The 7th Balkan Conference on Operational Research
"BACOR 05"
Constanta, May 2005, Romania

M-BANKING AND SECURITY

MIROSLAV MINOVIĆ
VELIMIR ŠTAVLJANIN
DUŠAN STARČEVIĆ
IGOR KOSTADINOVIĆ

Faculty of Organizational Science, Belgrade University

Abstract

This paper contains overview of key technologies for mobile banking client implementation and presents J2ME mobile banking client development. Banking applications follow a trend of mobile application expansion. Banking services on mobile device presents for current services new channels of distribution, implemented in order to satisfy ascending customer needs. This enables service access 24/7 from every place. This was enabled by new mobile devices, which possess excellent characteristics like large processor power, large amount of memory, enabled Java programming. On the other part, mobile providers implemented 2.5G networks, which use GPRS as information bearer. This type of networks enable greater amount of data to be transmitted between client and server, using much securer protocol and stable connection. User services were firstly realized using WAP applications, which had some disadvantages in terms of security. These applications were useful for less powerful mobile phones. Next generations of applications are client server mobile applications. These client applications need powerful mobile phones, because they are Java applications. These applications implement secure API and web service technology.

***Keywords**: mBanking, security, J2ME, mobile computing*

1. INTRODUCTION

Banking applications follow a trend of mobile application expansion. Banking services on mobile device presents for current services new channels of distribution, implemented in order to satisfy ascending customer needs. This enables service access 24/7 from every place. This was enabled by new mobile devices, which possess excellent characteristics like large processor power, large amount of memory, enabled Java programming. On the other part, mobile providers implemented 2.5G networks, which use GPRS as information bearer. This type of networks enable greater amount of data to be transmitted between client and server, using much securer protocol and stable connection.

User services were firstly realized using WAP applications, which had some disadvantages in terms of security. These applications were useful for less powerful mobile phones. Next generations of applications are client server mobile applications. These client applications need powerful mobile phones, because they are Java applications. These applications implement secure API and web service technology.

All this new features provide, for mobile phone users, new service which is mobile, secure and easy to use. This new concept is competitive advantage on the market. This competitive advantage is possibility of success.

2. PROBLEM DEFINITION

Problem definition was to develop client/server application which will enable mobile device users to perform following tasks:

- account checking,
- paying on predefined accounts (user defines maximum up to 5 accounts) and
- micro payments (for any account user defines, but reduced to maximum 20$, once a day on the same account).

Client application supports next functionalities:

1. Configuration:
   - user data,
   - web service address,
   - user accounts,
   - account payable,
   - accounts for micro payment.
2. Logging:
   - selection of accounts for payment,
   - selection of services we want to access,
   - selection of accounts payable,
   - amount of money for payment.

Server application supports functionalities:

1. User authorization,
2. Account checking, and amount of the last payment receivable,
3. Paying on predefined account,
4. Paying on predefined account number, maximum 20$ once a day, on the same account.

3. GENERIC ARCHITECTURE OF MOBILE BANKING APPLICATION

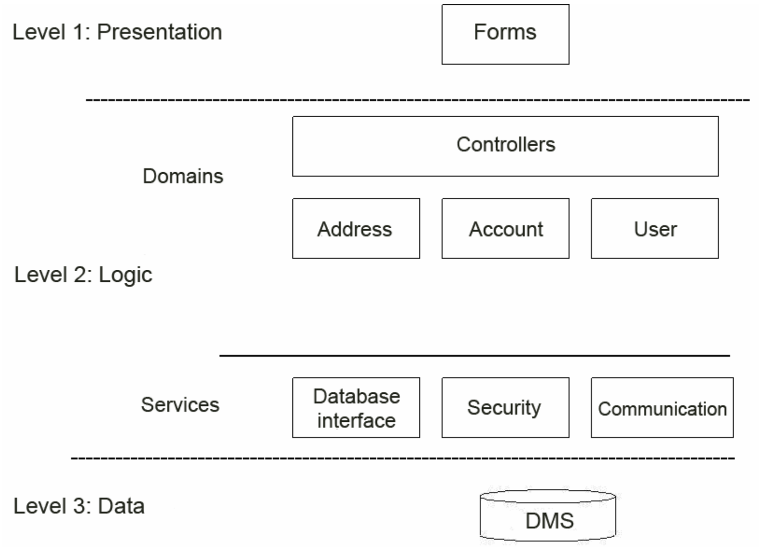

Figure 3.1 Architecture

Figure 3.1 presents multilayer architecture which enables component development, simple upgrade and simple application setup.

4. J2ME (Java 2 Micro Edition)

J2ME is a Sun's platform for mobile application development. It consists of Java Virtual Machine and a set of APIs chosen and adapted for mobile device application. Basis of the J2ME platform comprises three concepts: configurations, profiles and optional packages which are used to define ways of implementation of Java.

❖ **Configurations**

Configurations present complete Java executable environment composed of three parts:

- Java VM (*Java Virtual Machine*) which executes bytecode,
- interface between genuine code and implementation system,
- set of basic Java Runtime classes

J2ME platform defines two configurations, CLDC (*Connected Limited Device Configuration*) and CDC (*Connected Device Configuration*). CLDC is targeted toward less powerful devices. CDC uses Java Virtual Machine, and CLDC uses reduced Virtual Machine, because CLDC is used for less powerful devices, with small amount of memory and slower processor. Reduced Virtual Machine has reduced number of classes.

❖ **Profiles**

Profiles are a set of higher-level APIs that further define the application life-cycle model, the user interface, and access to device-specific properties. These APIs are set of new classes, new functionalities not introduced in basic configurations. These APIs include classes for user interface, persistent storage and networking. For example, most of profiles define classes for interactive user interface development.

Profile which is widely used on mobile devices is Mobile Information Device Profile (MIDP), based on CLDC configuration. MIDP specification defines hardware, software and network requirements as well as application management. MIDP is designed to be used on mobile devices with limited resources, like processor power, amount of memory, keyboard and screen. This specification has classes for HTTP protocol.

Two version of MIDP specification exists. First MIDP 1.0 known as JSR-37 (*Java Specification Request*) was introduced in September 2000, and second is MIDP 2.0 (JSR-118), revision of prior version. MIDP 2.0 has new features like: better user interface, support for multimedia and games, and better networking, etc.

Optional packages

Optional packages are set of APIs which maintain some additional functionality which exactly don’t exist in other specific configurations and profiles. Created to address very specific application requirements, optional packages offer standard APIs for using both existing and emerging technologies such as database connectivity, wireless messaging, multimedia, Bluetooth, and web services.

❖ **K Virtual machine (KVM)**

K Virtual Machine is a key component of J2ME architecture. It is a very portable JVM designed specially for devices with limited resources, which are always connected to network like mobile phone, pager and PDA (*Personal Digital Assistants*). KVM is very easy to implement on different devices with various processor power, memory capacity and characteristics.

5. SECURITY

❖ **BouncyCastle**

We used cryptographic package BouncyCastle for secure connections. BouncyCastle is an open source cryptographic package developed for Java platform. It has support for many cryptographic algorithms, as well as implementation for JCE 1.2.1. It’s supported by vast number of platforms, from J2SE to J2ME (including MIDP), and it is the only package which completely works under MIDP. The only problem which may arise from the usage of this package is lack of documentation.

❖ **Digest authentication**

Digest authentication uses combination of user name and password for identity check. Password is never sent in its plain form. Server instead, when it receives request from client, sends ‘Challenge’. Challenge consists from several parameters, which are used by client in combination with its own password, all in order to calculate signature (Digest) using one of the several non-reversible functions (MD5 is the most used one). Calculated signature is then sent by client to server using header of repeated request, so that server (which also has user password stored) can calculate the signature, and compare it with received signature. Authentication is completed only if these two signatures match. To prevent possible exploits of this algorithm, Challenge sent by server is only valid in short time period, and it’s different every time, so that, if anyone taps the communication, he can not acquire authentication data. Digest is very useful method of authentication for mobile applications, it is simple for implementation, and it provides good password protection.

Sending sensitive data over the network is another problem. We provide protection of confidential data which is transmitted by applying cipher. We use DES (*Data Encryption Standard*), symmetric algorithm implemented in BouncyCastle library. Length of the key is 56 bits. Today, industry standard is 3DES algorithm, which stands for triple usage of DES algorithm in sequence, but every time with different key. Length of the complete key in 3DES algorithm is 168 bits. But, when we look from different aspect, we can see that mobile devices have very limited processor power, and running cipher algorithms on these devices sometimes can become very big problem.

Let’s recapitulate: every single data which is sent from client to server and vice versa is pre-encrypted. Communication key is encrypted with user PIN code on user’s mobile device. After very detailed consideration of mBank system architecture and cost-benefit analysis, we decided to use symmetric algorithm. Main assumption is that mobile device is something that is very personal, something which is always carried around, and is protected by its owner, so that, when key is once entered in phone, there is no need to change it in near future. And if phone has been lost or stolen, user can inform central service, and his mBank application will be no longer usable. One of the next steps of the project we are working on would be development of the secure remote key change algorithm.

On the following picture we can see characteristic sequence diagram, for the use case named „Account status“. Diagram is generalized with the goal of showing concept of system work, and hence communication details are excluded.

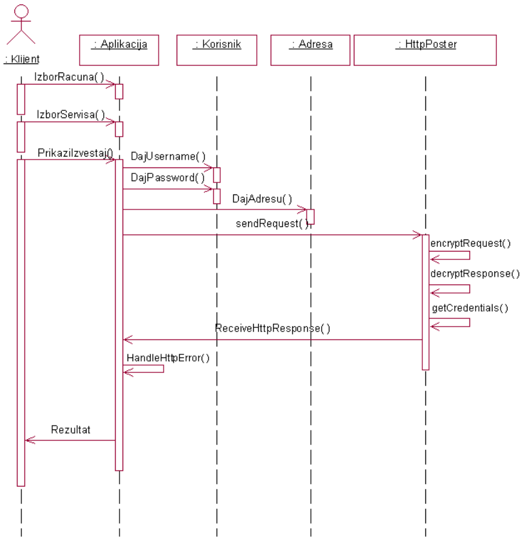

Figure 5.1 Sequence diagram for the use case „Account status“

From the main menu, user selects account for which he wants to check status, and then he chooses service named „Account status“ (Figure 5.1.).

Since user has pre-defined his/her user name, password and the address of service in Settings menu, values of these parameters are automatically read and sent by application. For every call of web service, class HttpPoster creates new thread, and so, application is not blocked and don’t need to wait for an answer.

When Web service is called for the first time without authorization, it adds X-WWW-Authenticate part in the Http header of reply, and as its value sets „Challenge“ which has been automatically generated and has been waited for response. Client receives connection error 401 UNAUTHORIZED, but with modified header. Now, client ciphers password using given Challenge, and again calls Web service, but this time with cipher. Then, client authorization commences, and if it is accepted, message content is being decrypted followed by execution of one specific method. Reply is also ciphered and returned to client along with Http response about successful authorization (200 HTTP_OK). That response is embraced by HttpPoster, who deciphers it and as a result returns text which is then sent to the Application. User is then presented with the report (picture 6.), and if some error has occurred, appropriate notification is shown.

6. CONCLUSION

Mobile client/server applications have an edge over WAP applications, which are instead easier for development. But, banking services are much more demanding when we look from the aspect of security, and for the time being, client/server applications are the only solution which can satisfy these standards.

The main weakness of these solutions is complexity of their architecture, as well as of the system, but also there is a lot of scepticism shown by common banking systems toward introduction of new technologies, which are not entirely acknowledged in practice.

With broader implementation of new versions of MIDP specifications in new mobile devices, J2ME platform will be more commonly used for development of banking services, which will bring development of mobile applications closer to development of classic client/server applications, and their larger practical application.

BIBLIOGRAPHY

[1] Miroslav Minović, Velimir Štavljanin, Dušan Starčević, Miodrag Dobranić, „ J2ME MOBILE BANKING CLIENT IMPLEMENTATION “, Proceedings of the symposium InfoTech 2004, 24.5. – 28.5.2004. Vrnjačka Banja, SCG;

[2] Marko Petrović, Miroslav Minović, Saša D. Lazarević, „Mobile devices and .NET web services“, Proceedings of the symposium YuInfo 2004, 8.3. – 12.3.2004. Kopaonik, SCG;

[3] Dušan Starčević, Velimir Štavljanin, Miroslav Minović, „ECML and Mobile wallet“, Proceedings of the symposium SYM-OP-IS 2003, 30.9. – 03.10.2003, Herceg Novi, SCG, ISBN 86-80593-33-8;

[4] Dušan Starčević, Velimir Štavljanin, Miroslav Minović, „ APPLYING WAP 2.0 TEHNOLOGIES IN BANKING" Proceedings of the symposium Info-Tech 2003, may 2003, Vrnjačka Banja, SCG;

[5] Obrenović Željko, Starčević Dušan, Štavljanin Velimir, Batalov Vladimir, "APPLYING WAP/SMS TEHNOLOGIES IN BANKING" Proceedings of the symposium TelFor 2002, 26-28. november 2002, Belgrade, SCG

[6] Jonathan Knudsen, Wireless Java: Developing with J2ME, Second Edition, Apress, March 5, 2003

[7] Martyn Mallick, Mobile and Wireless Design Essentials , John Wiley & Sons; 1 edition (March 23, 2003)

[8] Michael Juntao Yuan, Enterprise J2ME: Developing Mobile Java Applications , Prentice Hall PTR; 1st edition (October 20, 2003)

[9] John Poal Mueller, Using SOAP, Special Edition, Que, 2002

[10] Nokia Forum, A Brief Introduction to MIDP Clients for Web Services, April 1, 2003