A Data Synchronization Framework for Personal Health Systems

Davide Capozzi and Giordano Lanzola

Department of Computer and Systems Science
University of Pavia
Via Ferrata 1, 27100 Pavia, Italy
{davide.capozzi,giordano.lanzola}@unipv.it

Abstract. This paper illustrates the design of a multi-platform synchronization framework which is particularly useful for speeding up the implementation of Personal Health Systems on mobile devices. Those devices turn out to be of great help since in order to transfer any data available at the patient site to the clinic and vice-versa a solid networking infrastructure and data exchange protocol is needed. The framework we developed extends an open source platform available on the market by empowering it with new features that better decouple domain specific data from the underlying transport logic. In the last part of the paper two prototypes exploiting the framework are described.

Keywords: Healthcare telemetry and telemedicine, Measurement and monitoring technologies, Mobile devices for patient monitoring, Transmission of patient data.

1 Introduction

The increasing aging of the population combined with many unhealthy lifestyles being adopted nowadays and resulting in an augmented prevalence of obesity are acting as a modern plague in most of the western countries. In fact they are responsible for an increased incidence of chronic disorders such as coronary artery disease, congestive heart failure or diabetes which account for the majority of the medical expenses [1]. It is now clear that such a current trend cannot be sustained any longer [2] and new and more effective ways of coping with chronic diseases should be pursued in order to reduce long-run medical expenses and prevent the onset of those complications which frequently result into specific treatments and hospitalizations pressing on the health care budgets.

With respect to this concern, there is a growing interest about Personal Health Systems (PHSs) in the technological communities which has been stirred up recently. This term refers to devices made available by the joint achievements in micro-electronics and nanosciences and exploiting the Information and Communication Technologies (ICT) to provide applications supporting the personalization and individualization of the treatment process [3].

K.S. Nikita et al. (Eds.): MobiHealth 2011, LNICST 83, pp. 300–304, 2012.

2 Materials and Methods

Mobile phones, Personal Digital Assistants (PDAs), Smart-phones, and tablets nowadays have such a great variety of technical features that allows to choose each time the product fitting any given application at best, but it becomes a serious drawback inasmuch the plain connectivity is of concern.

According to the models advocating the decoupling and separation of concerns among the different components building up a system, we envisioned instead a layered architecture where data exchange is supported by an underlying layer shared among all platform and supporting their interoperation platforms [4].

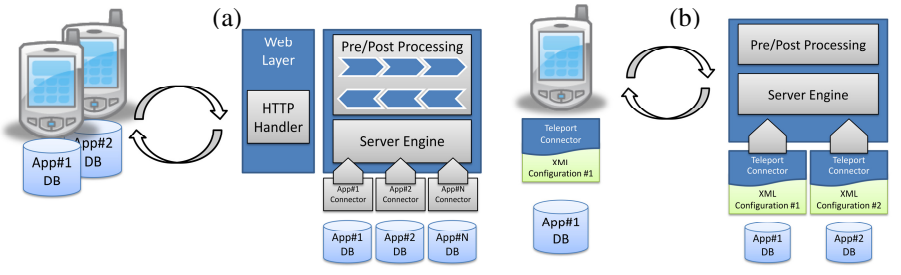

Fig. 1. (a) The synchronization framework. (b) The Teleport Connector within the framework.

What we need is a synchronization framework that can easily be accessed from different devices running different Operating Systems (OSs), store data on different formats and provide some programming facilities to allow us customizing and extending its basic functionalities. Figure 1(a) shows the idea of a web-based synchronization framework that could be exploited as a transparent two-ways-data-exchange layer by a PHS application. On the left there are two smartphones running two different PHSs having each one its own Data Base (DB); each time an application needs to be synchronized, it starts an HTTP connection towards the remote server that exposes an HTTP Handler in its Web Layer. The majority of the synchronization platforms available on the market adopt at this level an open communication protocol named SyncML [5]. SyncML [6] is an open industry initiative supported by hundreds of companies including Ericsson, IBM, Lotus, Matsushita, Motorola, Nokia, Openwave, and Starfish. It seeks to provide an open standard for data synchronization across different platforms and devices.

To minimize communication time, the standard assumes that each device maintains information about modification flags for each of its records with respect to every other device on the network. For this reason, in the architecture a Server Engine is needed that takes the burden of managing stored records in terms of handling record IDs, detecting and trying to resolve conflicts among records, as well as keeping trace of record modifications. The extensibility of the platform is represented by connectors plugged on the bottom of the server: each one for a different PHS application. Through the connector, the Server Engine interacts each time with a different

application DB, since its structure and the domain knowledge are enclosed in that component. Last but not least, a pre/post processing function block is needed in the architecture, in order to resolve any data format conflict between clients and server.

For the implementation of our synchronization framework addressing PHSs we chose the open source Funambol platform [7], since it better captures our needs and reflects the architectural features described above.

3 Results

The main goal of our synchronization framework is to decouple the transmission of data between client and server from the management and the storage of data on both sides. That important feature enables us to reuse the solution for any PHS application that, in this way, exploits the synchronization framework for exchanging its data transparently with a remote server anytime this is necessary, without the requirement to be connected continuously to the internet.

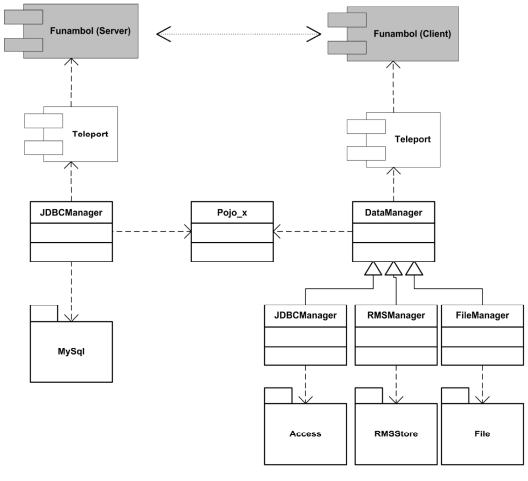

Fig. 2. The UML architecture of the Teleport Connector

We implemented that feature exploiting the extendibility of the Funambol platform, developing a generic connector named Teleport Connector (TC) that interfaces the whole architecture as displayed in Figure 1(b). TC can be exploited both on the client-side interfacing PHS applications and on the server-side connecting to PHS remote DBs. This component responds to the tasks of encapsulating the application data into a generic format, shared between client and server, sending them through the HTTP connection, described in the paragraph above, and accessing the application specific DB exploiting an XML Configuration file provided with the application itself.

For the data encapsulation we designed generic POJOs (Plain Old Java Object) that enclose data in a collection of key-value couples disregarding their types and thus gaining generality. POJOs are the only data type that TC can manage. In this way the TC could transmit and receive POJOs without being aware of what they contain.

In order to exploit the SyncML protocol for exchanging data, TC has been equipped with the capability to encode a POJO into a plain-text document complying with an XML based formalism, since XML can be natively managed by those mobile platforms running a Java Virtual Machine, such as J2ME or Android. For any other mobile platform we have provided a custom linear text encoding, independent of the XML formalism.

Furthermore, the TC is able to access the application DB without containing any wired knowledge about the domain specific data structure; this can be performed through the parsing of an application specific XML configuration file that encapsulates the definition of the data structure. Moreover, TC not only is totally independent from application data but also from the physical/logical support where information is stored. In order to save its generality, TC can't be dependent on those technologies, so that a data store abstraction layer has been introduced. Figure 2 shows in more details the last feature described.

On the topmost part of the figure the Funambol platform components (the server on the left side and the client on the right) are displayed. From those the TC architecture develops for both server and client, sharing the most part of the code located in the Teleport package. On the client side, the data store abstraction layer is represented by the component DataManager used directly from the Teleport package to access data and, in the meantime, extended by many custom DataManagers. Each particular DataManager, such as JDBCManager, RMSManager or FileManager, takes the burden of interacting with a specific data store technology, such as respectively an MS Access DB, an RMS Record Store or even simply a file. On the server side, since the data store technology has been established to be a MySQL database, we streamlined the architecture, so that just a JDBCManager is required to represent the data store abstraction layer.

4 Conclusions

This paper described the design of a multi-platform synchronization layer which is particularly useful for speeding up the implementation of PHSs. The synchronization layer described has been utilized for the implementation of two separate applications running on different devices. Since the overall focus of the paper is just on the synchronization layer and its constraints and does not allow for an extensive description of the applications, we just mention briefly each of those applications including a reference to a publication where more detailed descriptions are available [8].

The first prototype is meant for managing uremic patients who are experiencing renal failure as a major complication of diabetes and are thus undergoing Peritoneal Dialysis (PD). Thus it is mandatory for those patients to strictly control blood pressure and weight, and regularly keep informed their treating staff about any variations. The application for acquiring data is implemented on a Smart-phone

running Symbian-Os and supporting the J2ME development platform. It has been designed to acquire data directly from a blood pressure monitor and a scale exploiting a Bluetooth wireless connection and uses the device display just for a minimal interaction with its users which are likely to be elderly people.

The second prototype is meant instead to support in a medium sized randomized controlled trial for patients undergoing an Artificial Pancreas (AP) therapy. The requirements for the clinical trials see the patients undergoing an AP therapy at their domiciles while the treating staff at the clinic should be able to follow in almost real-time the evolution of the patient's clinical state. AP units are available so far only as research products and run on Personal Computers. Thus in that case decoupling the synchronization layer from the application one has been most useful for adding networking capabilities to the AP units without having to modify their code. With the only knowledge that AP units saved their data to a local database implemented with Microsoft Access, it was quite easy to establish a link with the server to ship those data to the clinic. On the way back in this case is sent information concerning directives for the AP unit and informational messages for the patient.

For both prototypes the synchronization server is running on a PC and the underlying database is implemented using MySql Server.

References

1. Levit, K., Smith, C., Cowan, C., Sensenig, A., Catlin, A.: Trends - Health spending rebound continues in 2002. Health Affairs 23(1), 147–159 (2002)
2. Fogel, R.W.: Forecasting the cost of US Health Care in 2040. Journal of Policy Modeling 31, 482–488 (2009)
3. Maglaveras, N., Bonato, P., Tamura, T.: Special Section on Personal Health Systems. IEEE Transactions on Information Tehcnology in Biomedicine 14(2), 360–363 (2010)
4. Lindholm, T., Kangasharju, J., Tarkoma, S.: Syxaw: Data Synchronization Middleware for the Mobile Web. Mobile Networks & Applications 14(5), 661–676 (2009)
5. Agarwal, S., Starobinski, D., Trachtenberg, A.: On the scalability of data synchronization protocols for PDAs and mobile devices. IEEE Network 16(4), 22–28 (2002)
6. SyncML Specifications, `http://www.syncml.org/downloads.html`
7. Fornari, F.: Funambol Mobile Open Source (Paperback), ch. 10. Packt Publishing (2009)
8. Capozzi, D., Lanzola, G.: Utilizing Information Technologies for Lifelong Monitoring in Diabetes Patients. J. Diabetes Sci. Technol. 5(1), 55–62 (2011)