User identification for cross-system personalisation

User Identification for Cross-System Personalisation

Francesca Carmagnola and Federica Cena

Department of Computer Science, University of Turin
Corso Svizzera 185, Torino, Italy
{carmagnola,cena}@di.unito.it

Abstract. Currently, there is an increasing demand for user-adaptive systems for various purposes in many different domains. Typically, personalisation in information systems occurs separately within each system. The recent trends in user modeling rely on cross-system personalisation, i.e., the opportunity to share information across multiple information systems in order to improve user adaptation. Cooperation among systems in order to exchange user model knowledge is a complex task. This paper addresses a key challenge for cross-system personalization which is often taken as a starting assumption, i.e., user identification.
In this paper, we describe the conceptualization and implementation of a framework that provides a common base for user identification for cross-system personalisation among web-based user-adaptive systems. However, the framework can be easily adopted in different working environments and for different purposes.
The framework represents a hybrid approach which draws parallels both from centralized and decentralized solutions for user modeling. To perform user identification, we propose to exploit a set of identification properties that are combined using our identification algorithm

Keywords: Intelligent Information Systems, User Modeling, Cross-System Personalisation, Personalised Systems

1 Introduction

Nowadays personalisation is regarded as crucial in many areas, such as e-commerce, e-learning, tourism and cultural heritage, digital libraries, travel planning, interaction in instrumented environments, etc. As a consequence, a large number of user-adaptive systems have been developed. In a user-adaptive system, the available personal information about a user (preferences and system's assumptions about the current user's state) are stored in a user model (UM). Applications can build a model of the user in different ways, e.g. exploiting the information the user has entered upon registration, clustering users in stereotype categories, tracking user behavior and reasoning about his/her interests and competencies, learning from interaction with the user, and allowing the user to inspect and change his/her model [10]. The model of the individual user is often conceived

as an overlay of the domain model. This allows the user's current state with respect to domain concepts to be recorded. On the base of such user information, the system uses reasoning strategies to derive further knowledge about the user from the user information, to update the model, and to choose adaptation strategies and techniques.
The proliferation of user-adaptive systems, especially on the Web, represents a chance for users to interact with many of them. This implies the possibility that a lot of data on a specific user (e.g. characteristics, preferences, knowledge, interests, goals, activities) are replicated over many applications. Thus, the user profile is inherently distributed.
The major challenge is to develop environments where user-adaptive systems cooperate in a "many-to-many paradigm" to exchange knowledge about the user [38].
This scenario represents cross-system personalization, i.e., the chance of sharing information across multiple information systems in order to improve user adaptation [32],asscheetal:06. Thus, information about a user, which is originally scattered across multiple systems, is combined to obtain maximum leverage and reuse of information. In recent years, researchers have extensively explored cross-system personalization for different purposes and in different domains. Research to date has focused mainly on user model representation, data integration, conflict resolution, application of semantic Web techniques for user modeling, privacy, security and trust, etc.
This paper addresses a key challenge for cross-system personalization that is often taken as a starting assumption, i.e., the identification of users across systems. In user-adaptive systems, identifying a user means to *authenticate* the user when accessing the system, e.g. through cookies, password management, registration forms, etc.
In cross-system personalisation, user identification refers to the process of identifying a subject by its characteristics.
In this paper, we describe the conceptualization and implementation of a framework that provides a common base to perform user identification for cross-system personalisation among user-adaptive systems. More specifically, we focus on those user-adaptive systems provided on-line, known as web-based systems. Notice that the user identification problem is not only limited to user-adaptive systems: it applies to all systems that need to identify users. More specifically, user identification may be relevant and needed in several domains, e.g. finance, telecommunications, manufacturing, etc. Typically, systems working in such domains need to perform user identification to make a financial transaction secure, or to ensure that a business contract is related to the corresponding person. Another domain in which user identification is particularly relevant is healthcare, where user identification is the identification of patients. It is evident that, in this specific domain, correct user identification is crucial.
Therefore, the framework we propose can be adopted easily in different working environments and for different purposes.
In Section 2 we present an overview of the cross-system personalisation issues.

Section 3 presents an application scenario showing the relevance for user identification for cross-system personalisation. The framework is presented in Section 4, which includes the conceptual description of the approach (Section 4.1, 4.2, and 4.3) and a description of the identification algorithm (Section 4.4). Some implementation remarks are made in Section 5; while Section 6 describes an instantiation for the case study presented in Section 3.
In Section 7, an evaluation of the algorithm is reported. Section 8 illustrates the privacy issue, while Section 9 offers an overview of the related work which inspired our research. Finally, Section 10 concludes the paper providing some future directions and open issues of the current project.

2 From User Models to Interoperable User Models

Recently, user-adaptive systems have been widely deployed in different areas and for different tasks.[1] Although these systems are heterogeneous, many of them concern similar domains. The knowledge about a user represented in the applications working in similar domains shows a partial overlap. This is because the knowledge represented in a system is strictly related to the domain in which the system acts. For this reason, even if the user model of a system is defined independently by the other systems, applications in the same domain may be similar in the information they maintain.
The great proliferation of user-adaptive systems means that a user is likely to interact with many such systems. Since each of them builds its own model of the user [33], the result is that the user profile is scattered across multiple systems. Thus, there is no common "memory" of all user activities, preferences and characteristics, which would allow effective adaptation to the user's current state. One way of achieving a rather complete picture of a user's experience is to allow systems to share user data. This is especially valuable when a user's interaction with an information system is part of a larger task that covers several interactions with different systems [20], [34]. This is cross-system personalization: the sharing of information across multiple systems to improve user adaptation [32], [40].
There are many advantages to cross-system personalization.[2]
Kobsa et al. [30] seek to use cross-system personalisation to to speed up the phase of the user model initialization. Vassileva states that cross-system personalisation relieves users from the pain of training new systems [42]. In fact, every time users interact with a system for the first time, they have to provide data they may have already provided to other applications. On the contrary, users typically do not appreciate wasting time to explicitly fill in their model for the applications they use. More generally, Berkovsky assumes that cross-system personalisation enhances the user model knowledge stored in a system (we refer this to *qualitative* improvement) [4]. Carmagnola et al. [12] and Schwartz et

[1] For an extensive survey on adaptive systems, the interested reader can refer to Kobsa [28].
[2] For a more detailed overview, see Carmagnola et al. [12].

al. [25] make reference to an increased amount of information about users, since there is the chance to benefit from the efforts led by other modelers and systems. Cross-system personalisation gives the model increased coverage, because more aspects can be covered by the aggregated user model, including the user model features that one system could not acquire by itself (we should refer to this as *quantitative* improvement).[3]
To summarize, the additional knowledge about the user, which comes from cross-system personalisation, allows systems to obtain a deeper understanding of the user, leading to more appropriate adaptation.
To benefit from distributed information, a system must be able to access and interpret information derived from multiple heterogeneous sources and to integrate this information into a model of proper granularity [42]. This is the so-called interoperability. Interoperability is defined as the "ability of two or more systems or components to exchange information and to use the information that has been exchanged".[4] Interoperability has been variously addressed in the literature. Greaves relates the concept of interoperability with two main issues [22].

1. **Syntactic interoperability**: the capability of different information systems to interpret the syntax of the delivered data in the same way. Syntactic interoperability can be achieved, for instance, through the definition of a common interchange formalism, or through APIs [3]. It is also referred as *language interoperability.*
2. **Semantic interoperability**: the possibility to bridge differences between information systems on the meaning level. It refers to the "capability for different systems to entail a co-ordination of meaning on the basis of shared, pre-established and negotiated meaning of terms and expressions"[43]. It is often achieved through multiple controlled vocabularies. It is also referred as *logical interoperability.*

The fulfillment of both syntactic and semantic interoperability is the *sine qua non* condition to ensure cross-system personalisation.
Reaching interoperability in an open environment like the Web is, in general, extremely difficult and requires a very high degree of alignment between the applications on syntax, structure [15] and semantics [1], [2]. This requires a lot of extra efforts in the design of such interoperable applications. For this reason, for a long time it has been very difficult or even impossible to share and exchange user profile data.
More recently, suppliers and consumers of user profiles have shown an increased awareness of the need for standards for representing and exchanging user profile data. However, the heterogeneity of user-adaptive applications in a open environment as the Web makes it impossible to easily create a unified user profile

[3] Notice that qualitative and quantitative improvements are connected. As stated by Vassileva "the more information is available, the more adequate the user model and consequently the adaptation process will be" [42].
[4] IEEE Standard Computer Dictionary: A compilation of the IEEE Standard Computer Glossaries, 1990.

infrastructure and especially, to enforce applications to use a prescribed syntax, structure and semantics [25]. However, the technological advances of the last few years, in particular in the context of the Semantic Web, provide languages and technologies useful for a basic interoperability of data since it deals with the development of a "common way to represent information in a sharable, expressive, semantic and machine-processable way. It provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries."[5] Semantic Web technologies allow data be structured in a syntax-independent way and offer mechanisms to define relations between those data structures, e.g. by ontologies.[6] Furthermore it provides useful instruments for the representation of semantics of the data structured in ontologies (e.g. semantic formalism like RDF[7], RDF Schema[8] and OWL[9]). In summary, on the one hand the Semantic Web provides a standardized structure to enable syntactic interoperability; on the other hand, it offers the means to achieve semantic interoperability.

Cross-system personalisation has been addressed, according with two major directions : centralized and decentralized approach.
According to the centralized approach, the knowledge-base storing the user model is separated from the internal application logic and it is stored as a central repository. The knowledge about the user is maintained by a server (User Model Server) that hosts the information and it is delivered to different applications through a flexible client-server architecture. Such information is made available for more than one application at the same time and user information acquired by one application can be employed by other applications and vice versa. Domain-independent and application-independent user's information can be stored with low redundancy and can be easily available to all systems accessing the server. Such a central repository of user information is in contrast with the redundant modeling of user characteristics within today's applications (including those on the Web). Using a central repository to store user data can significantly contribute to the definition of a more consistent environment that includes different user-adaptive applications. For mobile applications, this enables systems on devices with limited memory and computing power to access user models. Furthermore, it allows different applications to access the same knowledge and to make adaptation in a consistent way. User modeling servers seem to provide promising advantages for the deployment of user modeling systems. Despite the advantages of centralized user modeling, we believe that current and future usage scenarios for computing devices will require a more flexible and sophisticated architecture.

[5] http://www.w3.org/2001/sw/
[6] An ontology is a specification of a conceptualization [23]. In other words, an ontology is a data model that represents some knowledge and is used to reason about the objects in that domain and the relations between them.
[7] http://www.w3.org/RDF/
[8] http://www.w3.org/TR/rdf-schema/
[9] http:// www.w3.org/2004/OWL/

In fact, while integrated systems can rely on centralized user model servers, applications working in the market can not, since the centralized UM model is too restrictive. It imposes a set of user features that should be represented and a non-negotiable format of representation, APIs, and protocol. It also introduces a central point of failure, and while reliability can be increased by introducing mirrors or distributing the load on several servers, the problem of synchronization and coordination of the mirror servers increases the cost. Furthermore, there is the rise of ubiquitous computing and intelligent environments, where more and more interactions take place between humans and different stationary, mobile or web-connected IT-systems in daily life [46].
The ubiquitous environment is populated by next-generation mobile, distributed and autonomous applications, thus they are "accessible from everywhere" (e.g. PC, portable computers, mobile devices). Managing interactions in such environments populated by many systems (multi-agents) that interact with different users (multi-users) in different contexts (multi-context) through different devices is a particularly complex task.
This lead to the second direction to address interoperability among user-adaptive system, that is a decentralized approach for user modeling. In decentralized settings, each system maintains a user model for its own purposes of adaptation. However, systems can interact with other systems to exchange user information, and to be more up-to-date. In this way, through communication, each system can benefit from the UM efforts done by many modelers. Communication among the different applications in order to exchange user data may happen through several techniques; e.g. web services, agent-based technologies, dialogue, etc. The central user model, in which applications store their user data in a single, centralized repository, contrasts with the decentralized approach, where each application has its own UM. This supports the creation of a community of adaptive applications sharing user knowledge. The distributed approach is also more flexible in managing privacy issues than centralized approach, since each system may define which parts of user model to be shared and which ones to keep private.
In general the cooperation among user-adaptive systems is as complex process, composed of three tasks.

1. **Identification of the user whose data are exchanged**. How can the identification of the user whose data and information are shared across systems be achieved?
2. **Exchange of knowledge about the user**. How should knowledge about the user be represented to support interoperability? How can Semantic Web techniques be employed to manage any problem related to syntactic and semantic heterogeneity?
3. **Evaluation of the exchanged knowledge and data integration**. How can an application be sure that the user model data provided by another application are reliable? How can it be sure that the applications themselves are reliable? [11]

Comparing such subtasks with the research described in the literature, we noticed that the researchers' attention has almost exclusively focused on the sub-

task 2 (exchanging knowledge about users) which represents the core part of the cooperation process. On the contrary, this paper focuses on task 1 (user identification), which is usually taken as starting assumption. The management of user identification issue is crucial for an effective fulfillment of interoperability tasks. Assume two systems aim at sharing the user feature "gender". Even before reaching an agreement on the meaning of such a feature and before exchanging the value of this feature, systems need to agree on the identity of the user whose "gender" is being stored. In the next section we present an application scenario to show the relevance of the user identification for cross-system personalisation.

3 Scenario

As an example of the user identification problem among user-adaptive systems, let us consider the following scenario. We consider iCITY[10], an adaptive social mobile guide that provides personalized information about cultural events of the city of Torino [13]. Suppose a scenario showing iCITY wishes to decide whether to show an advertisement for an incoming rock concert to a novice user called Carlo. According to the internal working of iCITY, an event is recommended considering the level of user's interest (stored in the user model) in the category the event belongs to. In this specific case, iCITY recommends to Carlo an event related to rock music if Carlo's level of interest in rock music is almost "medium-high". Currently, the value for such a feature is computed directly by observing the user's actions on the system, e.g. considering how many times a user has selected links about rock events or asked for information about them. Therefore, a correct assumption about the user's interest in rock music can be estimated only after a reasonably high number of user's interactions with the system. This situation addresses the well known "cold start problem", which occurs when, at the beginning of the user's interactions, the user model stores little data about the user [35].
In a cross-system personalisation context, iCITY has the opportunity to cooperate with the other systems used by Carlo in order to gather a more reliable value for his interest in rock music.
The search for such a user model feature presents a twofold challenge. Before searching the specific value for the such a feature, iCITY needs to understand if Carlo interacts (or interacted) with other systems which store some knowledge about him - this is the *identification problem*, and discover, among them, those which store the required user feature, i.e., Carlo's "interest in rock music" (which is outside the scope of this paper).
The former is a challenging issue. Suppose Carlo is used to interact with another user-adaptive system, UbiquiTO. Assume Carlo uses different usernames to log into iCITY and UbiquiTO (he uses "Charlie" as a username in iCITY, and "Carletto" as a username in UbiquiTO). Therefore, in absence of a unique user identifier among iCITY and UbiquiTO for Carlo, how could iCITY recognize that the user with the username *Charlie* is the same user with the username

[10] http://icity.di.unito.it/dsa-dev/

Carletto in UbiquiTO? Moreover, what could happen if iCITY mistook Carlo for another user who exploits the username *Charlie* in UbiquiTO? In this last case, iCITY will gather in Carlo's user model the value for interest in rock music of another user. This is a problem since two different users have probably different features and preferences, and bad user model data will result in a bad adaption. The result is a decrease of the level of trust of the user in the system.
This is the foremost reason why we state that the problem of user identification in cross-system personalisation is extremely important and deservers a proper analysis and capable solutions.

4 Framework Conceptual Model

Identification can be defined as the process of using claimed or observed attributes of an entity to deduce who the entity is. If the entity is a user, as in cross-system personalisation context, the identification becomes the process of using user's attributes to deduce who the user is. In an environment of cross-system personalisation, user identification is the process that gives an understanding of whether users interacting with different user-adaptive systems are the same or different entities.
In a system that profiles users, each user is represented as an instance of his/her user model. Every instance of the user model is formed by a collection "property, value" (p, v) pairs. The user model of a user i can be represented as:

$$\text{UM(i)} = [(\text{p1, v1}), (\text{p2, v2}), (\text{p3, v3}), (\text{p4, v4}), ..., (\text{pn, vn})]$$

where (*p1, p2, p3, p4*) are the properties used to describe and profile the user, and (*v1, v2, v3, v4*) are the values that such properties assume. For example,

$$\text{UM(User\_x)} = [(\text{first\_name, Carlo}), (\text{second\_name, Bellini}), (\text{birth\_date, 19740130}), (\text{birth\_city, Torino})]$$

In a cross-system personalisation context, a user is considered the same entity which interacts with more systems whether the value (v) of property (p) in UM(i) of systems A matches the value of the same property (p) in UM(i) of systems B. In other words, a user is identified as "the same user" on two systems if the properties which are in common among them assume the same values.
However, it is quite unlikely that an entity will have common properties on heterogeneous systems. This is due to the fact that all the systems are independent and they all represent the concept "user" with different properties in their user model. This can be called "semantic heterogeneity". Moreover, systems can represent their users with the same user model properties but exploiting a different syntax. This can be addressed as "syntactic heterogeneity".
To reduce the semantic heterogeneity among the user model properties in different systems, we focus on those properties which can be more easily used for user identification. Our analysis of these properties is described in Section 4.1.

To overcome the syntactic heterogeneity of the user model properties assumed as adapted for user identification, we describe their syntax and format by means of proper meta-data. Such a description is made available to the designers of user-adaptive systems (Section 4.2).

4.1 Identification Properties

To reduce the semantic heterogeneity among the user model properties exploited by the different systems, it is necessary to find which properties are most useful for user identification. To this purpose, we analyzed which properties can best be used for user identification. Such properties should concern the user identity. Moreover, they should be context-, domain- and application-independent so that they are likely to be used in many existing user-adaptive systems [39].
To discover the properties suitable for user identification, we have been inspired by the user properties defined by the existing standards for personal data interchange.
The most important standard we considered is vCard[11], a file format standard proposed in 1995 by the Versit consortium for personal data exchange. vCards include information such as name, addresses (business, home, mailing, parcel), telephone numbers (home, business, fax, pager, cellular, ISDN, voice, data, video), email addresses and Internet URLs. They are often attached to e-mail messages, but can be exchanged in multiple ways on the Web.
Besides the analysis of vCard data, we also analysed the user data collected by other user-adaptive systems on the web. Typically systems identify their users exploiting the data gathered during the first registration of the user in the system, when the user provides the system with some basic data. We considered the data required in the registration forms of 25 user-adaptive systems. In order to collect those user model data which can be assumed to be application- and domain-independent, we analyzed systems working in different domains and with different purposes: from the tourist domain to e-commerce, from cultural heritage to e-learning.
Table 1 reports the result of this analysis. The user model properties we collected are sorted according to the number of their occurrences. As shown, *username*

	Username	**First Name**	**Last Name**	**Email**	**Birth Date**	**Birth City**
Occurences	22	19	17	16	8	6
Percentage	88%	76%	68%	64%	32%	24%

Table 1. Registration form data for 25 user-adaptive systems

was required most often, followed by *first name*, *last name* and *email*. *Birth date* and *birth city* have been required, respectively, in the 32% and 24% of the cases.

[11] http://www.imc.org/pdi/

Because these features are the most application-independent ones, they are assumed to be well adopted for the sake of user identification. In the rest of the paper these properties will be addressed as "identification properties".
Notice that the other user model data collected in the registration forms of the systems we analyzed are not referred as identification properties because are used in less than 8% of systems. As a consequence they are considered too application- and task-dependent, so not suitable for user identification.
As a final consideration, we should notice that there already exist some user properties that unambiguously identify a specific user. Such properties, like the national identifiers, are typically used in several domains, e.g. administration, government, police, e-commerce. Although they univocally address a specific user, they do not seem suitable for user identification since they are not domain independent. For instance, in e-commerce, users can be identified through the VAT number, while this is not used in different domains. Moreover, each country employs different identifiers, e.g. the identification number issued by the tax authority, the social security number, etc. Furthermore, users are not willing to provide this kind of information because of privacy concerns (see Section 8).
Such a variety of possibilities, led us to discard these identifiers from the identification properties.

4.2 The Identification Meta-Information Schema

Although we outlined some common properties typically used to identify users across systems, some syntactic and structural differences still remain. Systems can represent their user model properties in any format.
To give uniform syntax to identification properties and their values, we describe their syntax and format in a schema, which we call the **Identification Meta-Information schema**. The schema is a text file where the identification properties pointed out in Section 4.1 are described using the Dublin Core Qualifiers. The Dublin Core Meta Data Initiative states the need of a "common set of universal properties for describing any type of resource... Such universal collection lets knowledge [be] combined and shared across different entities".[12]
The Dublin Core qualifiers allow any kind of resource to be described using meta-data, also known as "universal properties". Designers can access the Identification Meta-Information schema through a Web page.[13]
In the following, an extract of the Meta-Information schema for the identification property "Birth Date" is reported.

Birth Date attribute
description *"The birth date of the user."*
label *"Birth Date"*
syntax *string*
schemes

[12] http://dublincore.org
[13] http://www.di.unito.it/∼carmagno/identification/imi.txt

"ISO8601" type

Notice that the qualifier "schemes" expresses the format that the value of the property may assume. In the example above, the wording *ISO8601* for the qualifier "schemes" indicates that the syntax of the instance of the identification property *Birth Date* should be expressed in compliance with the ISO8601 standard, i.e., yy/mm/dd.
If the information collected in the registration forms of the user-adaptive systems are in compliance with the Identification Meta-Information schema, the identification properties assume a data uniformity across systems.

The Identification Meta-Information also has the aim of making designers aware of the identification properties we defined. Through the schema, designers can check whether there is a semantic correspondence between the information collected in the registration form of their systems and those stored in the schema. We have identified a set of properties that are frequently used to identify users of user-adaptive systems and given a uniform syntax for them. When a system needs to identify a user, it must find out if they have interacted with any other systems by checking for matches between the values of properties shared with other systems. When a system needs to identify user, it should realize if there exists any other available systems the user interacted with. To this purpose, it checks the occurrence of any match among the values of the properties which are in common among systems.
The first task is described in the following section, while the second one is presented in Section 4.4.

4.3 The Identification Registry and the Identification System-Set

To start user identification, a system must find out what identification properties are used by other systems. Once it has found other systems using at least some of the required identification properties, a system can start to compare the values of these properties. To achieve this, systems advertise the properties from the Identification Meta-Information schema that they use in a registry that we call the "Identification Registry". The registry is accessibly only to authorised systems and its privacy policies are explained in Section 8.

The registry stores the following information about each system:

- the name of the system,
- the identification properties the system uses to identify users,
- the Web URI that stores the values of such properties.

Each requester[14] queries the registry to discover the systems use the same identification properties (or a sub-set of them). The requester puts the result of the query in the **Identification-Systems Set** (**ISS**), containing the list of the suppliers[15] that use the same identification properties. Each requester maintains

[14] We refer to the system that wishes to identify a specific user as the "requester".
[15] We refer to each system included in the Identification-System Set as the "supplier".

its own specific ISS. Notice that this does not imply that the systems in the ISS interacts with the user the requester wants to identify. This simply means that those systems exploit some identification properties which are in common with the requester, i.e., whose values can be compared.
Once the ISS has been created, the requester does not need to access the Identification Registry every time it has to identify a user. On the contrary it can directly query its local ISS. The requester will only access the external Identification Registry to obtain an up-to-date list of the systems it contains.
As said, in a cross-system personalisation context, a user is identified as "the same user" among system A and system B if the properties which are in common among A and B assume the same values. Using the ISS speeds user identification as it avoids consulting systems that do not use user identification properties that cannot be compared.

4.4 The Comparison of Values

When the requester needs to search for the other available user-adaptive systems storing some knowledge about a specific user, it queries its internal ISS. As a result, it gets the list of the identification properties exploited by each system reported in the ISS and the Web URI where such properties are instantiated with values. In our approach, the values of the identification properties are not present in the registry, but in what we call the **Public User Model (PUM)** repository. Each system maintains its own PUM repository including the *values* of the identification properties for all its users. Notice that systems are not compelled to make their entire user model available in the Public User Model. On the contrary, the Public User Model includes only the portion of the user model storing the identification properties[16] and, among them, only those ones users agree to make public.[17]
The requester needs to verify if a specific user interacts with any suppliers included in its ISS. To this purpose, the requester accesses the Public User Model repository of each supplier.
The sequence diagram for the user identification task from the requester's perspective is represented in Figure 1.

To compare the values of the identification properties that each supplier shares with the receiver, we develop a computational mechanism which we call the "Identification algorithm", described in the following.

[16] Notice that systems can internally represent the knowledge about the users as they prefer. In our approach, systems are required to express semantically only those parts of knowledge they wish to declare as public. This is different from other approaches [14] in which systems are required to semantically represent all their user, domain, and context data.

[17] We enable users to scrutinize their user model and associate their individual privacy constraints to each part of the model (if they can be released, to which, for how long, if they can be used, for which purposes) [26].

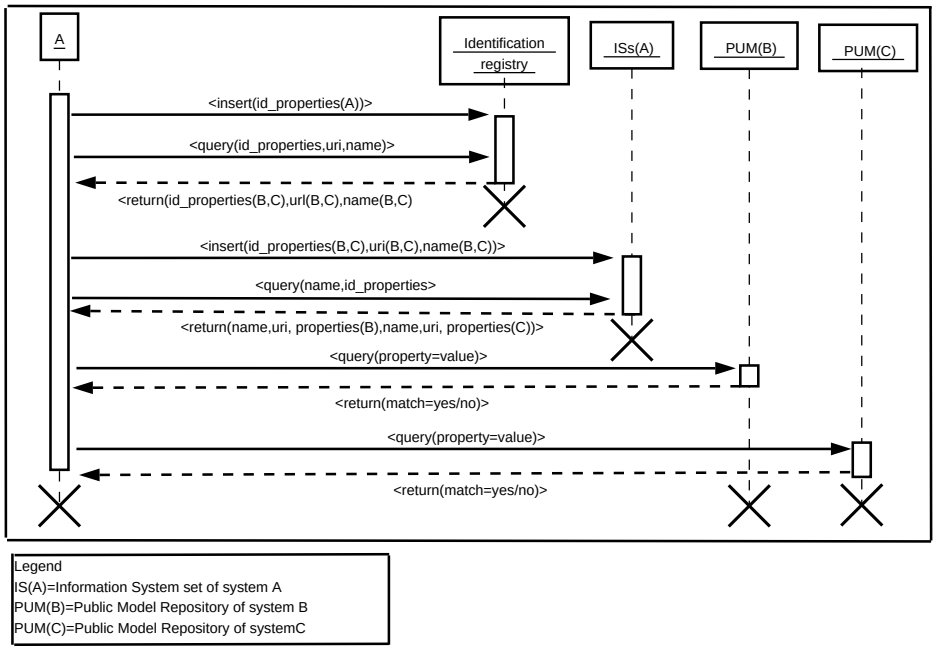

Fig. 1. Sequence diagram for the user identification task from the requester's perspective

Identification Algorithm

User identification is performed by comparing the values of the requester's identification properties with those stored by the suppliers listed in the ISS. If one of these matches succeeds, the user is said to be "identified".

But is a minimum number of matching values required? Are some identification properties more relevant than others for the user identification process?

To answer such questions, we analyzed the identification properties collected in the analysis described in Section 4.1. More specifically, each identification property is characterized by a level of **univocity** (**UL**)[18], which represents how much a feature may assume the same value across different users. This property is directly related to the capability of identifying the user.

However, such a parameter is not sufficient to perform a correct user identification. In fact, even if a user who interacts with the requester (system A) has the same *surname* (a property with a high UL) of a user who interacts with system B, the requester cannot state that the user has the same identity. This is even more problematic in an open environment where many users interact with

[18] Wordnet (http://www.wordnet.org) defines univocity as "unequivocal, univocal, unambiguous (having only one meaning or interpretation and leading to only one conclusion)"

multiple user-adaptive systems. In this context, there is a high probability that different users will have the same value for a specific property. This is the so called "false positive" situation, where different users are wrongly identified as the same entity.
Notice that there is also the opposite situation. A user might be not recognized as the same entity interacting with multiple systems if he/she uses different values for a specific property. This is the so called "false negative" situation.

To identify users for cross-system personalisation, we need to take into consideration all these challenges, deeply investigating the provision of knowledge about the user into a system.
In particular, we should consider that a user can provide different values in different systems for the same data, and that a user can provide systems with false values. Thus, we such considerations, we define two further parameters:

1. **Values per User**(**VpU**), representing the possibility for a feature, to be provided with different values for a unique user to the systems he/she interact with. For example, as seen in the use case, a user can typically use different usernames interacting with different systems. Thus, the feature "username" has a high VpU. Instead, a user uses typically only one value for the last name in the systems he/she interacts with. Thus, "last name" has a low VpU.
   This property is inversely related to the capability of identifying the user: the lower the VpU is, the more effective the identification will be.
2. **Misleading Level** (**ML**), expressing the probability, for a feature, to be provided with a false value. This should be taken into account since users could provide false data [18],[44].
   The ML is directly related to how much the feature regards sensitive data that the user might not like to disclose [17]. These data typically regard private aspects, e.g. gender, age, marital status, etc. For example, [44] demonstrated that 40% of users always provide false demographic data when asked to register for a web site, 7% do this often, 17% sometimes.
   This property is inversely related to the capability of identifying the user: the lower the ML is, the more effective the identification will be.

As shown in Table 2, each identification property is associated with a value (defined in a scale from 0 to 1) which expresses the relation between each feature and the above described parameters. For an overview of the preliminary test that led us to the definition of the values of UL, VpU and ML, see Section 7.

For example, the identification property "email" has the value of 1 in relation to the *univocity*, which indicates that two users with the same email values have for sure the same identity. "Email" has the value of 0.4 in relation to the *VpU*, which indicates that there is medium-low probability (0.4) that a unique user supplies different emails in the different systems she interacts with. Finally, "Email" has the value of 0.4 in relation to the *ML*, which indicates that there is medium-low probability (0.4) that a user provides false email values in different

Feature	email	last name	birth date	username	birth city	first name
UL	1	0.8	0.8	0.8	0.2	0.2
VpU	0.4	0.8	0.8	0.2	0.8	0.8
ML	0.4	0.8	0.6	0.2	0.2	0.2

Table 2. Identification properties in relation with UL, VpU and ML

systems.

UL, VpU and ML are differently relevant for the task of user identification. As said, the univocity level is the most relevant for such a task, since it is directly related to the capability of identifying the user. To define a scale of importance for *VpU* and *ML*, we performed a preliminary test (fully described in Section 7) on 80 users. The aim of the preliminary test was to discover those cases where the occurrences for *VpU* or *ML* were high or medium-high.
For what concerns the VpU, the test showed that users interacting with different systems provided different values for common features in just 37% of cases. As regards the ML, false values have been supplied in 12% of the cases. Such a low percentage with respect to data elicited by Wang and Kobsa (2007) can be justified by the fact that we restricted the analysis to user-adaptive systems [44]. In these systems it is quite rare for the users to use false data, since they are aware that system performance is directly related to the correctness of the data provided to the system.
From such considerations, UL, VpU and ML have been related to a weight, defined in a scale from 0 to 1. The weights are respectively, 0.45; 0.35 and 0.20.
According to such weights, the identification properties are further on combined to derive the Importance Factor (IF) expressing how much each feature is relevant for the process of user identification.
The Importance Factor of each identification property is a weighted average of the above mentioned variables combined with the value reported in Table 2. Because of VpU and ML inversely related to the capability of identifying the user, their are combined to the UL value as *(1-Value)*.
The combination the variables values/weights is obtained through the following formula:

$$IF = Value(UL) * 0.45 + Value(1 - VpU) * 0.35 + Value(1 - ML) * 0.20$$

Table 3 shows the identification properties ranked according with their IFs.

Feature	username	email	birth date	last name	birth city	first name
IF	0.8	0.78	0.51	0.47	0.32	0.32

Table 3. The identification properties and their corresponding IF

We define the following identification algorithm, that requesters can use to query suppliers' PUMs to find matching values of common identification properties and compute the IF of any properties found. A high Importance Factor indicates a higher probability of correct identification. In case the algorithm returns more than one matching value, the IF of all the matching properties are combined according to an additive formula:

*IF = p + (1 - p) * q*

where p and q represent the IF of each identification property whose values match.
For example, in case the algorithm returns that requester and supplier have the same values for the identification properties *username* (p = 0.8) and *birth date* (q =0.51), the application of the additive formula

*0.8 + (1 - 0.8) * 0.51*

results 0.902.

We established an **Importance Factor threshold** (**Thd**) that the IF must exceed for the user to be considered identified. Experiments described in Section 7 indicate that choosing Thd=0.74 results in correct user identification in 91.25% of cases. In the above example, the threshold is overcome so the algorithm returns that the user is identified.[19]
Once the algorithm has identified the user, the (system, user) pair is stored in the requester ISS. This is valuable for the next phase of the cross-system personalisation process, i.e., the user data exchange- since the search for a specific user model data may happen only for those systems which profile the same user. This makes the interoperability process more efficient.
Algorithm 1 describes the identification algorithm in pseudocode.

[19] For a complete argumentation of the results, see Section 7.

Algorithm 1 Identification Algorithm

```
1: access the ISS
2: (systems names, id properties) = getFromPUM()
3: common id props[] = getFromKB()
4: matching values = verify(common id props[], PUM)
5: wait for a result = true
6: i=0
7: matching values = 0
8: for (i=0; i<common id props[].length; i++) do
9:    actual prop = constrain(common id props[i], matching values)
10:   if (matching values) then
11:      FIF = ImportanceFactor(actual prop)
12:      if (FIF >= Threshold) then
13:         return = USER IDENTIFIED
14:      end if
15:   end if
16: end for
17: return = USER NOT IDENTIFIED
```

5 Implementation Remarks

From an architectural perspective, the framework we propose is versatile, since it can be used within many scenarios and by different user-adaptive systems, with little need for configuration.

Both the Identification Registry and the Public User Model repository are conceived as semantic repositories represented through the RDF syntax. We chose the RDF meta-data model because it allows XML tags to be expressed by means of URIs providing a standard for meta-data so that interoperability between applications that exchange machine-understandable information on the Web is made possible. Moreover, RDF is used in a variety of application areas; resource discovery, cataloging, intelligent software agents, etc.

Given the semantic representation of the Identification Registry and the Public User Model repository, systems need to be able to store and retrieve references to such semantic objects. To this purpose, we use Sesame[20], an open source Java framework that can be used as a database server which client applications can access through the HTTP protocol[21]. We hosted the Sesame Java servlet in the

[20] http://www.openrdf.org/

[21] The use of Sesame is suggested but not compulsory for the requestors and suppliers systems. Systems might use other semantic environments like Jena. We advice Sesame because it supports the storage, inferencing and querying of RDF data, which is the syntax we exploit to represent the knowledge in the framework. The main advantage of Sesame is that it can be deployed on top of a variety of storage systems, e.g., relational databases, filesystems, keyword indexers, etc. Moreover, it presents many tools to developers in order to leverage the power of RDF and RDF Schema, such as a flexible access API, which supports both local and remote access, and several query languages, like SeRQL and SPARQL

Apache Tomcat environment. The core module in the Sesame framework is the "repository", a Java object that stores RDF statements. The Identification Registry and the Public User Model repository are defined as an in-memory Sesame repository. To ensure the access to the Identification registry and the Public User Model repository, they are both required to be stored on remote web servers. To make systems connect to the servers, retrieve and manipulate all the data, Sesame offers APIs, which abstract from the storage format used and provide reasoning support. In particular, the APIs contain a set of available procedures that programmers can use to manage both the task of inserting RDF statements into systems repositories and the task of querying their internal repositories and those of the suppliers.

Access to the repositories hosted by Sesame is query driven: a SELECT-FROM clause is sent via HTTP to remote repositories and requires a short time answer. The queries are performed through SeRQL (Sesame RDF Query Language), a RDF/RDFS expressive query language, with many features and useful constructs [9]. Notice that the Java APIs that Sesame offers can be wrapped on different communication protocol, according to the implementation choices of every specific system, e.g. via peer-to-peer communication, using agent-based techniques, or using a constraint-based approach. We used the Sesame API to provide systems with the suitable procedures to insert RDF statements in the Identification Registry and in each system's Public User Model repository. Furthermore, we included our identification algorithm in the API used to query system's Public User Model repository, in order to exploit query results for the user identification.
Figure 2 shows the main components of the framework architecture.

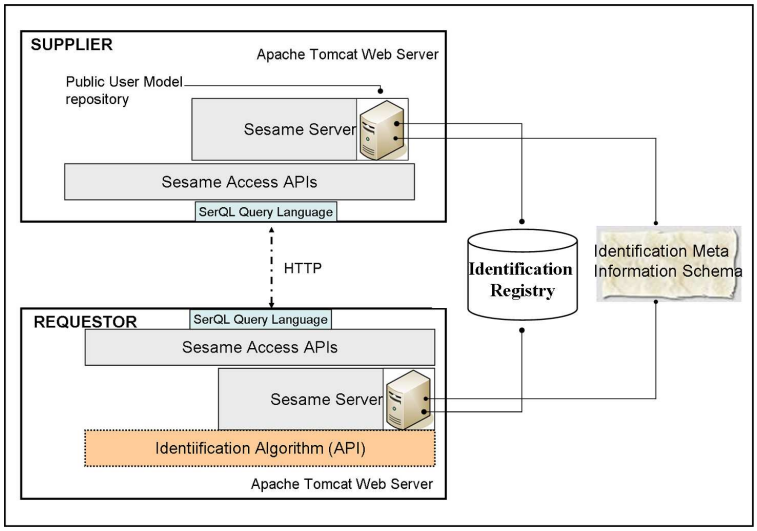

Fig. 2. Architecture of the framework

6 Example Implementation

In this section we describe the use of the framework for the case study presented in Section 3. In this use case, a user-adaptive system, iCITY, acts as requester. iCITY needs to know whether a user named Carlo interacts with other available user-adaptive systems. The identification algorithm will determine that Carlo has interacted with another system named UbiquiTO.

6.1 The Identification Registry — Insertion

Both iCITY and UbiquiTO (together with the other user-adaptive systems in the framework) expose in the Identification registry the identification properties they use. For each system, it reports the list of the identification properties, the name (*System_Name*) and the URI of the Public User Model repository where the values of the identification properties are stored (*PumUri*).
In the following the portion of the Identification Registry concerning iCITY, UbiquiTO and a third system named MASTROcarONTE [16] is presented.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
xmlns:reg="http://carmagno/reg#"
```

```
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

<rdf:Description rdf:about="http://carmagno/icity.rdf#id">
<reg:Username>Username</reg:Username>
<reg:Birth_date>Birth_date</reg:Birth_date>
<reg:Last_name>Last_name</reg:Last_name>
<reg:Birth_city>Birth_city</reg:Birth_city>
<reg:System_Name>iCITY</reg:System_Name>
<reg:PumUri>http://carmagno/icity.rdf</reg:PumUri>
</rdf:Description>

<rdf:Description rdf:about="http://carmagno/ubiquito.rdf#id">
<reg:Username>Username</reg:Username>
<reg:Email>Email</reg:Email>
<reg:Birth_date>Birth_date</reg:Birth_date>
<reg:System_Name>UbiquiTO</reg:System_Name>
<reg:PumUri>http://carmagno/ubiquito.rdf</reg:PumUri>
</rdf:Description>

<rdf:Description rdf:about="http://carmagno/mastrocaronte.rdf#id">
<reg:Email>Email</reg:Email>
<reg:First_name>First_name</reg:First_name>
<reg:Last_name>Last_name</reg:Last_name>
<reg:System_Name>MASTROcarONTE</reg:System_Name>
<reg:PumUri>http://carmagno/mastrovalues.rdf</reg:PumUri>
</rdf:Description>

</rdf:RDF>
```

As reported in the Identification registry, iCITY includes the identification properties: *username*, *last name*, *birth date* and *birth city*. UbiquiTO includes *username*, *birth date*, *email*. Finally, MASTROcarONTE includes *first name*, *last name* and *email*.
In order to support designers in uploading the RDF statements with such information, we exploit the Sesame Java APIs (see Section 5). Notice that this Java API simply requires designers to indicate the global web location where the Identification registry is hosted.

6.2 The Identification Registry — Query

As described in Section 4.4, a requester queries the Identification registry to verify if there is any system using the same identification properties as that requester.
The query performed by iCITY over the Identification Registry the following.

```
‘‘SELECT System_Name, PumUri, Username, Birth_city, Birth_date, Last_name’’+
  FROM {}
  ‘‘reg:System_Name{System_Name};’’+
  ‘‘reg:Pum_Uri{Pum_Uri};’’+
  ‘‘[reg:Birth_date{Birth_date}];’’+
  ‘‘[reg:Birth_city{Birth_city}];’’+
  ‘‘[reg:Username{Username}];’’+
  ‘‘[reg:Last_name{Last_name}]’’+
 ‘‘WHERE Birth_date = \‘‘‘‘+ BIRTH_DATE + ’\’’+
 ‘‘AND Birth_city = \‘‘‘‘+ BIRTH_CITY + ’\’’+
 ‘‘AND Username = \‘‘‘‘+ USERNAME + ’\’’+
 ‘‘AND Birth_place = \‘‘‘‘+ BIRTH_PLACE + ’\’’+
 ‘‘USING NAMESPACE reg = <http://carmagno/reg#>’
```

From the Identification registry, iCITY receives as answer:

- system name: UbiquiTO
- matching identification properties: Username, Birth Date
- Public User Model Repository URI: http://www.di.unito.it/˜carmagno/ubiquitovalue.rdf-xml

- system name: MASTROcarONTE
- matching identification properties: Last Name
- Public User Model Repository URI: http://www.di.unito.it/˜carmagno/mastrocarontevalue.rdf-xml

6.3 The Identification-Systems Set

This result constitutes the Identification-Systems set of iCITY, locally stored by iCITY.
When iCITY needs to verify if Carlo interacts with other user-adaptive systems, it extracts from its ISS the URI of the possible supplier systems (UbiquiTO and MASTROcarONTE). iCITY then accesses the Public User Model repositories of both of them to compare the values of the common identification properties. The following is a portion of the Public User Model repository of MASTROcarONTE.

```
<rdf:Description rdf:about="http://carmagno/mastrocaronte.rdf#id">
<reg:First_name>Carlo</reg:First_name>
<reg:Last_name>Bellini</reg:Last_name>
<reg:Email>bellini@yahoo.it</reg:Email>
<reg:System_Name>MASTROcarONTE</reg:System_Name>
<reg:PumUri>http://carmagno/mastrovalues.rdf</reg:PumUri>
</rdf:Description>
</rdf:RDF>
```

6.4 The Identification Algorithm

We describe here the execution of the identification algorithm by the systems in the case study.
First of all, iCITY accesses the Public User Model repository of UbiquiTO. It starts querying for the value of the common property "Birth Date=19740130". It finds a positive match, with the corresponding IF value of 0.51.
Since the IF value is lower than the threshold, it queries for the value of the other common identification property, "username=Charlie". The values match. The combination of IF(Birth Date) with IF(Username) results a value of 0.902. The IF is higher than the threshold, thus the algorithm returns "identified" and terminates.
The process is performed for the other supplier included in the ISS of iCITY, i.e., MASTROcarONTE. iCITY queries for the value of the common property "LastName" ("Last Name=Bellini"). The values match, and the corresponding IF value is 0.47. Since the IF value is lower than the threshold, iCITY verifies whether there are any other identification properties in common with MASTROcarONTE. There are no more common properties so the algorithm returns "not identified" and terminates.

As a result of the identification algorithm, iCITY knowns that there is another system, named UbiquiTO, which stores some knowledge about the user Carlo. iCITY may ask UbiquiTO for the user model data it requires to perform adaptation. With respect to the scenario presented in Section 3, iCITY can access UbiquiTO's user model to search for Carlo's "interest in rock music".

7 Evaluation

We conducted an experimental study to test the Identification algorithm. The test has been realized to simulate the behaviour of users who interact with multiple user-adaptive systems. More specifically, we analyzed the phase in which users register themselves with the system to take advantage of their adaptive services. The aim of the test was to verify if the algorithm we developed correctly performs user identification, i.e., correctly finds out whether two entities interacting with different systems have the same identity. For the evaluation, we considered iCITY as requester and UbiquiTO and MASTROcarONTE as suppliers.

Method
We tested the Identification algorithm selecting 80 users following an availability sampling strategy.[22] Users were selecting among the students of Computer

[22] The availability sampling is a non-random sample of convenience, based on subjects available to the researcher, often used when the population source is not completely defined.

Science courses in our university, 40 males and 40 females, 19-31 years old.
To avoid that the experiment biasing the results obtained, we decided not to provide the users with explanation or details about the goal of the experiment. On the contrary, users were simply asked to register themselves into the registration forms of 3 systems. However, users where required to provide their explicit consent to allow us to collect and analyze their sensitive data, as required by the Italian law.[23] [21].
As a test bench, we considered three user-adaptive systems developed in our research group: iCITY, UbiquiTO and MASTROcarONTE.[24]
All the 80 users were asked to fill in the registration form of iCITY. After one week, 64 users were invited to register themselves also in UbiquiTO system. After other two weeks, the same 64 users were asked to register themselves into MASTROcarONTE system as well. The remaining 16 users (already registered into iCITY) were divided into two groups of 8 users each. One group was asked to register only into UbiquiTO and the other group was asked to register only into MASTROcarONTE.
We chose to maintain a two week timeframe spent after every interaction to reproduce, as more faithfully possible, the real-life behavior of a user who interacts with various user-adaptive systems. Furthermore, if users were asked to fill the registration forms of all the 3 systems at the same time, probably the Misleading Level and the Values per User Level would have been affected by such a request.

Results
We applied our Identification algorithm to the selected sample. Since we asked 64 users to fill in the registration forms of all the available systems, we expected that the algorithm would result "identified" in 64 cases we have previously selected, and "not identified" for the remaining 16 cases. In other words, we expected that the execution of the algorithm supported iCITY in discovering that 64 users interacting with it also interact with UbiquiTO and MASTROcarONTE.
In fact, the execution of the Identification Algorithm on those 64 cases, returned "identified" in 59 cases and "not identified" in the remaining 5 cases. Thus, the Identification algorithm has failed in 5 cases: 5 users have been referred as "not identified" although they had the same identity.
Among the 16 users who interacted with UbiquiTO or with MASTROcarONTE we expected the result "not identified" for 16 cases. However, in 2 cases the algorithm had incorrectly resulted "identified".
As defined in Section 4.4, such errors can be respectively labeled as "false negative" and and "false positive".
To understand these motivations, we analyzed, for each case which generated an error, the relations occurring among the matching identification properties and the parameters outlined in Section 4.3.1. "False negative" errors occurred when the Univocity Level and/or the Misleading Values of the matching identification

[23] Italian Privacy Regulation, Law n. 675, December 31th, 1996.
[24] This choice allowed us to easily extract the data provided by the users during the test from the systems' databases.

properties were high.
"False positive" errors occurred when the Univocity Level of the matching identification properties was low (thus the chance for a user thought to be different among two systems was high), while the Values per Users and/or Misleading Values were high.
We used this test also to derive the value expressing the relation between the user model properties and UL, VpU and ML parameters and to refine the Importance Factor threshold which should to be overcome to achieve "identified" as a result of the identification process (Section 4.4).

8 Privacy Issues

Privacy and authorization must be considered when designing user identification processes for cross-system personalisation [5]. This is extremely important for user modeling systems in general, since they collect large amounts of information about users, to support personalized behaviour.
The main requirements for performing personalisation in relation with privacy are to obtain permission from users to collect and use their data, and to protect the collected data.
In cross-system personalisation, the privacy issue becomes even more critical. In addition to the aforementioned issues, privacy deals with the release of user data to third party systems [27]. The two main privacy constraints which can affect user modeling, and especially cross-system personalisation, are individual preferences and privacy laws [45].
Individual preferences. A user may have personal requirements on privacy dimensions. The main common user requirements for privacy in cross-system personalisation are: i) which part of the user model to made available to other applications; ii) how long the user model data are retained; iii) which applications can access the data; iv) the purpose of data sharing; v) the context of the information access [8], [47].
Privacy laws. Many user requirements are safeguarded by suitable privacy laws. Even though these laws differ from country to country, the common legal requirements are: i) proper data acquisition; ii) notification about the purpose of use, iii) permissible data transfer (to third parties and/or across national borders); iv) permissible data processing (organization, modification or destruction) [29] . These legal requirements affect the possibility of sharing user model data. In fact, they may forbid user modeling systems for suppling data to other systems if they use information for different purpose with the respect from the starting applications. Moreover, they can compel service providers to obtain the consent of the user to transmit data to third parties. Finally, they can forbid the combination of usage logs of different services and this can be an obstacle to the distribution of user models built by gathering data from different sites.
Privacy issue becomes even more relevant for *user identification* in cross-system personalisation. In this case, the user data which are exchanged across systems are sensitive data strictly related to the personal identity of the user.

A basic requirement of our framework is that systems must obtain users' explicit consent before sharing their user model data for sake of user identification.
Thus, there is the risk that users do not agree to such information exchange. Users are unlikely to tolerate the storage of large amounts of personally-identifiable data in a central user modeling server, even if the personalization benefits they receive are extremely valuable to them [31].
More recent work reveals that computer users are very concerned about their privacy on-line [29]. But Internet users often lack sufficient information to be able to make educated privacy-related decisions, and this often leads to an overvaluation of small but immediate benefits and an undervaluation of future negative privacy impacts.
On these considerations, we allowed users to inspect their profiles [26] and to ask their informed consent about which data can be exchanged for user identification. In other words, users are first informed that some data could be exchanged, then they are asked to decide which data can be provided. Moreover, users are required to indicate which systems are allowed to use their data.
Finally, the management of user privacy and authorization in our framework has been inspired by Van der Sluijs and Houben [41] and by CPExchange specifications. Following these projects, we provide a granular privacy and authorization model that is appropriate for aggregated and interchanged information .
Regarding the access to Identification Registry (see Section 4.3), we relied on the "*role-based access control*" (RBAC) policy[25] to filter communication between user-adaptive systems and the server. Only systems which are authorized "via role" [31] can access the registry. They can have different levels of freedom in accessing user model values according to the privacy policies established by the users.

9 Related Work

The approach we presented draws parallels from different research directions. The issue of identifying a user in her/his interaction over the Web has been recently addressed. Hardt[26] proposed the definition of "Identity 2.0" which denotes the move from an environment where user identity is recorded in each system to a user-centric environment where the same user identity is shared among a variety of web applications.
Starting from such a definition, researchers have proposed "OpenID", which is an open, decentralized, free framework for verifying users' online identity. In the

[25] In computer systems security Role-Based Access Control (RBAC) is an approach to restricting system access to authorized entities (both users or other systems). Since entities are not assigned permissions directly, but only acquire them through their role (or roles), management of individual entity rights becomes a matter of simply assigning the appropriate roles to the user, which simplifies common operations such as adding an entity, or changing an entity status [36].

[26] http://identity20.com/

OpenID approach, anyone can identify themselves on the Internet with a URI, as for websites. In OpenID Authentication, the username is used as a URI, and the password is safely stored on the users' OpenID Provider. On OpenID-enabled sites, users do not need to register themselves and manage a new account for every site before being granted access. They only need to be previously registered on a website with an OpenID "identity provider". At present the OpenID framework is not meant to be used on sensitive accounts. Since the knowledge in the user model often includes many sensitive data, it is clear that the OpenID framework cannot be used for our purpose. Even in case OpenID would manage sensitive data, it is still different from our approach which allows users to be identified without the need for a unique user identifier.

The only research aimed at identifying users among user-adaptive systems, to authors' knowledge, is Dolog's [19]. He presents the conceptualization and implementation of a framework that provides a common base for the exchange of learner profiles between several sources, through APIs designed and implemented to create,export and manipulate these data. Besides the general aim of his work, he focuses on the identification of profile fragments across systems. This is performed by unification of identification records maintained on different sites. Similarly to our approach, systems are allowed to use their local user identification schema. Differently to our approach, the mapping between the schemas is performed by a personal learning assistant. This agent is in charge of retrieving all the instances of the identification concept for the current user in the identification record. It then searches the instances of the user on systems references in each identification entry. The author performed the identification process in a complementary manner. For each specific user, the personal assistant maintains a list of systems that user interacted with, while in our framework each system maintains and publishes the list of users it interacted with.
Outside the user model interoperability context, many systems have been proposed to manage, in general, the identification of objects across different data sources. One of the most relevant is proposed by Guham, and defines negotiation strategies (in particular the use of shared keys) to manage the problem of matching objects across data sources (e.g. two web stores that want to exchange information about a particular CD album need to agree on a mutually comprehensible reference to the object) [24].
Our definition of the identification properties has been influenced in particular by the vCard specification for personal data interchange (presented in Section 4.1) and Customer Profile Exchange (CPExchange) [6]. The CPExchange specification aims to allow all the applications used by an organization to share customer information. This provides a comprehensive view of the customers who interact with more than one part of the organization, rather than being users of single applications. This allows the tracking of interactions with the customer over time, leading to better understanding of the customer. Using CPExchange to describe user profiles leads to the interoperability between all applications in an enterprise. CPExchange provides a comprehensive view of the customer, not

just as a user of a particular application, but also as an entity which interacts with multiple facets of an enterprise. It allows views of the customers activities over time, providing a cumulative historical record of events that enhances the enterprises understanding of the customer.[27]
Both vCard and CPExchange specifications are defined for the close environment of enterprises. Thus they can hardy be adapted to the open cross-system environment we described.
Another relevant project concerning the definition of standard for user identification is PIDS[28], the Person IDentification Service, developed by the CORBA Healthcare DomainTask Force.[29] PIDS defines a set of interfaces to organize person ID management functionalities in the healthcare domain. For example, the *IdentifyPerson interface* is a query used to send user properties (traits) to be matched and to receive the matching candidate(s). The *ProfileAccess interface* can be used to access and gather a person's profile (formed by a set of traits with the corresponding values).

10 Conclusions and Future Work

In recent years, a large number of user-adaptive systems, mainly available online, have been developed. User data is scattered across multiple, independent applications: user profiles are inherently distributed.
The big challenge is to develop environments where user-adaptive systems cooperate in a "many-to-many paradigm" to exchange knowledge about the user to improve user adaptation. The cooperation among systems to exchange user model knowledge can be seen to be a complex task.
This paper addresses a key challenge for cross-system personalization which is often taken as a starting assumption: user identification. We describe the conceptualization and implementation of a framework that provides user identification for cross-system personalisation among web-based user-adaptive systems, even in absence of a unique shared user identifier. The framework can be easily adopted in different working environments and for different purposes.
We define a set of identification properties and an algorithm that uses these properties to identify users. We have implemented the system as a Java API that system designers can use for user identification. Moreover, we developed a set of Java APIs to support designers and systems in performing such a task.

Our approach to cross-system personalisation is neither fully centralized nor decentralized. The identification registry is centralized but the PUM repositories are distributed and, therefore, decentralized. The Identification registry clearly

[27] http://www.CPExchange.com
[28] http://www.omg.org/technology/documents/formal/person_identification_service.htm
[29] CORBA, the Common Object Request Broker Architecture is a standard proposed by OMG for programming-language- and location-independent transfer of data between parts of applications.

adheres to a centralized approach, while the Public User Model repositories are distributed. Thus they rely on a decentralized solution.
Both centralized and a decentralized approaches have several advantages and limitations for user identification and for cross-system personalisation in general. A centralized approach implies a unique privacy policy which must be accepted by all the contributing systems to access data. However, in a decentralized approach, each system may define its own privacy policies about which part of user model to be shared, keeping private some portion of the user model and defining different rules of access according to requester reputations [7]. However, in decentralized approach for cross-system personalisation, systems do not rely on a common representation for their user models. Dealing with the heterogeneity of information is much harder among decentralized systems than in centralized ones. Different systems may represent the same information in different ways using different syntactic and conceptual structures and often also using different terminologies or different interpretations of the same terminology [37]. Therefore, the overcoming any syntactic and semantic heterogeneity is a critical problem and it requires many efforts.
For all these considerations we believe that a hybrid solution allows user information (and cross-system personalization in general) to be managed by exploiting the advantages of both centralized and decentralized approaches and overcoming many drawbacks of both of them.

The strength of our approach is its high flexibility, which allows it to be applied to many different user-adaptive systems, and intelligent information systems in general. Our framework can be adopted by existing systems without significant architecture changes.
Beside the great flexibility of our approach, it show some further challenges.
Our identification algorithm combines the Importance Factors of matched identification properties and classified a user as identified if this combined Importance Factor exceeds a threshold. The threshold is such that matches for "Last name" and "Birth city" are sufficient for the algorithm to decide that two users are the same entity interacting with different systems. This found an empirical confirmation in the experimental study described in Section 7. However, this ignores the existence of common last names in many countries. Moreover, the threshold assumes that match for "Email" is enough to establish that two users are the same entity. This is quite tricky as well, since users might have more than one email address. All these challenges need to be further on investigated.

Our main on-going effort is in making user identification more lightweight. At the moment, each requester has to use the API to access the identification algorithm; in order to lower the implementation cost, we are working towards a modular architecture, with a specific *ad hoc* module to which the client can delegate user identification.
We are also working on minimizing the disadvantages of the centralized Identification Registry. Specifically, we intend to replicate the registry across several

mirrors so that the registry is no longer a single point of failure.
As well as user identification, we are working on the definition of a complete domain-independent framework for cross-system personalization.

11 Acknowledgment

We would like to thank Prof. Luca Console for his valuable comments and constructive advices.

References

1. N. Arch-Int and P. Sophatsathit. A semantic information gathering approach for heterogeneous information sources on www. *Journal of Information Science*, 29(5):357–374, 2003.
2. Y. Arens, C.Y. Chee, C.N. Hsu, and C.A. Knoblock. Retrieving and integrating data from multiple information sources. *Journal of Information Science*, 2(2):127–159, 1993.
3. L. Aroyo, P. Dolog, G. Houben, M. Kravcik, A. Ambjorn Naeve, M. Nilsson, and F. Wild. Interoperability in personalized adaptive learning. *Educational Technology & Society*, 9(2):4–18, 2006.
4. S. Berkovsky. Ubiquitous user modeling in recommender systems. In *Proc. of International Conference on User Modeling*, pages 496–498, 2005.
5. E. Bertino, J. Byun, and N. Li. Privacy-preserving database systems. *Foundations of Security Analysis and Design. Lecture Note in Computer Science, Vol. 3655*, 2005.
6. K. Bohrer and B. Holland. Customer profile exchange. http://www.cpexchange.org/standard/cpexchangev1 0F.pdf, 2000.
7. A. Brar and J. Kay. Privacy and security in ubiquitous personalized applications. Technical report, School of Information Technologies, University of Sydney, 2004.
8. Kay J. Brar, A. Privacy and security in ubiquitous personalized applications. *Technical report number 561, University of Sydney, School of Information Technologies, NWS*, 2004.
9. J. Broekstra and A. Kampman. Serql: An rdf query and transformation language. In *Proc. of International Semantic Web Conference*, 2004.
10. P. Brusilovsky. From adaptive hypermedia to the adaptive web. In *Mensch & Computer*, 2003.
11. F Carmagnola and F. Cena. An approach for evaluating user model data in an interoperability scenario. In *Proc. of European Starting AI Researcher Symposium, STAIRS*, pages 251–252, Riva del Garda (TN), Italia, August 28-29 2006. IOS PRESS.
12. F. Carmagnola and F. Cena. From interoperable user models to interoperable user modeling. In *Proc. of International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, pages 409–413, 2006.
13. F. Carmagnola, F. Cena, L. Console, O. Cortassa, C. Ferri, M. Gena, A. Goy, M. Parena, I. Torre, A. Toso, F. Vernero, and A. Vellar. icity - an adaptive social mobile guide for cultural events. In *Proc. of Mobile Guide Workshop*, 2006.
14. F. Cena and L. Aroyo. A semantics-based dialogue for interoperability of user-adaptive systemsin a ubiquitous environment. In *In Proc. of User Modelling Conference*, 2007.

15. Hsu. Ching and S. Kao. An owl-based extensible transcoding system for mobile multi-devices. *Journal of Information Science*, 31(3):178–195, 2005.
16. L. Console, I. Torre, I. Lombardi, S. Gioria, and V. Surano. Personalized and adaptive services on board a car: An application for tourist information. *Journal of Intelligent Information Systems*, 21(3):249–284, 2003.
17. L. Cranor, J. Reagle, and M. Ackerman. Beyond concern: Understanding net users' attitudes about online privacy. *CoRR*, cs.CY/9904010, 1999.
18. P. De Bra, L. Aroyo, and V. Chepegin. The next big thing: Adaptive web-based systems. *Journal of Digital Information*, 5(1), 2004.
19. P. Dolog. Identifying relevant fragments of learner profile on the semantic web. In *Workshop on Applications of Semantic Web Technologies for E-learning, in International Semantic Web Conference(ISWC 04)*, 2004.
20. J.V. Fink. *User Modeling Servers - Requirements, Design and Evaluation*. PhD thesis, University of Duisburg-Essen, 2003.
21. C. Gena. Methods and techniques for the evaluation of user-adaptive systems. *Knowledge Engineering Review*, 20(1):1–37, 2005.
22. M. Greaves, H. Holmback, and J. Bradshaw. What is a conversation policy? In *Issues in Agent Communication*, pages 118–131, London, UK, 2000. Springer-Verlag.
23. T.R. Gruber. Toward principles for the design of ontologies used for knowledge sharing? *International Journal of Human-Computer Studies*, 43(5-6):907–928, 1995.
24. R. Guham. Semantic negotiation: Co-identifying objects across data sources. In *Proc. of Semantic Web Services Symposium*, 2004.
25. D. Heckmann, T. Schwartz, B. Brandherm, M. Schmitz, and M. von Wilamowitz-Moellendorff. Gumo - the general user model ontology. In *Proc. of International Conference on User Modeling*, pages 428–432, 2005.
26. J. Kay and B. Kummerfeld. Scrutability, user control and privacy for distributed personalization. In *Proc. of Workshop on Privacy-Enhanced Personalization, In International Conference on Human Factors in Computing Systems*, April 22-27 2006.
27. J. Kay and B. Kummerfeld. Scrutability, user control and privacy for distributed personalization. In *CHI 2006 Workshop on Privacy-Enananced personalization*, 2006.
28. A. Kobsa. Generic user modeling systems. *User Modeling and User-Adapted Interaction*, 11:49–63, 2001.
29. A. Kobsa. Privacy-enhanced web personalization. In A. Kobsa P. Brusilovsky and W. Nejdl, editors, *The Adaptive Web: Methods and Strategies of Web Personalization*, volume 4321 of *LNCS*. Springer Verlag, 2007.
30. A. Kobsa and et al. Koenemann, J. Personalized hypermedia presentation techniques for improving online customer relationship. *The Knowledge Engineering Review*, 16(2):111–115, 2001.
31. A. Kobsa and J. Schreck. Privacy through pseudonymity in user-adaptive systems. *ACM Transactions on Internet Technology*, 3(2):149–183, 2003.
32. B. Mehta, Hofmann T., and P. Fankhauser. Cross system personalization by sparse factor analysis. In *Proc. of Intelligent Technoques for Web Personalization Workshop, in National Conference on Artificial Intelligence*, July 16-20 2006.
33. M. Montaner, B. López, and J.L. de la Rosa. A taxonomy of recommender agents on the internet. *Artificial Intelligent Review*, 19(4):285–330, 2003.
34. C.J. Niederee, A. Stewart, B. Mehta, and M. Hemmje. A multi-dimensional, unified user model for cross-system personalization. In *Proc. of Workshop on En-*

vironments for Personalized Information Access, in Advanced Visual Interfaces International Working Conference, 2004.
35. G. Salton and M. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill Book Company, 1984.
36. R. Sandhu, E.J. Coyne, H.L. Feinsten, and C.E. Youman. Role-based access control models. *IEEE Computer Society*, 2:38–47, 1996.
37. S. Staab and H. Stuckenschmidt. *Semantic Web and Peer-to-Peer Decentralized Management and Exchange of Knowledge and Information.* Springer Verlag, Berlin, 2006.
38. N et al. Stash. Explicit intelligence in adaptive hypermedia:generic adaptation languages for learning preferencesc and styles visualization of route descriptions. *Cognitive Processing*, 2:323–345, 2001.
39. C. Stewart, A. Cristea, I. Celik, and E. Ashman. Interoperability between aeh user models. In *Proc. of International workshop on Adaptivity, personalization and the semantic Web, in Conference on Hypertext and Hypermedia*, pages 21–30, New York, NY, USA, 2006. ACM Press.
40. F. Van Assche, Duval E., Massart D., Olmedilla D., B. Simon, Sobernig, S. Ternier, and F. Wild. Spinning interoperable applications for teaching & learning using thesimple query interface. *Educational Technology & Society*, 9(2):51–67, 2006.
41. K. van der Sluijs and G.J. Houben. Towards a generic user model component. In *Proc. of Workshop on Decentralized, Agent Based and Special Approaches to User Modelling , In International Conference on User Modelling*, July 2005.
42. J. Vassileva. Distributed user modelling for universal information access. In *Internation Journal of Human Compueter Interaction*, pages 122–126, 2001.
43. K.H. Veltman. Syntactic and semantic interoperability: New approaches to knowledge and the semantic web. *The New Review of Information Networking*, 2001.
44. Y. Wang and A. Kobsa. Respecting user's individual privacy constraints in web personalization. In *To appear in proceedings of International Conference on User Modeling 2007*, June 2007.
45. Y. Wang and A. Kobsa. Respecting users' individual privacy constraints in web personalization. In *Proceedings of the 11th International Conference on User Modeling*, 2007.
46. M. Weiser. The future of ubiquitous computing on campus. *Commun. ACM*, 41(1):41–42, 1998.
47. W. Woerndl and M. Koch. Privacy in distributed user profile management. In *WWW2003*, 2003.

Figure Captions

Fig. 1. Sequence diagram of the user identification task from the requester perspective

Fig. 2. Architecture of the framework

Algorithm Caption

Algorithm 1. Identification Algorithm

Tables

Table 1. Registration form data for 25 user-adaptive systems and corresponding number of occurrences

Table 2. Identification properties in relation with UL, VpU and ML

Table 3. The identification properties and their corresponding IF