# A Flexible Architecture for Mobile Collaboration Services

Thomas Springer, Daniel Schuster, Iris Braun & Jordan Janeiro
TU Dresden, Germany
thomas.springer@tu-dresden.de

Markus Endler
Departamento de Informatica
PUC-Rio
Rio de Janeiro, Brazil
endler@inf.puc-rio.br

Antonio A.F. Loureiro
DCC/UFMG
Belo Horizonte, Brazil
loureiro@dcc.ufmg.br

## 1. INTRODUCTION

There is already a multitude of collaborative applications available in mobile environments. Although they share a good mount of common functionality, most of them are built from scratch, or are tailored to a specific device platform using proprietary libraries. An open and customizable environment for mobile collaborative applications is still missing.

A technology survey comprising the top 50 proposals of Google's Android Developers Challenge illustrates the need for platform support for mobile collaboration. These applications can be considered as a sample regarding user needs and market relevance. According to our analysis 41 out of 50 applications (i.e., 82%) provide collaborative functionality. That means, they could use at least one of the services we propose in section 2. The collaboration features comprise shared editing of images, chatting, or discussing music or prices of products. The most important aspect is location-awareness, 52% use it, followed by communication features (26 %), geo-tagging (22 %) and content sharing (20 %). 30 % of the applications combine at least 3 collaborative features.

Collaborative middleware platforms for mobile development are already covered to some degree by various articles and elaborations. Proem [5], the George Square System [2] and BuddySpace [3] are examples of middleware platforms focusing on different aspects of collaboration. Proem includes services realized as a high-level Java API that manage areas like presence, community management and a common data space. However, the underlying communication protocols are application specific and directly define the syntax of messages. The George Square System is a map based collaborative system for tourism. It primarily analyzes how tourists work together in groups and collaborate around maps. The BuddySpace instant messenger is a collaboration tool for desktop computers with an enhanced Presence notion and thus, does not consider mobile collaboration. Moreover, free collaboration frameworks like the Extensible Messaging and Presence Protocol (XMPP) [1] or NaradaBrokering [6] exist, which can be used as a basis for implementing higher-level collaboration services.

In the following we present a conceptual architecture for mobile collaborative systems. We identify common services that may be helpful for multiple applications. These include map-based visualization, multimedia geo-tagging, communication functions and comprehensive context management. The evaluation is based on the implementation of a mobile collaborative tourist guide system using three different mobile platforms.

## 2. MOBILIS ARCHITECTURE

Our Mobilis conceptual architecture for mobile collaboration consists of four layers as depicted in Figure 1. The *Device OS layer* represents the functionality provided by the operating system of the device which can be for instance Windows Mobile, Symbian OS or the OS layer in Android. The *Basic Services layer* contains services which are implemented dependent on the operating system functionality. The *Mobilis Services layer* contains the services for supporting mobile collaboration. At this layer all services are interconnected by a service bus and can thus potentially share their functionality with all other services of that layer. For instance, the Group chat service uses the Group Management service, and most of the Mobilis services access the context service to control internal adaptation processes. The *Application layer* is the layer where the applications reside. In the following we will focus on the Mobilis Services layer. We describe all services and their interdependencies with other Mobilis and Basic services.

The **Context Management Service** is responsible for sharing context between different participants of a collaboration session (e.g., device capabilities, connectivity, physical context like noise and light level, and user activity). It uses basic services for request/response and pub/sub communication and Mobilis services for group management and security to deal with privacy concerns of context distribution.

The **Geo-location Service** provides location information and answers proximity and range queries. This includes the access of geographic information systems and lower layer positioning functionality based on local hardware like a GPS device or a WLAN adapter.

The **Multimedia Tagging Service** supports the attachment of multimedia content to locations. The service uses the basic Event Detection and Pub/Sub services and Geo-location, Media Sharing and Context Management for representing tags by overlaying maps.

The **Group Chat Service** allows the creation of chat rooms and the exchange of asynchronous chat messages in the scope of particular rooms. It builds on the session man-

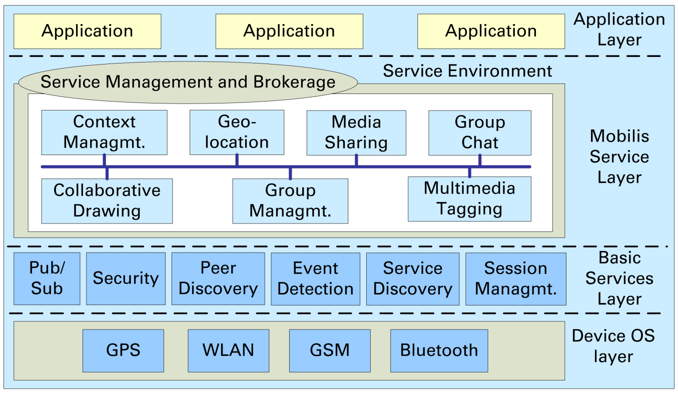

**Figure 1: The Mobilis conceptual architecture.**

agement and Group and Context Management Service.

The **Group Management Service** enables the creation of groups and the control of adding and removing members to and from groups. The service uses the basic Pub/Sub, Event Detection and Session Management services.

The **Media Sharing Service** offers the functionality to search for, access and offer multimedia content in the scope of groups. Therefore, the service builds on the pub/sub and session management basic services and the Mobilis services for Group and Context Management. Context is used to adapt content to connectivity and device capabilities.

The **Collaborative Drawing Service** allows to edit a common canvas in a group. Especially, an overlay with other content is supported, for instance for drawing the shortest route between two locations on a map provided by the Geo-location Service. The service builds on the Event Detection, Session, Group, and Context Management services.

## 3. IMPLEMENTATION EXPERIENCES

In the context of a tourist application the Mobilis services were implemented and combined to support several features. The tourist application allows all members of a tourist group to be aware about the others activities based on a map. The map is provided by the **Geo-location Service** and the tourists locations are shared using the **Context Management Service**. Each tourist can create so called *Fun flags* at defined locations which are presented on the map. For instance a nice building can be marked by a Fun flag containing photo of that building provided by the **Multimedia Tagging Service** in combination with the **Media Sharing Service**. According to presence and context information provided by the **Context Management Service**, for instance user activity, battery level and connectivity status, the right way to communicate with other tourists is determined. By selecting the icon of a tourist on the map, he can be contacted using the **Group Chat Service** or a direct phone call. Moreover, tourists can share any sort of content object either locally or remotely in a transparent way by using the **Media Sharing Service**. In this way, photos, videos and sounds of the trip can be circulated.

We implemented our middleware concept and tourist application using the three platforms Java ME, Java SE and Android [4]. For providing collaboration functionality we adopted functionality of the XMPP collaboration framework, encapsulated into the specified services. XMPP is a family of protocols for collaborative environments based on a client/server architecture. A set of XEPs (XMPP Extension Protocols) provide further functionality, e.g. presence exchange, pub/sub, group management and group chat. One major requirement was the use of Java, because it is supported by the majority of device platforms. For XMPP and XEPs a set of libraries exists which implement the protocols on client site. The most comprehensive one is the Smack API, the API we choose for the implementation. It is the only library with support for XEPs for pub/sub and Group Management. For the implementation of the Geo-location service the *Google Maps API* was adopted.

## 4. SUMMARY

Our technology survey illustrates the need for a middleware platform for mobile collaboration. We have presented the Mobilis platform as our solution and discussed the services as well as their interrelations. While the Mobilis platform is conceptual it can be implemented on top of various operating systems. We have presented our implementation experiences using Java ME, Java SE and Android. As a result we discovered our assumption about missing support for collaborative features in the platforms to be correct. Just basic features are provided and the capabilities of the platforms are quite different. With Java ME we failed to implement the entire functionality because of version problems of Java and XMPP libraries. Some XMPP functionality had to be reimplemented. The Java SE version was fully implemented but is just runnable on Laptops and powerful PDAs. The Android platform provided the best support for our implementation. Especially, the concepts for creating UIs and the easy to use APIs for Google Maps access have arisen to be helpful. In future we want to extend the services of our platform and plan to find new and innovative application scenarios with new requirements for our platform such as knowledge sharing within a group or P2P location-aware product search on mobile devices.

## 5. ACKNOWLEDGMENTS

This work was partially funded by CNPq (Brazil) and BMBF (Germany). The authors would like to thank all members of the Mobilis project for their contributions.

## 6. REFERENCES

[1] Xmpp standards foundation, 2008. http://xmpp.org.
[2] B. Brown and E. Laurier. Designing electronic maps: an ethnographic approach. In *Map Design for Mobile Applications*. Springer Verlag, 2004.
[3] M. Eisenstadt and M. Dzbor. Buddyspace: Enhanced presence management for collaborative. In *Learning, Working, Gaming and Beyond. In JabberConf Europe*, 2002.
[4] Google. Android - An Open Handset Alliance Project, 2008. http://code.google.com/android/.
[5] G. Kortuem. Proem: a middleware platform for mobile peer-to-peer computing. *SIGMOBILE Mob. Comput. Commun. Rev.*, 6(4):62–64, 2002.
[6] S. Pallickara and G. Fox. NaradaBrokering: A Distributed Middleware Framework and Architecture for Enabling Durable Peer-to-Peer Grids. In *in Proceedings of Middleware Conference 2003, Rio de Janeiro*, pages 41–61, 2003.