# An Agent-Based Approach for Privacy-Preserving Recommender Systems

Richard Cissée
DAI-Labor, TU Berlin
Franklinstrasse 28/29
10587 Berlin
richard.cissee@dai-labor.de

Sahin Albayrak
DAI-Labor, TU Berlin
Franklinstrasse 28/29
10587 Berlin
sahin.albayrak@dai-labor.de

## ABSTRACT

Recommender Systems are used in various domains to generate personalized information based on personal user data. Preserving the privacy of all participants is an essential requirement of the underlying Information Filtering architectures, because the deployed Recommender Systems have to be accepted by privacy-aware users as well as information and service providers. Existing approaches neglect to address privacy in this multilateral way.

We have developed an approach for privacy-preserving Recommender Systems based on Multi-Agent System technology which enables applications to generate recommendations via various filtering techniques while preserving the privacy of all participants. We describe the main modules of our solution as well as an implemented application based on this approach.

## Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information Filtering*; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent Systems*

## General Terms

Management, Security, Human Factors, Standardization

## Keywords

Multi-Agent Systems, Recommender Systems, Information Filtering, Privacy, Trust

## 1. INTRODUCTION

Information Filtering (IF) systems aim at countering information overload by extracting information that is relevant for a given user out of a large body of information available via an information provider. In contrast to Information Retrieval (IR) systems, where relevant information is extracted based on search queries, IF architectures generate personalized information based on user profiles containing, for each given user, personal data, preferences, and rated items. The provided body of information is usually structured and collected in provider profiles. Filtering techniques operate on these profiles in order to generate recommendations of items that are probably relevant for a given user, or in order to determine users with similar interests, or both. Depending on the respective goal, the resulting systems constitute Recommender Systems [5], Matchmaker Systems [9], or a combination thereof.

The aspect of privacy is an essential issue in all IF systems: Generating personalized information obviously requires the use of personal data. Users can be expected to be less reluctant to provide personal information if they trust the system to be privacy-preserving with regard to personal data, according to surveys indicating major privacy concerns of users in the context of Recommender Systems and e-commerce in general [22]. Similar considerations also apply to the information provider, who may want to control the dissemination of the provided information, and to the provider of the filtering techniques, who may not want the details of the utilized filtering algorithms to become common knowledge. A privacy-preserving IF system should therefore balance these requirements and protect the privacy of all parties involved in a multilateral way, while addressing general requirements regarding performance, security and quality of the recommendations as well. The following section lists some approaches with similar goals, but none of these provide a generic approach in which the privacy of all parties is preserved.

We have developed an agent-based approach for privacy-preserving IF which has been utilized for realizing a combined Recommender/Matchmaker System as a deployed application supporting users in planning entertainment-related activities. In this paper, we focus on the Recommender System functionality. Our approach is based on Multi-Agent System (MAS) technology because fundamental features of agents such as autonomy, adaptability and the ability to communicate are essential requirements of our approach. In other words, the realized approach does not merely constitute a solution for privacy-preserving IF within a MAS context, but rather utilizes a MAS architecture in order to realize a solution for privacy-preserving IF, which could not be realized easily otherwise.

The paper is structured as follows: Section 2 describes related work. Section 3 describes the general ideas of our approach. In Section 4, we describe essential details of the

modules of our approach and their implementation. In Section 5, we evaluate the approach, mainly via the realized application. Section 6 concludes the paper with an outlook and outlines further work.

## 2. RELATED WORK

There is a large amount of work in related areas, such as Private Information Retrieval [7], Privacy-Preserving Data Mining [2], and other privacy-preserving protocols [4, 15], most of which is based on Secure Multi-Party Computation [25]. We have ruled out Secure Multi-Party Computation approaches mainly because of their complexity, and because the algorithm that is computed securely is not considered to be private in these approaches.

Various enforcement mechanisms have been suggested that are applicable in the context of privacy-preserving Information Filtering, such as enterprise privacy policies [16] or hippocratic databases [1], both of which annotate user data with additional meta-information specifying how the data is to be handled on the server side. These approaches ultimately assume that the provider actually intends to protect the privacy of the user data, and support him in this regard, but they are not intended to prevent the provider from acting in a malicious manner. Trusted computing aims at realizing a trusted systems by increasing the security of open systems to a level comparable with the level of security that is possible in closed systems. It is based on a combination of tamper-proof hardware and various software components. Some example applications, including peer-to-peer networks, distributed firewalls, and distributed computing in general, are listed in [12].

There are some approaches for privacy-preserving Recommender Systems based on distributed collaborative filtering, in which recommendations are generated via a public model aggregating the distributed user profiles without containing explicit information about user profiles themselves. This is achieved via Secure Multi-Party Computation [6], or via random perturbation of the user data [19]. In [18], various approaches are integrated within a single architecture. In [9], an agent-based approach is described in which user agents representing similar users are discovered via a transitive traversal of user agents. Privacy is preserved through pseudonymous interaction between the agents and through adding obfuscating data to personal information. More recent related approaches are described in [17].

In [3], an agent-based architecture for privacy-preserving demographic filtering is described which may be generalized in order to support other kinds of filtering techniques. While in some aspects similar to our approach, this architecture addresses at least two aspects inadequately, namely the protection of the filter against manipulation attempts, and the prevention of collusions between the filter and the provider.

## 3. PRIVACY-PRESERVING INFORMATION FILTERING

We identify three main abstract entities participating in an information filtering process within a distributed system: A user entity, a provider entity, and a filter entity. Whereas in some applications the provider and filter entities explicitly trust each other, because they are deployed by a single party, our solution is applicable more generically because it does not require any trust between the main abstract entities. In this paper, we focus on aspects related to the information filtering process itself, and omit all aspects related to information collection and processing, i.e. the stages in which profiles are generated and maintained, mainly because these stages are less critical with regard to privacy, as they involve fewer different entities.

### 3.1 Requirements

Our solution aims at meeting the following requirements with regard to privacy:

- *User Privacy*: No linkable information about user profiles should be acquired permanently by any other entity or external party, including other user entities. Single user profile items, however, may be acquired permanently if they are unlinkable, i.e. if they cannot be attributed to a specific user or linked to other user profile items. Temporary acquisition of private information is permitted as well. Sets of recommendations may be acquired permanently by the provider, but they should not be linkable to a specific user. These concessions simplify the resulting protocol and allow the provider to obtain recommendations and single unlinkable user profile items, and thus to determine highly requested information and optimize the offered information accordingly.
- *Provider Privacy*: No information about provider profiles, with the exception of the recommendations, should be acquired permanently by other entities or external parties. Again, temporary acquisition of private information is permitted. Additionally, the propagation of provider information is entirely under the control of the provider. Thus, the provider is enabled to prevent e.g. the automatic large-scale extraction of information.
- *Filter Privacy*: Details of the algorithms applied by the filtering techniques should not be acquired permanently by any other entity or external party. General information about the algorithm may be provided by the filter entity in order to help other entities to reach a decision on whether to apply the respective filtering technique.

In addition, general requirements regarding the quality of the recommendations as well as security aspects, performance and broadness of the resulting system have to be addressed as well. While small trade-offs may be acceptable, the resulting system should reach a level similar to regular Recommender Systems with regard to these requirements.

### 3.2 Outline of the Solution

The basic idea for realizing a protocol fulfilling the privacy-related requirements in Recommender Systems is suggested by allowing the temporary acquisition of private information: User and provider entity both propagate the respective profile data to the filter entity. The filter entity provides the result information, and subsequently deletes all private information, thus fulfilling the requirement regarding permanent acquisition of private information.

The entities whose private information is propagated have to be certain that the respective information is actually acquired temporarily only. Trust in this regard may be established in two main ways:

- *Trusted Software:* The respective entity itself is trusted to remove the respective information as specified.

- *Trusted Environment:* The respective entity operates in an environment that is trusted to control the communication and life cycle of the entity to an extent that the removal of the respective information may be achieved regardless of the attempted actions of the entity itself. Additionally, the environment itself is trusted not to act in a malicious manner (e.g. it is trusted not to acquire and propagate the respective information itself).

In both cases, trust may be established in various ways. Reputation-based mechanisms, additional trusted third parties certifying entities or environments or trusted computing mechanisms may be used. Our approach is based on a trusted environment realized via trusted computing mechanisms, because we see this solution as the most generic and realistic approach. This decision is discussed briefly in Section 5.

We are now able to specify the abstract information filtering protocol as shown in Figure 1: The filter entity deploys a *Temporary Filter Entity (TFE)* operating in a trusted environment. The user entity deploys an additional relay entity operating in the same environment.

Through mechanisms provided by this environment, the relay entity is able to control the communication of the TFE, and the provider entity is able to control the communication of both relay entity and the TFE. Thus, it is possible to ensure that the controlled entities are only able to propagate recommendations, but no other private information. In the first stage (steps 1.1 to 1.3 of Figure 1), the relay entity establishes control of the TFE, and thus prevents it from propagating user profile information. User profile data is propagated without participation of the provider entity from the user entity to the TFE via the relay entity. In the second stage (steps 2.1 to 2.3 of Figure 1), the provider entity establishes control of both relay and TFE, and thus prevents them from propagating provider profile information. Provider profile data is propagated from the provider entity to the TFE via the relay entity. In the third stage (steps 3.1 to 3.4 of Figure 1), the TFE returns the recommendations via the relay entity, and the controlled entities are terminated. Taken together, these steps ensure that all private information is acquired temporarily only by the other main entities. The problems of determining acceptable queries on the provider profile and ensuring unlinkability of the result information are discussed in the following section.

Our approach requires each entity in the distributed architecture to have the following five main abilities: The ability to perform certain well-defined tasks (such as carrying out a filtering process) with a high degree of autonomy, i.e. largely independent of other entities (e.g. because the respective entity is not able to communicate in an unrestricted manner), the ability to be deployable dynamically in a well-defined environment, the ability to communicate with other entities, the ability to achieve protection against external manipulation attempts, and the ability to control and restrict the communication of other entities.

MAS architectures are an ideal solution for realizing a distributed system containing all features outlined above, because they provide agents as entities that are actually characterized by autonomy, mobility and the ability to communicate [24], as well as agent platforms as environments providing means to realize the security of agents. In this context, the issue of malicious hosts, i.e. hosts attacking agents, has to be addressed explicitly. Additionally, existing MAS architectures generally do not allow agents to control the communication of other agents. It is possible, however, to expand a MAS architecture and to provide designated agents with this ability. For these reasons, our solution is based on a FIPA[10]-compliant MAS architecture. The entities introduced above are mapped directly to agents, and the trusted environment in which they exist is realized as agent platforms.

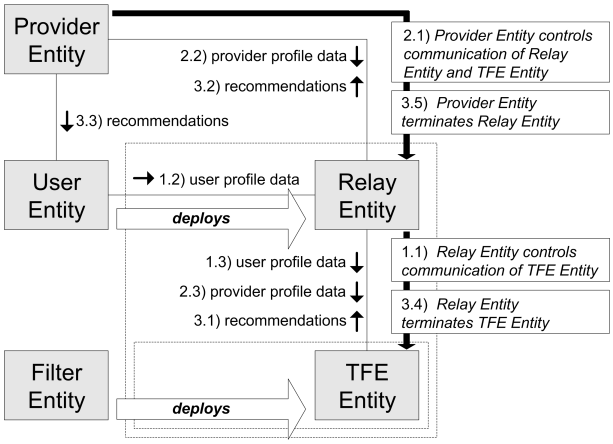


**Figure 1: The abstract privacy-preserving information filtering protocol. All communication across the environments indicated by dashed lines is prevented with the exception of communication with the controlling entity.**

In addition to the MAS architecture itself, which is assumed as given, our solution consists of the following five main modules, each of which basically contains ontologies, agent services, and internal functionality:

- The **Controller Module** described in Section 4.1 provides functionality for controlling the communication capabilities of agents.

- The **Transparent Persistence Module** facilitates the use of different data storage mechanisms, and provides a uniform interface for accessing persistent information, which may be utilized for monitoring critical interactions involving potentially private information e.g. as part of queries. Its description is outside the scope of this paper.

- The **Recommender Module**, details of which are described in Section 4.2, provides Recommender System functionality.

- The **Matchmaker Module** provides Matchmaker System functionality. Its description is outside the scope of this paper.

- Finally, a separate module described in Section 4.4 provides **Exemplary Filtering Techniques** in order to show that various restrictions imposed on filtering techniques by our approach may actually be fulfilled.

The trusted environment introduced above encompasses the MAS architecture itself and the Controller Module, which have to be trusted to act in a non-malicious manner in order to rule out the possibility of malicious hosts.

# 4. MAIN MODULES AND IMPLEMENTATION

In this section, we describe the main modules of our approach, and outline the implementation. While we have chosen a specific architecture for the implementation, the specification of the module is applicable to any FIPA-compliant MAS architecture. A module basically encompasses functionality provided by agents via agent services.

Regarding notation conventions used in this paper, $\{m\}_{K_X}$ denotes a message $m$ encrypted via a non-specified symmetric encryption scheme with a secret key $K_X$ used for encryption and decryption which is initially known only to participant $X$. A key $K_{XY}$ is a key shared by participants $X$ and $Y$. A cryptographic hash function is used at various points of the protocol, i.e. a function returning a hash value $h(x)$ for given data $x$ that is both preimage-resistant and collision-resistant. We denote a set of hash values for a data set $X = \{x_1, .., x_n\}$ as $H(X) = \{h(x_1), .., h(x_n)\}$, whereas $h(X)$ denotes a single hash value of a data set[1].

## 4.1 Controller Module

TODO

TODO - REF to environment here!

TODO

As noted above, the ability to control the communication of agents is generally not a feature of existing Multi-Agent System architectures but at the same time a central requirement of our approach for privacy-preserving Information Filtering. The required functionality cannot be realized based on regular agent services and/or components, because an agent on a platform is usually not allowed to interfere with the actions of other agents in any way. Otherwise, the security of agents would be severely compromised. Therefore, we add additional infrastructure providing the required functionality to the MAS architecture itself.

Controlling the communication capabilities of an agent is realized by restricting via rules, in a manner similar to a firewall, but with the consent of the respective agent, its incoming and outgoing communication to specific platforms or agents on external platforms as well as other possible communication channels, such as the file system. Consent is required because otherwise the overall security would be compromised, as attackers could arbitrarily block various communication channels. Our approach does not require controlling the communication between agents on the same platform, and therefore this aspect is not addressed. Consequently, all rules addressing communication capabilities have to be enforced across entire platforms, because otherwise a controlled agent could just use a non-controlled agent on the same platform as a relay for communicating with agents residing on external platforms. Various agent services provide functionality for adding and revoking control of platforms, including functionality required in complex scenarios where controlled agents in turn control further platforms.

[1]For the implementation, we have used the Advanced Encryption Standard (AES) as the symmetric encryption scheme and SHA-1 as the cryptographic hash function.

**Table 1: The basic information filtering protocol with participants $U$ = user agent, $P$ = provider agent, $F$ = TFE agent, $R$ = relay agent, based on the abstract protocol shown in Figure 1. $UP$ denotes the user profile with $UP = \{up_1, .., up_n\}$, $PP$ denotes the provider profile, and $REC$ denotes the set of recommendations with $REC = \{rec_1, .., rec_m\}$.**

| Phase. Step | *Sender* → *Receiver* | Message or Action |
|---|---|---|
| 1.1 | $R \rightarrow F$ | establish control |
| 1.2 | $U \rightarrow R$ | $UP$ |
| 1.3 | $R \rightarrow F$ | $UP$ |
| 2.1 | $P \rightarrow R, F$ | establish control |
| 2.2 | $P \rightarrow R$ | $PP$ |
| 2.3 | $R \rightarrow F$ | $PP$ |
| 3.1 | $F \rightarrow R$ | $REC$ |
| 3.2 | $R \rightarrow P$ | $REC$ |
| 3.3 | $P \rightarrow U$ | $REC$ |
| 3.4 | $R \rightarrow F$ | terminate $F$ |
| 3.5 | $P \rightarrow R$ | terminate $R$ |

The implementation of the actual control mechanism depends on the actual MAS architecture. As most MAS architectures are based on Java, we suggest utilizing methods provided via the Java Security Manager as part of the Java security model. Thus, the supervisor agent is enabled to define custom security policies, thereby granting or denying other agents, which are executed as threads in the JVM, access to resources such as files or sockets for TCP/IP-based communication. Apart from denial of service attacks in which agents may attempt to disrupt the execution of other agents on the same platform by seizing large amounts of available resources, all other malicious actions and communication attempts may be blocked via this mechanism. In MAS architectures realizing agent communication via a designated Agent Communication Channel, only this channel has to be controlled as long as agents are not able to communicate in any other way.

## 4.2 Recommender Module

The Recommender Module is mainly responsible for carrying out information filtering processes, according to the protocol described in Table 1. The participating entities are realized as agents, and the interactions as agent services. We assume that mechanisms for secure agent communication are available within the respective MAS architecture. Two main issues have to be addressed in this module: The relevant parts of the provider profile have to be retrieved without compromising the user's privacy, and the recommendations have to be propagated in a privacy-preserving way.

Our solution is based on a threat model in which no main abstract entity may safely assume any other abstract entity to act in an honest manner: Each entity has to assume that other entities may attempt to obtain private information, either while following the specified protocol or even by deviating from the protocol. According to [14], we classify the former case as honest-but-curious behavior (as an example, the TFE may propagate recommendations as specified, but may additionally attempt to propagate private information), and the latter case as malicious behavior (as an example, the

**Table 2: The updated second stage of the information filtering protocol with definitions as above. $PP_q$ is the part of the provider profile $PP$ returned as the result of the query $q$.**

| Phase. Step | *Sender* → *Receiver* | Message or Action |
|---|---|---|
| repeat 2.1 to 2.3 $\forall$ $up \in UP$: | | |
| 2.1 | $F \rightarrow R$ | $q(up)$ (a query based on $up$) |
| 2.2 | $R^{anon} \rightarrow P$ | $q(up)$ (R remains anonymous) |
| 2.3 | $P \rightarrow R^{anon}$ | $\{PP_{q(up)}\}_{K_P}$ |
| 2.4 | $P \rightarrow R, F$ | establish control |
| 2.5 | $P \rightarrow R$ | $K_P$ |
| 2.6 | $R \rightarrow F$ | $PP_{q(UP)}$ |

filter may attempt to propagate private information instead of the recommendations).

### 4.2.1 Retrieving the Provider Profile

As outlined above, the relay agent relays data between the TFE agent and the provider agent. These agents are not allowed to communicate directly, because the TFE agent cannot be assumed to act in an honest manner. Unlike the user profile, which is usually rather small, the provider profile is often too voluminous to be propagated as a whole efficiently. A typical example is a user profile containing ratings of about 100 movies, while the provider profile contains some 10,000 movies. Retrieving only the relevant part of the provider profile, however, is problematic because it has to be done without leaking sensitive information about the user profile. Therefore, the relay agent has to analyze all queries on the provider profile, and reject potentially critical queries, such as queries containing a set of user profile items.

Because the propagation of single unlinkable user profile items is assumed to be uncritical, we extend the information filtering protocol as follows: The relevant parts of the provider profile are retrieved based on single anonymous interactions between the relay and the provider. If the MAS architecture used for the implementation does not provide an infrastructure for anonymous agent communication, this feature has to be provided explicitly: The most straightforward way is to use additional relay agents deployed via the main relay agent and used once for a single anonymous interaction. Obviously, unlinkability is only achieved if multiple instances of the protocol are executed simultaneously between the provider and different users. Because agents on controlled platforms are unable to communicate anonymously with the respective controlling agent, control has to be established after the anonymous interactions have been completed. To prevent the uncontrolled relay agents from propagating provider profile data, the respective data is encrypted and the key is provided only after control has been established. Therefore, the second phase of the protocol described in Table 1 is replaced as described in Table 2.

Additionally, the relay agent may allow other interactions as long as no user profile items are used within the queries. In this case, the relay agent has to ensure that the provider does not obtain any information exceeding the information deducible via the recommendations themselves. The cluster-based filtering technique described in Section 4.3 is an example for a filtering technique operating in this manner.

### 4.2.2 Recommendation Propagation

**Table 3: The updated final stage of the information filtering protocol with definitions as above.**

| Phase. Step | *Sender* → *Receiver* | Message or Action |
|---|---|---|
| 3.1 | $F \rightarrow R$ | $REC, \{H(REC)\}_{K_{PF}}$ |
| 3.2 | $R \rightarrow P$ | $h(K_R), \{\{H(REC)\}_{K_{PF}}\}_{K_R}$ |
| 3.3 | $P \rightarrow R$ | $K_{PF}$ |
| 3.4 | $R \rightarrow P$ | $K_R$ |
| repeat 3.5 $\forall$ $rec \in REC$: | | |
| 3.5 | $R \rightarrow P$ | $\{rec\}_{K_{UR_{rec}}}$ |
| repeat 3.6 $\forall$ $rec \in REC$: | | |
| 3.6 | $P \rightarrow U$ | $h(K_{P_{rec}}), \{\{rec\}_{K_{UR_{rec}}}\}_{K_{P_{rec}}}$ |
| repeat 3.7 to 3.8 $\forall$ $rec \in REC$: | | |
| 3.7 | $U \rightarrow P$ | $K_{UR_{rec}}$ |
| 3.8 | $P \rightarrow U$ | $K_{P_{rec}}$ |
| 3.9 | $U \rightarrow F$ | terminate $F$ |
| 3.10 | $P \rightarrow U$ | terminate $U$ |

The propagation of the recommendations is even more problematic mainly because more participants are involved: Recommendations have to be propagated from the TFE agent via the relay and provider agent to the user agent. No participant should be able to alter the recommendations or use them for the propagation of private information. Therefore, every participant in this chain has to obtain and verify the recommendations in unencrypted form prior to the next agent in the chain, i.e. the relay agent has to verify the recommendations before the provider obtains them, and so on. Therefore, the final phase of the protocol described in Table 1 is replaced as described in Table 3. It basically consists of two parts (Step 3.1 to 3.4, and Step 3.5 to Step 3.8), each of which provide a solution for a problem related to the *prisoners' problem* [21], in which two participants (the prisoners) intend to exchange a message via a third, untrusted participant (the warden) who may read the message but must not be able to alter it in an undetectable manner. There are various solutions for protocols addressing the prisoners' problem. The more obvious of these, however, such as protocols based on the use of digital signatures, introduce additional threats e.g. via the possibility of additional subliminal channels [21]. In order to minimize the risk of possible threats, we have decided to use a protocol that only requires a symmetric encryption scheme.

The first part of the final phase is carried out as follows: In order to prevent the relay from altering recommendations, they are propagated by the filter together with an encrypted hash in Step 3.1. Thus, the relay is able to verify the recommendations before they are propagated further. The relay, however, may suspect the data propagated as the encrypted hash to contain private information instead of the actual hash value. Therefore, the encrypted hash is encrypted again and propagated together with a hash on the respective key in Step 3.2. In Step 3.3, the key $K_{PF}$ is revealed to the relay, allowing the relay to validate the encrypted hash. In Step 3.4, the key $K_R$ is revealed to the provider, allowing the provider to decrypt the data received in Step 3.2 and thus to obtain $H(REC)$. Propagating the hash of the key $K_R$ prevents the relay from altering the recommendations to $REC'$ after Step 3.3, which would be undetectable otherwise because the relay could choose a key

$K_{R'}$ so that $\{\{H(REC)\}_{K_{PF}}\}_{K_R} = \{\{H(REC')\}_{K_{PF}}\}_{K_{R'}}$. The encryption scheme used for encrypting the hash has to be secure against known-plaintext attacks, because otherwise the relay may be able to obtain $K_{PF}$ after Step 3.1 and subsequently alter the recommendations in an undetectable way. Additionally, the encryption scheme must not be commutative for similar reasons.

The remaining protocol steps are interactions between relay, provider and user agent. The interactions of Step 3.5 to Step 3.8 ensure, via the same mechanisms used in Step 3.1 to 3.4, that the provider is able to analyze the recommendations before the user obtains them, but at the same time prevent the provider from altering the recommendations. Additionally, the recommendations are not processed at once, but rather one at a time, to prevent the provider from withholding all recommendations.

Upon completion of the protocol, both user and provider have obtained a set of recommendations. If the user wants these recommendations to be unlinkable to himself, the user agent has to carry out the entire protocol anonymously. Again, the most straightforward way to achieve this is to use additional relay agents deployed via the user agent which are used once for a single information filtering process.

## 4.3 Exemplary Filtering Techniques

The filtering technique applied by the TFE agent cannot be chosen freely: All collaboration-based approaches, such as collaborative filtering techniques based on the profiles of a set of users, are not applicable because the provider profile does not contain user profile data. Instead, these approaches are realized via the Matchmaker Module, which is outside the scope of this paper. Learning-based approaches are not applicable because the TFE agent cannot propagate any acquired data to the filter, which effectively means that the filter is incapable of learning. Filtering techniques that are actually applicable are feature-based approaches, such as content-based filtering (in which profile items are compared via their attributes) and knowledge-based filtering (in which domain-specific knowledge is applied in order to match user and provider profile items). An overview of different classes and hybrid combinations of filtering techniques is given in [5]. We have implemented the following two generic content-based filtering approaches that are applicable within our approach:

A direct content-based filtering technique based on the class of item-based top-$n$ recommendation algorithms [8] is used in cases where the user profile contains items that are also contained in the provider profile. In a preprocessing stage, i.e. prior to the actual information filtering processes, a model is generated containing the $k$ most similar items for each provider profile item. While computationally rather complex, this approach is feasible because it has to be done only once, and it is carried out in a privacy-preserving way via interactions between the provider agent and a TFE agent. The resulting model is stored by the provider agent and can be seen as an additional part of the provider profile. In the actual information filtering process, the $k$ most similar items are retrieved for each single user profile item via queries on the model (as described in Section 4.2.1, this is possible in a privacy-preserving way via anonymous communication). Recommendations are generated by selecting the $n$ most frequent items from the result sets that are not already contained within the user profile.

As an alternative approach applicable when the user profile contains information in addition to provider profile items, we provide a cluster-based approach in which provider profile items are clustered in a preprocessing stage via an agglomerative hierarchical clustering approach. Each cluster is represented by a centroid item, and the cluster elements are either sub-clusters or, on the lowest level, the items themselves. In the information filtering stage, the relevant items are retrieved by descending through the cluster hierarchy in the following manner: The cluster items of the highest level are retrieved independent of the user profile. By comparing these items with the user profile data, the most relevant sub-clusters are determined and retrieved in a subsequent iteration. This process is repeated until the lowest level is reached, which contains the items themselves as recommendations. Throughout the process, user profile items are never propagated to the provider as such. The information deducible about the user profile is the same information deducible via the recommendations themselves (because essentially only a chain of cluster centroids leading to the recommendations is retrieved), and therefore it is not regarded as privacy-critical.

## 4.4 Implementation

We have implemented the approach for privacy-preserving IF based on JIAC IV [11], a FIPA-compliant MAS architecture. JIAC IV integrates fundamental aspects of autonomous agents regarding pro-activeness, intelligence, communication capabilities and mobility by providing a scalable component-based architecture. Additionally, JIAC IV offers components realizing management and security functionality, and provides a methodology for Agent-Oriented Software Engineering. JIAC IV stands out among other MAS architectures as the only security-certified architecture, since it has been certified by the German Federal Office for Information Security according to the EAL3 of the Common Criteria for Information Technology Security standard [13]. JIAC IV offers several security features in the areas of access control for agent services, secure communication between agents, and low-level security based on Java security policies [20], and thus provides all security-related functionality required for our approach.

We have extended the JIAC IV architecture by adding the mechanisms for communication control described in Section 4.1. Regarding the issue of malicious hosts, we currently assume all providers of agent platforms to be trusted. We are additionally developing a solution that is actually based on a trusted computing infrastructure.

# 5. EVALUATION

For the evaluation of our approach, we have examined whether and to which extent the requirements (mainly regarding privacy, performance, and quality) are actually met. Privacy is directly addressed by the modules described above. Performance is a critical issue, mainly because of the overhead caused by creating additional agents and agent platforms for controlling communication, and by the additional interactions within the Recommender Module. Overall, a single information filtering process takes about ten times longer compared to a non-privacy-preserving information filtering process with the same results, which is considerable but still acceptable under certain conditions, as described in the following section.


**Figure 2: The Smart Event Assistant, a privacy-preserving Recommender System supporting users in planning entertainment-related activities.**

## 5.1 The Smart Event Assistant

As a proof of concept, and in order to evaluate performance and quality under real-life conditions, we have applied our approach within the Smart Event Assistant, a MAS-based Recommender System which integrates various personalized services for entertainment planning in different German cities, such as a restaurant finder and a movie finder [23]. 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. All services are accessible via mobile devices as well[2]. Figure 2 shows a screenshot of the intelligent day planner's result dialog. The Smart Event Assistant is entirely realized as a MAS system providing, among other functionality, various filter agents and different service provider agents, which together with the user agents utilize the functionality provided within the modules of our approach.

Recommendations are generated in two ways: A push service delivers new recommendations to the user in regular intervals (e.g. once per day) via email or SMS. Because the user is not online during these interactions, they are less critical with regard to performance and therefore, the protracted duration of the information filtering process is acceptable. Recommendations generated for the intelligent day planner, however, have to be delivered with very little latency because the process is triggered by the user, who expects to receive results promptly. In this scenario, the overall performance is substantially improved by setting up the relay agent and the TFE agent offline, i.e. prior to the user's request, and by an efficient retrieval of the relevant part of the provider profile: Because the user is only interested in items, such as movies, available within a certain time period and related to specific locations, such as screenings at cinemas in a specific city, the relevant part of the provider profile is usually small enough to be propagated entirely. Because these restrictions are not seen as privacy-critical (they are not based on the user profile, but rather constitute a short-term information need), the relevant part of the provider profile may be propagated as a whole, with no need for complex interactions. Taken together, these improvements result in a filtering process that takes about three times as long as the respective non-privacy-preserving filtering process, which we regard as an acceptable trade-off for the increased privacy. Table 4 shows the results of the performance evaluation in more detail.

[2]The Smart Event Assistant is accessible online via http://www.smarteventassistant.de.

**Table 4: Time complexity of typical privacy-preserving (PP) vs. non-privacy-preserving (NPP) filtering processes in the realized application. In the non-privacy-preserving version, an agent retrieves the profiles directly and propagates the result to a provider agent.**

| scenario | "push" | | "day planning" | |
|---|---|---|---|---|
| version | NPP | PP | NPP | PP |
| profile size (retrieved/total amount of items) | | | | |
| user | 25/25 | | 25/25 | |
| provider | 125/10,000 | | 500/10,000 | |
| elapsed time in filtering process (in seconds) | | | | |
| setup | n/a | 2.2 | n/a | offline |
| database access | 0.2 | 0.5 | 0.4 | 0.4 |
| profile propagation | n/a | 0.8 | n/a | 0.3 |
| filtering algorithm | 0.2 | 0.2 | 0.2 | 0.2 |
| result propagation | 0.1 | 1.1 | 0.1 | 1.1 |
| complete time | 0.5 | 4.8 | 0.7 | 2.0 |

In these scenarios, a direct content-based filtering technique similar to the one described in Section 4.3 is applied. Because equivalent filtering techniques are applied successfully in regular Recommender Systems [8], there are no negative consequences with regard to the quality of the recommendations.

## 5.2 Alternative Approaches

As described in Section 3.2, our solution is based on trusted computing. There are be more straightforward ways to realize privacy-preserving IF, e.g. by utilizing a centralized architecture in which the privacy-preserving provider-side functionality is realized as trusted software based on trusted computing. However, we consider these approaches to be unsuitable because they are far less generic: Whenever some part of the respective software is patched, upgraded or replaced, the entire system has to be analyzed again in order to determine its trustworthiness, a process that is problematic in itself due to its complexity. In our solution, only a comparatively small part of the overall system is based on trusted computing. Because agent platforms can be utilized for a large variety of tasks, and because we see trusted computing as the most promising approach to realize se-

cure and trusted environments providing these platforms, it seems reasonable to assume that these respective mechanisms will be generally available in the future, independent of specific solutions such as the one at hand.

## 6. CONCLUSION & FURTHER WORK

We have developed an agent-based approach for privacy-preserving Recommender Systems. By utilizing fundamental features of agents such as autonomy, adaptability and the ability to communicate, by extending the capabilities of agent platform managers regarding control of agent communication, by providing a privacy-preserving protocol for information filtering processes, and by utilizing suitable filtering techniques we have been able to realize an approach which actually preserves privacy in Information Filtering architectures in a multilateral way. As a proof of concept, we have used the approach within an application supporting users in planning entertainment-related activities.

We envision various areas of future work: To achieve complete user privacy, the protocol should be extended in order to keep the recommendations themselves private as well. Generally, the feedback we have obtained from users of the Smart Event Assistant indicates that most users are indifferent to privacy in the context of entertainment-related personal information. Therefore, we intend to utilize the approach to realize a Recommender System in a more privacy-sensitive domain, such as health or finance, which would enable us to better evaluate user acceptance. Finally, users should have actual control over and access to their personal agents, a feature that is not implemented in the Smart Event Assistant yet.

## 7. ACKNOWLEDGMENTS

We would like to thank our colleagues Andreas Rieger and Nicolas Braun, who co-developed the Smart Event Assistant. The Smart Event Assistant 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.

## 8. REFERENCES

[1] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Hippocratic databases. In *28th Int'l Conf. on Very Large Databases (VLDB), Hong Kong*, 2002.

[2] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proc. of the ACM SIGMOD Conference on Management of Data*, pages 439–450. ACM Press, May 2000.

[3] E. Aïmeur, G. Brassard, J. M. Fernandez, and F. S. M. Onana. Privacy-preserving demographic filtering. In *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing*, pages 872–878, New York, NY, USA, 2006. ACM Press.

[4] M. Bawa, R. B. Jr., and R. Agrawal. Privacy-preserving indexing of documents on the network. In *Proc. of the VLDB, 2003*, 2003.

[5] R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.

[6] J. Canny. Collaborative filtering with privacy. In *IEEE Symposium on Security and Privacy*, pages 45–57, 2002.

[7] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *IEEE Symposium on Foundations of Computer Science*, pages 41–50, 1995.

[8] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Trans. Inf. Syst.*, 22(1):143–177, 2004.

[9] L. Foner. *Political artifacts and personal privacy: The yenta multi-agent distributed matchmaking system*. PhD thesis, MIT, 1999.

[10] Foundation for Intelligent Physical Agents. Fipa abstract architecture specification, version l, 2002.

[11] S. Fricke, K. Bsufka, J. Keiser, T. Schmidt, R. Sesseler, and S. Albayrak. Agent-based telematic services and telecom applications. *Communications of the ACM*, 44(4), April 2001.

[12] T. Garfinkel, M. Rosenblum, and D. Boneh. Flexible OS support and applications for trusted computing. In *Proceedings of HotOS-IX*, May 2003.

[13] T. Geissler and O. Kroll-Peters. Applying security standards to multi agent systems. In *AAMAS 2004 Workshop: Safety & Security in Multiagent Systems*, 2004.

[14] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proc. of STOC '87*, pages 218–229, New York, NY, 1987. ACM Press.

[15] S. Jha, L. Kruger, and P. McDaniel. Privacy preserving clustering. In *ESORICS 2005*, volume 3679 of *LNCS*. Springer, 2005.

[16] G. Karjoth, M. Schunter, and M. Waidner. The platform for enterprise privacy practices: Privacy-enabled management of customer data. In *PET 2002*, volume 2482 of *LNCS*. Springer, 2003.

[17] H. Link, J. Saia, T. Lane, and R. A. LaViolette. The impact of social networks on multi-agent recommender systems. In *Proc. of the Workshop on Cooperative Multi-Agent Learning (ECML/PKDD '05)*, 2005.

[18] B. N. Miller, J. A. Konstan, and J. Riedl. PocketLens: Toward a personal recommender system. *ACM Trans. Inf. Syst.*, 22(3):437–476, 2004.

[19] H. Polat and W. Du. SVD-based collaborative filtering with privacy. In *Proc. of SAC '05*, pages 791–795, New York, NY, USA, 2005. ACM Press.

[20] T. Schmidt. *Advanced Security Infrastructure for Multi-Agent-Applications in the Telematic Area*. PhD thesis, Technische Universität Berlin, 2002.

[21] G. J. Simmons. The prisoners' problem and the subliminal channel. In D. Chaum, editor, *Proc. of Crypto '83*, pages 51–67. Plenum Press, 1884.

[22] M. Teltzrow and A. Kobsa. Impacts of user privacy preferences on personalized systems: a comparative study. In *Designing personalized user experiences in eCommerce*, pages 315–332. Kluwer Academic Publishers, Norwell, MA, USA, 2004.

[23] J. Wohltorf, R. Cissée, and A. Rieger. Berlintainment: An agent-based context-aware entertainment planning system. *IEEE Communications Magazine*, 43(6), 2005.

[24] M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10(2):115–152, 1995.

[25] A. Yao. Protocols for secure computation. In *Proc. of IEEE FOGS '82*, pages 160–164, 1982.