Programming Android Client for Smart-M3 Applications: SmartRoom Case Study

Pavel Kovyrshin*, Dmitry Korzun*†
*Petrozavodsk State University (PetrSU), Russia
†Helsinki Institute for Information Technology and Department of Computer Science and Engineering, Aalto University, Helsinki, Finland
{kovyrshi, dkorzun}@cs.karelia.ru

***Abstract*—Smart-M3 is an open platform for constructing smart spaces service environments. SmartSlog is a Smart-M3 SDK for ontology-driven programming; it targets a wide range of computing devices. In this abstract we consider programming issues for Android OS with SmartSlog and introduce a scheme for developing mobile clients on the Smart-M3 platform. This generic scheme is evaluated within a particular Smart-M3 application—the SmartRoom system, which supports such collaborative activities as conferencing. SmartRoom clients run on end-user mobile devices (e.g., Android smartphones) and serves as personal access points to the system.**

Keywords*—Smart-M3, SmartSlog, Collaborative environment, Mobile programming, Android OS.

The Smart Spaces paradigm defines a smart space as a logical entity consisting of smart objects and enabling them to cooperate with each other [1], [2], [3]. A smart object is a physical/digital object that becomes, being augmented with network capabilities, applicable to interact within a smart space, understanding and reacting to the environment. A wide class of smart objects can be constructed as client software running on end-user mobile devices such as smartphones [4].

Our reference case is the SmartRoom system, which provides a shared smart space—SmartRoom space—for assisting collaborative work activities such as conferencing [5], [4], [6]. A SmartRoom client makes the host mobile device a personal access point to the system and its services. It allows a user to join and leave the SmartRoom Space. The basic services for SmartRoom users are Agenda and Presentation that provide the most essential information part of conferencing, see Fig. 1 for screenshot examples. For the chairman additional functions are available, e.g., using SmartRoom client the activity can be started or finished, the Presentation slide show can be manually controlled (in addition to the control from the speaker), or the agenda is changed (talks insertion/update/removal).

Smart-M3 platform is used to construct a smart space [7], [2]. SmartSlog SDK supports ontology-driven programming of Smart-M3 applications [8]. As target end-user mobile devices we consider smartphones and tablets with Android OS, which are popular nowadays. In Smart-M3 terms, the software part of a smart object is a knowledge processor (KP).

Android OS is Linux-based. For applications, the primary programming language is Java. Essentially that GUI programming is typically Java-based. Nevertheless, native code (C/C++ language) is supported for application logic. This feature opens a way for programming Smart-M3 applications using SmartSlog and its support of ANSI C programming language[1]. Also the recent development progress of Qt framework for Android OS leads to expectation that in near future GUI development for Android application can be performed totally with Qt, thus using C++ programming.

Fig. 1. Agenda and Presentation services: example representation in GUI on the client side.

We assume that the client KP operates using the terms of the problem domain. The latter is described by ontology. For programming a SmartSlog-based Android client we propose the following scheme (Fig. 2).

1) Client GUI is written in Java. GUI elements are connected with the application logic using Java Native Interface (JNI).
2) Client chunk of the KP application logic is written in C using data structures that directly correspond to the problem domain entities from the ontology and smart space access operations parameterized with those entities.
3) Application logic code uses the ontology library (generated by SmartSlog): data structures and variables are related with the ontology entities, the library provides functions for operations on the problem domain data and for operations with the smart space.

[1]SmartSlog supports also .NET/C# programming, which seems less suitable for Android.

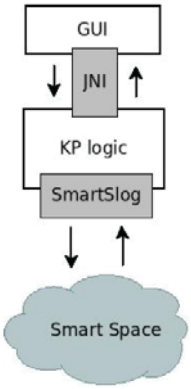

Fig. 2. Structure of Android client KP code: Java-based GUI is connected with C-written client application logic by JNI, SmartSlog ontology library provides domain-oriented data structures and operations for accessing the smart space.

KP application logic is C-written and uses data structures and function of the SmartSlog ontology library. Integration of the KP logic into the Android client uses NDK toolset. All source files are added to the Android application project in some directory (e.g., `jni/`). To compile them it is necessary to prepare makefile `Android.mk`. It defines all sources and flags needed for compilation and linking. The resultant object files are assembled into a target library, which then can be loaded by the following Java code:

```
static {
   System.loadLibrary("library name");
}
```

JNI enables Java code to call or to be called by native applications. All methods of the KP logic must be described in a Java class with `native` modifier. JNI states specific naming conventions, including project name and method name as described in Java class. To implement these methods in KP it is necessary to include corresponding header file containing their prototypes. `Javah` utility is the best way to generate the header file. Besides, it is important to include the SmartSlog library header file to access functions of the library. It could be done by including `generic.h` directly to KP module or by *ontology file header* which contains such a declaration. For linking GUI with native methods it is enough to write usual class method call in widget implementation.

Let us consider the SmartRoom reference case (conference in our example). A SmartRoom participant may join the SmartRoom space as a registered user (one of the conference speakers), a spectator (anonymous participant), or a chairman (organizer of the activity). After performing login the mobile client subscribes to Agenda and Presentation services [5]. That is, if some changes occur the client will receive the notification and react correspondingly.

The conference activity is controlled by its chairman, e.g., she/he starts and ends the conference. When the conference started the clients update the conference agenda on own displays. The current speaker is highlighted in the list. The speaker's mobile client receives the control on the Presentation service and shows the first slide. The speaker can switch the slides using the functions like "next slide" and "previous slide". The same function can use the chairman if needed. When the speaker (or chairman) ends the current presentation the control is moved to the SmartRoom client of the next speaker. Similarly, all participants from their clients can watch the current agenda or browse slides of any available speaker, including the current presentation online. On the level of KP logic, the above functions are based on subscription and update queries to the SmartSpace and implemented with appropriate smart space access primitives from the SmartSlog ontology library.

The Android client for the SmartRoom system is available at http://sourceforge.net/projects/smartroom/files/clients/android/. The proportion of Java and C code is about 60:40. It includes proper application code: the code is not automatically generated by SmartSlog SDK and lines with comments and blanks only are excluded.

REFERENCES

[1] S. Balandin and H. Waris, "Key properties in the development of smart spaces," in *Proc. 5th Int'l Conf. Universal Access in Human-Computer Interaction. Part II: Intelligent and Ubiquitous Interaction Environments (UAHCI '09)*. Springer-Verlag, 2009, pp. 3–12.

[2] D. G. Korzun, S. I. Balandin, V. Luukkala, P. Liuha, and A. V. Gurtov, "Overview of Smart-M3 principles for application development," in *Proc. Congress on Information Systems and Technologies (IS&IT'11), Conf. Artificial Intelligence and Systems (AIS'11)*, vol. 4. Moscow: Physmathlit, Sep. 2011, pp. 64–71.

[3] E. Ovaska, T. S. Cinotti, and A. Toninelli, "The design principles and practices of interoperable smart spaces," in *Advanced Design Approaches to Emerging Software Systems: Principles, Methodology and Tools*, X. Liu and Y. Li, Eds. IGI Global, 2012, pp. 18–47.

[4] A. Vdovenko, S. Marchenkov, and D. Korzun, "Mobile multi-service smart room client: Initial study for multi-platform development," in *Proc. 13th Conf. of Open Innovations Association FRUCT and 2nd Seminar on e-Tourism for Karelia and Oulu Region*, S. Balandin and U. Trifonova, Eds. SUAI, Apr. 2013, pp. 143–152.

[5] I. Galov and D. Korzun, "Smart room service set at Petrozavodsk State University: Initial state," in *Proc. 12th Conf. of Open Innovations Association FRUCT and Seminar on e-Tourism*, S. Balandin and A. Ovchinnikov, Eds. SUAI, Nov. 2012, pp. 239–240.

[6] D. Korzun, S. Balandin, and A. Gurtov, "Deployment of Smart Spaces in Internet of Things: Overview of the design challenges," in *Proc. 13th Int'l Conf. Next Generation Wired/Wireless Networking and 6th Conf. on Internet of Things and Smart Spaces (NEW2AN/ruSMART 2013)*, ser. LNCS 8121. Springer-Verlag, Aug. 2013, pp. 48–59.

[7] J. Honkola, H. Laine, R. Brown, and O. Tyrkkö, "Smart-M3 information sharing platform," in *Proc. IEEE Symp. Computers and Communications (ISCC'10)*. IEEE Computer Society, Jun. 2010, pp. 1041–1046.

[8] D. Korzun, A. Lomov, P. Vanag, J. Honkola, and S. Balandin, "Generating modest high-level ontology libraries for Smart-M3," in *Proc. 4th Int'l Conf. Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM 2010)*, Oct. 2010, pp. 103–109.