Management of Ambient Media Preferences in Distributed Environments for Service Personalization

M. Anwar Hossain, Pradeep K. Atrey and Abdulmotaleb El Saddik
Multimedia Communications Research Laboratory (MCRLab)
School of Information Technology and Engineering (SITE)
University of Ottawa
{anwar, patrey, abed}@mcrlab.uottawa.ca

Abstract

Due to the increasing need for service personalization in pervasive environments, the management of user’s contextual media preferences becomes an important issue. In this paper, we present a framework for managing ambient preferences, which is a continuously updated set of attributes reflecting user’s need and expectations for different media services in different contexts. These attributes are dynamically collected and updated based on the interactions of the users in distributed environments. The proposed framework provides mechanisms to store and share these preferences in order to provide personalized services to the user according to the different contextual situations. We demonstrate the development of the framework, its implementation and usage in distributed environment settings.

1 Introduction

An ambient and ubiquitous environment is a distributed interconnected environment, where a user needs ambient media services in a personalized manner. The ambient media services are those media services that can be provided in the surrounding environment depending on the context the user is in. Due to the availability of numerous heterogeneous media services, providing the user with a personalized set of ambient media services becomes a challenging task.

In the distributed ubiquitous environment, a user normally has different preferences in different contexts that include location, time, activity, and other factors. For example, a user while driving would like to listen to music instead of watching a movie. Moreover, the user’s ambient media preferences change over time, and hence it need to be updated regularly based on his/her interaction with the environment. For example, if a user has seen a movie today, he or she may not like to see it again in near future. Therefore, it is very important to address the management issues of the contextual media preferences and their changes over time.

Our work in this paper is motivated by the existing research in this area [1, 2, 3, 4, 5]. These works present various mechanisms to build a user model that captures the user’s preference for different services including media services. Compared to these, we particularly focus on the management of media service preferences and propose a framework for such management. The proposed framework is based on a simple yet comprehensive preference model that can integrate the user’s context and need for media services. We show how the ambient media preferences evolve over time based on the user’s past interaction with the environment. Besides, our framework also addresses other management issues such as storage, distribution and sharing of ambient media preferences among disparate locations to include the user’s mobility and presence.

The remainder of this paper is organized as follows. Section 2 comments on some related research followed by Section 3, which elaborates on the proposed management framework. In Section 4, we present experimental results. Finally, Section 5 concludes the paper with a discussion on future research issues.

2 Related work

Existing research in the area of user profile management, personalization and user modeling are closely related to the work presented in this paper. In this section, we comment on some of these works.

McBurney et al. [5] presented a rule-based approach to define users’ preferences for service selection, call redirection and network/device selection. The approach is adopted in the Daidalos project aiming to develop a pervasive service platform and provide flexibility in service selection. This work, however, does not explicitly show how the con-
textual media preferences can be set and maintained over time.

A relevance-feedback based preference learning mechanism has been presented in [6] for multimedia personalization. In this approach, the higher capability device (master) is used for learning the users preference based on the feedback provided by the low-capability portable devices (slave) by observing the users behavior. This work is related to ours in that it enables the user’s preference, specifically the multimedia preference, to be updated over a period. However, unlike ours, it does not specify how the preferences are maintained among different contexts.

In a recent work [3], the authors described a profile management approach to enable context-awareness in personalizing applications and services in the context of the EU-IST MobiLife project [7]. The proposed mechanism adopts the Doppelganger user modeling system [8] by defining domain and conditional submodels. The domain submodels are associated with a specific application as a default profile and the conditional submodels are the contextual counterparts of the domain submodels. We can draw a relationship with this work to ours in that our model represents the user’s ambient media preferences based on context, which is logically viewed as conditional submodels. On the other hand, the summarization of the contextual ambient media preferences can be treated as domain submodels. However, the work in [3] does not provide any formal description of the way the profile values are updated dynamically. Also, there is no indication of the distributed management of user profiles and preferences.

Authors in [4] presented a profile management framework for customizing smart home services in a context-sensitive manner. In this work, user profiles are managed as a set of pre-defined and on-demand profiles, where the former represents user preference and device related information and the latter represents current contextual information such as sensory information, device status and so on. The authors also talk about the dynamic evolution of profiles and address some conflict resolution approaches they adopt for the multi-user smart environment. A slightly different work [9] in the context of mobile service provisioning also addresses several aspects of profile and preference management issues. However, both [4] and [9] do not explicitly elaborate how the profiles are represented and managed. Therefore, it is not clear how these approaches can be adapted for the management of ambient media preferences in the pervasive environment.

Heckmann [1] introduced the General User Model Ontology (GUMO) for improving service personalization in a ubiquitous computing environment. The GUMO allows uniform interpretation of distributed situational information using semantic web technologies. This work mostly focuses on interoperability issues by allowing the exchange of partial models between applications through the use of UserML - a user model markup language. Contrary to this approach, we describe a simple model that can be adopted to personalize the user’s media preferences in a distributed environment context. We also develop a framework for the management of ambient media preferences and elaborate several related issues such as representation, update, storage, and distribution of preferences.

3 Proposed framework

The proposed management framework for ambient media preferences enables the creation, update and distribution of media preferences in distributed environments. In order to explain the management of ambient media preferences, we need to first elaborate the context and ambient media preference models that we use. The following section explains these models.

3.1 The context model

In the proposed framework, context \( C \) is an important factor for service personalization. This is because a user’s preference changes over time based on the context that she/he is in. The context can be characterized based on the existing works in context-awareness, particularly the work in [10], [11]. Accordingly, we define context that can be represented by different attributes such as where, when, what, with whom, and what mood. The where attribute represents the user’s location \( l \), when refers to time of presence \( t \), what refers to the user’s activities \( a \), with whom refers to the user’s companion \( c \) and mood refers to the user’s psychological state \( m \). Each of these context attributes \( l, t, a, c, m \) can have different values, thereby referring to many contextual situations. For example, the value of the location attribute \( l \) for the distributed environments can be different such as home, office or any other specific place, which would mean different contexts. We will show in Section 3.2 that based on the different contexts, a user may have different preferences for the media services in their ambient environment.

3.2 Ambient media preference model

The ambient media preference (or simply AMP) is the evolving part of the general user profile, which reflects the user’s preference of media services in different context. This is dynamically updated based on the user’s interaction in distributed environments. The other part of the user’s profile contains static user-related parameters such as name, age, sex, address and so on. These parameters are considered private and will not be shared among applications without the approval of users. Unlike ambient media
preferences, the static user-related information does not frequently change over time, if any.

We model AMP in a particular context $C_i$ for a user $U_r$ as $AMP(C_i, U_r) = \{p_1, p_2, \ldots, p_k\}$, where $\{p_1, p_2, \ldots, p_k\}$ refers to the set of preferences on the different attributes of media services. Each preference element $p_j$, $1 \leq j \leq k$ represents a tuple $<MT, A_q, S_q>$, where $MT$ is the type of media service, $A_q \in U$ is the attribute of interest (e.g. genre), which will have different data items in its value domain (e.g. action) and is represented as $dm(A_q)$. Each data item will have an associated score as a value of $S_q$. These scores are maintained as a normalized percentile scale. The value of $A_q$ and $S_q$ may change over time depending on the user’s changing preference, which is obtained based on interaction history. Note that the AMP scores do not explicitly refer to the preference of a media service; rather it provides an abstraction to the preferences of a user for all media services from a meta level. Note the different types of media services (e.g. movie, music, RSS feed, camera feed) are usually represented by corresponding metadata attributes. Fig. 1 shows a portion of the AMP of a user in a particular context.

The AMP of the user changes over time for different contexts; so do the scores associated with the preference attributes. Such dynamics are quite natural in real life. However, adapting these changes in distributed environments in a manageable way is not a trivial issue and raises several management challenges. Such as 1) How to dynamically update the AMP? 2) Where to store the AMP? 3) How to globally update AMP based on local updates? 4) How to summarize the AMPs of all users in different contexts? 5) How to distribute and share AMP? In the following section, we address these issues in turn.

$$p_j = <MT, A_q, S_q> = <MT, genre, score>$$

Sample value domain of $p_j$:

<table>
<thead>
<tr>
<th>MT</th>
<th>genre</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movie</td>
<td>Action</td>
<td>50%</td>
</tr>
<tr>
<td>Movie</td>
<td>Comedy</td>
<td>20%</td>
</tr>
<tr>
<td>Movie</td>
<td>Romantic</td>
<td>30%</td>
</tr>
</tbody>
</table>

$$p_x = <MT, A_q, S_q> = <MT, feedType, score>$$

Sample value domain of $p_x$:

<table>
<thead>
<tr>
<th>MT</th>
<th>feedType</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSS Feed</td>
<td>Football</td>
<td>40%</td>
</tr>
<tr>
<td>RSS Feed</td>
<td>Hockey</td>
<td>40%</td>
</tr>
<tr>
<td>RSS Feed</td>
<td>Cricket</td>
<td>20%</td>
</tr>
</tbody>
</table>

Figure 2. Example scenario of movie service usage over time

3.3 AMP management

Here we sequentially go through the abovementioned challenging issues of AMP management.

3.3.1 Dynamic update of AMP

For setting up the ambient media preference for a user, a primitive approach is to show the user a visual interface and ask them to explicitly provide the AMP in different contexts. This approach serves the initial purpose; however, it requires a mechanism to dynamically update the AMP over time. To handle this case, many techniques can be adopted. In this paper, we only show one example on how to gradually update the AMP based on the user’s interaction data referring to media service usage.

Let us consider Fig. 2, which shows a sample scenario of movie service usage in a smart environment. In order to add a new data element or update existing data in AMP, we first define a relative weight factor ($\delta$) based on service usage data. The $\delta$ is defined as the ratio of the number of times a data item of a particular attribute of a media service appears in the service usage to the total number of occurrences of that particular type of media service. For example, after time instance $t_4$, $\delta_{\text{action}} = 2/4$, $\delta_{\text{comedy}} = 1/4$, $\delta_{\text{romantic}} = 1/4$. The normalized scores of each of the data items within each attribute are computed using Eq. 1.

$$w'_x = \frac{w_x + \delta_x}{\sum_{x=1}^{n} (w_x + \delta_x)}$$

where, $n$ = total data items in each attribute. $w'_x$ and $w_x$ are the updated and existing score of each of the data items, respectively. Using eq. 1 we can gradually update a preference element $p_i$ (e.g. genre) with the data item action, comedy and romantic and their scores. Similarly, we can also update other preference elements such as actress and actor. Note, it is a design choice to determine whether to update AMP after every instance or several instances of service usage. Accordingly, a window of interval can be defined and its size can be adjusted depending on the specific application in mind.
3.3.2 Storage of AMP

The AMPs that are dynamically updated in different contexts also inherently refer to distributed sites. This is due to the location attribute in the context definition. Therefore, based on the location of the user’s surroundings, the AMP will be stored. Based on the analysis of the user’s mobile location and the computing capability in the surroundings, different AMP repositories can be created - local and global. Fig. 3 shows this case. Note that, the AMP repository at one location may also be used zone-wise, referring to the sharing of repositories by the distributed sites in a certain geographical area.

3.3.3 Update of the global AMP

The global update of the AMP is an important functionality of the proposed AMP management framework, which enables the distributed pervasive environment to obtain the latest changes in the AMP in a particular context. The global update can also be termed as AMP Synchronization. In the proposed framework, we maintain a global AMP repository that is often consulted by the systems in distributed sites to retrieve the AMP that matches a particular context. Therefore, the global update functionality is used to update the global AMP repository with the local updates that happened in the distributed sites. This would keep the AMP synchronized among different repositories for different sites.

3.3.4 Creating AMP summary

As discussed, AMP can be stored in different repositories in distributed sites. Also, recall that the AMP is associated with a particular context and user. Therefore, an interesting challenge is how to summarize the AMPs available on all contexts for all users by considering the AMPs stored in the distributed repositories. To address this, we first decide on a summary model for the AMP, which is similar to that of the AMP model as described in Section 3.2. However, this summary is not generated based on interaction data, rather the different contextual AMPs are aggregated together to obtain this summary. Several aggregation strategies can be followed for this case. Here, we use a simple average to merge the scores of the data items of the AMP attributes.

This AMP summarization approach can be adopted at different levels of abstraction. For example, we can make a zone-based summarization of AMPs to know more about the media preferences of different users in a zone. Also, we can produce a global summary, a context-wise summary, a group-wise summary and so on. Note that the summarization of AMPs provides efficiency and flexibility in different situations in addition to higher-level understanding of media service usages. For example, consider a new user is identified by the ambient system in a particular context for which there is no matching AMP. In such a case, the system can prompt the user by providing a suitable form of AMP summary, which can be set as the user’s initial preference upon his/her approval. Another important note is that the summarized AMP could be considered as the higher-level default AMP, which is dynamically evolved and can be provided to the user when no matching AMP is available.

3.3.5 Distribution and sharing of AMP

In a distributed environment, there are many cases when the AMP from one site needs to be distributed among other sites. For example, a mobile user might want to access his AMP from the remote site, which would require the remote site to send the matching AMP to the mobile context. In other cases, the unavailability of the local AMP repository would require access to the AMP from the global repository located in a different site. Therefore, AMP distribution and sharing becomes an important issue. However, one thing need to be clear that the AMP is all about the user’s media preferences and hence requires mechanisms to handle them in a privacy-preserving manner.

3.4 System architecture

Fig. 3 shows a high-level architecture of the proposed framework. It is based on a service-oriented architecture, where the functionalities are developed as a service. Every local site, where the user is located, logically represents a distributed site in the proposed framework due the changing location of the user. In general, the computing facility
of the local site is capable of performing several functionalities. In the following, we first describe the services that are executed at any local site. We then mention the special case when the local site architecture requires global support.

In Fig. 3, the Context Capture and Identification Service at the local SITE is responsible for capturing the different context parameters using inputs from multiple sensors (e.g., proximity sensor, GPS, and camera). As sensory data often provides uncertain readings, there are cases when the system will only be able to identify a few context parameters. The detail of a context capture mechanism is out of the scope of this paper. Nevertheless, the identified context parameters are provided as inputs to the Local AMP Retrieval Service, which extracts the matching AMP from the Local AMP Repository. The retrieved AMP is then sent to the Local Media Recommendation Service, which determines the best set of media services to be provided in the given context. Note, in case of imprecise context information, the system retrieves a set of AMPs from the Local AMP Repository. These are used by the Local Media Recommendation Service to determine the best matching AMP to select the media services that would maximize the user’s gain.

The recommendations that are made by the Local Media Recommendation Service is passed to the Ambient Media Execution Service, which follows a suitable strategy to execute the recommended media service. The Ambient Media Execution Service uses different criteria, such as whether or not two media can be scheduled together, to personalize the user’s media preferences. While the media services are used by the system, their usage history is logged by the Usage Log Service. The usage log history is used by the Local AMP Update Service to update the AMP of that particular user and context. The update follows the procedure described in Section 3.3.1. Note the Local AMP Update Service does not consider all the usage history to update the AMP, as some usage logs do not actually reflect the true usage of a media service. The Local AMP Update Service follows different criteria to ensure this, for example, the service usage duration.

The local site architecture as described above would be the ideal case for any distributed sites. However, there are cases when the local site cannot perform the intended functionalities due to lack of computing facility. Besides, it might need the support of global services in different situations. For example, a user with a mobile device might be in a context for which there is no matching AMP at the local site. The Local AMP repository might not be deployed or is empty. Therefore, the local site’s services need to communicate with the global site for obtaining the user’s preference. In this case, the Local AMP Retrieval Service would request the AMP Summary Retrieval Service for an updated copy of the matching AMP based on the current context. Note the AMP Summary Retrieval Service can provide a summary AMP from different levels of abstraction, which can be used to suggest the initial AMP for new users for unknown contexts. The Local Media Recommendation Service can also communicate with the Global Media Recommendation Service. Both local and global recommendation services can recommend media services for the user based on the retrieved AMP and the available media services. The metadata of the media services are registered in the Local and Global Media Service Repositories, respectively. In addition, the Local AMP Update Service can communicate with the Global AMP Update Service to keep a copy of the preferences in the Global AMP Repository and subsequently update them over time. In Fig. 3, communication with the local and global site is shown with the dotted lines.

4 Experimental results

We have implemented a prototype of the proposed framework by implementing the basic functionalities as services. The Microsoft .Net framework has been used for the development. User information, context parameters and media service metadata are stored in a MySQL database. The AMPs are stored in an XML database. In the following we show a test case where the mobile system identifies a user at a particular context and sends the context parameters to the global AMP retrieval service to fetch the AMP for that particular user. Note, the context identification is not always straightforward, meaning the sensory system at the mobile site might be able to identify only partial context parameters. Therefore, based on the partial context parameters, the global AMP retrieval service obtains the best matching AMP and makes recommendations for appropri-
Figure 5. Snapshot of the service usage logs

rate media services. The AMP and the recommendations are sent back to the requested parties, where the ambient media execution service executes the media.

Let us consider Fig. 4(a) as an example of the fetched AMP of the particular user. Among different preference attributes, the figure only highlights the feedtype preference for the RSS Feed and genre preference for movie service. Based on the preferences in Fig. 4(a), the system makes recommendation to execute the different media services. However, the user has full control to change the scheduled media execution. In any case, the media usages are recorded in log files. These logs are analyzed by the local AMP update service to update the initially fetched AMP. Fig. 5 shows an example of the logs created from the user’s interaction history. Note, the interaction history logs presented here are collected at different timestamps, but in similar contexts.

The data in Fig. 5 are used to update the AMP using eq. 1. The updated AMP is shown in Fig. 4(b). Note, to obtain the AMP in Fig. 4(b) from the initial AMP in Fig. 4(a), we have applied the update procedure after every few instances of service usage. The updated AMP scores would be different if the number of instances (window of interval) is varied. The number of instances after which the AMP update procedure should be executed can therefore be experimentally determined. It also depends on the specific application scenario. Nevertheless, the aforementioned results are presented to show an example of how the AMP may change over time based on the usage data.

5 Conclusion

This paper presents a distributed ambient media preference or AMP management framework for providing personalized media services to the user. We have given the model of the AMP and described several issues that arise for managing such a preference. In particular, we have addressed the issues such as dynamic update, storage, summarization, distribution and sharing of AMP. We have presented experimental results to demonstrate how AMP evolves over time based on the user’s interaction data. Overall, the proposed work shows new possibilities. However, there are areas for further improvement, such as, to extensively test the developed functionalities and conduct large scale user testing.

References