A MOBILE CHESS GAME

Boyan Bontchev
Faculty of Mathematics and Informatics, Sofia University, Bulgaria
Nickolay Gabarev, Hristo Pavlov
Skala Soft Ltd., Sofia, Bulgaria

Abstract: Mobile gaming is growing alongside voice communication and exchanging text messages into an essential area of application for mobile devices. The paper presents architectural and user features of a new chess game, which can be played simultaneously in all modern Internet communication lines for mobile phones: through Web browsers, WAP devices and SMS mobile phones. The software architecture is based on a game engine residing on the server side, and a client side chess viewers and verifiers. The HTTP client is a Java applet, while the WAP client is implemented by WML cards containing the verifier done by WML script.
Key words: mobile games, WAP, WML, Java

1. INTRODUCTION

The number of mobile devices sold on the market is already outstripping all forecasts with a certain degree of regularity. Their modern communication channels as SMS (Short Message System) and WAP (Wireless Application Protocol) appeal in particular to young users around the world as generators of fun and games [1, 2].

Nowadays, game portals provide users with access to various mobile and mixed games with different *application business scopes, client device types,* and *complexity*. Regarding business scope, there are various games played just for fun, as a competition, or for betting in horse races, Internet and WAP casinos, etc. The client device type determines the channel access to the game. Game players may access the game server through one of the following ways or, using any combination of them:

- HTTP browser
- WAP browser
- SMS mobile phone
- Palm clients The complexity of the games servers determines what king of game architecture should be adopted in order to design an flexible, extendible and scalable solution. The server architecture is a crucial issue for both Internet and mobile games.

The paper describes the software architecture of the Mobile Chess game (called m-Chess) and shows parts of its user interface. The m-Chess is a Java server-side application for multi-user playing the traditional chess game. The mobile chess game is available not only through WAP channels but as well through HTTP browsers. After discussing the implementation issues of the current m-Chess, we describe our plans for future extension of the game via SMS user interface preserving the current functionalities.

2. M-CHESS SOFTWARE ARCHITECTURE

M-Chess is accessible for authorized users through Web browsers (MS IE and Netscape Navigator) and via mobile telephones with WAP support. Moreover, the game will be played in an off-line mode by SMS interaction flow between users and the game server. Fig. 1 shows the access channels to the game server. HTTP browsers connect to the game server directly through Internet while connections to

WAP browsers go via a WAP gateway. Similarly, SMS clients receive and send SMS messages from and to the game server via a SMS center (SMSC).

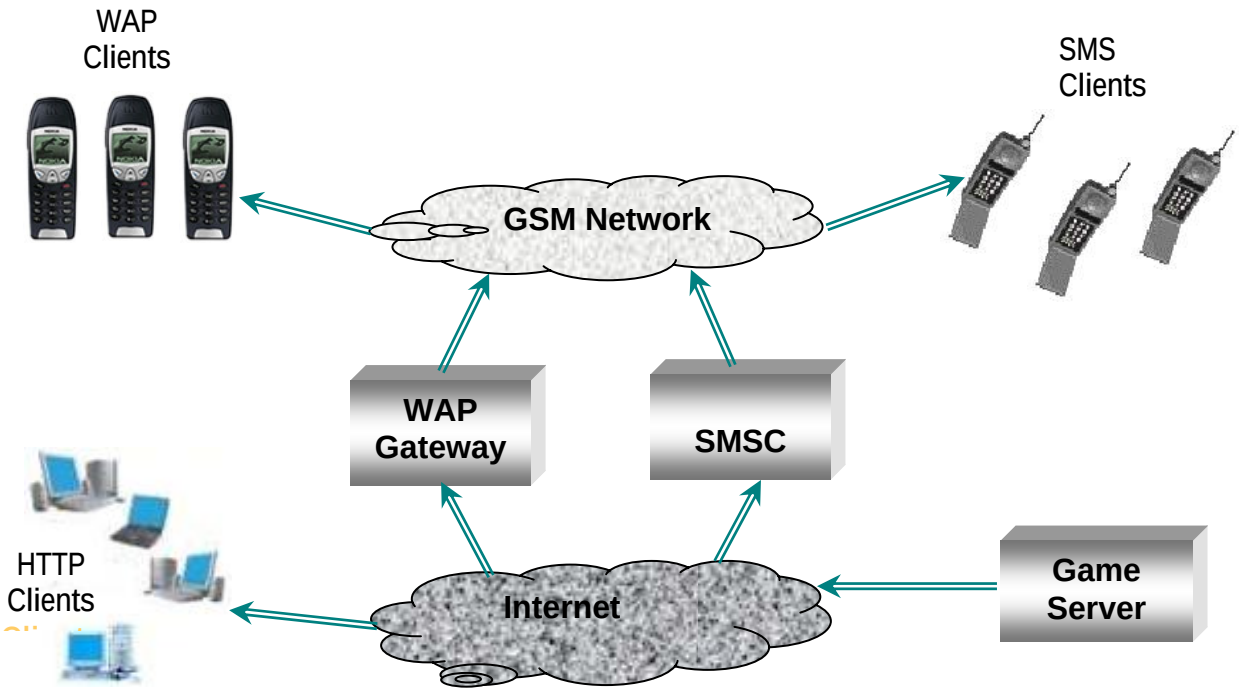

Fig. 1: Access channels to the m-Chess game server

The game architecture depends on the complexity both on client and server side. It is an important issue for multi-user games where designers should treat with scalability and synchronization among the concurrent players. This is the case of the m-chess game, which is real multi-user game.

Depending on the way of game engine design concerning its eventual incorporation within an application server there are two main architectural types:

1. embedded game server - relatively simple, incorporated into a JSP application and running under a Java application server, any-ML clients but not applets. This architecture is based on the principles described in [3].
2. standalone game server - more complex, the engine runs as a Java application with TCP socket connections to any clients including applets.The standalone game engine provides several important advantages for design of complex multi-user games:

- there is no need of any application servers
- faster and reliable communication between clients and the server
- promotes complex inter-communication protocols and allows usage of applet clients
- it is highly scalable, robust and easy for maintenance

Following these advantages, we have chosen standalone game engine for the m-Chess game. The game architecture is presented in fig. 2. The game server is the core component of the game. It is a pure Java application and incorporates the following:

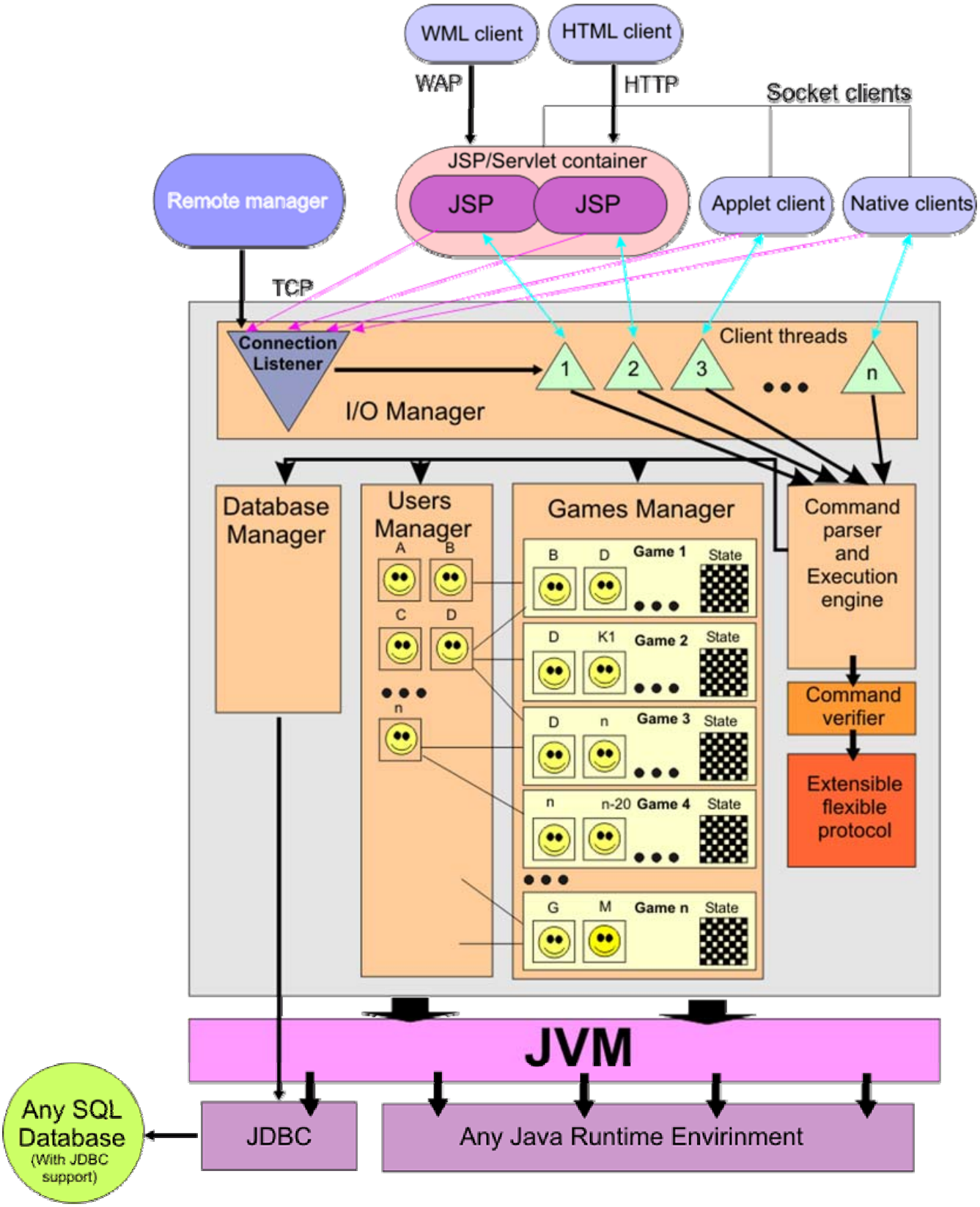

Fig. 2: The m-Chess game architecture

- games manager - controls all active chess games among different users
- user manager - responsible for user registration, login, and maintenance of the chess players
- I/O manager - it is a multithreaded socket server which takes care about client connections (via Java sockets), handles client requests and drives data via clients
- command parser and execution engine - includes flexible and extensible protocol based on very simple commands exchanged between the game server and the clients. As well, it contains a server-side chess verifier, which verifies client moves. It implement all of chess rules
- database manager - supports a single database connection via JDBC

The m-Chess database stores users' information, chess parties in use, unfinished parties and some statistics. It can be any kind of SQL database with a JDBC driver available. The database can reside on other machine, different than machine with game server (due to the JDBC connectivity).

For remote administration of the game server, there is designed a special remote manager, which provides a GUI based tool for implementing and sending server command, server log, visual error alert, client count and other useful information.

The applet clients connect with server vie TCP/IP socket and talk directly to server like a native Java client application. They incorporate a built-in chess move verifier. The WML and HTML clients are based on markup language generation by means of JSP (Java Server Pages), which provide connections, verifications and other issues. In fact, the applets, native clients and JSP's are the socket clients for the game server assuring high reliability and scalability by means of TCP socket connections. A JSP generates page contents depending on data and request. The user input forms and graphics (WBMP format for WAP clients) are dynamically generated with Java images and texts.

3. GAME IMPLEMENTATION AND USER INTERFACE

The m-Chess implementation follows closely the architecture discussed in the previous chapter. In the development process of the WML client we found various problems and their solutions:

- the WAP 1.1 protocol does not support any push from the server side [4] – it was solved by client pulling of new context after a timeout
- WAP does not support cookies - our solution is based on session support via URL rewriting
- mobile devices normally cash the presentation –solved by URL rewriting with a time stamp
- there are restrictions with presentation size –solved by tricky layouts and WML scripts
- image layout varies with different mobiles – in order to avoid differences in picture layouts, we implemented on-the-fly generation of composite images

All the clients contain build-in chess move verifiers. Both applet clients and HTML/JavaScript clients provides some enhanced features as drag-and-drop, however, WAP clients are rather simpler due to the WML restrictions.

The game is designed to be used by both the observers and authorized players (within given party of chess). It matches users with given skills in the chess game and starts for each couple a chess party. The party can be played asynchronously, simultaneously in Web and WAP (on-line) and, in the future, in off-line mode (by SMS alerts). Fig. 3 shows the game process using a WAP browser (a) and a view of the applet gaming (b). All the clients provide easy user internationalization.

a. b.

Fig. 3: Playing m-Chess by WAP client (a) and by Web browser (b)

4. CONCLUSIONS

The authors believe this game and similar table games will have success on the market, as players are not restricted to play by a given client device only. They will be able to change the client device dynamically and to play with other persons without matter what is the opposite client device.

The future work planned for the m-Chess enhancement is in two main directions:

- support of SMS client interface based on the current command exchange protocol between the game server and the clients
- incorporating of a chess engine for a single player mode - there are many free chess engines which can be ported from any language to ANSI C (for a faster server response)

The market target of m-Chess are any mobile Internet companies– from GSM operators and ISP providers till Web publishers. There is a huge community of chess players, thus, providing them with Web/WAP/SMS chess companies will gain opportunities for effective advertisement attacks.

REFERENCES

1. Siemens Ltd. Gaming for the mobile world. Siemens white paper, *http://www.siemens.com/*.

2. S. Peterson. Mobile Games Get Muscle, *OS Opinion Journal*, http://www.osopinion.com/perl/story/16613.html, March 5, 2002.

3. Bontchev B., S. Ilieva. Middleware Service Support for Modern Application Presentations. *Proc. of 16-th Int. SAER Conference*, 2002.

4. Nokia Mobile Internet Toolkit 3.0 User's Guide. *http://americas.forum.nokia.com/wap/*, 2002.