A Realization of Context Management Facilitating the Usage of Partial Identities

Elke Franz, Benjamin Engel
Dresden University of Technology
01062 Dresden, Germany
{ef1|be020578}@inf.tu-dresden.de

ABSTRACT

Privacy-Enhancing Identity Management (PIM) provides a feasible approach to preserve users' privacy despite the need for gathering personal data in order to support personalization or collaboration. Complex or collaborative applications require even to partition within one single application [1]. Since intra-application partitioning (IAP) must be done additionally to the actual tasks in the application, a reasonable support for users is essential. This paper introduces a possible realization of context management which we suggest as a means to motivate the use of PIM as well as to support its reasonable use. Problems encountered during realization point out possible pitfalls and general limitations. Generally, considering usability and performance is of fundamental importance for the acceptance and usage of IAP but approaches to improve these aspects must be carefully evaluated since they might endanger the achievable privacy.

INTRODUCTION

Current applications increasingly provide a personalized working environment or support collaboration. However, these goals require gathering of personal data what introduces privacy risks since we cannot completely exclude the misuse of data once collected. Therefore, collecting and processing personal data should be transparent to users so that they can assess what others already know about them and what these potential observers can additionally learn from observing their actions.

PIM enables users to control which personal data they disclose to whom in which situation [5]. Thereby, personal data are partitioned into subsets called partial identities (pIDs). PIM is mainly used for inter-application partitioning. However, applications containing complex scenarios or supporting collaboration impose the need for IAP [1]: Since users act in various scenarios or interact with different user (groups), a lot of information about them accrue in the course of time which might be used to create detailed user profiles.

But even if users are increasingly aware of privacy risks, they do not necessarily behave according to their privacy requirements in real-life scenarios [9]. Within an environment as assumed here, this discrepancy might even get worse: It seems to be especially questionable whether users will partition their data using existing PIM functionality when they have to perform the necessary steps additionally to their actual tasks. Since the accruement of data is not necessarily obvious for users, they might not care about possible consequences of their actions for their privacy and focus solely on the tasks they want to perform in the application.

Therefore, a continual awareness of privacy risks is a necessary precondition for motivating users to apply PIM at all. Furthermore, a suitable support in applying PIM seems to be essential for achieving its reasonable use. Particularly, the system should make suggestions considering the use of PIM according to rules users have defined in advance or apply such rules automatically if possible. The context management discussed here focuses on this aspect: It aims at *supporting users* in their decision under which pID they should perform an action and, consequently, which personal data they should disclose to whom after considering their current situation. The techniques needed for PIM are not in the scope of this paper, we rather utilize functionality provided by a privacy-aware platform.

As an example application, we consider the eLearning application BluES'n[1] which aims at supporting a collaborative environment. Since awareness of other users is relevant for collaboration as well as for a rational IAP in such an environment, it is not possible to consider processing of personal data at client-side only. A discussion of problems encountered during implementation points out the necessity to find a balance between privacy on the one hand and performance and usability on the other hand for a feasible solution.

After a short overview on the example environment, we discuss which personal data is disclosed and point out requirements on context management. The next section describes the realization which has been implemented within BluES'n. After a discussion of possible pitfalls and limitations and a short overview on related work, a summary and outlook concludes the paper.

BASIC CONSIDERATIONS

Example Application

We are realizing context management within the privacy-enhanced eLearning application BluES'n. This application uses functionalities provided by a privacy-aware platform currently developed within the project PRIME[2] such as an-

---

[1] www.blues-portal.de

[2] www.prime-project.eu.org

onymous communication or functionality to manage pseudonyms [2].

BluES'n aims at supporting a collaborative and flexible working environment which users can adapt to their individual needs. The application structure is based on the concept of workspaces [4]. Necessary functionality is provided by functional modules. Users can create new workspaces using existing templates.

The different workspaces reflect the objectives a user is working on within BluES'n, for example, learning or teaching in different classes or participating in dynamically established learning groups. Functional modules provided within a workspace can be used to describe subtasks such as performing tests, editing material, or participating in a discussion.

BluES'n is based on a client-server architecture. Depending on the defined policies, some actions might be performed anonymously. If it is necessary to deliver a pseudonym in order to make actions accountable, the PIM server requests this pseudonym during a negotiation phase. Furthermore, it could also be necessary to deliver user data for getting access to requested resources. The context management has the task to assist users in selecting a suitable pID what might include the suggestion to switch to another pID.

Disclosing Personal Data
Attributes assigned to pIDs are personal data which might be disclosed. Within BluES'n, a pID comprises a *pseudonym* as identifier of the pID as well as an *alias* assigned to the pseudonym for simplifying its handling [3]. Further attributes are *roles* assigned to a pID, a *history* of actions, and *additional information* about users such as interests, end devices, or age.

If users are aware of possible privacy risks, they will disclose as less personal data as possible. The wish to get assistance and to cooperate with other users motivates to disclose personal data despite such concerns: The history enables a tutor to provide individual assistance to learners of his class. Delivering examination results is necessary to get certificates confirming the successful participation in classes. Users might disclose additional information to be contacted by others. Moreover, accessing additional information of other users might require to disclose such information to others as well. For synchronous communication, users might additionally want to get information about the online state of other users.

Disclosing explicitly known attributes might be restricted by policies, but users cannot prevent that circumstances of their actions can be observed: When do users perform actions, how often are they online, how long they usually work, etc. However, especially the accruement of information about the circumstances of actions is less obvious for users. Since these aspects allow to draw further conclusions about users, they also need to be considered.

Feature Space of Contexts
We can conclude from the considerations about disclosure of attributes that any support for users in applying IAP must be based on the evaluation the user's context within the application [1]. A context describes the current situation and environment of a user by considering a number of features [6, 12].

Particularly, a rational management of pIDs within a collaborative environment requires to be aware of other users since a user has to know which information he has received or disclosed and in which context. This information enables users to assess the linkability of their pIDs for other users. Altogether, our context comprises the following main features:

1. *Current task*: This feature can be basically derived from the application structure (workspace/functional module) and by considering the initiated action.

2. *Others' possible knowledge (degree of knowlegde of own pIDs)*: Particularly, it has to be evaluated which pIDs have been already used while working on this task, how often they have been used, and whether they have been additionally used for working in other workspaces and, thus, possibly within other user groups.

3. *Knowledge about others (degree of knowledge of others' pIDs)*: For example, we have to consider users who are currently working within the same workspace since we assume that the BluES'n system provides awareness information about users to others in order to support collaboration. Obviously, these users potentially can observe the user who wants to perform an action.

4. *Physical environment*: Finally, also environmental conditions such as time or location could be considered.

According to [13], these features can be devided into human factors (1-3) and features related to the physical environment (4). The context changes if one of the components of its feature space changes.

Requirements on a Feasible Context Management
In order to achieve user acceptance and to enable them to rationally utilize context management for IAP, the following important requirements need to be considered:

- *Privacy:* Users should be enabled to soundly partition their personal data while acting within the application. Their pIDs must not be linkable by others.

- *Performance and usability of application:* Functionalities provided by the application must not be affected by the context management. Furthermore, a comfortable working should be possible: The context management must not disturb users in performing their actual tasks within the application and should not noticeably decrease *performance*. Otherwise, users will either not rationally use the context management or just switch it off.

- *Usability of context management:* It need to be intuitively usable and its functionality should be easy to understand.

Users should be able to adapt it to their individual preferences. This requirement is not in the scope of this paper.

Even if integrating context management will obviously affect the application, usability and performance must not be significantly influenced.

REALIZATION

Basic Components

The context management comprises three main components. Fig. 1 gives an overview on interactions between these components and the application.

- *Context Configuration* enables users to configure how the features of the context should be evaluated and how the system should react in case of an observable action. Obviously, any network communication is observable. However, even local actions may become observable, for example, if they are logged and transmitted to the server for assessment.

- *Context Monitoring and Logging* manages the context information during a client application session, especially, local actions of a user and observable actions which include information about the selected pID.

- *Context Evaluation* should suggest a suitable pID to be used for an observable action. Thereby, it selects the corresponding context configuration made by the user. Depending on the configuration, it might also evaluate former actions by using the history.

These components were realized within BluES'n considering the requirements summarized above. According to [1], a reasonable combination of prospective, collateral and retrospective configuration is necessary for a viable realization. Until now, prospective configuration which can be used to emulate collateral processing is realized.

Context Configuration

Generally, all features of a context listed above can be used for the configuration. Our current solution focuses on the description of the task. Thereby, we consider the workspace and the functional module in which the user initiates an action as a suitable description. For example, a user can define that a switch to another pID is necessary when he starts an observable action in another workspace. Depending on the situation, even a new observable action in the same functional module could imply a switch to another pID (e.g., asking a question within a discussion forum). Configuration can consider solely the workspace and functional module in which the initiated observable action should be performed but also appropriate information about formerly performed actions. In the following, "working on a task" refers to "working within a specific workspace".

The configuration is structured according to workspaces and functional modules which are objects within BluES'n. It is necessary to distinguish the following cases regarding this part of the task description:

- the user works for the first time on this task,

- the user has already worked on this task, but he has not opened the appropriate workspace within this application session,

- the user starts an observable action belonging to the active workspace.

The context management checks whether the user has already worked on this task by means of the history. A user could work for the first time on a task when he creates the appropriate BluES'n object or opens an object already existing on the BluES'n server for the first time.

Users can define the following reactions of the system for observable actions:

- switch to a specified pID, notify user about this switch,

- start user dialogue with specified settings,

- act anonymously or generate new pID that is not used again (i.e., act under a transaction pseudonym [10]).

Settings for user dialogues contain a suggestion for a suitable pID to be used for the initiated action or a list of already existing pIDs from which the user should select (without excluding possibilities for other choices). This list could contain, e.g., pIDs already used when acting in the workspace the initiated action should be performed.

There are basic configurations for the different workspace templates and functional modules. Users can adapt these settings for actual objects. Thus, they could define different settings for different classes which are based on the same workspace template. Of course, default settings always aim at supporting maximum privacy. A general default configuration ensures that also newly introduced object types can be processed in a reasonable manner: In this case, an observable action initiates a user dialogue suggesting to create a new pID.

Context Monitoring and Logging

Monitoring observes all events within the GUI-layer and determines the current context features. Logging keeps track of all observable actions and corresponding context information received from the context evaluation component.

The context monitoring component registers an event listener which receives every event and keeps a track of the current context by means of a simple state machine concept. Transitions between the states are implied by events on the GUI-layer which either change the context or influence the selection of the pID. For example, activating a workspace is only a local action but implies a change of the state machine. Whenever an observable action occurs it is known which workspace is active and, therefore, in which context the action will be performed.
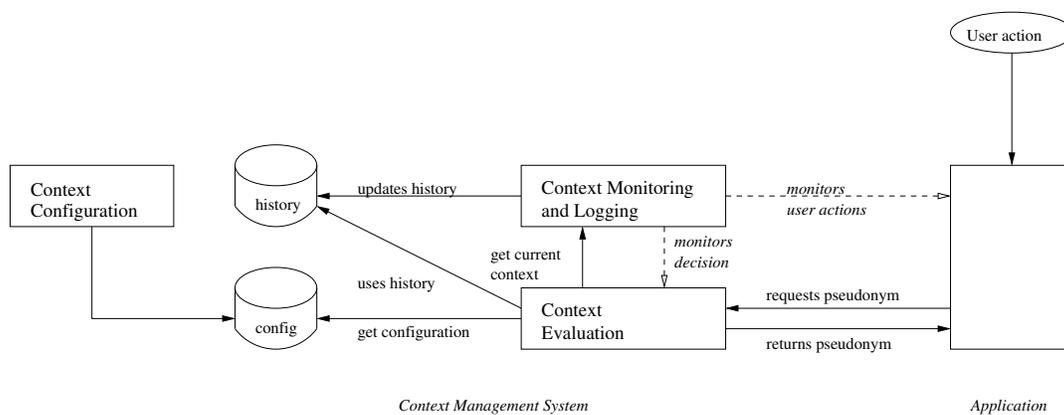
Context Configuration

history

updates history

Context Monitoring and Logging

User action

monitors user actions

get current context

monitors decision

uses history

config

get configuration

Context Evaluation

requests pseudonym

returns pseudonym

Context Management System

Application

Figure 1. Interactions between components of context management.

Context Evaluation

Initiating an observable action requires figuring out which pID should be used for this action. Since we are currently considering solely actions implying communication via the network as observable, this component is invoked by the network layer of the BluES'n client. Particularly, the context evaluation component performs the following steps:

- First, it evaluates the received observable action and determines the workspace and functional module in which the invoked action should be performed.

- Subsequently, it selects the corresponding configuration for this "target context" or the suitable default configuration, respectively.

- According to the configuration, it possibly requests further context features from the context monitoring component or the history.

- It evaluates all relevant information in order to select an existing pID or generate a new one. This step might initiate an interaction with the user.

- The pseudonym assigned to this pID is delivered to the network layer which sends the request.

- Finally, the result of the evaluation process as well as the observable action is passed to the logging component which updates the history.

Figure 2 shows the integration of the monitoring and logging component as well as on the evaluation component into the example environment.

Example

A simple example shall illustrate how context management finally works. More detailed examples are given in [7]. For this example, we assume that the user wants to work within an authoring workspace and a self-study class providing the possibility to work with different learning material, perform tests, and participate in a chat. The user decides in advance that he would prefer to work under one and the same pID within the authoring workspace since he wants to be recognized by others working also in this workspace. In contrast, he wants to have the possibility to generate new pIDs when he starts to work in the self-study class. This enables him to start working in an unbiased environment. Finally, he prefers to be asked by the context management before sending a message in the chat so that he has also the possibility to ask questions without the necessity that they are linkable to his other actions.

The user configures the context management for ensuring that he really will act according to these preferences during his every-day actions in the application. For the authoring workspace he defines that the context management shall start a user dialogue suggesting to create a new pID when he wants to open this workspace for the first time. When he opens the workspace again, the context management should automatically select this pID. The system should just show a notification about this choice for enabling the user to be aware of his current privacy state.

For the self-study class, the user makes the same configuration for the case "open for the first time". For opening this workspace again, he wants the context management to start a user dialogue which

- provides the pID used when acting for the last time in this workspace as default pID and

- shows a list of all pIDs already used within this workspace.

The user can accept the suggested pID or select one of the pIDs provided in the list. However, he can also create a new pID, e.g., if he wants to start learning on a new unit without being recognized by others, or select any other pID which is not explicitly suggested. For the functional module chat, the user configures to select the pID before sending a message.

Thus, the user defines in advance in which situation he would like to work under which pID or when it could be useful to switch to another pID. During the actual work, the context management initiates the predefined dialogues reminding the user that he should consider the selection of a pID at this point. Without such predefinition, the user might send messages in the chat without selecting a suitable pID.
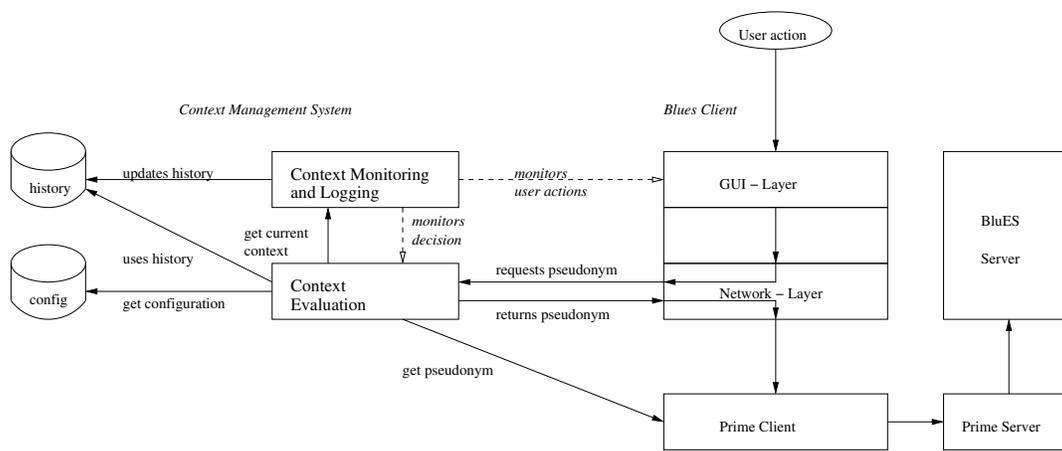
Figure 2. Integration of context management into application.

The suggestions defined in advance help the user to make reasonable decisions when he is actually occupied with performing the other tasks in the application. Furthermore, the interactions invoked by the context management give the user feedback about his privacy state.

POSSIBLE PITFALLS AND LIMITATIONS

Generally, there are always trade-offs between performance and usability on the one hand and privacy on the other hand. First implementations and tests have pointed out that achieving a high level of privacy affects usability and performance significantly. It has to be carefully considered whether approaches to improve the situation influence the achievable privacy. Some encountered problems during implementation shall illustrate this.

Thus, using caching to improve performance might undermine the achieved privacy, e.g., due to unintentionally reuse of transaction pseudonyms. Caching and prefetching might also influence service requests what could be utilized for drawing conclusions about relations between different pIDs. Finally, observable actions might even be omitted if a needed object is still in cache. If the context configuration refers to a sequence of actions, this must be considered. Consequently, influences of caching need to be explicitly modelled.

Another example is communicating awareness information to users. Since the server must not know which pIDs belong to one and the same user, it will send this data to all pIDs. Thus, each user receives this data quite often. Establishing specific channels between clients and the server seems to resolve the problem, but threatens privacy due to possible timing analysis. We suggest to register some of the currently used pIDs for receiving awareness information. Moreover, it is necessary to use them for varying durations, especially to prevent that all pIDs used to receive awareness information are signed off from the channel at the same time when the user closes his application session.

Approaches to increase usability might reveal chronological dependencies. For example, opening all workspaces used in the last application session while starting the application client surely provides the user a comfortable working environment. However, the pIDs used for opening the workspaces in consecutive service requests can be definitely linked and, therefore, former careful partitioning becomes useless. Consequently, we do not automatically open these workspaces. Rather, the user gets appropriate information which is managed solely at client side. A workspace is not opened before the user starts the first observable action in it.

Furthermore, we also have to be aware of possible linking due to the actions themselves. First, an observer could analyze the contents of messages in order to link different pIDs by utilizing typical syntactical or semantical peculiarities [8,11]. Second, the circumstances of observable actions such as their time and frequency might divulge information about users' behavior. The former analysis require a fairly amount of messages. However, both problems can be treated by extending the context feature space by an appropriate attribute. The definition of suitable features concerning the circumstances of actions requires further investigations.

RELATED WORK

There are several approaches known which aim at supporting users in managing their personal data. However, to the best of our knowledge, there are no solutions which could be directly utilized to support IAP.

Some products allow to define different profiles or identities within an application, e.g., browsers such as Mozilla, tools designed for email such as KMail[3] or tools related o chats such as Konversation[4]. However, since communication is a main goal of these applications, support for users in selecting the best suitable identity or profile is not in focus.

P3P[5] (Platform for Privacy Preferences Project aims at supporting a standardized format for describing data handling issues policies. Service providers can specify their privacy

---

[3]http://kmail.kde.org
[4]http://konversation.kde.org
[5]http://www.w3.org/P3P/

policies, and users can define their preferences. Appropriate tools can check whether the policies of a provider match the user's preferences. This aspect can be considered as contextual information, and it could be included in the context feature space especially when we consider a distribution of the application server on different nodes with various privacy policies. However, a main part of the context management discussed here is the possible evaluation of the contextual information, i.e., to define the concluding reactions of the system. This is not in focus of P3P.

Closer to the problems discussed here is the DRIM[6] project (Dresden Identity Management) since it aims at supporting the selection and creation of pseudonyms used within communications based on user defined rules. The definition and evaluation of user defined rules is obviously of interest for the context management. The currently considered rules, however, are not sufficient for IAP since they only consider information about the (technical) communication partner, i.e., the service provider. We are currently checking whether the rule evaluation can be utilized for context management. Nevertheless, there are still tasks which need to be solved by context management: Handling information about the context feature space, measuring the defined features, enabling users to configure context management based on the selected features as well as integrating the context management in the application and realizing the user interactions.

SUMMARY AND OUTLOOK

The proposed realization of context management supports users in partitioning their personal data. Default settings prevent that unexpected situations might cause privacy problems. To reduce disturbances during work, interactions are reduced as much as possible. Particularly, only observable actions can invoke dialogues at all. More experienced users might reduce dialogues to simple notifications about automatic decisions if possible. As a result, usual work flows within the application are less interrupted.

Future work has to be done in order to consider further features of contexts, e.g., information about other users. Timing aspects has also to be thoroughly investigated for answering the question to which degree different pIDs can be more easily linked if they are repeatedly used within short periods of time. In the long run, it could become even possible to link pIDs which belong to one and the same user. We have at least to consider probabilistic conclusions. Currently, we assume that acting under different pIDs within one application session cannot be linked and that it is sufficient to evaluate the sequence of actions assigned to the corresponding pIDs.

User trials are planned in order to assess to which degree the context management described here improves users' continual awareness of privacy problems, whether it motivates the use of PIM at all and its reasonable use.

Acknowledgments

REFERENCES

1. K. Borcea, H. Donker, E. Franz, K. Liesebach, A. Pfitzmann, and H. Wahrig. Intra-application partitioning of personal data. In *Proc. of PEP, Edinburgh, UK*, pages 67–74, 2005.

2. K. Borcea, H. Donker, E. Franz, A. Pfitzmann, and H. Wahrig. Towards Privacy-Aware eLearning. In *Proc. of PET, Dubrovnik, Croatia*, 2005. LNCS, to appear.

3. K. Borcea, E. Franz, and A. Pfitzmann. Usable presentation of secure pseudonyms. In *Proc. of DIM, Alexandria, Virginia, USA*, 2005.

4. K. Borcea-Pfitzmann, K. Liesebach, and A. Pfitzmann. Establishing a privacy-aware collaborative elearning environment. In *Proc. of EADTU*. Rome, 2005.

5. S. Clauß and M. Köhntopp. Identity Management and its Support of Multilateral Security. *Computer Networks*, (37):205–219, 2001.

6. A. K. Dey, G. D. Abowd, and D. Salber. A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications. *Human-Computer Interaction (HCI) Journal*, 16(2-4):97–166, 2001.

7. E. Franz and K. Borcea-Pfitzmann. Intra-application partitioning in an elearning environment – a discussion of critical aspects. Accepted for ARES Workshop Security in E-Learning, 2006.

8. J. Novak, P. Raghavan, and A. Tomkins. Anti-Aliasing on the Web. In *Proc. of the 13th Int. Conference on the World Wide Web*, pages 30–39, 2004.

9. E. Perik, B. de Ruyter, and P. Markopoulos. Privacy & personalization: Preliminary results of an empirical study of disclosure behaviour. In *Proc. of PEP, Edinburgh, UK*, pages 15–22, 2005.

10. A. Pfitzmann and M. Hansen. Anonymity, Unlinkability, Unobservability, Pseudonymity, and Identity Management — A Consolidated Proposal for Terminology. http://dud.inf.tu-dresden.de/ Anon_Terminology.shtml, Feb. 2006.

11. J. R. Rao and P. Rohatgi. Can Pseudonymity Really Guarantee Privacy? In *Proc. of the Ninth USENIX Security Symposium*, pages 85–96. USENIX, 2000.

12. B. N. Schilit, N. Adams, and R. Want. Context-Aware Computing Applications. In *Proc. of WMCSA*, pages 85–90. IEEE, 1999.

13. A. Schmidt, M. Beigl, and H.-W. Gellersen. There is more to context than location. *Computer & Graphics*, 23(6):893–901, 1999.

---

[6]http://drim.inf.tu-dresden.de/