Where to locate user profiles of personalized applications? - A user profile management agent -

Tomonari Kamba and Laurent Freléchoux

C&C Research Laboratories, NEC Corporation
4-1-1, Miyazaki, Miyamae-ku, Kawasaki,
Kanagawa, 216
Japan
{kamba | laurent}@mmp.CL.nec.co.jp

ABSTRACT This paper describes a user profile management agent on the World Wide Web. To help web users to get relevant information efficiently, various sites are providing personalized services and the number of such sites is rapidly increasing. However, deciding where to locate user profiles of personalized applications is a difficult problem. To date, there have been two solutions to this problem: storing user profiles on the server side or storing them on the client side, which is a terminal that runs a web browser. Both these methods have advantages and disadvantages. When the user profiles are stored on the server side, it is impossible for the users to use the same profiles to access other personalized servers. When they are stored on the client side, it is difficult for the server to compare different users' profiles and to provide a so-called collaborative filtering service. Our method solves these problems by storing the user profiles on an agent. The agent keeps the user profiles of a group of users and mediates the communications between the users and personalized servers. When a server requests a user profile, the agent gives it to the server. When a user profile needs to be updated, it is done so on the agent. This agent not only discharges the personalized servers from the user profile management tasks but also gives users the possibility of getting new functions such as collaborative filtering using their own user profiles. We have implemented the user profile management agent as an extended proxy server of the web and have demonstrated that the proxy server is useful for sharing user profiles both between different personalized servers and between different users.

KEYWORDS user profile, personalization, agent, World Wide Web, proxy server

1. INTRODUCTION

We often hear the word "personalization" these days, especially within the context of the World Wide Web (WWW). The word is similar to "customization" but there seems to be a slight difference between the two. For example, the appearance attributes of the X-Window system are called customizable because the user can change them by editing an attribute file. Automatic customization is important for the so-called adaptive user interface. For example, if a system can automatically find an operation sequence that a user does often and if it can create a short-cut for the operation sequence spontaneously, the user interface of the system will be called "automatically customizable". In these cases, most people will agree that we should call the mechanism behind them "customization" rather than "personalization".

Human–Computer Interaction: INTERACT'97 S. Howard, J. Hammond & G. Lindgaard (editors)
Published by Chapman & Hall

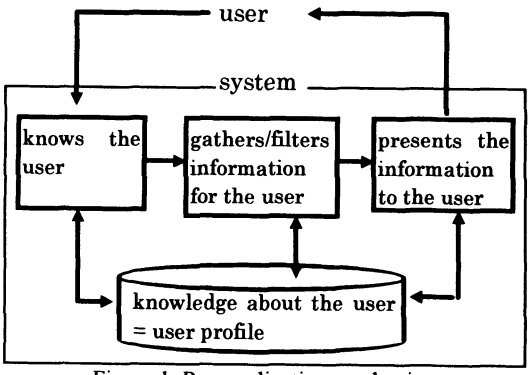

Figure 1. Personalization mechanism

On the other hand, there are some cases that are usually called "personalization", not "customization". One of the typical examples is a personalized news system. It is a system that picks up articles for each user from a large amount of sources so that the user can read relevant articles in a short time.

When we compare these examples, it seems that "customization" is related to the syntactic aspect of the system, and "personalization" is related to more semantic aspects. When the appearance or operations is adapted to each user, it will usually be called customization; however, when the information content is adapted to each user, it will be called personalization. We think the reason why personalized services are so popular now is that people really need them. Contents accessible on the Web are very widespread and heterogeneous, and thus people need a way to filter them. Search engines such as AltaVista (http://altavista.digital.com/) are useful when looking for documents about a specific topic. However, when they want to get relevant (but not so pinpoint) information constantly, as when people read newspapers everyday, a personalization mechanism is useful to filter the information on the WWW.

Then, from the technical point of view, what is needed for personalization? First, a personalization engine has to know the user (see Figure 1). The knowledge about users is usually called the user profile. Then, the engine has to filter or gather information considering the user profile. Lastly, it has to present the gathered or filtered information to the user. As can be seen in Figure 1, one of the key factors of personalization is the user profile.

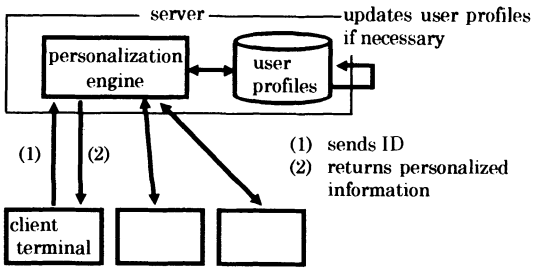

Figure 2. Centralized user profiles on the server

It is of course important to discuss what information should be included in the user profile with what format, but the answer to this question depends on the applications. Sometimes we need the users' names for personalization, and sometimes we may need to know which color they like. On the other hand, where to locate user profiles is a more common problem.

To provide a personalized service, we can build a system that stores user profiles on the server side. It is also possible to store the user profiles on the client side, which is, a terminal that runs a web browser. As we will discuss in the next section, the difference of these two architectures directly affects the system functions, and both have advantages and disadvantages.

In this paper, we will propose another architecture based on this discussion, which is to build a user profile management agent. This agent is placed between servers and clients and manages the user profiles. We have implemented such an agent as an extended proxy server of the WWW and will show how it works.

2. RELATED WORK AND PROBLEM STATEMENT

2.1. Having user profiles on the server

One way to build a personalized service is to manage all the user profiles on the server (see Figure 2). When the user accesses the server, the server identifies the user in some way, retrieves the user's profile from the database, gathers or filters information considering the user profile, and presents the personalized information to the user. The identification of the user is usually done

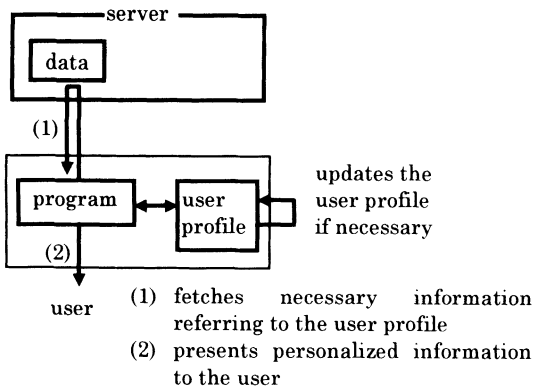

Figure 3. User profiles stored and managed on the client side

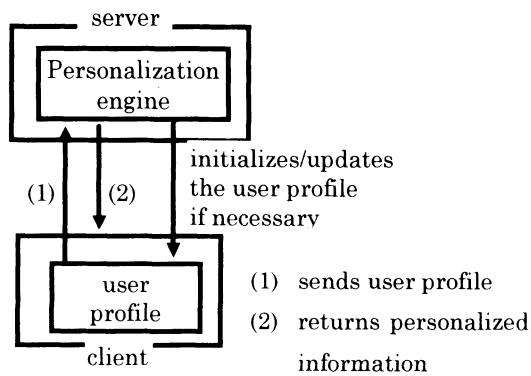

Figure 4. User profiles stored on the client side and managed by the server

by asking the user to type in an ID and password. We can find a lot of applications built with this architecture on the WWW.

The Krakatoa Chronicle developed by Kamba et al. is one example (Kamba, 1995). In this system, the user profiles of all the users are stored on the WWW server. The user can access it with a WWW browser. When a user gives an ID and password to the system, the system scores newspaper articles using the user profile and sends the article contents and article scores to the client. The appearance of the newspaper is created on the client side using the article scores.

2.2. Having user profiles on the client

We will explain the following two methods of storing user profiles on the client side.

- The user profile is stored and managed on the client side.
- The user profile is stored on the client side but is managed by the server.

An architecture for the first method is shown in Figure 3. The server works just as a document database. Considering the user profile stored on the client side, the client program fetches the necessary pieces of information from the server and presents them to the user. This architecture seems to have been adopted by PointCast Network (http://www.pointcast.com/). On the client side, the users can select news categories, areas where they want to know the weather, companies on which they want to get related news, and so on. The files that store such information can be called a user profile. The client program automatically accesses the server and gets articles related to the genres specified in the user profile.

An architecture for the second method is shown in Figure 4. In this architecture, the personalization engine is on the server, and the server manages the user profile but uses a client terminal to store a user profile. When the user accesses the server, the client sends the user profile to the server, then the server personalizes information using the received profile and sends the personalized information to the client. The server also sends a modified user profile to the client if necessary. Personalization with this method can be implemented on the WWW using the Cookie mechanism (http://home.netscape.com/newsref/std/cookie_spec.html). Cookie defines an interface between the server and the client (WWW browser) that manages a special file on the client, called the cookie data. In the cookie data file, the address of a personalized web server is associated with a set of attributes and values that should be sent to the server. Whenever the browser accesses a personalized web site listed in the cookie data file, the browser sends the corresponding attributes and values to the server. When the server sends a message that includes the "Set-Cookie:" string to the browser, the browser modifies the cookie data file based on it.

2.3. Problem statement

Before discussing the problems of current architectures, we will point out one of the popular trends of personalized services. To provide a personalized service to a user, his or her user profile is of course needed. However, it is not enough. Other users' profiles are also very useful for personalization. This is called collaborative filtering or social filtering. Let us look at an example. If a person X likes a book A, the system can analyze the content of the book and deduce what kind of information person X likes. Then, the system can find documents similar to book A and show them to person X. This technique is called content-based filtering and is often used to provide personalized services. Collaborative filtering is quite different from this. When a person X likes a book A, the system looks for other users who also enjoyed book A, and finds what other books are popular among the users who like book A. Based on this information, the system can propose a book that will be relevant to person X without analyzing the content of book A at all. This collaborative filtering has been demonstrated successfully in various systems (Hill, 1995; Shardanand, 1995) and is also an important and powerful technique for personalization. It is important to notice that personalization by this method is quite difficult unless the user profiles are centralized.

The advantages and disadvantages of each personalization architecture will now be discussed.

- **Storing and managing user profiles on the server:** The advantage here is that both content-based and collaborative filtering can be easily performed on the server. One of the disadvantages is that the server needs to have a large amount of disk space to store all the user profiles. Another disadvantage is that the user profile cannot be shared between different web services. For example, if keywords that a user is interested in were stored on the client side, the user would be able to access various personalized services, such as personalized newspapers and magazines, by using the same profile. In addition, there are some privacy concerns associated with this method. In other words, not only the personalized service but also user profiles are controlled by the server with this architecture. This is not desirable from the users' point of view.
- **Storing and managing user profiles on the clients:** With this method, the user can control the user profile and can also use it for another objective if necessary. However, it is quite difficult to get a collaborative filtering service. Since all the user profiles are on the client side, no one can gather all the users' profiles to compare or group them.
- **Storing user profiles on the clients and managing them on the server:** For practical purposes it is almost impossible to do collaborative filtering with this method too. Since the server controls personalization, it would be possible in theory to gather all the users' profiles and do collaborative filtering. However, in comparing and grouping user profiles the server will need to store the same profiles as those on the clients and guarantee the synchronization between the profiles on the server and the clients. This is quite redundant and needs much work on the server.

In short, there is no way of performing both content-based and collaborative filtering services easily, while ensuring the users have reasonable control over their own profiles.

3. A USER PROFILE MANAGEMENT AGENT

3.1. Concept

To solve the problem described in the previous section, we propose a user profile management agent. It stores user profiles of a group of people and mediates the communications between personalized servers and the users within the group (see Figure 5).

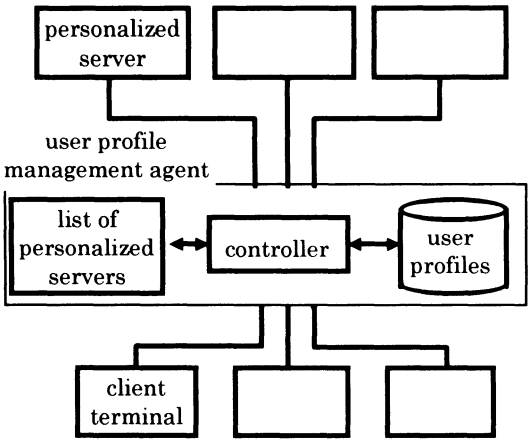

Figure 5. A user profile management agent

The following list shows the functions provided by the agent.

- When the server requires a user profile from the agent, the agent returns the requested profile if it is allowed by the user beforehand.
- When the server asks the agent to forward personalized information to a specific user, the agent does so.
- When the server asks the agent to update a user profile, the agent does so if it is allowed by the user beforehand.
- When a user requires a user profile from the agent, the agent returns the requested profile if it is allowed by the owner of the profile beforehand.
- When a user asks the agent to send his or her profile to a specific server, the agent does so.
- When a user asks the agent to update his or her profile, the agent does so.

In other words, the agent exists in the middle of the users and personalized servers, and both of them can read/edit the stored user profiles if allowed to by the owners beforehand. Therefore, the servers can share the user profiles and the users can share the profiles within a group of people.

3.2. Implementation as an extended proxy server

To show how a user profile management agent works, we have implemented one as an extended proxy server of the WWW. It mediates the communications under the Hypertext Transfer Protocol (HTTP) and manages Cookie-based personalized services. All the user profiles are stored on the extended proxy. The server can set a user profile on the extended proxy in the same way as when it sets the profile on a client terminal using Cookie. When a user tries to access a personalized service that needs the user's profile, the extended proxy server automatically sends the user profile to the server by adding it on a message from the user to the server. Since this extended proxy is transparent for the communication between the users and non-personalized services, it can be used as a gateway between an Intranet within a company and the Internet open to the world. By using this extended proxy server, people within a company can share user profiles. They can even add a collaborative filtering function to the personalized services provided by an outside service provider. This proxy server can also be used by information providers. If a provider provides multiple personalized services, it will be desirable to share customers' profiles among different services. By using our extended proxy server, the user profiles can be centralized on the user profile management agent. Implementation details are described below.

Our extended proxy server is implemented on a UNIX platform using the proxy server developed at CERN as a base (http://www.w3.org/pub/WWW/Daemon). We call it a user profile extended proxy. It works as a regular proxy for any web address (URL) that does not provide a personalized service using Cookie. Whenever the proxy receives an HTTP request for a personalized server, it requires the user to type in an ID and password, then it looks in its database to see if the "URL-user ID" pair is registered. If it is already registered, the content of the user profile is retrieved, added to the HTTP request in the Cookie-format header, and sent to the web server that provides the personalized service (Figure 6). When the server needs to update a user profile, it will include the updated profile again as a header within the regular HTTP response. As this response goes through the proxy, the content of the user profile is extracted and

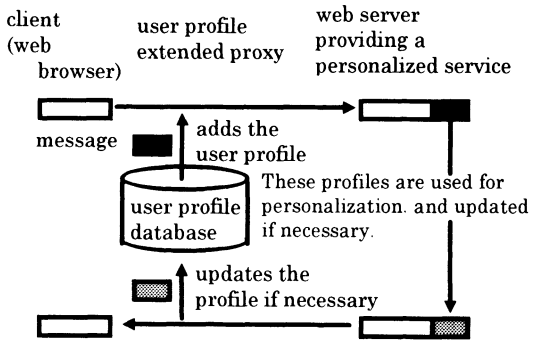

Figure 6. Communications through the extended proxy

stored in the database on a proxy. After extracting the content of the user profile, the proxy forwards the HTTP response to the client. This process assures complete transparency both for the client and the server: they send and receive the same messages as when they do so without an extended proxy.

When a user accesses a personalized service for the first time, the HTTP request will not contain any user profile. The server then informs the proxy, by including a Cookie-format header in the HTTP response, that the server in fact provides a personalized service and needs to store a new user profile. The proxy then adds an entry to the database and stores the user profile. This user profile extended proxy has the following advantages.

- **Ease of server maintenance:** discharges the web service providers from the user profiles management tasks and simplifies the programming of the web services since they are not required to interface with a large database of user profiles. The management tasks are distributed onto multiple proxies.
- **Ease of authentication:** centralizes and simplifies the authentication of the user. Once the user logs in to the proxy, all the log-ins to personalized servers are managed by the proxy. This also allows the user to use different terminals to access personalized web services with the same user profile as long as the access to the Internet is done through the same proxy.
- **User profiles sharing among servers:** enables the user profiles to be shared among different web services. Once a user profile is created for a personalized service, another service can also use it if the owner of the profile allows it.
- **User profiles sharing among users:** enables users to implement collaborative filtering services on the proxy since the profiles of a group of people are centralized on it.

3.3. Application examples

We have implemented two applications that share the same user profiles using our user profile extended proxy. One is a personalized newspaper service and the other is a personalized technical report. These two services are implemented on different web sites and provide different contents, but both use the user profiles of the same format within the mechanism of Cookie. In these applications, a user profile is a set of keywords and their weights. Since a Cookie data file consists of "attributes" and "values", our applications just interpret attributes as keywords and values as weights. When a user is interested in documents that include a specific keyword, the word's weight is high in the user's profile. Each of these systems gathers newspaper articles or technical reports from external sites, extracts keywords from them, and stores them on the server. When a user accesses the server, the server compares the user's profile with the keywords of each document, and scores all the documents. The more keywords in the user profile are included in a document, the higher the score the document gets. When a document gets a high score, it is shown at a former position within a list of document titles (Figure 7).

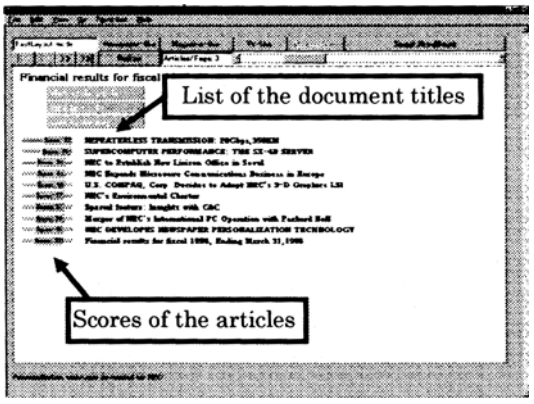

Figure 7. An example screen of our personalized technical reports

If there is no extended proxy, the Cookie file is stored on the client terminal, and the two similar personalized services cannot share the user profile written in this file. When the user profile extended proxy is used, however, the two services can share the user profile. This is useful for both services. Even when a regular user of one of the services accesses the other service for the first time, he or she can get the personalized information because the user profile is already on the extended proxy. For example, if the user has already registered the word "Internet" as a keyword for the newspaper, the user will get technical documents related to the Internet on the other server without setting his or her profile for the personalized technical reports service.

In addition, the user profiles can be shared among multiple users on the proxy. As an example, we have implemented a primitive collaborative filtering service. When this collaborative filtering does not work, the scores of the articles for a user are computed using only the user's profile as described above. When a program to support collaborative filtering exists on the proxy, the program automatically creates a group profile which is the average of all the users' profiles. When the proxy sends a user profile to the server, the program automatically mixes the group profile with the user's profile before it is sent to the server. Generally, a group of people who use the same proxy server often have similar interests, and they exchange ideas to get new information. This collaborative filtering function automatically supports such activities.

It should be noticed that this collaborative filtering function can be added by the users, not by the personalized service provider. By using the extended proxy, even when the server provides only a content-based filtering service, the group of users can start a collaborative filtering service by themselves as long as the user profile format is open to the users and Cookie is used for transporting the user profiles.

4. DISCUSSION AND FUTURE WORK

4.1. Security and privacy

Personalization and privacy are closely related issues. When we want to get personalized services on the Web, there is some possibility that our personal data will be stolen by a malicious third party. However, storing the user profiles at the proxy can be considered secure in the sense that they are stored within a usually trusted organization. In many companies, proxies are used to implement a firewall, and therefore access to the user profiles is impossible from the outside. In any case, storing personal data on the proxy will be better than letting outside providers keep the data.

In addition, in our system, access to the user profile is protected by a password so as to prevent any abuse from the inside. As long as the user is not requesting a URL associated with a user profile, no authentication is required. The first time the user requests a URL associated with a user profile, the proxy requires the user to authenticate him or herself with his or her user ID and password. This happens only once per browser's session, as it does when accessing a restricted web server.

4.2. Storage

As the proxy will centralize the storage of the user profiles, the amount of data to be stored can be significant. If it is considered that the users will use more and more personalized services, the size of each user's profile will tend to increase, but rarely decrease. This problem will certainly be accentuated by users surfing on the Web, accessing personalized services that require storage of a user profile, and never accessing these services again. To avoid storage problems, the proxy

must be able to recycle the disk space for unused user profiles.

One solution that enables recycling is to ask web applications to specify an expiration date for each user profile that they want to be stored. This mechanism is provided by the Cookie. The disadvantage of this solution is that it relies on the "good will" of web applications for setting reasonable values for the expiration.

Another solution, and complementary to the first one, is to stamp each user profile with the current date/time when it is accessed. This stamping is done by our extended proxy and consequently does not rely on the web applications. Whenever a storage shortage occurs, the system starts deleting the oldest user profiles, considering that they are not relevant anymore. In addition to this stamping method, a counter is associated with each user profile, and this counter is increased each time when the profile is accessed. This prevents the omission of a profile that is accessed regularly, for example every month, even if the stamp is the oldest.

4.3. Future work

To date, we have implemented only a primitive collaborative filtering service on our extended proxy, but we are planning to design various filtering services. For example, we are planning to do grouping of our extended proxy users automatically. Even in our laboratory, whose members share fairly consistent interests, there are some groups that have different research themes. Each group member will have similar but different interests. To get group scores, we should consider which group a user belongs to.

In order to make our extended proxy widely useful, it is required that more personalized service providers make the format of their Cookie files open to public. Then, similar personalized services can share the user profiles, and the users need not register their profiles for each service. The opening of the format of the Cookie file will be necessary also when we consider the privacy issue.

5. CONCLUSION

We have pointed out the problems of current user profile management architectures and have proposed a user profile management agent that mediates the communication between personalized servers and users. This agent stores user profiles in itself and manages them. We have implemented the agent as an extended proxy server of the WWW. By using this extended proxy server, different personalized services can share the same user profiles, and at the same time, a collaborative filtering service can be implemented on the proxy server.

Personalization is an important technology to help people to get relevant information efficiently, but it should be implemented deliberately since it relates to the privacy issue. We believe our user profile management agent helps web service providers to manage personalization services more easily than now and helps users to get useful services such as collaborative filtering under their own control.

ACKNOWLEDGMENT

We would like to thank Shiro Sakata and Yoshiyuki Koseki for their constant support during our research. We would also like to thank Hidekazu Sakagami for his help in building our personalized applications.

REFERENCES

Hill, W., Stead, L., Rosenstein, M., and Furnas, G. (1995) Recommending and Evaluating Choices in a Virtual Community of Use, *CHI'95 Conference Proceedings*, pp.194-201.

Kamba, T., Bharat, K., and Albers, M.C. (1995) The Krakatoa Chronicle - An Interactive Personalized Newspaper on the Web, *Proceedings of the Fourth International Conference on the World Wide Web*, pp.159-170.

Shardanand, U. and Maes, P. (1995) Social Information Filtering: Algorithms for Automating "Word of Mouth", *CHI'95 Conference Proceedings*, pp.210-217.