Electronic Communications of the EASST
Volume 11 (2008)

Proceedings of the First International DisCoTec Workshop on Context-aware Adaptation Mechanisms for Pervasive and Ubiquitous Services (CAMPUS 2008)

Context-Aware Adaptation of Mobile Multimedia Presentations

Diana Weiß, Stefan Helas, Johannes Martens

6 pages

Guest Editors: Romain Rouvoy, Mauro Caporuscio, Michael Wagner
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Context-Aware Adaptation of Mobile Multimedia Presentations

Diana Weiß, Stefan Helas, Johannes Martens

Mobile & Distributed Systems Group, Department of Computer Science,
Ludwig-Maximilians-Universität München, Germany
diana.weiss@ifi.lmu.de, helas@cip.ifi.lmu.de, johannes.martens@ifi.lmu.de

Abstract: With increasing capabilities of mobile devices, the number of mobile multimedia services grows steadily. Usage of multimedia services and especially the presentation of multimedia content is more challenging in a mobile environment than on stationary devices due to the diversity of mobile devices and their parameters, the sparse resources of the air interface or the changing context.

This position paper introduces an approach for context-aware adaptation of mobile multimedia presentations. For this, a middleware architecture is proposed that performs an automatic and distributed adaptation process: while the server provides a pre-processing of the content, the client completes the overall adaptation. To support this task, the *Adaptation Rule Language* (ARL) was developed which is introduced in this paper as well.

Keywords: context-awareness, adaptation, multimedia, middleware

1 Introduction

Because of a constant development over the past years, present-day mobile devices provide more sophisticated capabilities like faster CPUs, colored and bigger-sized displays, and increased memory size. Additionally, mobile network operators launched 3G networks in many countries that provide higher bandwidth and support extended multimedia services. Compared to stationary terminals, mobile devices pose new challenges to the usage of these services: the battery longevity delimits usage time, smaller displays limit presentation potential, the sparse resources of the air interface restrict the content size etc. The diversity of mobile devices causes a diversity of existing device parameters in a mobile environment and tightens this problem. Besides these device related limitations, the user's mobility and the changing context also affect the usage of mobile multimedia services: based on the current situation the user may prefer different types of presentation. A user of a mobile video service e.g. may want to have subtitles instead of audio when using public transportation or just audio information when battery power is low.

This position paper introduces an approach to adapt the presentation of mobile multimedia content according to device parameters, user preferences and especially to the current context. For this, a middleware architecture is proposed that distributes the adaptation process to the server and client part. On the one hand, servers have much more processing power and can achieve content adaptation much more efficiently. Additionally, server-side processing prevents the transmission of unnecessarily large media files over the air interface that are converted by the mobile device into a much smaller file. On the other hand, user's privacy concerns and scalability militate for adaptation at the mobile device. To combine those advantages, we decided for

distributed processing: the server pre-processes the content and the client completes the overall adaptation process. To support the middleware, the XML-based *Adaptation Rule Language* (ARL) is introduced as well. This language allows the definition of rules to specify content adaptation based on device parameters, user preferences and context information.

The remainder of the paper is structured as follows: an example scenario is given in Section 2. It shows requirements the proposed approach has to meet. Section 3 gives an overview of related work. The middleware and ARL are introduced in Section 4. In the last section, the paper provides a short conclusion and overview of future work.

2 Scenario and Requirement Analysis

Alice is sitting in the university library and wants to get informed about the latest news on the stock market. Thus, she accesses her favorite multimedia news portal using her PDA. Alice chooses an audio-visual report. Since she is sitting in a library, spoken text is automatically replaced by subtitles, so other people are not disturbed. Later at home, she watches a stock report on her laptop, this time getting a fullscreen high definition video and audio output. Suddenly the video is replaced by a smaller sized slideshow, because the battery power is getting low. Alice plugs in the power cord and regains the fullscreen high definition video.

This scenario shows, that several requirements need to be fulfilled by the proposed approach. First of all, **static device parameters, user preferences as well as the current context have to be taken into account** when adapting the multimedia content. Static device parameters are based on the given characteristics of the mobile device like the screen size, available codecs or installed applications. User preferences describe how the user prefers the content to be adapted, like bigger font size or every text should be read out. Context refers to all dynamic parameters of the device like the remaining battery power and the user's surrounding like the current location. Static device parameters, user preferences and the context determine the current situation. **Situations are processed rule-based**: *if* a certain situation appears, *then* something has to be done, e.g. *if* battery power is low, *then* high definition videos have to be avoided. Rules should be given as default rules as well as specified by the user.

Since it is not very efficient to store a particular file for all possible types of presentation, **content should be adapted** with respect to device parameters, user preferences and the context. This can be achieved by different methods that have to be supported by our approach. *Scaling* can be used to fit content to the device's screen, while *conversion* is needed, when the original multimedia format cannot be presented by the device, e.g. due to missing video codecs. Other methods adapt multimedia content by *removing*, *replacing* or *adding* parts of the content.

3 Related Work

The adaptation of mobile multimedia presentations has been taken up in several systems:

Opera Mini [Ope], a mobile web browser, uses a proxy to pre-process requested web pages. The adaptation technique is incorporated in the Extensible Rendering Architecture [Wil05]. It supports adaptation based on device parameters, particularly the screen size, but not on any context information or user preferences.

The Niccimon platform [Bal04] is a modular system supporting a flexible and rapid development of location-aware mobile applications. It considers a few user preferences for the adaptation process, but no context information or device parameters.

MobileMM4U [SB04], a framework based on MM4U [Bol03], enables personalized mobile multimedia presentations with focus on the special needs and characteristics of mobile devices. All adaptation parameters described in Section 2 are taken into account. In contrast to our requirements, for all possible types of presentation, a single media file is stored. The system does not actually perform adaptation of content but selects the file matching the parameters best.

Jannach et al. introduce in [JLTH06] a knowledge-based approach for building multimedia adaptation services based on the MPEG-7 and MPEG-21 standard respectively. Although this approach is technically quite mature in the adaptation processing, it does not support context.

The introduced systems are a first step toward adaptation of multimedia presentations, but do not fully meet the requirements determined in Section 2. Either context information is not considered or no real adaptation is performed like in our approach.

4 Context-Aware Adaptation of Mobile Multimedia Presentations

In this section, a middleware architecture that supports context-aware adaptation of mobile multimedia presentations is introduced. As already mentioned in Section 1, the adaptation is distributed among the server- and the client-side part of the middleware. While the server provides a pre-processing of the content based on information sent by the client, the client completes the overall adaptation. The middleware is located between the applications and services respectively and the connection management (see Figure 1). It was designed to allow integration into existing architectures and offers according interfaces to services, applications and the connection management. Due to a lack of space, applications, services and the connection management are not further considered in this paper, but the middleware components of client and server are introduced. Subsequent to this, the Adaptation Rule Language (ARL) is presented that supports the middleware's adaptation process. This language was designed for defining rules that specify content adaptation based on device parameters, user preferences and context information.

4.1 Client-side Middleware Components

The **Presentation Proxy** is the client-side communication interface of the middleware and keeps the adaptation process transparent to the applications (see Figure 1). It makes sure that all communication with the multimedia service is performed exclusively via the middleware. Additionally, the Proxy can request the application's current status which may be important for the adaptation process. Communication between Presentation Proxy and application is to the application the same as a direct communication to the service itself.

The adaptation process is based on device parameters, user preferences and context respectively. Each of these three adaptation parameters is managed by a particular middleware component to allow access to these different information each by a single interface. Device parameters like screen resolution, installed codecs, available applications etc. are stored by the **Feature Registry**. This information is mostly static but may change e.g. with the installation of new

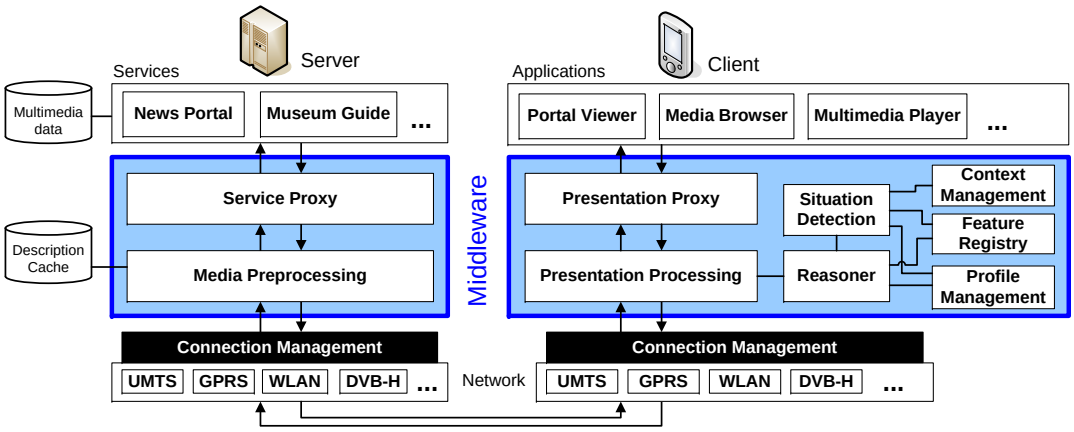

Figure 1: Client- and Server-side Parts of the Middleware Architecture

software or the establishment of a connection to peripheral devices like an external display. User and device context information is sourced and managed by the **Context Management**. Additionally, it should manage maps and floor plans of buildings, allow the definition of several user zones and perform the mapping of user's coordinates to these zones, but this is still part of future work. The **Profile Management** manages all user preferences. These specify e.g. if the user prefers subtitles instead of audio or if she prefers higher video quality at the expense of an increased transmission time. The user can also specify the behavior of the service in different context as well as her privacy preferences given as rules similar to W3Cs APPEL [LCM02]. For simple handling of preferences, the Profile Management allows import of default preferences and provides a user interface to change existing preferences or enter new ones.

Device parameters, user preferences and context are processed by the **Situation Detection** to determine the given situation. This information is passed to the **Reasoner** that selects matching rules and creates guidelines with respect to privacy preferences and device parameters e.g. *what kind* of content has to be adapted *in which way*. Situations and rules are specified using ARL (see below).

The actual decisions are made by the **Presentation Processing** that matches the guidelines given by the Reasoner, the application's status retrieved by the Presentation Proxy and the content. When starting a specific service, the Presentation Processing sends static information (as device parameters) to the server that is stored during the overall service usage time. Dynamic data that is expedient and not classified as private by the user's privacy preferences is packed into an *Adaptation Description* and sent whenever content is retrieved. For all sending processes, the Presentation Processing has access to the connection management. Additionally, it performs the actual client-side adaptation based on the given rules.

4.2 Server-side Middleware Components

The **Service Proxy** is the server-side communication interface of the middleware (see Figure 1). The task of pre-adaptation is realized by the **Media Preprocessing** component. Every content

processing is based on the information that was sent by the client on registration and stored in the **Description Cache**, and the *Adaptation Description*s received when content is accessed by the application. Depending on this data, the server can start pre-processing like removing audio or scaling the resolution of the content and sends the pre-processed content to the client using its interface to the connection management.

4.3 Adaptation Rule Language (ARL)

For the definition of rules specifying a specific content adaptation processing based on device parameters, user preferences and context information, ARL, an XML-based description language was designed. A simple example of ARL can be seen in Listing 1.

```
<Situation id="lowBatteryPower">
 <SituationInformation> <Context type="batteryPower"
    scale="Ah" value="0,075" constr="max"/>
 </SituationInformation>
</Situation>
<Behavior id="subtitlesInsteadOfAudio">
 <Connector name="and">
  <Function name="remove"><Object name="audio"/></Function>
  <Function name="add"><Object name="subtitles"/></Function>
 </Connector>
</Behavior>
<Rule bref="subtitlesInsteadOfAudio" sref="lowBatteryPower" priority=
    "medium"/>
```

Listing 1: Situation, Behaviour and Rule Examples

`Situations` contain one or more `SituationInformation` tags. These tags contain `DeviceParameter`, `UserPreference` or as shown in Listing 1 `Context` tags. According to the Aspect-Scale-Context model [SLF03], context is modeled using the attributes `type` specifying the context type (e.g. batteryPower), `scale` setting the used scale (e.g. `Ah` for ampere hour) and `value` providing the current value. The optional attribute `constr` can be set to restrict context values. If the attribute is e.g. set to `max`, the actual value may not be higher than the given value to belong to the corresponding situation. Different `SituationInformation` tags can be merged using boolean `Connectors` like `and` as well as `or` as seen within the `Behavior` tag in the listing. So far, `UserPreference` tags are designed simply, but already include privacy preferences similar to APPEL. The most important `DeviceParameters` like screen size and codecs are already considered as well. Extension of these descriptions is part of future work.

In different situations, the middleware should process presentations in different ways. The type of adaptation is specified by the `Behavior` tag. Within this element, several `Function` tags can be merged using different boolean `Connectors`. These descriptions of functions are mapped to the actual adaptation functions in the implementation. Every function refers to a specific `Object` and can have `Parameters` given to it (not shown in the listing).

The `Rule` tag relates a defined `Situation` to a specific `Behavior` by referencing to both elements. The `priority` of the rule can be set by assigning the according value `high`,

`medium` or `low` to the corresponding attribute. Further conflict handling of contradictory rules is part of the future work as well.

5 Conclusion and Future Work

In this position paper, an approach was introduced that allows adaptation of mobile multimedia presentations according to device parameters, user preferences and the current context. For this, a middleware architecture is proposed that allows distributed processing of the media adaptation and the XML-based language ARL to support this task.

Since this contribution describes work in progress, some tasks still have to be completed. These contain amongst other extension of ARL and conflict handling mechanisms of contradictory rules as described in Section 4.3, the mapping of user's coordinates to locations and efficiency considerations towards the distributed adaptation process. A main task can be found in the completion of the prototype implementation that is based on the .NET Framework and SMIL 2.1 as an open multimedia presentation format.

Bibliography

[Bal04] Baldzer, J., et al. Location-Aware Mobile Multimedia Applications on the Niccimon Platform. In *Proc. of 2nd Symposium on Informationssysteme für mobile Anwendungen (IMA)*. Pp. 318–334. 2004.

[Bol03] S. Boll. MM4U - A Framework for Creating Personalized Multimedia Content. In *Proc. of Int. Conf. on Distributed Multimedia Systems (DMS)*. Pp. 12–16. 2003.

[JLTH06] D. Jannach, K. Leopold, C. Timmerer, H. Hellwagner. A knowledge-based framework for multimedia adaptation. *Applied Intelligence* 2:109–125, 2006.

[LCM02] M. Langheinrich, L. Cranor, M. Marchiori. APPEL: A P3P Preference Exchange Language. W3C Working Draft, 2002.
http://www.w3.org/TR/P3P-preferences/

[Ope] Opera Software ASA. Opera Mini™.
http://www.operamini.com

[SB04] A. Scherp, S. Boll. MobileMM4U - Framework Support for Dynamic Personalized Multimedia Content on Mobile Systems. In *Proc. of Techniques and Applications for Mobile Commerce (TAMoCo)*. Pp. 204–215. 2004.

[SLF03] T. Strang, C. Linnhoff-Popien, K. Frank. CoOL: A Context Ontology Language to Enable Contextual Interoperability. In *4th IFIP WG 6.1 Int. Conf. on Distributed Applications and Interoperable Systems (DAIS)*. Pp. 236–247. 2003.

[Wil05] Wilton-Jones, T. (Opera Software). Extensible Rendering Architecture. Technology White Paper, 2005.
http://www.opera.com/docs/whitepapers