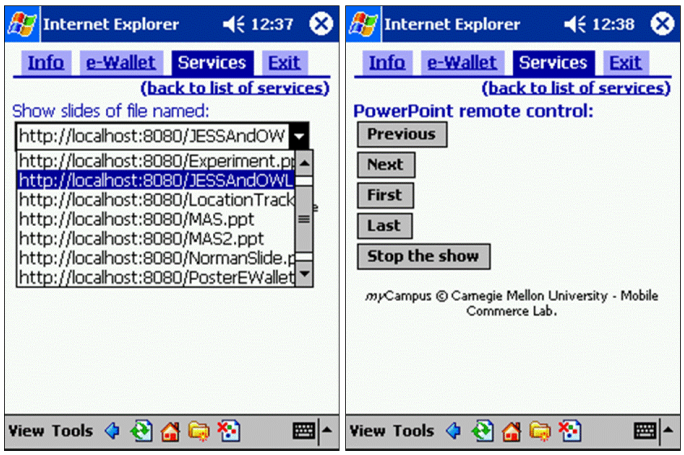


Figure 11 A remote slide controller enables users to selectively share presentations with others and allow them to use their PDAs to view the slides on nearby projectors.

- **Community applications:** Community applications have proved extremely popular among campus users and have been shown to have the potential to genuinely enhance everyday campus life. Applications we have developed range from mundane calendar scheduling applications and people locators to more sophisticated virtual, context-aware poster applications. Our calendar scheduling application and people locator application enable users to specify privacy policies, both in the form of access control preferences and obfuscation preferences. For instance, calendar scheduling users can customize their preferences to disclose different schedule details to different people (e.g. indicating they are busy to some versus disclosing the actual activity they already have scheduled versus pretending they are unavailable). Figure 12 illustrates the messages exchanged when Jim invokes the calendar scheduling application (from the "Interface" client on his PDA) for a meeting with Mary. The application accesses both Jim's and Mary's calendar information subject to privacy preferences they each specified in their e-Wallets.

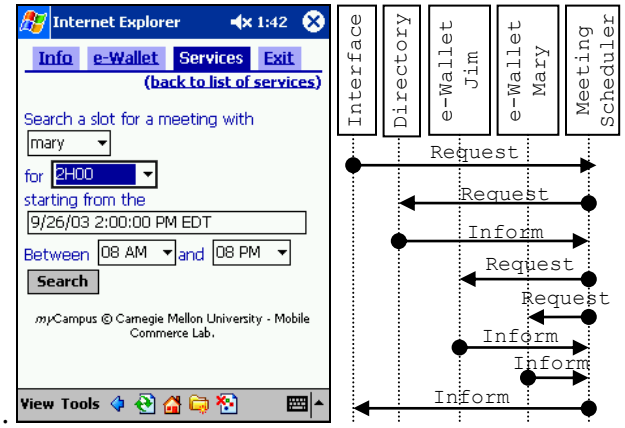


Figure 12 Scheduling meetings subject to e-Wallet-based privacy policies

Similarly, access control and obfuscation policies specified in a user's e-Wallet enable her to selectively disclose her location to different people (e.g. class mates, room mates, friends, etc.) under different conditions and with different levels of accuracy (Figure 13).

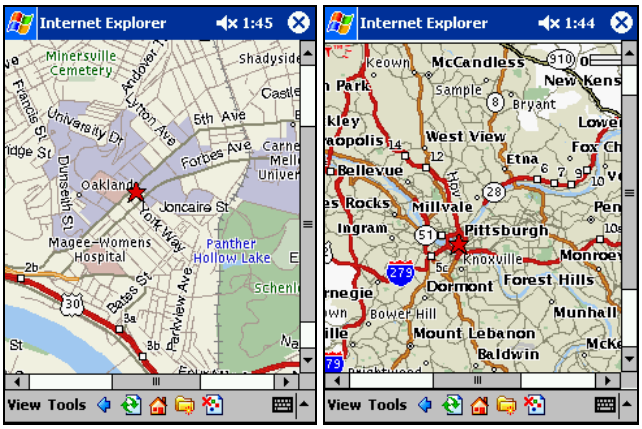


Figure 13 People locator with different obfuscation preferences: one user is willing to disclose the city block she is on while the other only discloses the city she is in.

An example of a more sophisticated community application is a **context-aware poster service** (called "InfoBridge"), enabling students to annotate, post and retrieve posters based on both interests and contextual information. This application stemmed from two observations:

- Placing posters all around campus is tedious, expensive and not particularly pleasing to the eye
- People's centers of interest can often be correlated with different contextual attributes. For instance, students majoring in different areas will typically have different daily routes through campus (Figure 14)

In InfoBridge, users can publish virtual posters that are annotated with information about the type of activity advertised (i.e. type and topic) as well as relevant contextual information. Users with an interest in the topic or whose contextual attributes match the announcement, will have it added to their collection of posters for them to check at their convenience (Figure 15).

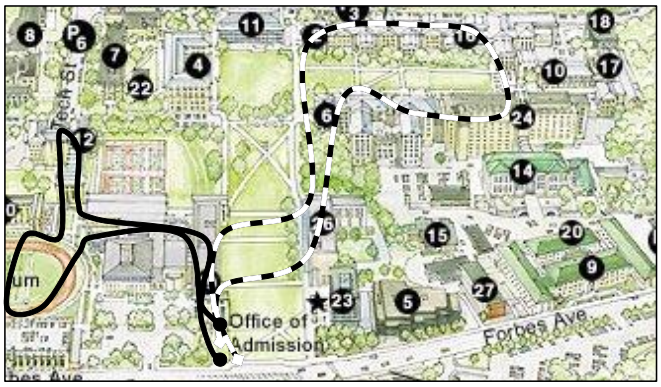


Figure 14 Typical routes of a design student and a computer science student at Carnegie Mellon.

Figure 15 Screenshots of InfoBridge

A typical InfoBridge usage scenario is summarized in Figure 16.

Jane knows that a group of students organizes a debate on free software at the cafeteria at 4PM. She is currently at the cafeteria and uses her PDA to place there a virtual poster containing information on the event. Thomas, who is an enthusiastic defender of Linux, receives the poster immediately even though he is not at the cafeteria. Later, John passes by the cafeteria. While John has not indicated interest or disinterest for the subject, he also receives the poster as he gets close to the cafeteria. His PDA rings/vibrates. He can consult the poster immediately or ignore it for the moment and look at it later along with other posters he will have collected as he moved around campus. When he later views the poster he can opt to add the event to his diary, to ignore it or specify that he does not want to receive any more posters on this topic. Once the date of the debate is passed, the poster is automatically purged.

Figure 16 Typical InfoBridge Usage Scenario

MyCampus Development Experience

When relevant contextual information has already been wrapped in the form of web services, the time it takes to build a first prototype is rather minimal (e.g. from a few hours to a few days

depending on the sophistication of the underlying logic). In situations where sources of contextual information had not been wrapped, our experience has often been that the resulting services end up being re-used across other applications. As a result, once time and resources had been invested in developing our infrastructure and in wrapping an initial set of context services, we found that most of our time was spent in identifying, designing, evaluating and refining services with groups of users. One lesson we have learned over and over again is that the time it takes to turn a seemingly promising idea into a working and usable prototype should never be underestimated. While several of our prototypes went through multiple significant refinement cycles and were eventually seen as being quite useful by users, few of them would really qualify as being truly "ready for prime time". Another key lesson from our work with members of the Carnegie Mellon campus community has been their constant concerns about privacy. While many of them see benefits in our context-aware applications, they also all express their desire to control who has access to their contextual information - as we had anticipated. They all generally see value in the flexibility and expressiveness of our Semantic eWallet technologies, though most require assistance with the eWallet settings. In Section 7, we report in more detail on a series of experiments conducted with MyCampus users at Carnegie Mellon University, looking at the benefits of context-aware Ambient Intelligence applications.

6 Empirical Evaluation

The MyCampus environment and its applications have been evaluated through a series of experiments in which users were observed and data collected over periods of several days at a time. This data was complemented by additional in-depth interviews with users to elicit additional information and develop a better understanding of how they interacted with the environment, what worked and what did not. In this section we briefly summarize results of a 3-day experiment conducted in early 2003 with a group of 11 users. The experiment involved observing how students interact with their e-Wallets to specify different sets of preferences and evaluate the benefits of two specific applications: a version of our context-aware restaurant concierge and a context-aware message filtering agent. This was complemented by an additional study to evaluate the applicability of case-based reasoning (CBR) to learning individual users' context-aware message filtering preferences.

Over a period of three days, each user's message filtering agent was used to process a total of 44 messages and the restaurant concierge was systematically used by participants in the experiment to decide where to eat, selecting from a total of 23 web services created for restaurants on or near campus. Messages involved a mix of campus-specific news (e.g. announcements of talks, meetings, social events, movies, etc.) and general news (e.g. news headlines, weather forecast, etc.). Users were allowed to specify both static and context-sensitive message filtering preferences ("a priori preferences") for different categories of messages (e.g. "messages about social events should be placed in my mailbox for later inspection", "emergency messages should be shown to me right away", "general news messages should be shown to me when I'm not busy" and "I don't care for sport-related news"). As part of the experiments, participants were later asked to review each individual message they had received and indicate what the ideal filtering action for that message should have been – "a posteriori preference" (See Figure 17). By collecting a posteriori preferences for individual messages we were able to determine how well these preferences were captured by the more limited set of options available to users when they configured their message filtering agent ("a priori preferences"). As can be seen in Figure 17, actual messages sent as part of the experiment covered a wide range of topics. A posteriori preferences for seemingly related topics are often different (e.g. "movies" and "symphony"), illustrating the complex and nuanced nature of many user preferences and the difficulty in capturing these preferences through a limited set of a priori preference options.

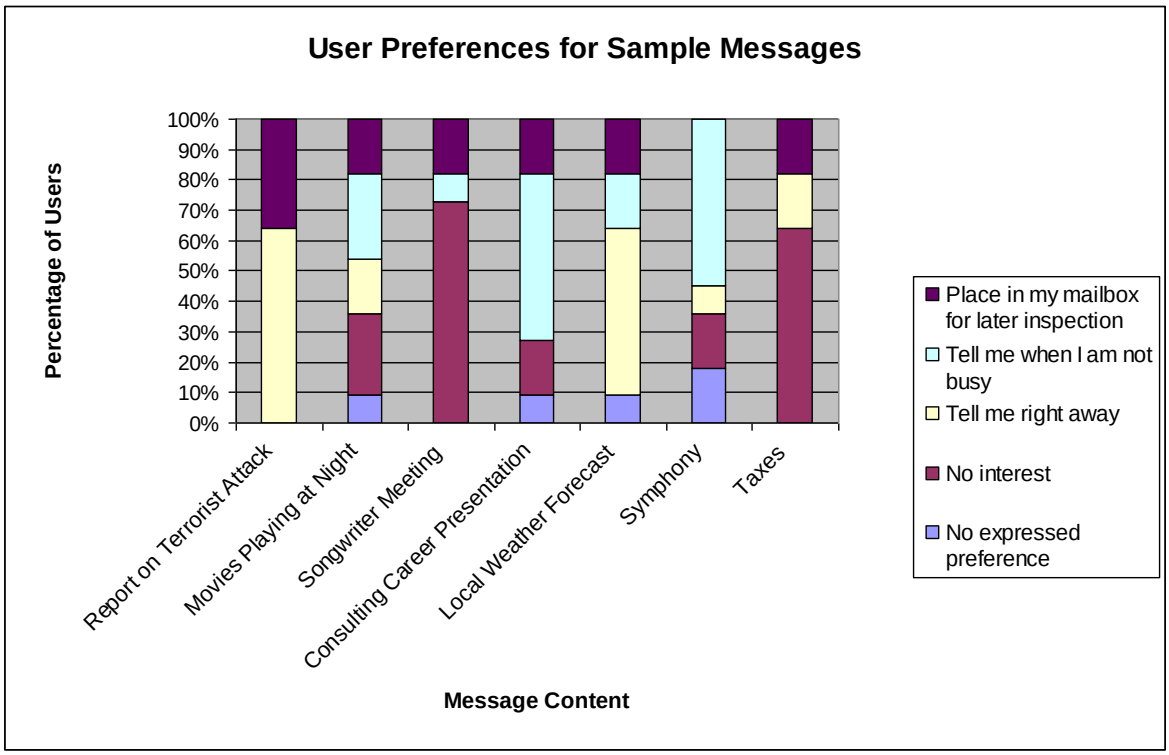


Figure 17 A posteriori message filtering preferences as collected from user feedback for a sample of 7 (out of 44) messages sent over a 3 day periods. Statistics in this figure are for a sample of 11 users.

Analysis of results collected during the experiment confirmed the difficulty of properly capturing user preferences based on a limited set of a priori preference options. Participants indicated they were only satisfied with the way in which a little over 50% of the messages they had received had been processed. At the same time, results showed that performance would have been even worse had it not been for the context-sensitive options users were allowed to specify as part of their a priori preferences. On the whole, 70 percent of the messages processed as desired had benefited from the presence of context-sensitive elements in the users' a priori preferences.

The complexity of users' actual ("a posteriori") preferences, as illustrated in Figure 17, begs the question of whether these preferences could possibly be learned over time for individual users, thereby enabling the message filtering agent to progressively improve its decisions. To test this hypothesis, a simple Case-Based Reasoning (CBR) [40, 41] module was implemented to attempt to learn more nuanced context-sensitive message filtering preferences for individual users based on their feedback. In CBR, past "cases" are used to guide decision-making in new situations. In the message filtering agent, a case corresponds to a particular message and the a posteriori preference expressed by the user for processing that message. Cases are collected based on user feedback. As new messages come in, prior cases can be retrieved and matched against the new message according to various attributes ("indices"). Closely matching cases can be used to help decide what to do with the new message. Indices used to retrieve and match cases in the implementation of our CBR module included: the type of message (e.g. meeting, class announcement, food specials, etc.), the sender of the message, the user's current calendar activity, the user's current location, as well as the weather. Cases were matched, using Aha's Nearest Neighborhood Algorithm [42], as adapted by Cercone and Chan [43]. Experiments were

conducted, using the first 33 messages received by each individual user as training cases and the remaining 11 messages as test cases. Results could only be computed for a subset of participants as contextual attributes in some user logs had been corrupted. While more extensive experiments would need to be carried out to clearly establish the potential of CBR, our results showed a significant improvement in the quality of the filtering decisions, which jumped from an average accuracy of a little over 50% (with a priori preferences) to an accuracy of over 80% with the CBR module – accuracy here is measured as the percentage of filtering decisions that exactly match the a posteriori preference indicated by the user for each message. These results suggest that context-sensitive message filtering preferences may be too complex to be correctly specified upfront and that, instead, user feedback may need to be collected over time to develop finer models. These results are illustrated in Figure 18, where we show the improvement observed with the CBR module over the version of our message filtering agent relying on a priori user preferences. The results are for the two most extreme participants in our study, one who was nearly satisfied already with a priori preferences and one whose satisfaction significantly improves with CBR.

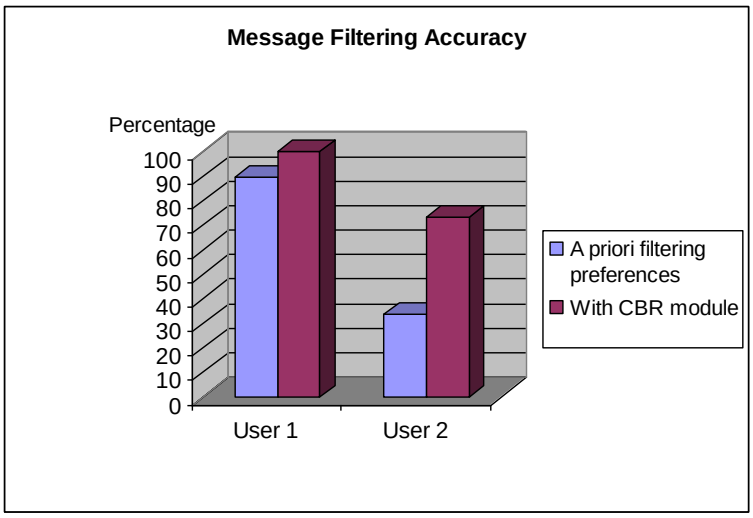


Figure 18 Comparing the accuracy of context-sensitive message filtering with a priori user preferences and with the CBR module for two extreme participants.

7 Summary and Concluding Remarks

In this article, we presented an overview of *Ambient Intelligence* work conducted over the past five years by the MyCampus group at Carnegie Mellon University. The project has drawn on multiple areas of expertise, looking at issues of usability while focusing on the development of an open Semantic Web infrastructure for context-aware and privacy-aware service provisioning. We reviewed key architectural elements of the MyCampus Semantic Web infrastructure with a special emphasis on its introduction of Semantic e-Wallets that act as both clearinghouses and gatekeepers to a user's personal information. Another important element of this infrastructure is the way in which sources of contextual information are modeled as Semantic Web services. These services can automatically be identified and accessed to supplement an e-Wallet's local knowledge about its user. Our experience tailoring this architecture to different environments (everyday campus life applications, office applications, and a museum tour guide) has shown that the reusability of sources of contextual information modeled as Semantic Web Services can substantially reduce the time it takes to develop and refine new context-aware applications. This is particularly helpful in environments, where the objective is to develop a growing collection of context-aware applications over time – as has been the case with our work on Carnegie Mellon's

campus. Despite these benefits and the positive feedback obtained from a number of users on the value of Ambient Intelligence applications we developed, our work with users has also shown that it typically takes numerous iterations before an application can be considered ready for prime time.

Our work with users at Carnegie Mellon University has also confirmed our initial intuition that privacy issues are central to user acceptance. It has shown that user's privacy preferences are often complex and nuanced. Capturing these preferences as well as other relevant context-sensitive preferences remains a major impediment to the broad acceptance of Ambient Intelligence technologies. Our initial work using Case-Based Reasoning suggests that some of these preferences can be learned over time, based on user feedback, thereby alleviating the initial burden that would be placed on users if they had to specify all their preferences upfront. This is an area we are continuing to research.

8 Acknowledgements

The work reported herein has been supported in part under DARPA contracts F30602-02-2-0035 ("DAML initiative") and F30602-98-2-0135. Additional support has been provided by IBM, HP, Symbol, Boeing, Amazon, Fujitsu, the EU IST Program (SWAP project), and the ROC's Institute for Information Industry. Over the years, the MyCampus project has benefited from the contributions of a number of participants. Besides the authors, past and present members of the project include: Jinghai Rao, Enoch Chan, Linh Van, Hirohiko Yamamoto, Kazuaki Takizawa, Yoshinori Shimazaki, Rahul Culas, Huntington Howe, Paul Ip, Ruth Lee, Mithun Sheshagiri, Shih-Chun Chou, Wen-Tai Hsieh, Matt Chang, Andrew Li, Polly Ng, Sumat Chopra, Srini Utpala, and Wilson Lau. The US Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

9 Additional Sources of Information

Additional information, including a video on the MyCampus project, can be found at http://www-2.cs.cmu.edu/~sadeh/mycampus.htm. Some of the code developed as part of the project has also been released on SemWebCentral (see http://projects.semwebcentral.org/projects/rowl/). This includes:

- ROWL – Rule Language in OWL and translation engine for Jess
- A standalone version of our Semantic e-Wallet, including a development environment.

10 References

[1] OASIS: UDDI: Executive Overview: Enabling Service Oriented Architecture, http://uddi.org/pubs/uddi-exec-wp.pdf, 2004

[2] W3C: Web Services Description Language (WSDL) 1.1, Note 15 March 2001 http://www.w3.org/TR/wsdl

[3] Want, R., Hopper, A., Falcao, V., Gibbons, J.: The Active Badge Location System, *ACM Transactions on Information Systems* 10(1) (1992) 91-102.

[4] Schilit, W.: A System Architecture for Context-Aware Mobile Computing, Ph.D. Thesis, Columbia University, 1995.

[5] Schilit, B., Adams, N., Want, R.: Context-Aware Computing Applications. *Proc. of the Workshop on Mobile Computing Systems and Applications*, IEEE Computer Society, Santa Cruz, CA, (1994) 85-90

[6] Dertouzos, M.: The Future of Computing, *Scientific American*, August (1999)

[7] Garlan, D., Siewiorek, D., Smailagic, A., Steenkiste P.: Project Aura: Towards Distraction-Free Pervasive Computing, *IEEE Pervasive Computing*, Special Issue on Integrated Pervasive Computing Environments, Vol. 1, Number 2, April-June (2002) 22-31 Sardinia, Italy, June 2002.

[8] Hong, J., Landay, J.: A Context/Communication Information Agent, *in Personal and Ubiquitous Computing*, Special Issue Situated Interaction and Context-Aware Computing, Vol. 5(1) (2001) 78-81

[9] Hong, J., *An Architecture for Privacy-Sensitive Ubiquitous Computing*, Unpublished PhD dissertation, University of California at Berkeley, Berkeley, CA, 2004.

[10] Dey, A.K., Abowd, G.D.: Toward a Better Understanding of Context and Context-Awareness, *GVU Technical Report* GIT-GVU-99-22. College of Computing, Georgia Institute of Technology, (1999)

[11] Dey, A., Salber, D., Futakawa, M., Abowd, G.: An Architecture to Support Context Aware Computing, GVU *Technical Report* GIT-GVU-99-23. College Computing, Georgia Institute of Technology, Nov. (2000)

[12] W3C: The Platform for Privacy Preferences 1.0 (P3P1.0) Specification, Recommendation 16 April 2002, http://www.w3.org/TR/P3P/

[13] OASIS: Security Assertion Markup Language (SAML), Technology Reports, April 14 (2003) http://xml.coverpages.org/saml.html

[14] OASIS: Extensible Access Control Markup Language (XACML), Technology Reports, March 28 (2003) http://xml.coverpages.org/xacml.html

[15] Schunter, M., Powers, C., The Enterprise Privacy Authorization Language (EPAL 1.1), IBM Research Laboratory, http://www.zurich.ibm.com/security/enterprise-privacy/epal/

[16] Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web, *Scientific American*, May (2001)

[17] Hendler, J.: Agents on the Web, *IEEE Intelligent Systems*, Special Issue on the Semantic Web, Vol. 16, No. 2, pp. 30-37, March/April, (2001)

[18] DAML Joint Committee: DAML+OIL language, 27 March 2001, http://www.daml.org/2001/03/daml+oil-index.html

[19] W3C: OWL Web Ontology Language Reference, Working Draft 31 March 2003, http://www.w3.org/TR/owl-ref/

[20] DAML Services Coalition: DAML-S: Web Service Description for the Semantic Web, First International Semantic Web Conference, ISWC'02, Sardinia, Italy, LNCS 2342, (2002) 348-363

[21] DAML Services Coalition, "OWL-S: Semantic Markup for Web Services", http://www.daml.org/services/owl-s/1.1/overview/, 2004.

[22] Feier, Cristina and Domingue, John. WSMO Primer, WSMO, April 2005. Available at http://www.wsmo.org/TR/d3/d3.1/v0.1/#S44

[23] M. Ackerman. “Privacy in Pervasive Environments: Next Generation Labeling Protocols”. *Pervasive and Ubiquitous Computing* , Vol 8, 430-439, 2004 http://dx.doi.org/10.1007/s00779-004-0305-8

[24] U. Hengartner, and P, Steenkiste. Implementing access control to people location information. In *9th ACM Symposium on Access Control Models and Technologies (SACMAT'04)*, Yorktown Heights, June 2004.

[25] L. Kagal, T. Finin, and A. Joshi. A policy language for a pervasive computing environment. In *Collection of IEEE 4th International Workshop on Policies for Distributed Systems and Networks*, June 2003.

[26] M. Langheinrich. A privacy awareness system for ubiquitous computing environments. In *Proceedings of Ubicomp 2002*, pages 237-245, 2002.

[27] S. Lederer, J. I. Hong, X. Jiang, A. K. Dey, J. A. Landay, and J. Mankoff. Towards everyday privacy for ubiquitous computing. Technical Report UCB-CSD-03-1283, *Computer Science Division, University of California*, Berkeley, October 20, 2003. http://www.cs.berkeley.edu/projects/io/publications/privacy-techreport03a.pdf

[28] Norman M. Sadeh, Ting-Chak Chan, Linh Van, OhByung Kwon and Kazuaki Takizawa. "Creating an Open Agent Environment for Context-aware M-Commerce", in "*Agentcities: Challenges in Open Agent Environments"*, Ed. by Burg, Dale, Finin, Nakashima, Padgham, Sierra, and Willmott, LNAI, Springer Verlag, pp.152-158, 2003

[29] F. Gandon, and N. Sadeh. “Semantic Web Technologies to Reconcile Privacy and Context Awareness”. Web Semantics Journal, 1(3), 2004.

[30] N.Miller, G. Judd, U. Hengartner, F. Gandon, P. Steenkiste, I-H Meng, M-W Feng and N. Sadeh, Context-Aware Computing Using a Shared Contextual Information Service**,** *Pervasive 2004*, "Hot Spots", Vienna, April 2004.

[31] FIPA, Specifications (2002) http://www.fipa.org/repository/fipa2000.html

[32] M. Sheshagiri, N. Sadeh and F. Gandon, “Using Semantic Web Services for Context-Aware Mobile Applications” *MobiSys 2004 Workshop on Context Awareness*, Boston, June 2004.

[33] S-C Chou, W-T Hsieh, F. Gandon and N. Sadeh, “Semantic Web Technologies for Context-Aware Museum Tour Guide Applications”, *International Workshop on Web and Mobile Information Systems (WAMIS’05)*, IEEE Computer Society, 2005.

[34] F. Gandon, and N. Sadeh. A semantic e-wallet to reconcile privacy and context awareness. *In Proceedings of the Second International Semantic Web Conference (ISWC03)*, Florida, October 2003.

[35] F. Gandon, and N. Sadeh. ROWL: Rule language in OWL and translation engine for JESS. *Mobile Commerce Laboratory*, Carnegie Mellon University, 2004. http://mycampus.sadehlab.cs.cmu.edu/public_pages/ROWL/ROWL.html

[36] Friedman-Hill, E.: Jess in Action: Java Rule-based Systems, *Manning Publications Company*, June 2003, ISBN 1930110898, http://herzberg.ca.sandia.gov/jess/

[37] W3C: RDF Vocabulary Description Language 1.0: RDF Schema, Working Draft 23 January (2003) http://www.w3.org/TR/rdf-schema/

[38] W3C: XSL Transformations (XSLT) Version 1.0, Recommendation 16 November 1999, http://www.w3.org/TR/xslt

[39] Bellifemine, F., Caire, G. Poggi, A. and Rimassa, G. "JADE : A While Paper", in EXP magazine, Telecom Italia, Vol. 3, No. 3, September 2003. Available at: http://exp.telecomitalialab.com/upload/articoli/V03N03Art01.pdf

[40] J. Kolodner. *Case-Based Reasoning*. Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[41] David Leake. *Case-based reasoning – Experiences, lessons and future directions*. AAAI Press/The MIT Press, 1996.

[42] Aha, D. W., Kibler, D., & Albert, M. K. . "Instance-based Learning Algorithms". *Machine Learning*, Vol. *6*, 37-66, 1991.

[43] Cercone, N., An, A., and Chan C., "Rule-Induction and Case-based reasoning: hybrid architectures appear advantageous," *IEEE Transactions on Knowledge and Data Engineering*, 11 (1), 1999, pp. 166-174.