An Agent-Based Framework for Secure and Privacy-Preserving Personalized Information Services

Richard Cissée Andreas Rieger Nicolas Braun Sahin Albayrak[*]

Abstract

Providers of next-generation ubiquitous services are facing increasing competition. In order to attract and satisfy customers, these services must offer added value e.g. by delivering personalized information in a seamless and multi-modal way. Personalized services, however, bring about additional requirements related to privacy and security, which have to be addressed if the services are to become widely accepted. The ability to design, implement and deploy personalized information services in a secure, privacy-friendly and at the same time highly efficient way is becoming a key success factor for service providers. Nevertheless, current service development architectures usually fail to cover all relevant aspects of the service development process, resulting in an unnecessary development overhead on the side of the service provider.

We introduce an agent-based Serviceware Framework assisting service providers in developing personalized information services, thus improving user acceptance and reducing the time-to-market of the resulting applications. We describe the utilization of different modules of the framework, which offer functionality for context-aware services at large, focusing on the module for personalization including privacy enhancing technologies. In addition, we present the Smart Event Assistant, a prototypical application for personalized, seamless and ubiquitous planning of entertainment activities, which we have implemented based on this Serviceware Framework.

Keywords*:* *personalization, privacy, security, multi-agent systems, context-awareness.*

1 Introduction

Providers of next-generation ubiquitous services are facing increasing competition. In order to attract and satisfy customers, these services must offer added value. Context-aware services fulfill these requirements because they integrate functionality related to three main aspects of mobile services: Provision of personalized information, ubiquitous and device-independent access, and location-based services functionality. While there are a number of solutions for device-independent and location-based services, many existing personalized information services have failed to meet the requirements of service users, resulting in a lack of acceptance of the respective services in general. The main reason for this lies in the inherent conflict between the primary goal of the service user (the goal to receive personal recommendations based on individual preferences) and additional requirements related to privacy and security aspects (the necessity to protect sensitive personal information against various threats).

[*] DAI-Labor, Technische Universität Berlin; Franklinstrasse 28/29, D-10587 Berlin, Germany; {richard.cissee, andreas.rieger, nicolas.braun, sahin.albyrak}@dai-labor.de

Providers of context-aware services have to cover these aspects, and at the same time need to develop services in a highly efficient way. In this paper, we introduce an agent-based Serviceware Framework assisting service providers in developing context-aware services in general, and personalized information services in particular. Based on the framework, we have developed a prototypical application, the Smart Event Assistant, providing several context-aware services related to the planning of entertainment activities in various cities. The implemented personalized information services meet all typical requirements related to personalization, privacy and security.
The paper is structured as follows: In the next section, we provide a problem description illustrated by a typical scenario taken from the domain of personalized information services. Section 3 introduces our approach of a framework for context-aware services. Section 4 focuses on the personalization module and its filtering techniques. Section 5 describes our solutions related to privacy and security aspects. Section 6 outlines the implementation of the framework and describes the Smart Event Assistant, an application based on the framework. Section 7 discusses related work. Section 8 concludes the paper with an outlook and possible further work.

2 Problem Description

Personalized information services offer added value to the user, but they bring about additional requirements related to privacy and security. We illustrate this problem by describing a scenario for a typical personalized information service based on an application, which supports users in planning entertainment activities in a certain area.

In our scenario, a Columbian tourist is visiting Berlin. Because he has never visited the German capital before, the tourist is not familiar with local entertainment activities. Because of his tight schedule, he does not want to familiarize himself with search engines and other standard information sources (such as newspapers, TV, or radio). Instead, the user prefers to receive personalized recommendations for entertainment activities, based on preferences and information he has already provided, e.g. for different cities he has visited before. He uses an application, which helps him to plan his activities by providing personalized recommendations.

Having arrived and checked in at his hotel, the user plans his evening entertainment activities by using the application via an HTML-based browser on a WiFi-enabled notebook. As he is visiting Berlin on his own, he is interested in meeting other users with similar interests. Having proceeded with the input, the system recommends a restaurant and a following theater visit, two activities matching the tourist's preferences. He therefore accepts the restaurant and theater recommendations, thus confirming the tentative reservations that have been automatically generated by the application. According to his schedule, the Columbian tourist leaves the hotel and uses the application's integrated routing information system via his third generation cellular phone to find his way to the first venue.

This scenario raises various privacy and security issues: The system requires private user information in order to create personalized recommendations and to determine users with similar interests. From the user's point of view, this information should never be used for additional purposes nor be passed on to third parties. In addition, private user information must be protected against unauthorized external access or attacks, e.g. during the communication between the user and the system as well as between different system components. Due to the ubiquitous character of the application, the user information cannot be stored on the user's device itself, but has to be accessible online, increasing the required amount of protection.

3 Framework for Context-Aware Services

We have developed a framework for context-aware services integrating functionality for personalized information services with functionality for device independence and location-based services. Additional management functionality is contained within a separate infrastructure module.

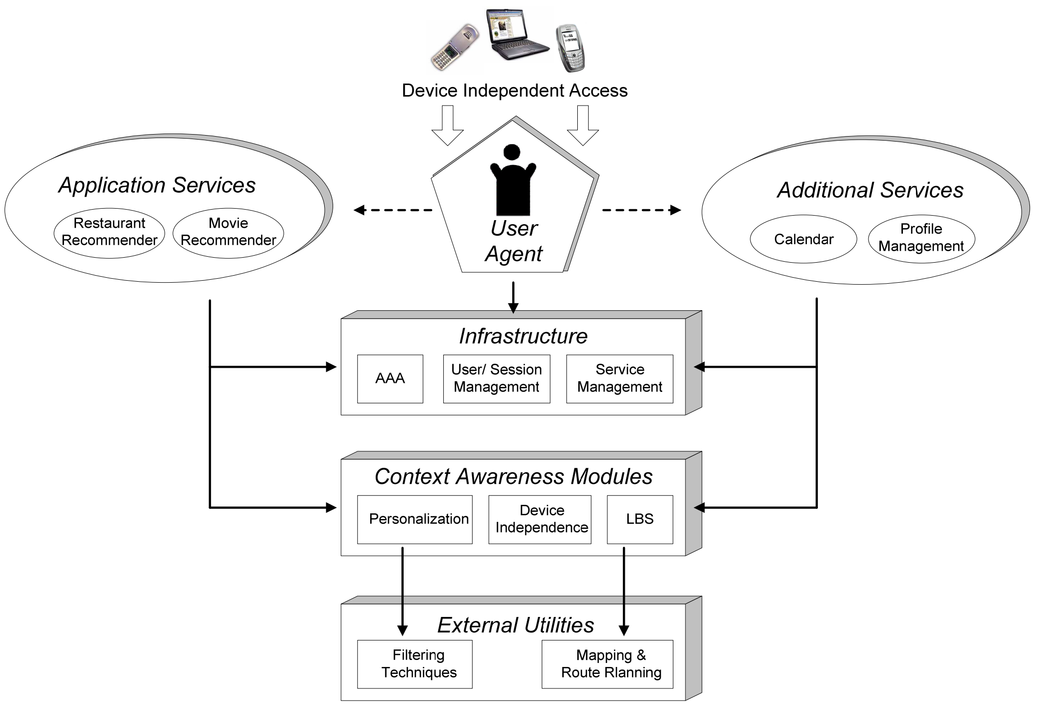


Fig. 1. The Framework for Context-Aware Services

Typically recurring functionality such as services for user profile management, a calendar for managing appointments etc. are provided as well. Fig. 1 gives an overview of the entire framework.

3.1 Framework Architecture

Our approach of a Serviceware Framework for context-aware services is based on Multi-Agent System (MAS) technology. For an introduction to MAS technology, we refer to [1]. Basically, MAS architectures consist of agents encapsulating specific functionality and offering services to exchange data with other agents. All agents exist in specific environments, the agent platforms. Agent interaction is based on ontologies, which define a common vocabulary. Agents offer services with specific effects usable by other agents, thus providing a well-defined interface. Dedicated agents fulfill infrastructure tasks, such as providing white and yellow pages services for agent and service discovery. Finally, agents are characterized by a certain degree of autonomy.

We have decided to use MAS technology because it fulfills many requirements of frameworks for context-aware services. An overview of these requirements, the respective solution provided by MAS technology, and the relevance for personalized information services in particular is given in Table 1. There are other approaches for distributed systems addressing these requirements in a similar manner, some of which are discussed in Section 7. The advantage of MAS technology lies

Table 1. Requirements of frameworks for context-aware services.

Requirement	Multi-Agent System Technology-Based Solution	Relevance for Personalized Information Services
Modularity: Typically, not all framework functionality is required in any given scenario. It should be possible to use only those parts of the framework that are actually required.	MAS-based applications are mainly configured by selecting and defining the participating agents. Therefore different modules made up by groups of agents may be changed easily.	In order to develop and deploy personalized information services, only the respective modules of the framework will have to be used.
Scalability: The framework should be usable for small, non-public systems as well as for applications with a large target audience.	Scalability is mainly achieved by duplicating the agents responsible for critical tasks, thus distributing the load between multiple identical agents and removing bottlenecks.	Many personalized services rely on information about other users, and thus require a large amount of users.
Adaptability: The functionality provided by the framework should not be static, or it would be outdated soon. Therefore, it should be possible to add, remove, or replace parts of the framework without requiring changes on the framework user's side.	MAS-based applications may be reconfigured even at runtime, i.e. agents may be added or removed to adapt the functionality provided. The newly offered services may be used immediately.	Probable reconfigurations would be to add agents providing e.g. knowledge-based (domain-specific) filtering techniques.
Distribution: It should be possible to distribute the framework e.g. in order to enable load balancing, or to increase the overall security.	Mobile agents have the ability to migrate between platforms, which may be located on different servers, e.g. for reasons of load-balancing.	A large amount of computation depends on the actual users at any given time and therefore is rather unpredictable.
Security: The framework should contain various security features to prevent unauthorized use of its functionality and resources, and to protect systems based on the framework against attacks.	MAS architectures include security features preventing unauthorized service usage and prohibiting agents from attacking other agents or platforms.	Sensitive personal information is especially susceptible to external threats. This is discussed in more detail in Section 5.1.
Privacy: In order to enhance the acceptance and trustworthiness of the respective systems from the intended users' point of view, privacy-enhancing technologies should be built into the framework.	Privacy aspects regarding personal information are addressed by encapsulating sensitive information within dedicated personal user agents.	Privacy is the essential requirement for personalized services. This is discussed in more detail in Section 5.2.

in its network independence and versatility. Additionally, certain aspects of our framework architecture (see Section 5.1.) require autonomous, communicating entities and may therefore ideally be realized via MAS technology.

3.2 Framework Modules

The framework for context-aware services consists of three main modules related to the main aspects of context awareness, i.e. personalization, device independence, and location-based services. Furthermore, an infrastructure module is provided covering different management tasks. The services provided by the agents in each module represent the interface of the respective module.

3.2.1 Personalization Module

The personalization module contains all functionality required for information filtering processes and is described in detail in Section 4.

3.2.2 Module for Device Independence

The device independence module provides functionality for generating user interfaces for different devices, such as cellular phones, PDAs or laptops, without having to change the underlying functionality of the respective services. This is mainly achieved by using the Multi-Access Service Platform (MASP) [2], which separates device-dependent layout aspects from general user interaction aspects for each dialog. Layout aspects are specified in an abstract way by the service developer and mapped to a specific layout optimized for the current device display. Moreover, the MASP offers multi-modal, multi-lingual and multi-media-based user interfaces, thus providing the most appropriate dialog based on the user's situation. Finally, the MASP includes an intelligent session management. The user may switch modes, devices and networks freely, and is therefore able to use the respective information service seamlessly and comfortably. For a detailed description of this module, we refer to [3]. In the context of personalization, the module for device independence is responsible for providing customized user interfaces, based on the user's preferred language and mode of interaction. Thus, accessibility and internationalization issues are covered.

3.2.3 Module for Location-Based Services

The module for Location-Based Services supports the localization of users, and additionally supplies ontologies for processing location information. It provides functionality for mapping and routing between different locations, and suggests specific Points of Interest (e.g. tourist attractions or pharmacies) within a given region.

3.2.4 Infrastructure Module

The infrastructure module supports, among other aspects, the management of users, sessions, and services. It provides AAA functionality for authentication, authorization and accounting.

4 The Personalization Module

The personalization module contains all functionality required for information filtering processes. Basically, an information filtering process generates personalized recommendations for a user, based on his personal user profile and an information service provider profile, i.e. all potential recommendations, by applying a specific filtering technique. Each user profile is stored and controlled by a dedicated user agent, while the provider profile, which is typically much larger, is stored and controlled by the service provider, usually in a database management system. There are filtering techniques for distributed information filtering (an example realized in our framework is Distributed Collaborative Filtering, see Section 4.4.1), in which the provider profile actually represents the profile of another user. The structure of the overall information filtering process, however, is similar to the regular case.

The personalization module consists of three submodules, which are responsible for information collection, information processing and information filtering, as shown in Fig. 2. While these submodules do not interact directly, they are responsible for specific tasks related to operations on the user profile. Subsequently, they are typically used by different services, but it is also feasible for a single integrating service to utilize all three submodules. Finally, it should be noted that the submodules for information processing and information filtering contain adaptable and complementary functionality, i.e. the specific filtering techniques.

4.1 Information Collection Submodule

The Information Collection submodule collects information offered by an information service provider, i.e. the information based on which personalized recommendations are generated and propagated to users. Additionally, it collects personal information about the user for whom recommendations are to be generated. While obtaining the service provider information is usually a straightforward task, because in most cases it is already available in an organized structure, such as a database, obtaining information about the user's preferences is actually rather difficult in many

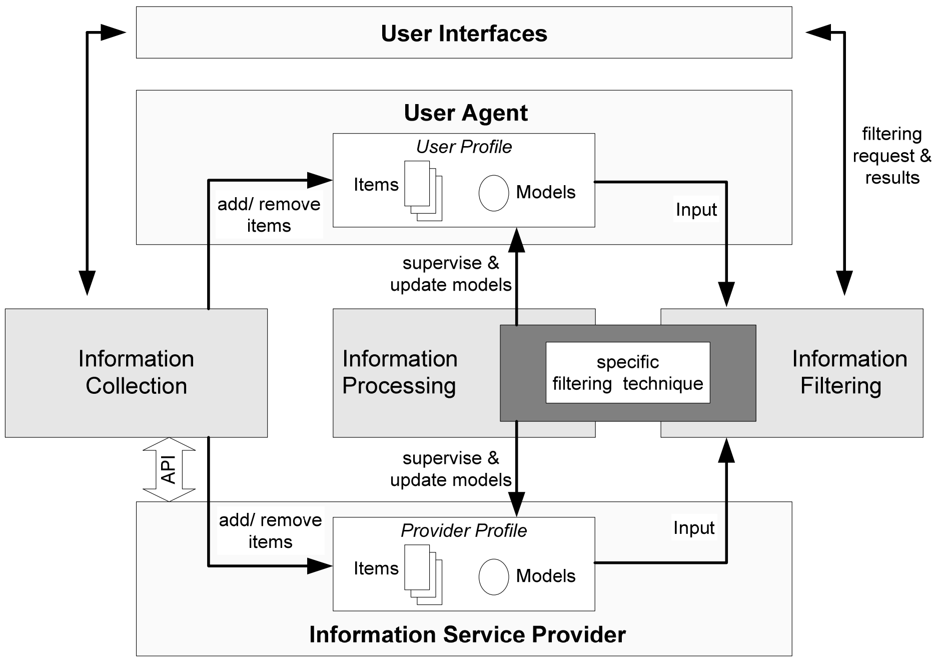


Fig. 2. Architecture of the personalization submodule and related framework elements

cases because of various reasons: The user may not be willing to provide the information, simply because it is too inconvenient; the user may not be able to provide the information because he cannot describe his preferences adequately; or the user provides, perhaps unintentionally, incorrect information.

For these reasons, the information collection submodule supports different methods for obtaining feedback from the user. In the most basic case, explicit feedback is provided by the user (e.g. the rating for a specific object or a list containing general preferences) and simply collected. Furthermore, this submodule supports services in collecting implicit user feedback by monitoring the user's activities (e.g. the time spent viewing specific items, actions performed in specific dialogs, etc.). While explicit feedback expresses the user's preferences more exactly, resulting in more accurate recommendations, users in general have been found to accept slightly less accurate recommendations as a tradeoff for not having to provide explicit feedback. In all cases, the gathered information is propagated to the respective user agent and directly stored as a part of the user profile. Additionally, the user may view and update his personal profile directly via the information collection submodule, e.g. in order to remove outdated items or simply to obtain an overview of all information that has been collected.

4.2 Information Processing Submodule

The information processing submodule converts the information stored in both the user and the provider profile, aggregating single items into models used for information filtering. The structure of a model depends on the filtering technique it is used for. Models may, e.g. represent neural networks, decision trees, or results of machine learning algorithms. While the information stored in profiles via the information collection submodule may be used directly by simple filtering techniques, it has to be processed further for other, more advanced filtering techniques. Some

Table 2. Exemplary filtering techniques and the required profile structures. Examples are based on a restaurant recommender system.

Filtering Technique	User Profile Items	User Profile Models	Provider Profile Items	Provider Profile Models
Direct Feature-Based Filtering	general preferences (e.g. preferred cuisine, price range etc.) & single restaurant items	single restaurant items	single restaurant items	*(none required)*
Clustered Feature-Based Filtering	single restaurant items	*(none required)*	single restaurant items	Hierarchical clusters of similar items
Decision Tree-Based Filtering	single restaurant items	general preferences as decision tree	single restaurant items	*(none required)*
Distributed Collaborative Filtering	single restaurant items	items weighted e.g. by age (to account for changes in user's preferences)	single restaurant items	items weighted e.g. by age

exemplary filtering techniques, all of which are included in the realized framework, are listed in Table 2. The filtering techniques themselves are described in the following section. In any case, the item models have to be created and updated in accordance with the filtering technique using them afterwards. Therefore, the agent providing filtering technique itself is responsible for offering specific information processing functionality as well.

4.3 Information Filtering Submodule

The information filtering submodule is responsible for creating recommendations based on the information stored in the user and provider profile, by applying a specific filtering technique. The choice of filtering technique mainly depends on the structure of the information stored in the profiles, especially if it has been processed and aggregated into a specific model, on the character of recommendations the user wants to receive (as an example, he may be interested in items similar to the items collected in his profile, or he may explicitly want to receive dissimilar yet still relevant items), and on privacy considerations, as will be discussed in Section 5.

The filtering techniques applicable within our framework may be classified by the following three main categories. For a further discussion of filtering techniques and their categories, we refer to [4].

4.3.1 Feature-Based Filtering

Feature-based filtering techniques (also known as content-based filtering) determine recommendations by comparing the features or attributes of items. In the most basic case, items are compared directly, and the provider profile items most similar to the user profile items are returned as recommendations. For large profiles, however, this approach is rather inefficient.

For this reason, feature-based filtering techniques are generally based on models of items in order to speed up the process of generating recommendations. Feature-based filtering techniques are applicable whenever the respective items may be described adequately and uniformly by distinct attributes. They are not applicable in cases where recommendations have to be generated from heterogeneous items characterized by only a few or no common attributes, or from items without any well-definable attributes.

4.3.2 Collaborative Filtering

Collaborative filtering techniques determine recommendations indirectly by comparing entire user profiles in order to detect users with similar preferences. The recommendations comprise the users themselves, if the user is mainly interested in contacting other users with matching interests, or they are taken from the profiles of similar users. While this approach facilitates serendipitous recommendations, and is applicable whenever user profiles at least partially overlap, it has several drawbacks mainly related to the fact that it is often difficult to reliably determine similar users.. Hence, collaborative filtering techniques are often used in combination with other approaches, resulting in hybrid techniques [5].

4.3.3 Knowledge-Based Filtering

Knowledge-based filtering techniques determine recommendations by applying domain-specific knowledge to profile items in order to create extensive models. As an example, the similarity of items may not be determined implicitly, as in the learning-based approaches described above, but stated explicitly instead. While the quality of recommendations is potentially highest for knowledge-based filtering techniques, they require additional effort for modeling domain-specific knowledge, and are therefore often impractical. Examples for knowledge based filtering techniques are given in [6].

4.4 Realized Filtering Techniques

In our framework for context-aware services, we have realized several filtering techniques, as described in the following subsections and listed in Table 2. The architecture allows an easy inclusion of additional filtering techniques by providing a standardized interface.

4.4.1 Direct Feature-Based Filtering

Direct feature-based filtering is included as the most fundamental filtering technique. It compares all user profile items with all provider profile items, returning as recommendations the most similar items not already contained in the user profile. It may be directly applied on the items themselves i.e. item models are not required. However, it is also possible to derive a user profile item model containing prototypical items based on the user's general preferences, and subsequently use these as additional items.

4.4.2 Clustered Feature-Based Filtering

Clustered feature-based filtering organizes the provider profile items into clusters of similar items, where each cluster is represented by one designated item indicating the center of the cluster. User profile items are then compared with these designated items in order to determine relevant clusters, from which the recommendations are taken. For large provider profiles, a hierarchical cluster structure is used, in which the cluster elements on each but the lowest level are sub-clusters and not the items themselves.

4.4.3 Decision Tree-Based Filtering

In decision tree-based filtering, which is another type of a feature-based filtering technique, a decision tree is created by machine learning algorithms based on the user profile items. This decision tree is mapped to a collection of queries on the provider profile data structures afterwards. Recommendations are generated by executing the queries.

4.4.4 Distributed Collaborative Filtering

Finally, a distributed collaborative filtering technique is included in order to provide recommendations representing users with similar preferences. Potentially similar users are determined based on single matching elements in the according user profiles, and the similarity of user profiles is then determined by comparing the user profile items, which additionally may be weighted, e.g. in order to indicate changes in a user's preference.

5 Privacy & Security Aspects

Our framework for context-aware services especially aims at addressing privacy and security aspects with a special focus on user information. Surveys [7] indicate that users are generally rather reluctant in providing a large amount of personal information, because there usually is no way for them to control the further dissemination of personal information. Existing mechanisms, such as privacy policies indicating the intended use of personal information by service providers, have turned out to be insufficient, because they are not legally binding and often not adhered to[*]. Hence, additional privacy enhancing technologies are required. Moreover, personal data, and data collected in general, has to be protected against security threats, such as external attacks.

5.1 Privacy-Preserving Information Filtering

Existing recommender system architectures are characterized by an incompatibility of personalization and privacy: In order to receive personalized recommendations, the user has to provide (and subsequently give up control of) personal information. We introduce an approach for Privacy-Preserving Information Filtering (PPIF) that overcomes this dilemma. Basically, this is achieved by granting the user exclusive control over his personal information, which is encapsulated within a dedicated user agent, as seen in Fig. 2. Whenever the personal information has to be accessed by other roles, i.e. for information processing and information filtering, the respective agents, who are potentially untrustworthy, are supervised, i.e. controlled with regard to their communication capabilities. Thus, it is ensured that personal information is not propagated and used for unintended purposes. The concept of PPIF is described in detail in [9].

5.1.1 The Basic Filtering Process

To illustrate the concept of supervising agents, we outline an exemplary filtering process. The User Role represents the user agent, offering a service via which a human user may initiate the information filtering process. The Provider Role represents the service provider, whereas the Filter Role represents the information filtering submodule. Fig. 3 shows the communication flow between the roles in the PPIF architecture, the dashed lines indicating

[*] As a recent example, several millions of customer records have been provided by an US airline to a third party in explicit violation of its publicly stated privacy policy [8].

the platforms, which are under the control of supervising agents. The numbers in parentheses in the following paragraph indicate the succession of the actions, which is shown in the figure as well.

A filtering process is initiated by the user role by instantiating a user role supervisor (URS) agent on any platform (1) and sending a request for recommendations to the provider role (2). The provider role instantiates a provider role supervisor (PRS) agent on any platform (3) and requests the filter role to provide a temporary filter instance (TFI) agent (4). After the TFI agent is instantiated (5), it queries the user role via the URS agent

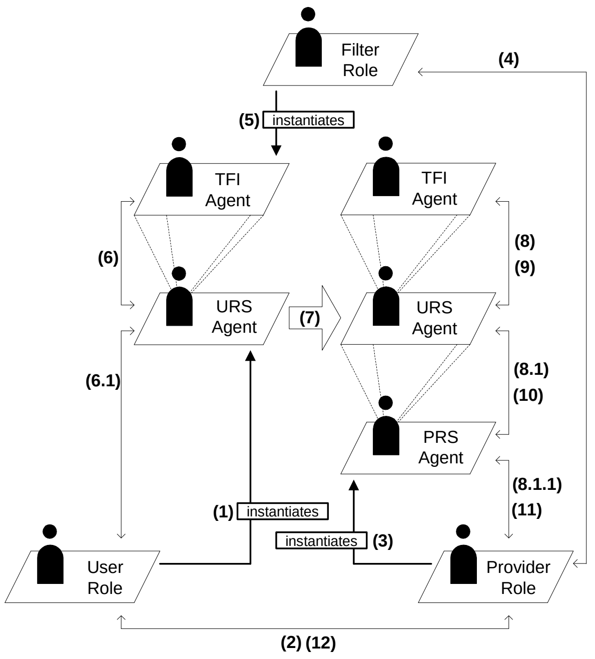


Fig. 3. An exemplary privacy-preserving information filtering process

(6 and 6.1) for the user profile. When the TFI agent has received all necessary information, the URS agent, including the TFI agent, migrates to the platform controlled by the PRS agent (7). After the migration has been completed, the TFI agent queries the provider role via both supervising agents (8, 8.1, and 8.1.1) for the provider profile. Having determined the recommendations, the TFI agent returns them to the provider role, again via the supervising agents (9, 10, and 11), which in turn returns the recommendations to the user role (12).

In this architecture, the supervising agents ensure that no critical part of the respective profile is communicated. The TFI, for example, cannot propagate user profile information at all, and the provider profile in turn is protected by the PRS agent.

5.1.2 Distributed Collaborative Filtering

In Distributed Collaborative Filtering user profiles are compared among themselves, instead of comparing a user profile with a provider profile. The basic PPIF architecture, as described above, may be applied in this case as well, simply by substituting the provider role with a second user role.

If the collaborative filtering is applied to determine similar users, the TFI returns a similarity measurement instead of or in addition to the recommendations themselves.

Another aspect remains to be addressed: Collaborative filtering should be based on a large number of users, because it is usually difficult to find sufficiently similar users in a small set of users. It is infeasible, however, to carry out the filtering process between all pairs of users. Hence, a subset of all pairs of users containing only potentially promising candidates has to be determined. In our architecture, this is done by linking each profile element to a list of users, who have added this element to their profiles. Whenever a user adds a new element to his profile, he may obtain the respective list and may contact potentially similar users.

Obviously, these lists constitute a privacy risk because it would be possible to reconstruct an entire user profile by aggregating all available information. Therefore, the list entries are pseudonymized and a mechanism to communicate with a user role known under different pseudonyms is included in the architecture.

5.1.3 Suitable Filtering Techniques

Not all filtering techniques are suitable for PPIF. In cases where the entire provider profile is too large to be completely transferred to the temporary filter instance, parts of the profile have to be selected. On the one hand, this selection has to be made based on the user profile, or the quality of the resulting recommendations would be compromised. On the other hand, the selection should be made in a manner that ensures only minimal information about the user profile is given away. In the ideal case, the provider is not able to deduce more information about the user profile than he is able to do anyway, based on the recommendations themselves. With regard to the filtering techniques included in the framework, as described in Section 4, the direct feature-based filtering technique and the distributed collaborative filtering technique are both obviously suitable because they are based on entire profiles. The clustered feature-based filtering technique may be used in a way ensuring that no additional information is provided and is therefore ideally suitable as well. The decision tree-based filtering technique leaks some additional information about the user profile, but it still prevents the provider from determining the specific items contained in the user profile.

5.2 Security Features

Our framework for context-aware services provides countermeasures against different security risks inherent to distributed systems. These countermeasures are provided by the underlying Multi-Agent System (MAS) technology-based architecture. In this paper, we focus on threats related to personal and other kinds of information.

5.2.1 Illegal Access of Information

An attacker, i.e. an agent or external entity, may try to access information in an illegal manner. In the context of personalized services, agents deployed by competing service providers may try to access data contained within each other, or eavesdrop on communication channels, in order to acquire additional information provided by the attacked service. Moreover, attackers may attempt to obtain user profile information in a similar manner. Countermeasures against this threat are provided by using security mechanisms for agent platform management, which prevent the access of data contained within other agents, and by utilizing mechanisms for secure communication (i.e. an SSL or an equivalent application layer protocol).

5.2.2 Unauthorized Access of Resources

An attacking agent may also attempt to utilize the interfaces provided by other agents, i.e. agent services, to access information in an unauthorized way. Alternatively, an attacking mobile agent may try to migrate to a platform in an unauthorized way, e.g. in order to acquire information via the usage of restricted services. Countermeasures against this threat are provided by using Service Control Lists (SCLs), i.e. access control lists containing rules for deciding whether a service usage request should be granted or rejected.

5.2.3 Malicious Hosts

While agents and platforms are adequately protected against attacks of other agents and external entities, the possibility of attacks by a host against agents residing on its platform remains. There are several approaches suggesting various countermeasures [10][11], but none of these solutions is entirely feasible, apart from deploying agent platforms on tamper-proof hardware. In our architecture, however, a host does not have to be provided by one of the participating roles, but may be provided by an external party instead. Therefore, it is reasonable to assume the host to be trustworthy.

5.2.4 Threats Related to User Interaction

In our architecture, the human user has to communicate with an agent located within a remote environment throughout all service interaction, because the requirement of device independence prevents the possibility of deploying software for user interaction on the user's device. All communication channels between human users and agents are based on the Multi–Access Service Platform (MASP) [3]. The MASP offers secure communication for certain interfaces (e.g. an HTTPS connection for HTTP-based communication), while other interfaces, such as the voice interface, cannot be secured. The user is made aware of this risk, however, and may switch modalities seamlessly when communicating sensitive information.

6 The Smart Event Assistant

The framework for context-aware services has been implemented in conjunction with prototypical application services for the entertainment domain, resulting in the Smart Event Assistant (SEA). In this section, we briefly describe the underlying MAS architecture with a focus on its privacy and security features, followed by the implemented services and an exemplary scenario illustrating the usage of the Smart Event Assistant, with a focus on the personalized services and their implications regarding privacy and security. The Smart Event Assistant as well as further aspects of the framework for context-aware services are discussed in [12][13][*].

6.1 JIAC

The implementation of the framework has been carried out based on the FIPA-compliant MAS-architecture Java Intelligent Agent Componentware (JIAC) [14][15]. JIAC integrates fundamental aspects of autonomous agents regarding pro-activeness, intelligence, communication capabilities and mobility by providing a scalable component-based

[*] This work is based on a project funded by the German Federal Ministry of Education and Research under Grant No. 01AK037, and a project funded by the German Federal Ministry of Economics and Labour under Grant No. 01MD506. The Smart Event Assistant is accessible online via <http://www.smarteventassistant.de>.

architecture. Additionally, JIAC offers components realizing management and security functionality, and provides a methodology for Agent-Oriented Software Engineering (AOSE).

6.1.1 Privacy Features

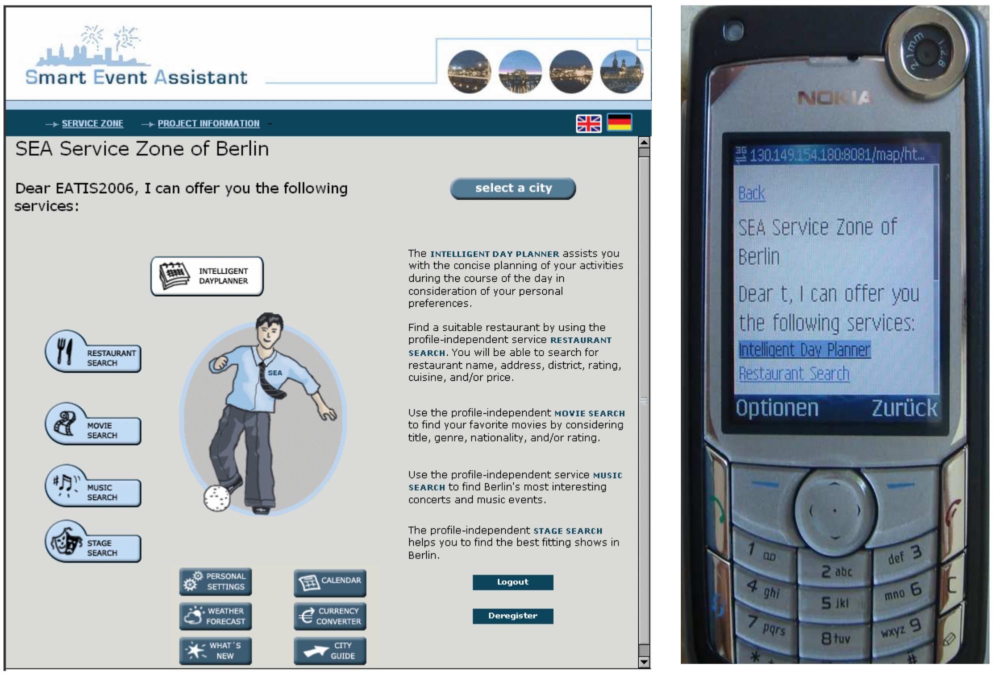
JIAC supports the application of Privacy-Preserving Information Filtering by providing mechanisms to control the communication capabilities of agents on specific platforms. The supervising agents of the PPIF architecture, as described in Section 5, are realized by adding specific functionality to the platform manager agents, who are subsequently responsible, in a manner similar to firewalls, for controlling all communication between agents on the respective platform and agents on different platforms.

6.1.2 Security Features

JIAC has been certified by the German Federal Office for Information Security (BSI) according to the Evaluation Assurance Level 3 (EAL3) of the Common Criteria for Information Technology Security standard [16]. JIAC offers several security features in the areas of access control regarding agent services, secure communication between agents, and low-level security based on Java security policies: Access control regarding agent services in JIAC IV is based on authenticated users or X.509 certificates belonging to agents. JIAC IV also offers means to secure the communication channel between agents. This is either achieved by using the SSL protocol on the transport layer or, if this is not possible, e.g. because a FIPA compliant exchange of speech acts via the Agent Communication Channel (ACC) is required, an application layer protocol similar to SSL is used to protect speech acts. X.509 certificates are used for access control and for protecting the communication channel. Finally, Java security mechanisms [17] are utilized to protect agents from attacks performed by other agents within the same Java Virtual Machine. Java security mechanisms are also used to represent human users as subjects within the Java Authentication and Authorization (JAAS) architecture [18].

6.2 SEA Services

As a prototypical application based on the framework for context-aware services, we have developed the Smart Event Assistant, which integrates various personalized services for entertainment planning in different German cities, such as a restaurant finder and a movie finder. Additional services, such as a calendar, a routing service and news services complement the information services. An intelligent day planner integrates all functionality by providing personalized recommendations for the various information services, based on the user's preferences and taking into account the location of the user as well as the potential venues.



SEA Service Zone of Berlin

Dear EATIS2006, I can offer you the following services:

The INTELLIGENT DAY PLANNER assists you with the concise planning of your activities during the course of the day in consideration of your personal preferences.

Find a suitable restaurant by using the profile-independent service RESTAURANT SEARCH. You will be able to search for restaurant name, address, district, rating, cuisine, and/or price.

Use the profile-independent MOVIE SEARCH to find your favorite movies by considering title, genre, nationality, and/or rating.

Use the profile-independent service MUSIC SEARCH to find Berlin's most interesting concerts and music events.

The profile-independent STAGE SEARCH helps you to find the best fitting shows in Berlin.

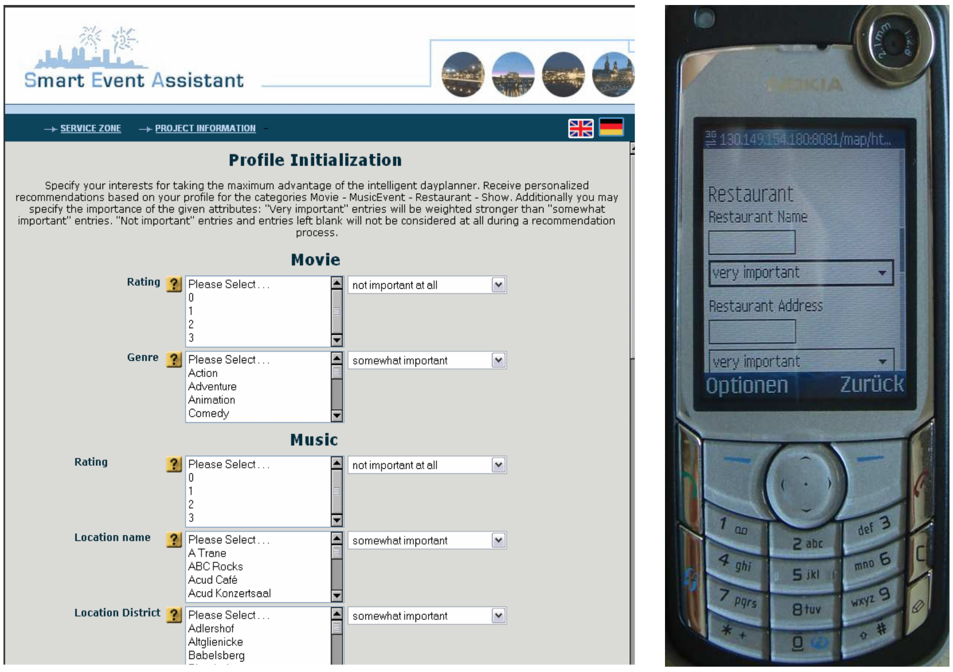
SEA Service Zone of Berlin

Dear t, I can offer you the following services:

Intelligent Day Planner

Restaurant Search

Fig. 4. The main interface of the Smart Event Assistant (standard and mobile client GUI)



Profile Initialization

Specify your interests for taking the maximum advantage of the intelligent dayplanner. Receive personalized recommendations based on your profile for the categories Movie - MusicEvent - Restaurant - Show. Additionally you may specify the importance of the given attributes: "Very important" entries will be weighted stronger than "somewhat important" entries. "Not important" entries and entries left blank will not be considered at all during a recommendation process.

Movie

Rating	Please Select . . . 0 1 2 3	not important at all
Genre	Please Select . . . Action Adventure Animation Comedy	somewhat important

Music

Rating	Please Select . . . 0 1 2 3	not important at all
Location name	Please Select . . . A Trane ABC Rocks Acud Café Acud Konzertsaal	somewhat important
Location District	Please Select . . . Adlershof Altglienicke Babelsberg	somewhat important

Restaurant

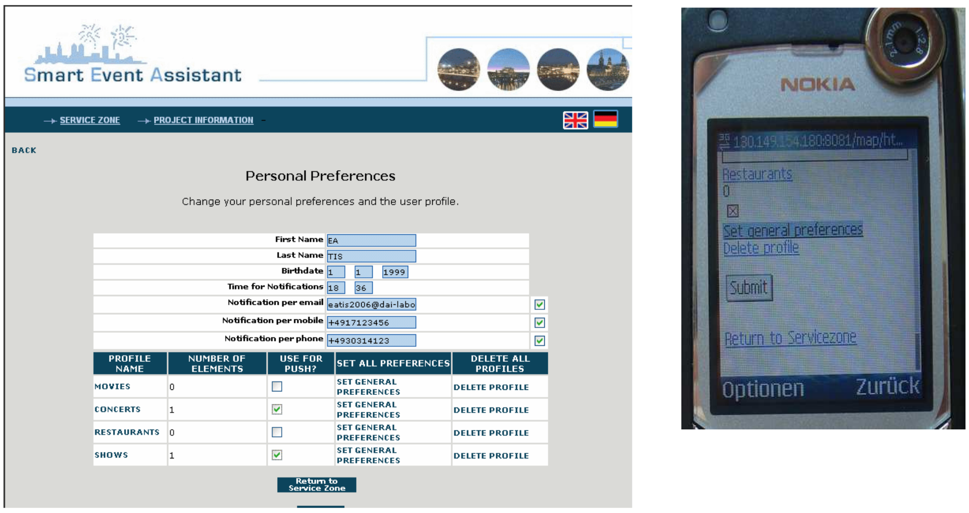
Restaurant Name

very important

Restaurant Address

very important

Fig. 5. The profile initialization dialog of the Smart Event Assistant (standard and mobile client GUI)



Personal Preferences

Change your personal preferences and the user profile.

First Name	EA	
Last Name	TIS	
Birthdate	1 1 1999	
Time for Notifications	18 36	
Notification per email	eatis2006@dai-labo	✓
Notification per mobile	+4917123456	✓
Notification per phone	+4930314123	✓

PROFILE NAME	NUMBER OF ELEMENTS	USE FOR PUSH?	SET ALL PREFERENCES	DELETE ALL PROFILES
MOVIES	0	☐	SET GENERAL PREFERENCES	DELETE PROFILE
CONCERTS	1	✓	SET GENERAL PREFERENCES	DELETE PROFILE
RESTAURANTS	0	☐	SET GENERAL PREFERENCES	DELETE PROFILE
SHOWS	1	✓	SET GENERAL PREFERENCES	DELETE PROFILE

Restaurants
0
☒
Set general preferences
Delete profile

Fig. 6. The profile management dialog of the Smart Event Assistant (standard and mobile client GUI)

Fig. 4 shows a screenshot of the browser-based main interface of the Smart Event Assistant. All dialogs are accessible via mobile devices as well. For personalized services, such as the intelligent day planner, newly registered users may initialize their personal profile, as shown in Fig. 5. Thus, personalized services are immediately accessible even for users who have not provided feedback about their preferences in another way. Via the profile manager service, the user may access his personal agent and directly and explicitly update his personal profile, as shown in Fig. 6.

It should be noted that the profile initialization and management dialogs are provided by the user agent itself, and not by the respective information services, the information is related to. All user interaction uses the Hypertext Transfer Protocol Secure (HTTPS). Thus, privacy and security aspects are adequately addressed by the personalized information services comprising the Smart Event Assistant.

7 Related Work

For the discussion of related work, we focus on the following two main areas: Privacy and security aspects of other frameworks for context-aware and mobile services, and approaches related to privacy-preserving information filtering. There are a number of frameworks offering similar functionality:

- The ***Mobile*** project has developed a framework for context-aware services with a special focus on mobile users. The framework is based on the SeMoA agent platform [19]. It heavily relies on mobile agents, and therefore focuses on mobile agent security. Additional security mechanisms include secure communication between agents and code signing.
- Another agent-based approach, the ***e-Wallet*** project [20], focuses on privacy aspects and the protection of the users' personal information. Users may protect their personal profiles by defining access and obfuscation rules, thus ensuring that personal information can only be accessed by approved services and third parties.
- The ***Oracle Application Server Wireless*** [21] is a commercial product offering a platform for the development of mobile and voice-operated services. Mobile services are secured by

offering support for Wireless Transport Layer Security (WTLS), Secure Sockets Layer (SSL), Virtual Private Networks (VPN) and a Public Key Infrastructure (PKI). Additional features are Access Control Lists (ACL) on the application layer and data storage mechanisms based on encrypted data.

Approaches related to privacy-preserving information filtering have been mainly researched in the following areas:

- The goal of ***private information retrieval*** approaches is to provide a mechanism for querying databases containing certain information without revealing the query to the respective information provider. An overview is given in [22].
- In ***secure multi-party computation*** protocols, a number of participants collaborate in order to determine the result of a publicly known function, which is applied on the private data of each of the participants, without revealing the private data to other participants. An overview is given in [23].
- In the area of information filtering, approaches for ***distributed collaborative filtering*** have been suggested [24], focusing, however, on providing recommendations based on similar user profiles. They cannot be applied for finding similar users directly.

8 Conclusion and Further Work

We have developed an agent-based Serviceware Framework, which assists service providers in developing context-aware services in general, and personalized information services in particular. Based on the framework, we have developed a prototypical application, the Smart Event Assistant, providing various context-aware services related to the planning of entertainment activities in various cities, thus demonstrating that the framework may in fact be utilized for the fast and efficient development of personalized information services addressing all relevant privacy and security aspects.

The biggest challenge we faced implementing the framework was to ensure adequate performance of the resulting application. Due to the significant amount of overhead within multi-agent system architectures, response times especially in the area of user interaction are more critical than in conventional approaches. We have addressed this issue mainly by performing operations offline, i.e. independent of the user interaction, as far as possible, e.g. when determining recommendations or users with similar preferences. Thus, we were able to ensure acceptable response times (a few seconds on average) in the resulting application.

Another challenging area was the selection of filtering techniques applicable to privacy-preserving information filtering. We had expected a tradeoff between the quality of the recommendations and the actual privacy of the personal information. It turned out, however, that there are in fact filtering techniques applicable to privacy-preserving information filtering without a significant loss of quality regarding recommendations.

We are currently working on the implementation of additional community services for connecting users with similar interests and preferences. As a next step, we intend to deploy the Smart Event Assistant as a commercial application in the context of major sports events, such as the 2006 FIFA World Cup in Germany, the 2008 Olympic Games in Beijing or the 2010 World Expo in Shanghai.

References

[1] S. Albayrak. Introduction to Agent Oriented Technology for Telecommunications. In: S. Albayrak (ed.), *Intelligent Agents for Telecommunications Applications*, IOS Press, pages 1 – 18, 1998.

[2] J.-H. Böse and S. Feuerstack. Adaptive User Interfaces for Ubiquitous Access to Agent-based Services, Workshop on Human-Agent Interaction, Agentcities ID3, February 2003.

[3] A. Rieger, R. Cissée, S. Feuerstack, J. Wohltorf and S. Albayrak. An Agent-Based Architecture for Ubiquitous Multi-Modal User Interfaces. In *Proceedings of AMT 2005, International Conference on Active Media Technology*, Takamatsu, Japan, May 2005.

[4] R. Burke. Hybrid Recommender Systems: Survey and Experiments. User *Modeling and User-Adapted Interaction* 12(4): 2002.

[5] D. Pennock, E. Horvitz, S. Lawrence, and C. L. Giles. Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence, UAI 2000*, Stanford, CA, pages 473–480, 2000.

[6] R. Burke. Knowledge-based Recommender Systems. In A. Kent (ed.), *Encyclopedia of Library and Information Systems*, 69(32): 2002.

[7] L. Cranor, J. Reagle, and M. Ackerman. Beyond Concern: Understanding Net Users' Attitudes about Online Privacy. Technical report TR 99.4.3, AT&T Labs-Research, April 1999.

[8] A.I. Anton, Qingfeng He, D.L. Baumer. Inside JetBlue's privacy policy violations. *IEEE Security & Privacy Magazine* 2(6): 2004.

[9] R. Cissée. An Architecture for Agent-Based Privacy-Preserving Information Filtering. Sixth International Workshop on Trust, Privacy, Deception and Fraud in Agent Systems, 2003.

[10] W. Jansen. Countermeasures for Mobile Agent Security. *Computer Communications, Special Issue on Advances in Research and Application of Network Security*, November 2000.

[11] T. Sander and C.F. Tschudin. Protecting Mobile Agents Against Malicious Hosts. Springer-Verlag, Lecture Notes in Computer Science #1419, June 1998.

[12] J. Wohltorf, R. Cissée, A. Rieger, H. Scheunemann. BerlinTainment - An Agent-Based Serviceware Framework for Context-Aware Services. In *Proceedings of 1st International Symposium on Wireless Communication Systems, ISWCS*, June 2004.

[13] J. Wohltorf, R. Cissée, A. Rieger. BerlinTainment: An Agent-Based Context-Aware Entertainment Planning System. In *IEEE Communications Magazine* 43(6): 2005.

[14] S. Fricke et al. Agent-based Telematic Services and Telecom Applications. In *Communications of the ACM* 44(4): 2001.

[15] R. Sesseler and S. Albayrak. Service-ware Framework for Developing 3G Mobile Services. The Sixteenth International Symposium on Computer and Information Sciences, ICSIS XVI, November 2001.

[16] T. Geissler, O. Kroll-Peters. Applying Security Standards to Multi Agent Systems. AAMAS Workshop: Safety & Security in Multiagent Systems, 2004.

[17] Li Gong, M. Mueller, H. Prafullchandra, and R. Schemers. Going Beyond the Sandbox: An Overview of the New Security Architecture in the Java Development Kit^{TM} 1.2. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, Monterey, California, December 1997.

[18] C. Lai, Li Gong, L. Koved, A. Nadalin, and R. Schemers, User Authentication and Authorization in the Java^{TM} Platform. In *Proceedings of the 15th Annual Computer Security Applications Conference*, Phoenix, Arizona, December 1999.

[19] U. Pinsdorf, J. Peters, M. Hoffmann, and P. Gupta. Context-aware Services based on Secure Mobile Agents. In *Proceedings of 10th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, 2002.

[20] F.L. Gandon, N.M. Sadeh. Semantic Web Technologies to Reconcile Privacy and Context Awareness. *Journal of Web Semantics* 1(3): 2004.

[21] Oracle application server wireless. http://www.oracle.com/technology/products/iaswe/index.html. May 1, 2006.

[22] A. Iliev, S.W. Smith. Protecting client privacy with trusted computing at the server. *IEEE Security & Privacy Magazine* 3(2): 2005.

[23] W. Du and M. J. Atallah. Secure MultiParty Computation Problems and Their Applications: A Review and Open Problems. in *Proceedings of the 2001 Workshop on New Security Paradigms*, pp. 13-22, 2001.

[24] J.F. Canny. Collaborative Filtering with Privacy. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, Berkeley, California, 2002.