MobAppGen: A Mobile Applications Generic Framework

PIERRE VINCENT
National Institut of Telecomm., GET-INT/MOBKIT
9, rue Charles Fourier- 91011 Evry
FRANCE / MEXICO

Abstract: The development of mobile telephony, combined to the fact that mobile applications are becoming indispensable for users, has caused that a need of a fast and effective implementation of new services appears. There are tools and kits that facilitate developers’ tasks but, considering that in mobile applications there are many concerns that cut across several aspects, the implementation of new applications may result in duplicated code within the same project or in similar ones. The existing way of development is also a tedious and error prone task that may produce hard to maintain applications. To remedy these concerns we propose a framework called MobAppGen which allows code reuse by modularizing mobile applications and that is able to develop client and server sides. In this paper we introduce the framework and we present some examples of its operation.

Key-Words: Mobile computing, Middleware, Java, Software engineering, Databases, Reusability.

1 Introduction

Nowadays mobile devices have an immense impact on our life, not only for the communication capacity that they offer but also for the increasing development of new applications provided by different developers every day.

Since mobile applications are becoming indispensable for users, developers’ work is gaining complexity. They must address users’ specific needs, but in a general way to reach most users. For this reason facilitate developers work has become an important task. This task must consider: code reuse, components maintenance and a common yet flexible computing and communication environment that can be fitted for devices of different makes and capabilities.

Frameworks are forms of high-level design reuse, through domain generic solutions expressed in a programming language [1].

Considering that most mobile applications require similar basis, including screen management, data persistence and network communication, it is possible to conceive a generic framework to construct applications from reusable software components.

In this paper we introduce a framework to generate client-server mobile applications, called MobAppGen (Mobile Applications Generator). To develop our framework we choose a deployment environment that runs on as many types of devices as possible as well as open standards to allow framework’s future extension and code mobility.

2 Framework Description

MobAppGen is a framework designed to facilitate the generation of mobile applications. It can be used as a command line code generator or with an incorporated web style tool which is friendlier. This software architecture for mobiles comprises three main components: the clients, the server and a database.

To develop the client side of the mobile applications, called MIDlets, we choose Java 2 Micro Edition, since it offers platform independence by running inside a virtual machine [2]. J2ME architecture is composed of three modular layers: a Virtual Machine (JVM), Configurations, and Profiles.

The configuration layer defines a minimum set of JVM features and core Java class libraries available on a particular category of devices. The configuration that defines small mobile devices is called the
Connected Limited Device Configuration (CLDC). The profile layer defines the minimum set of application programming interfaces on a particular group of devices, so applications written for a particular profile should therefore port to any device that conforms that profile. The only profile currently developed for the CLDC configuration is the Mobile Information Device Profile (MIDP). The MIDP is a set of Java API’s that, together with the CLCD, provides a complete J2ME application runtime environment targeted at mobile information [3].

CLDC requires implementations to support the distribution of Java applications using compressed Java Archive (JAR) files. Whenever a Java application intended for a CLDC device is created, it must be formatted into a JAR file and class files within a JAR file must contain the application attributes. To enable distribution of MIDlets, developers must generate a metadata file and generate a Java Application Descriptor (JAD) along with JAR file [4].

MobAppGen is able to generate J2ME source files for the MIDlets, as well as the JAR and JAD file which will be downloaded in the mobile.

On the other hand, MobAppGen generates also the server part. In this case we choose PHP which is a widely-used general-purpose scripting language that is especially suited for Web development, but also for many other applications including communication client-server and remote database management.

2.1 Framework Operation
The main functions of MobAppGen are:

- To generate proprietary code, based on user’s information.
- To generate source J2ME classes for clients.
- To generate PHP scripts for the server side.
- To compile and preverify source files.
- To generate JAR and JAD files for clients.
- To prepare all files to be downloaded by the user in a light format.

Each part that the user may add to projects is defined in the framework as a module. Modules are composed of J2ME classes and may include PHP scripts if they require communication with the server. Each module has different attributes. The user defines the modules required by the project, plus its specific attributes. The framework generates the code corresponding to these modules and their attributes.

The Class Generator is the part of the framework that interprets the code and generates all J2ME classes using the Velocity Templates [5] and the PHP scripts that each module requires.

If the project contains server applications, the Generator includes also HTML files to manage members’ information. Depending on the user’s needs one project may have only one MIDlet corresponding to one module, many MIDlets corresponding to many modules, or even different MIDlets corresponding to the same module but with different attributes. Modules actually included in the framework are:

- TextBox editor
- RMS MIDlet
- Members operations
- Member Messages
- DBRecord MIDlet

MobAppGen may be easily extended just by adding more modules. Figure 1 shows the main components of our framework. It is important to clarify that having multiple MIDlets corresponding to the same module doesn’t mean that the generator will repeat the same classes for each one. The principal purpose of the framework is reuse of code, for that reason all classes templates are predefined to be shared by MIDlets doing similar tasks.

2.2 Description of the Modules
When the application is ready to be used, all access to the server from mobile devices will require authentication. Authorized users are called members. All members’ information is registered in a database which may be managed by a PHP and HTML tool called Administrator Tool and that is also generated by MobAppGen.

The modules included in MobAppGen are described in the next sections.

2.2.1 TextBox Editor
A TextBox editor module provides the functions of a text editor in a mobile device. The display object used is a TextBox. The module adds the main functions required to manage information and provides data persistent storage. The functions incorporated allow the client to: add, edit, cut and copy text like any text editor does. The attributes for this module are TextBox length, TextBox name, record’s order and title length for each record.
2.2.2 RMS MIDlet
This module generates all the classes needed for an application that offers data persistent storage using the Record Management System provided within the MIDP. The user defines the objects that he requires, for example TextFields or DateFields, and all their attributes. For the records the user may define record’s order, title length and record title.

The generated classes contain all the functions required to manage the records including: add, delete and edit operations.

2.2.3 Members Operations
This module deals with data persistent storage in the remote server. The members’ accounts creation is done using the Administrator tool. When an account is created all the information of the member is registered in a database within the server and a storage space is assigned.

Members’ operations include save and recovery data from the server. All records generated with any RMS MIDlet may be saved in the server using the MIDlets generated with this module. The member requires a login and a password to access the server from his mobile. Access to the server is performed using HTTP.

Members name and password are only necessary for the communications with the server. Once the information is recovered in the mobile it is available using the corresponding RMS MIDlet.

2.2.4 Member Messages
The functions incorporated in this module allow all members to send and receive messages thru the server. Since each declared member has a dedicated space of safeguard, it is also used to maintain an inbox. All messages from other members are stored in the inbox and the member may access it from any mobile. After the inbox is recover, it is maintain locally using the device RMS system, this allows the member to read his messages any time without connecting to the server. This module generates also the MIDlets to send messages to others members, as well as the operations for inbox edition.

The attributes for messages and members modules are: server URL, administrator information, and directory name to create members spaces.

2.2.5 DBRecord MIDlet
This module is similar to the RMS MIDlet module, but in this case all records information is registered in a remote database in the server. Communication with the server is done using HTTP. The modules attributes allow that the user to determine which database field will correspond to each TextField or DateField in the mobile application.

Other attributes include the TextField and DateField information, records order, record title, and the database host information necessary to establish communication with the server.

3 Using the Framework
The first step to create a project is to generate the source code. MobAppGen requires specific source code that will be interpreted by the Class Generator. Source code is based in the predefined modules and may be created easily by the user if he knows the MobAppGen’s syntax. On the other hand, using the framework web-style tool the user doesn’t require to enter code, in this case, the user chooses the modules and their attributes among the different framework components and the tool generates the code.
After the code is created the Class Generator reads it and defines the templates required to create the Java source files. The Velocity templates are merged with the information provided by the user to generate an application according to his needs. When all source files are generated, the Generator compiles and preverifies them. Finally it generates the JAD and JAR files as well as PHP scripts if required. Figure 2 shows a screen of the web-style tool for a DB Record module. Here the user capture the information required for this specific module.

A fragment of source code that will be generated to create an application from a DBRecord module will be the next:

```java
Library
DBRecord
no order
http://127.0.0.1/MobAppGen/Test/db_books
books
book.isbn,books.title,books.author
TextField ISBN isbn: / 10 ANY t Books.isbn
TextField Title title: / 60 ANY t Books.title
TextField Author author: / 30 ANY t books.author
End
```

The lines in this fragment of code correspond to: MIDlet name, module’s name (DBRecord in this case), and all the attributes required for this module, as TextFields’ information, records’ order, server URL and the remote database information to register records.

Some classes, among others, that will be generated for this application are:

-Library: Main MIDlet.
-LibraryRecord: Record that contains and manages all generated TextFields.
-HttpDBinterface: Communication with the remote database.
-DataBase: Database interface operations to read, save, update and delete records.
-AbstractRecord: Abstract class that includes functions to manage any record with different fields or a TextBox.

Figure 3 shows a MIDlet screen used to add a record using the class LibraryRecord. The record will be stored in the remote database thru HttpDBinterface. If another application using the module DBRecord is added to this project, for example PetShop, only specific classes for this application will be added. In this case the Generator will add PetShop MIDlet class and PetShopRecord, containing the records attributes. Other classes like HttpDBinterface, DataBase and AbstractRecord will only be updated.

![Fig. 2: Web-Style MobAppGen screen](image)

4 Related Work

The MobCon approach (Cepa V., 2005) is closely related to our framework, in that both frameworks are based in templates. The main differences are code generation and modularization. In MobCon the user or developer must first familiarise with the specific tags used in the framework to be able to create the code that will be later interpreted. In our framework the code may be generated using the web-style tool, thus the user must only know what modules and what attributes requires. Other important difference is that Modules in MobCon incorporate specific tasks while in MobAppGen a module is a set of classes for a particular task. There are also others commercial tools that help mobile applications development, but we consider that MobAppGen is friendlier as well as easy to use and can be easily extended.

5 Conclusions and Future Work

MobAppGen framework is a mobile applications generator that facilitates developer’s work using predefined modules and code reuse. Our framework
targets mobile computing devices included in the J2ME MIDP profile, and generates also PHP scripts for the server when required.

We plan to optimize and extend MobAppGen to contain more modules including multimedia ones. Other important aspect is to include and evaluate security aspects using the framework as well as different communication protocols.

References: