Mobilis - Comprehensive Developer Support for Building Pervasive Social Computing Applications

Daniel Schuster, Robert Lübke, Sven Bendel, Thomas Springer, Alexander Schill
Computer Networks Group
Technische Universität Dresden
Dresden, Germany
{firstname.lastname}@tu-dresden.de

Abstract—**Pervasive Social Computing (PSC) is an ongoing research trend of merging the two worlds of pervasive computing and social computing. While there are already lots of promising PSC applications, building such systems is still very sophisticated and error-prone. The Mobilis framework provides developer support for mobile social apps (native Android and HTML5), a service environment for dynamic deployment of services, authentication and robust communication over unreliable networks as well as an emulation environment to be able to mimick user behavior and to test applications in a lab environment before field tests. The system is available as open source software on Github.**

I. Introduction

The tremendous success of mobile apps on smartphones has changed the focus of research in pervasive computing. As social features get more and more important, this creates what is called Pervasive Social Computing (PSC) as a new class of systems and applications as defined in [1] and [2]. Accordingly, the requirements to an infrastructure for PSC applications are shifting to the support of social and environmental awareness in combination with rich features for collaboration and interaction between users, potentially in real-time.

There are already some frameworks for pervasive computing which address at least some of these issues. But none of them significantly reduces the overwhelming complexity of building such distributed applications. With our practical experiences in the research field in mind, we expect a framework to:

- provide easy authentication and connectivity among all entities
- provide support for building and deploying services used by mobile apps
- provide easy APIs at a number of platforms including Android and HTML5
- provide an emulation environment where to test complex applications

With all these requirements in mind, we created the Mobilis platform, an ongoing open source effort to ease and accelerate development of PSC applications.

II. Pervasive Social Computing based on XMPP

The eXtensible Messaging and Presence Protocol (XMPP) is a family of open, highly extensible communication protocols designed for Internet-scale messaging and presence. It is standardized by the IETF as a bunch of core protocols and a large set of XMPP Extension Protocols (XEPs). All resources are identified by an XMPP ID (e.g., alice@xmpp.org/iphone). Using the XMPP ID all resources communicate based on asynchronous messages (XMPP stanzas) represented in XML and exchanged as XML-Streams via TCP.

We adopted XMPP as the heartbeat of our infrastructure as can be seen in the example scenario in Figure 1. Standard XMPP servers can be used as a black box. XMPP clients can connect and authenticate to an XMPP server where they are registered. Thereafter, XMPP clients can start to exchange messages and to track presence. Extension protocols add further features to XMPP servers. In our infrastructure we especially consider: Session Discovery (XEP-0030) for finding and exploring the Mobilis Server, Multi-User Chat (XEP-0045) for session management and group messages, Publish-Subscribe (XEP-0060) for context distribution, and File Transfer (XEP-0096) for media sharing.

To create a pervasive service infrastructure, client-side apps as well as server-side services are integrated as XMPP clients and thus communicate via standard XMPP protocols requiring no modification at the XMPP servers. This homogeneous setup ensures interoperability across highly heterogeneous devices and service platforms. Scalability is supported with the ability to federate a set of XMPP servers which distributes the effort for account management, message routing, etc. Basic awareness functions are covered by the exchange of presence information and pub/sub support.

III. The Mobilis Framework

The main components of the Mobilis framework can be seen in Figure 1. The **Mobilis Server** offers a hosting environment for PSC services (called Mobilis services). Services are based on a service description. We created the Mobilis Service Description Language (MSDL) for this purpose. It offers WSDL-like capabilities with some added features (especially XMPP support). With MSDL it is possible to describe the network protocol of the PSC application and to create stubs for each node type which eases development of the apps and services. Once a service is changed, it can easily be uploaded to the service environment during runtime. Clients connected to the old version keep running while new clients are redirected to the new service version during service discovery.

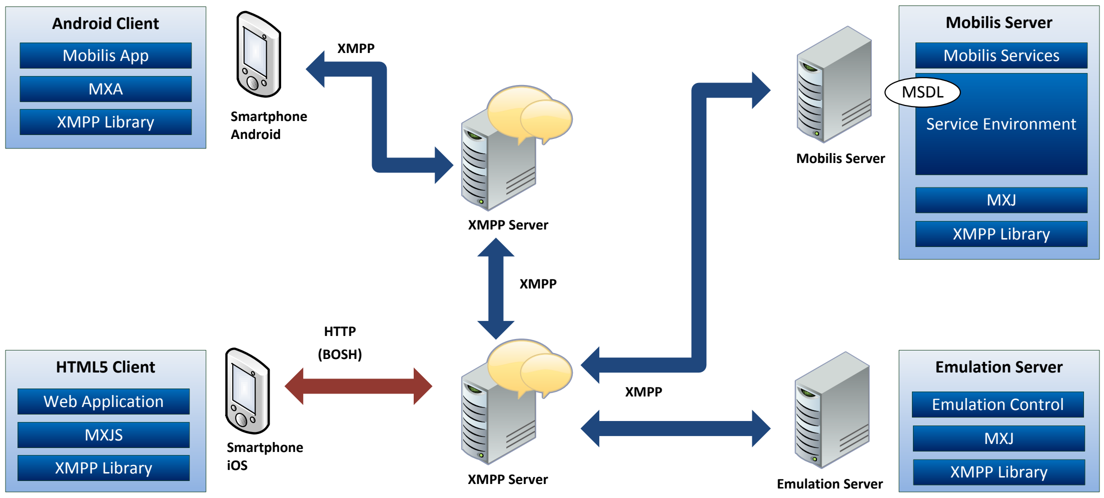

Fig. 1. Mobilis example scenario

We created **Mobilis Libraries** for each target platform to provide an abstraction layer for all network communication. These are called *Mobilis XMPP for X*, where *X* stands for the specific platform. We currently provide implementations for Android (MXA), pure Java (MXJ) and JavaScript/HTML5 (MXJS). They all include easy-to-use APIs based on MSDL descriptions and handle all the XMPP-based messaging between the apps and services including reliability mechanisms. Bidirectional-streams Over Synchronous HTTP (BOSH) are used to encapsulate XMPP into HTTP for HTML5 applications. This will be replaced by WebSockets in a later release.

The third component is the **Emulation Server**, which provides means to emulate the behavior of complex PSC applications. Clients register themselves at the Emulation Server and receive scripting commands. Thus, a central script can be created to describe a complex application scenario including location changes and waiting conditions between users.

Furthermore, a number of **Demo Applications** is available at our Github repository [3]. The most prominent application is Mobilis XHunt [4], a location-based game where multiple agents hunt a wanted criminal using public transport of a real city. It is yet our most complex and demanding application as participants dynamically form groups, exchange location events in high frequency and experience certain variations in network quality while they move through the city.

IV. Demo

The intended demo will show an emulated game of XHunt using different Android, iOS and Windows Phone devices as shown in Figure 2. Real movement of players will be replaced by explicit *SetLocation()* commands for demo purposes. An additional notebook will be used on site to show the Emulation Server output (scripting events) as well as the XMPP traffic on the Mobilis Server. Besides the emulated demo, attendees will optionally be able to create and play their own XHunt games using their own smartphones.

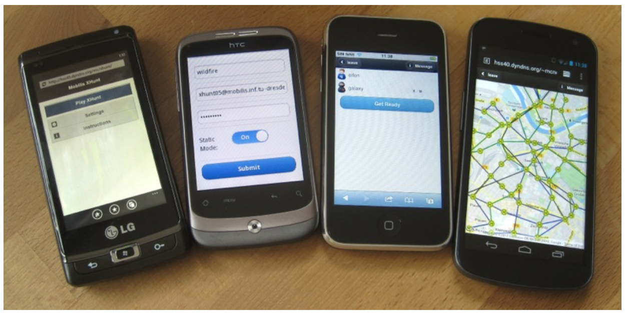

Fig. 2. Different devices running the XHunt demo app

V. Platform Requirements and Licensing

All software components described here are available with full source code under Apache license at our Github repository [3]. The software is now in a mature state and has already been used in a number of research projects. We counted more than 1,500 downloads of the software source code by now. It is well documented in the Github wiki attached to the project.

Acknowledgment

The authors would like to thank the many student contributors of the Mobilis project, especially István Koren, Marc Löchner, Danny Kiefner, Thomas Walther, Christian Magenheimer and Niko Jansen.

References

[1] S. B. Mokhtar and L. Capra, "From pervasive to social computing: algorithms and deployments," in *Proceedings of the 2009 international conference on Pervasive services*, ser. ICPS '09, London, UK, 2009.
[2] J. Zhou, J. Sun, K. Athukorala, and D. Wijekoon, "Pervasive social computing: Augmenting five facets of human intelligence," in *Proc. UIC/ATC*, 2010.
[3] TU Dresden, "Mobilis platform," https://github.com/danielschuster/mobilis, 2012.
[4] D. Schuster, D. Kiefner, R. Lübke, T. Springer, P. Bihler, and H. Mügge, "Step by step vs. catch me if you can - on the benefit of rounds in location-based games," in *3rd IEEE Workshop on Pervasive Collaboration and Social Networking (PerCol), Lugano, Switzerland*, 2012.