Architecture for profile translation

Bjoern Wuest, Olaf Drögehorn, Klaus David

Abstract—**User profiles are typically stored locally within proprietary personalisation architectures at service providers. Replicating the same information of the user profile, e.g. email address, across multiple independent service providers decreases consistency of user profiles.**

Centralising or exchange profile information increases consistency of profiles. Law issues, company policies and proprietary profile structures prevent successful profile exchange between service providers. Existing solutions for centralised profile storage like Microsoft Passport or Liberty Alliance are limited to specific information in the user's profile, e.g. authentication information.

This paper presents a profile architecture allowing to keep the user's full profile in the user domain. Service providers access the user profile by defined translations between the service providers profile structure and the user's profile structure. The translations defined are adaptable to reflect modifications in the different profile structures, e.g. Amazon, eBay etc. The concepts developed are especially interesting for facilitating future mobile applications.

Index-Terms—**Data conversion, Internet, Personalisation**

I. Introduction

PERSONALISATION has the potential to increase the value of web services for the user. Examples of personalised web services are Amazon, eBay, New York Times etc. Profiles are a cornerstone of personalisation. Profiles keep the user's preferences and gathered habits. Web services use the information in profiles to adapt to the user [1,2].

Personalisation architectures, where profiles are a part of, are mostly proprietary systems. Interoperability and exchange of profile information between personalisation architectures is not foreseen. Instead, a profile database with the user's preferences and gathered habits is kept with each service provider. This storage of the user's profile with the service provider raises issues concerning privacy and consistency of profile information. Privacy is a concern since information of and about the user is stored outside of the user's domain and control. Consistency is affected by possibly replicating the same information throughout a number of service providers, each using their own proprietary personalisation architecture.

A simple scenario may illustrate the two problems above of privacy and consistency. Assume two service providers offering personalised web services, one online book store and one online auction system. Both services provide a newsletter service with latest information being sent to registered users via email. Further, both web services support adaptation of font style and font size for different text styles and require the user's credit card information for offering paid services. Both service providers act independent and have no interest to share their data. Privacy policies and laws may also prohibit data sharing. A user registered at both services must setup his email address and preferred font settings with both web services at both service providers. The user must trust the service providers that their system is safe from hacking so his credit card information, email address and font settings remain with the book store and auction system only. Further, if any information or preference of the user is changing, e.g. his email address or credit card information, he must update his information with both service providers. In a real world scenario the Internet user may not only register with two systems but with dozens, possibly hundreds of services, needing the trust of the user into each single system and increasing the work of the user to keep his information up to date among all of these services.

The Microsoft Passport service [3] and the Liberty Alliance [4] are efforts to keep a service independent user profile. However, both are limited to user authentication and related information such that font settings and the like are not covered by these two efforts.

To translate between different profile structures filters and translation languages [5] may be applied. A profile structure filter is comparable to a test or graphics filter known from word processing systems. The use of filters and translation languages perform a static translation and copy the data of profiles. When a date in a profile is changed all translations have to be performed again.

Synchronisation mechanisms such as SyncML [6] or xmiddle [7] reconcile distributed profile information. However, synchronisation of profile information is impractical when used for synchronising and distributing profile information between different independent service providers. Reasons for this impracticality are local privacy policies and laws, competition and incompatible profile structures due to missing standards. We understand that synchronisation mechanisms are still necessary, but only within the user's domain, where privacy issues are of no concern. We do not agree that synchronisation is applied

Manuscript received February 04, 2005. This work was supported in part by the European Union Information Society Technology in the Sixth Framework Programme under the contract number of FP6-507993 (mCDN) and by the German Ministry of Research bmb+f Wireless Internet Zellular project. The paper represents the work and contribution of individual parties involved in the project. We thank our project partners for the good and fruitful discussions.

Bjoern Wuest, Olaf Drögehorn and Klaus David are with University of Kassel, Chair of Communication Technology, Wilhelmshöher Allee 73, 34121 Kassel, Germany (e-mail: {bjoern.wuest, droegehorn, david}@uni-kassel.de)

between two parties without the owner of the data being involved, in our case the user of the service providers' services; although our approach does not enforce avoidance of such synchronisation.

In this paper we introduce an approach to address consistency of information and to improve privacy for the scenarios above. Both goals are achieved by storing profile data within the user's domain. Service providers link to the information in the profile. Increasing privacy by restricting access to profiles stored in the Internet is presented in [8]. Keeping profile information within the user's domain is proposed in [9] but their approach lacks of the flexible profile structure expected by the different service providers. In profile technology there is no previous work known that aims at improving data integrity, consistency and privacy of profile information all together with flexibility in profile structure as done within this paper. Thus, the approach, definitions and translation technique presented are novel for profile technology in personalisation architectures. All this is especially interesting for scenarios with mobile devices.

In section II we give a clear and strict definition about how a profile is organised. We back this definition by a formal description. This formal description is tailored towards separation of profile information and profile structure. Its possibilities and advantages are discussed in section III. Section IV provides guidance on how to utilise the separation introduced in real world solutions, followed by the description of a tool capable of translating profiles between different standards for different personalisation architectures in section V. Finally the conclusion is given.

II. DEFINITION OF PROFILES

A profile is a collection of information. In a user profile this information reflects the preferences and gathered habit of the user. The information is accessed by a well known key. To represent a key a keyword or description or any other form able to uniquely identify information may be used. A pair of key and information is called a profile entry. A profile consists of several profile entries. A telephone book is such a profile. The contacts in the telephone book profile are pairs of names and telephone numbers. These pairs form the profile entries with one profile entry per contact. The name of a profile entry is the key to the profile entry while the telephone number is the information of the profile entry.

The formal description following is a strict definition of profiles. For the description we use the operators and functions defined in table 1. The formal description is used in a later section to derive a profile translation formalism.

Symbol	Meaning
Z	Element of Z.
≡	Definition assignment operator.
≠	Inequality operator.
Z_i	The i-th element of a group of Z-elements.
∀	Do for all items.
Y_i Z_i	Element Y_i is directly followed by element Z_i.
$\{Z_i\}$	Set of Z items forming a group of Z-elements.
n	Unspecified number.
=	Equality or identity operator.

TABLE 1 OPERATORS AND FUNCTIONS USED IN FORMAL DESCRIPTION OF PROFILE DEFINITION.

Let **P** be a profile and **E** be a single profile entry then a profile is defined as $\mathbf{P \equiv \{E_1, E_2, \ldots, E_n\}}$ with the constraint of $\mathbf{E_i \neq E_j \forall i \neq j}$. The telephone book would be a profile while every single contact in the telephone book would be a single profile entry. A single key/value mapping in a profile entry $\mathbf{E_i}$ is defined as $\mathbf{E_i \equiv K_i\ D_i}$ with $\mathbf{K_i}$ being the key of the profile entry. In the telephone book contact, the key would be the name of the contact. $\mathbf{D_i}$ is the value of the profile entry, e.g. the telephone number of the contact in the telephone book. The value $\mathbf{D_i}$ may be empty if there is no value set for the key $\mathbf{K_i}$. Keys in a profile are unique such that constraint $\mathbf{K_i \neq K_j \forall i \neq j}$ must be hold. This is a complete formal description of profiles as described in the first paragraph of this section.

Profiles may contain Meta information to augment information. Such Meta information could be used to constrain the information, e.g. to enforce a proper format of phone numbers and dates. Other Meta information could limit access to profile information. Meta information may also semantically describe profile information. Augmenting the profile with attributes provide support for Meta information in a profile. The augmentation of profile entries with attributes **A** results in the extension of the definition of $\mathbf{E_i}$ to $\mathbf{E_i \equiv K_i\ D_i\ \{A_{i,1}, A_{i,2}, \ldots, A_{i,n}\}}$ with $\mathbf{A_{i,j}}$ being a single attribute augmenting the profile. In the telephone book the annotation, or meta information, of 'home' or 'business' or 'mobile' are attributes. The simplicity of attribute augmentation proposed in this paper is sufficient for the purposes presented later in the paper. More advanced attribute augmentation schemes may be defined without affecting the results of this paper.

III. SEPARATION OF PROFILE STRUCTURE AND PROFILE INFORMATION

To increase data consistency and address privacy of profile information we regard it to be useful to keep the profile information in one profile, within the domain of the user, under his control. Yet, this raises the problem of how proprietary personalisation systems of service providers, which are used to keep their own profile, retrieve and use information in the user's profile. The identified challenges of storing the user's profile within the user's domain are unavailability of the user's profile when *disconnected* from the user on the one hand and the different standards of profile structure when the user's profile is stored in only one profile structure on the other hand. We address the problem of *disconnection* from the user in section IV and concentrate on how to solve the gap arising with different profile structures in this section.

In this section we make the assumption that service providers have access to the profile information stored in the user's domain every time they require access. In the following section we drop this assumption but for now it simplifies the considerations following.

As given in the scenario of the online book store and the auction system both base on different personalisation systems using their own proprietary profile structure

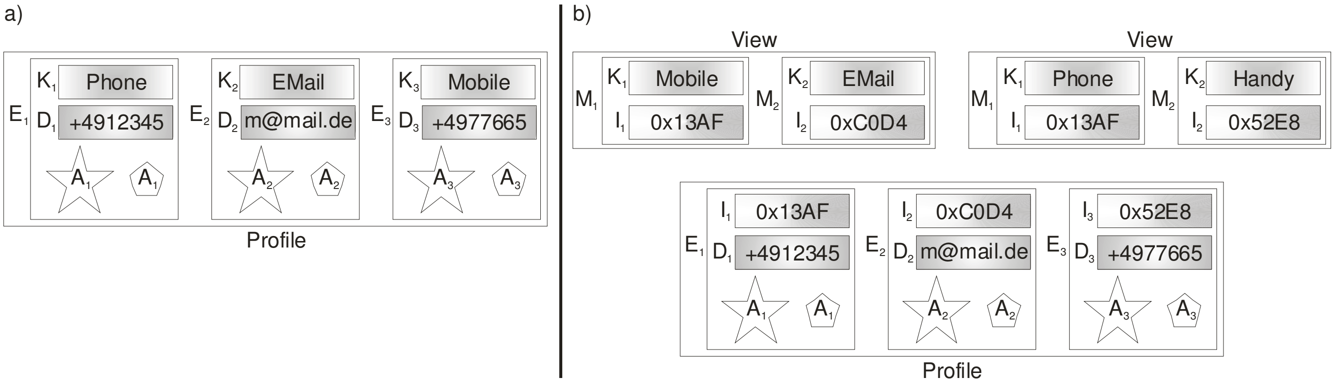

Figure 1: Schematic of profiles. In sub figure a) on the left there is the classic definition of profiles using keys, values, entries and meta-information (attributes). Sub figure b) shows the introduction of identifiers to profile entries and the resulting separation of profile structure from profile information into a profile and views defined on and linked to the profile.

scheme. The different profile structures would require for different, independent profiles. This requirement becomes obsolete once the structural structure of a profile is independent of the information stored in the profile. This independence would allow defining several different structures on the same information stored in a profile. Instead of duplicating information stored in the profile the information is stored only once and within the user's domain. The structure of profile structure is kept with the service providers. A linkage between the profile structure located with the service provider and the profile information located with the user is established. This linkage uses a plain, simple yet sufficient mapping between structure and information. This linkage is transparent to both the user and the service provider. Further the linkage is free of context and semantics. Context and semantics of profile information is addressed by the profile structure used by the service provider. Meaning of profile information is brought in by the profile information by itself. The following formal specification will derive the separation of profile structure and profile information and clearly defines what the linkage between both, structure and information, is.

Given the formally specified profiles from the previous section the independence of structure from the stored information would require a decoupling of profile keys ($\mathbf{K_i}$) and profile values ($\mathbf{D_i}$). Instead of a key $\mathbf{K_i}$ as key to a profile entry $\mathbf{E_i}$ we introduce an identifier $\mathbf{I_i}$ as key to $\mathbf{E_i}$. This decoupling then allows us to define numerous keys on the same profile entry by mapping the different keys onto the same identifier while the identifier is always mapped to the same value in the referenced profile entry. Thus, the definition of $\mathbf{E_i}$ changes to $\mathbf{E_i \equiv I_i\ D_i}$. To connect $\mathbf{K_i}$ to the value $\mathbf{D_i}$ of any $\mathbf{E_i}$ we introduce a mapping $\mathbf{M_i}$ of $\mathbf{K_i}$ to the identifier $\mathbf{I_i}$ of $\mathbf{E_i}$ such that $\mathbf{M \equiv K_i\ I_i}$. The mapping $\mathbf{M_i}$ actually defines the structural information of a profile $\mathbf{P}$. The profile still consists of profile entries such that $\mathbf{P \equiv \{E_1, E_2, \ldots, E_n\}}$ with the constraint of $\mathbf{I_i \neq I_j \forall i \neq j}$.

To group mappings which belong to each other we introduce the term of *view*. A view groups all mappings where the keys follow one structure of a profile. Thus, a view $\mathbf{V}$ is defined as $\mathbf{V \equiv \{M_1, M_2, \ldots, M_n\}}$ with the constraint of $\mathbf{K_i \neq K_j \forall i \neq j}$. Further, for a view being valid, for each mapping $\mathbf{M_i}$ there must be a profile entry $\mathbf{E_i}$ in $\mathbf{P}$ the view is defined on so that $\mathbf{I_i \in V = I_i \in P \forall i}$. A drawing of profiles and views using the telephone book example is shown in figure 1.

The definition of the view limits the view to one profile. Further research may investigate if it is useful and suitable to extend the definition of a view to be backed by multiple profiles. For the targeted scenario where the user's profile information of several service providers is pooled in one profile and different profile structures are defined on this one profile this limitation has no negative effect.

The remaining of the paper discusses how the proposed profile architecture with separating profile information from the profile structure may be used in the real world. The increase in privacy with reduced requirements towards trust and gained consistency of profile information is demonstrated. Further, a user controlled yet automatic algorithm for translating between different profile structures, the views, is given.

IV. APPLICATION IN THE REAL WORLD

In the previous two sections we postulated that a separation of profile structures and profile information is beneficial towards privacy and consistency of profile information. In this section we demonstrate the advantages of the separation presented above by applying it to the scenario of proprietary and isolated personalisation architectures in the introduction.

In our scenario there are two service providers offering web services. One service provider offers an online book store whereas the other service provider offers an online auction system. Both service providers keep behavioural and configured preferences of the user in profiles and personalise their service towards the user's preferences. Both service providers use their own proprietary personalisation architecture which is optimised to their offered service. Yet, both service providers depend on identical information about the user like his email address, postal address and credit card information. An exchange of information between both service providers is not possible, not wanted and not allowed due to technical incompatibilities, company and privacy policies and laws.

Traditionally the service providers keep the information about the user in their domain. We move the information about the user from the service provider's location into the user's domain, e.g. his terminal he uses to access the offered web services. The service providers keep the structural information about profile information within their domain. This separation fosters privacy of profile information of the

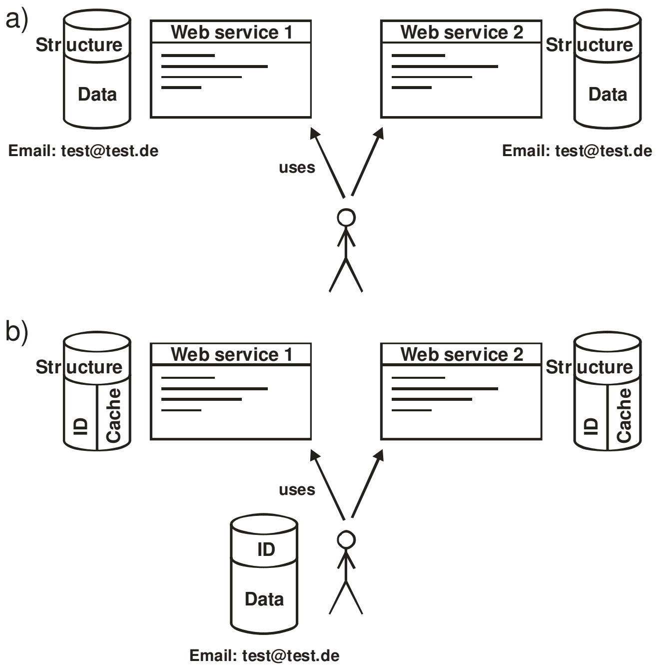

Figure 2: Traditional profile location (a) compared against the proposed separation of profile information and profile structure (b). In (b) service providers implement a cache to first reduce necessary data exchange between the user and the service provider when the service is used by the user and second to keep information demanded for advertisement and maintenance activities, e.g. email addresses) local with the service provider.

user and structural profile information of the service provider since both information is stored with the owner of the information. Further data consistency is improved since profile information about the user is kept only in one place, in the user's domain, and not spread and duplicated among all service providers the user is registered to. The classic case of storing user's profile information within the domain of the service provider and in contrast the proposed scheme where profile information is separated from structural profile information to keep the user's profile in the user's domain is depicted in figure 2.

The proposed separation of profile information and profile structure increases consistency of profile information by avoiding duplication and distribution of identical information across a number of locations and domains. Further, privacy is increased since the user does not need to trust that the service provider does not distributes stored user information like his email address since the service provider does not keep such information persistently. Questions arise on the case where service providers have interest to keep information about the user, e.g. his email address, to send email notifications or advertisements. A common solution to this problem is caching of such interested information. For updating the cache information we foresee the use of a synchronisation middleware like xmiddle [7] every time the user connects to the service of the service provider. This automatic update of cache information might also be a faster and more reliable solution than to ask the user to manually navigate to every single service he is registered with and update his information. However, caching neglects the advanced privacy of the user's profile information since content of the user's profile is stored with the service provider. Yet we do not understand caching of profile information as a disadvantage in means of privacy since it is common practice to keep such information with the service provider anyways.

As seen in this section the separation of profile information and profile structure has huge impact on how profile data may be managed and maintained. Privacy may be increased by avoiding user's profile information stored with service providers. Caching may neglect this advantage. Consistency of profile information is increased even caching is used by avoiding duplication of identical profile information among all service providers the user is registered with. Further, automatic profile reconciliation increases the convenience of use of personalised systems by leveraging the user from updating his profile information manually in many places. Instead the user updates his information in only one place, e.g. in his domain.

Yet unaddressed are concerns about how new profile structures can be defined on existing profile information. The next section deals with this open issue by providing a solution for semi-automatic mapping of profile structures.

V. TRANSLATING PROFILES – THE CUCA TOOL

As discussed in the previous section views defined on profiles avoid profile duplication and keep profile information consistent. However, there must then be a mechanism to translate between the views representing a profile definition.

For translation we introduce the term of "attribute dictionaries". Attribute dictionaries show similarities of attributes. Entries in a profile with attributes from the same attribute dictionary indicate a kind of relation. The similarities aid services and applications to process attributes and to design appropriate search strategies for profile entries where the key to the profile entry is unknown. Following the grammar in section III, "Separation of profile structure and profile information", an attribute dictionary **T** consists of a set of attributes and is defined as $\mathbf{T \equiv \{A_1, A_2, \ldots, A_n\}}$.

Attributes represent the keys in profile definitions and augment profile entries. Attribute dictionaries representing a profile definition summarise these attributes representing the keys in the profile definition. To provide automatic

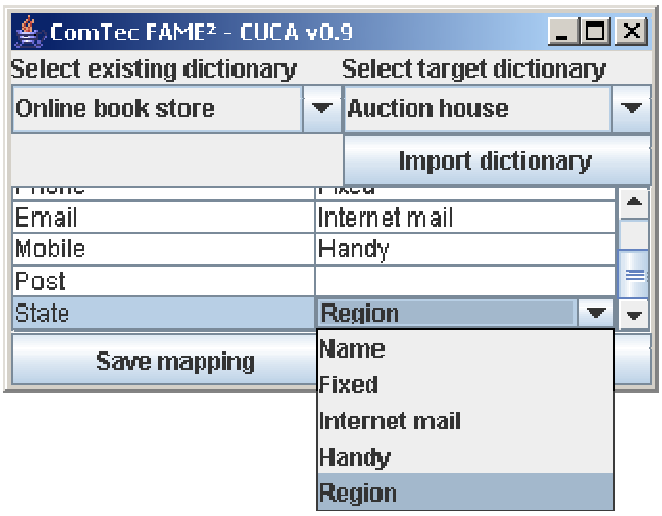

Figure 3: Screenshot of the CUCA, setting up a mapping between the RDF/XML vCard profile definition and the vCard 3.0 profile definition.

translation of a profile between different profile definitions attributes of different attribute dictionaries are mapped such that $\mathbf{A_i \in T_1 = A_j \in T_2 \forall i,j}$. If a profile **P** contains profile entries with attributes from $\mathbf{T_1}$ a view $\mathbf{V_1}$ can be created. For all profile entries in the profile **P** augmented with attributes from attribute dictionary $\mathbf{T_1}$ a view $\mathbf{V_2}$ for the attribute dictionary $\mathbf{T_2}$ can be created automatically where attribute mappings in the form of $\mathbf{A_i \in T_1 = A_j \in T_2 \forall i,j}$ exist.

The *ComTec User Centred Application Add-On for Profiling*, CUCA, is a graphical user interface for setup of attribute dictionary mappings. CUCA creates one view on the profile for every attribute dictionary present. An attribute dictionary is present when the profile contains entries augmented with attributes from that attribute dictionary. Figure 3 shows a screenshot of CUCA to setup an attribute mapping between the imaginary online book store and the auction house. CUCA defines one view per attribute dictionary such that $\mathbf{V_c \equiv \{M_1, M_2, \ldots, M_n\}}$ where $\mathbf{M_i.I = E_j.I \forall E_j.A \in T_k, i,j,k}$.

VI. Conclusion

To increase the value of web services, personalised services adapt themselves to the user. This adaptation bases on profile information about the user. Personalisation architectures typically keep profile information at the service provider. Exchange of user profiles between service providers is not done because of legislative provision, company policies and proprietary profile structures. Centralising of profile information, e.g. Microsoft Passport and Liberty Alliance, address only very specific parts of the user profile.

In the paper we propose to keep user profiles in the user domain and separating the profile data from the profile structure.

Keeping user profiles in the user's domain and let service providers access the user profile overcomes law and company policy restrictions. Separating the profile data from the profile structure allows defining different profile structures, used by different independent service providers, on the same profile data. This separation of profile data from the profile structure benefits consistency of data by avoiding redundancy, foster privacy by being able to keep user information within the user's domain as long as there is no caching used and improves interoperability between different proprietary personalisation architectures by providing a flexible translation technique based on attribute dictionaries. A formal description of profiles is used to show avoidance of redundancy of profile information. Further, profile representation translation is derived with the formal description, resulting in the possibility of adopting the presented and discussed research results in real world systems.

The viability of separating profile information from profile structure and the translation between different profile structures is demonstrated by the CUCA (*ComTec User Centred Application Add-On for Profiling*) tool presented in this paper. With CUCA, users define mappings of different profile structures of different service providers and link their profile information to the profile structures.

Comparable solutions of profile translation are rather static approaches of translation language documents (e.g. XSLT) and filters, which base on knowledge of profile structure. These solutions fail as soon as the profile structure changes.

References

[1] H. Hirsh, C. Basu, B. D. Davison, “Learning to personalize”, *In Communications of the ACM*, Vol.43.8, p. 102-106, August 2000

[2] I. Cingil, A. Dogac, A. Azgin, “A broader approach to personalization”, *In Communications of the ACM*, Vol.43.8, p. 136-141, August 2000

[3] Microsoft .net Passport, http://www.passport.net/

[4] Liberty Alliance Project, http://www.projectliberty.org/

[5] eXtensible Markup Language Stylesheet Language Transformation (XSLT), http://www.w3.org/TR/xslt/

[6] The Open Mobile Alliance, *SyncML*, http://www.openmobilealliance.org/syncml/

[7] C. Mascolo, L. Capra, S. Zachariadis, W. Emmerich, “xmiddle: A Data-Sharing Middleware for Mobile Computing”, *In Proceedings of the 24th International Conference on Software Engineering*, Orlando, Florida, USA, 2002

[8] R. M. Arlein, B. Jai, M. Jakobsson, F. Monrose, M. C. Reiter, “Privacy-preserving global customization”, *In Proceedings of the 2nd ACM conference on Electronic commerce*, Minneapolis, USA, p. 176-184, October 2000

[9] S. Riche, G. Brebner, M. Gittler, "Client-Side Profile Storage: a means to put the user in control", Public Technical Report, Hewlett Packard Laboratories Grenoble, November 2001