Multiplayer Mobile Game IDE

Ma. Suzanne Correa, Edward Mangaliman, Kathrina Anne Molera, John Lendl Yco, Danny C. Cheng

correa_kris@yahoo.com, etmangaliman@yahoo.com, kmolera10@yahoo.com, jolenz03@yahoo.com,
chengd@dlsu.edu.ph
College of Computer Studies De La Salle University 2401 Taft Ave. Manila Philippines 1004

Today, a number of mobile phones are equipped with different networking technologies such as Bluetooth, GPRS and 3G with the ability to connect to other mobile phones and play games. With this, mobile games are becoming increasingly popular. There is a boost in the demand for game designers and developers to create more mobile games. However, game development on mobile devices still requires low level programming and compatibility checks on actual devices. Thus, the project aims to minimize the need for such level of knowledge requirement in order to expand the developer base. The proponents designed and developed use an Integrated Development Environment (IDE) to help developers and non-developers to create two-dimensional, multiplayer games with top and side views for mobile devices. The research provided ways on how the connection between mobile devices may be implemented to cater to multiplayer games. This is all done without the need for low level programming, thus the developer can concentrate on the game play itself. The produced system may be used by any developer to create multiplayer games limited to phones powered by Windows Mobile. Further studies may include developing games for other platforms such as Symbian and Java.

1. INTRODUCTION

Multiplayer games allow two or more players play with each other or against each other on a same game [1]. On the other hand, single-player mobile games have dominated the mobile world because of their inherent single-user in nature status [2]. These mobile games have provided users a means of entertainment and provided added features on mobile phones. However, these games could only be acquired in mobile phones by downloading, which would usually require a fee. Another option of having them is to develop these games, which would require time and knowledge in implementing them.

Developing a certain mobile game requires programming knowledge, and skill. Not only that, developers should be able to familiarize themselves with the different tools, development environments and programming languages. Companies that focus on mobile game development uses different IDEs in order to create their desired games. They use these IDEs to design the different game elements (i.e. Characters, Scenes, etc.) and their ways of interacting with each other.

Current mobile devices are now readily equipped with different networking technologies that allow them to connect to another mobile device or to a computer. This allows for the developer of mobile games that are now capable of interacting with other players over the network.

Networking architectures should be taken into consideration while developing multiplayer mobile games. Similar to PC networking, a client-server architecture exists in order to gain interactivity between devices. In this manner, a stationary mobile phone can act as a server and one or more phones as the client [3]. Since most individual handsets do not have any IP addresses, the data traffic is routed through a tower and over the operator’s equipment [4].

Nokia sees possible strengths on implementing mobile games. One of these is having large audiences for mobile games since a big proportion of the population now owns a mobile phone. The idea of being able to play anytime and anywhere is possible because mobile phones are innately portable and networked. Networking capabilities are already inherent to mobile phones having the multiplayer games feasible given other limitations [5].

With the strengths stated, there lies a need to allow for more developers to be able to tap into this growing market more easily and be able to develop games with less effort.

2. SYSTEM OVERVIEW

The GMobile system is a multiplayer mobile game IDE (Integrated Development Environment) or tool which allows a user to create either a single or a multiplayer mobile game using their personal computers. The different games that may be produced using the IDE are turn-based games and synchronized (real-time) games.

GMobile users may be able to create two-dimensional, tile-based games with top-view perspective. The users are not required to have prior knowledge on mobile programming and networking to be able to produce a mobile game using the IDE. The IDE provides a way for users to be able to add different objects on their workspace and to associate events on how they want their objects to interact in their game.

GMobile IDE has provided pre-defined set of events, and type of characters that the developers may use in creating their games. The system makes use of the drag-and-drop feature as a substitute for programming. Interfaces were
provided for users to be able to apply the game rules, define the events and set the object strategies in the game. All the rules are inserted into pre-defined game templates in order to produce the game of the user.

Users of GMobile also have the option to incorporate networking capabilities in their created games. Having the same game session, the players may be able to interact with other players through the multiplayer feature with the use of their respective mobile devices.

Networking capabilities of GMobile is restricted to cellular or wireless LAN and Bluetooth. To trigger this networking feature, developers just need to enable the multiplayer mode of the game.

Players may experience problems (i.e. connectivity, lags, jitters, and the like) during their game play. Problems mentioned are already innate in a networking environment; hence, algorithms were implemented to retain consistency of states. The games designed with GMobile have features to interpret the types of messages coming from other mobile devices. With this, games were provided with exception handling methods and state management techniques to properly execute the different messages from the devices.

3. SYSTEM ARCHITECTURE

This section shows the designs and architectures of the system. It includes designs for the IDE, Game Skeleton and Networking. These designs were used in the implementation of the system.

4.1. Integrated Development Environment (IDE)

The IDE has thirteen (13) core functions, as seen in Fig. 1: Manage Projects, Manage Image Resource, Manage Sound Resource, Manage Levels, Manage Scenes, Manage Objects, Manage Object Types, Manage Variables, Manage Events, Manage Actions, Manage Networking, Build Game and Run Game. All Manage functions (except for Manage Projects which handles creation of a new project and opening and saving of a project) includes adding, modifying and deleting data. For example, Manage Objects means adding, modifying and deleting game objects.

Fig. 1. IDE Use Case Diagram

It must be noted that management of events and actions can only be done when managing an object or an object type. Actions can only be managed through events. Scenes can be added, modified or deleted only if a level exists and objects can only be added if a scene exists.

Architecture

Fig. 2. IDE Architectural Diagram

The IDE is composed of seven (7) packages: the GUI, Elements, Actions, ActionGUI, Handler, Events and Code Generator. These packages interact with one another to direct the development and execution of the game. Their connectivity is shown on Fig. 2.

4. SYSTEM FUNCTIONS

Creating a Game

Since the different parts of the IDE have been discussed already, then creating a game will now be tackled.
IDE Activity Flow

Fig. 3 shows the flow of the creating a game in the IDE. First the user must create a level then a scene, then add objects in a scene. Objects cannot be created without a scene and a scene cannot exist without a level. Objects can be grouped into certain objects types and then the objects may be assigned events then actions. Actions cannot be assigned without events. Events added to objects types are applicable to objects under the object type but events for an object will only be applicable to the object.

Once all the elements are added, the user can opt to make his game multiplayer. If so, he could do this by configuring the networking and enabling the multiplayer mode. The user can enable or disable networking any time in the creation of his game.

After adding everything in the game: levels, scenes, objects, object types, events, actions, networking, the user could build and run the game. He may build and run the game any time in the creation of his game but it is recommended that the game be built and run after all elements has been added.

The user can add, modify, and delete elements (objects, scenes, levels, object types, events, actions) all over again depending on his preference.

Levels
GMobile IDE starts with a new project which contains a default level and a default scene. Users can opt to create additional new levels and new scenes in levels.

Scenes
Levels have at least one scene. These scenes hold the different scenarios that can happen during the game. Scenes can be added through the Game sub-window. Scenes’ properties can be adjusted to match itself with the game play such as background effects (through image or color), the name, the dimensions, and others.

Creating Objects
A user may opt to create an object through the functionality found in the Menu Bar, Toolbar through the Game sub-window. The creation of an object requires 3 fields in order to exist. These are its type (i.e. PlayerCharacter, Non-Player Character, or defined by the user), name, and the image that will be associated to it.

Image Association
Associating an image on an object or even a particular scene background can be done by importing an image from the Graphics Manager. Users can create different level designs with the use of the different objects they will use. Created objects will appear on the scene currently being used.

Creating Events and Actions
Game objects may interact with other objects.

Creating and Associating Events
Using the Event Properties window for each object, users may create and select the appropriate events or movements that they want for their objects. Users can also associate their created events with the appropriate actions that they want for their objects in the Event Properties Window.

Incorporating Multi-player Capabilities
Multiplayer capabilities may be incorporated into the game being produced through the Configure Network Window in the IDE where users may choose what networking technology to sue and how many players can use their game handle.

5. RESULTS AND OBSERVATION

6.1. Network Testing

Bluetooth Profiling
Two (2) devices were used by the proponents to test the connection for the Bluetooth technology. One acted as the client and the other as the server. The communication between the two devices has been profiled and random messages were selected from the log file produced. Table 1 shows the summary of the result of the log file.

<table>
<thead>
<tr>
<th>Bluetooth</th>
<th>Received</th>
<th>Processed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>24.1261 sec</td>
<td>0.1615 sec</td>
</tr>
</tbody>
</table>

Table 1. Network Profiling Summary for Bluetooth Technology (2 Devices)

Table I shows the summary of the total time of all the messages received and processed. For the received column, it shows that for a Bluetooth connection wherein two devices are connected and one acts as a server and the other as the client, the total time from the sending to the receipt of the messages for the client is 24.1261 seconds and the total processing time of all the messages
is **0.1615 seconds**. The results show that there was an identified long amount of lag for the Bluetooth technology but once the message has been received, it is processed at once, as seen with the processing time which is less than a second.

**IP-Based (Wireless LAN Profiling)**

Table II. Network Profiling Summary for Wireless LAN Technology (2 Devices)

<table>
<thead>
<tr>
<th>Wireless LAN</th>
<th>Received</th>
<th>Processed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>0.2251 secs</td>
<td>0.1829 secs</td>
</tr>
</tbody>
</table>

Table III. Network Profiling Summary for Wireless LAN Technology (3 Devices)

<table>
<thead>
<tr>
<th>Wireless LAN</th>
<th>Received</th>
<th>Processed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client1</td>
<td>0.7187 secs</td>
<td>0.0442 secs</td>
</tr>
<tr>
<td>Client2</td>
<td>0.7078 secs</td>
<td>0.0352 secs</td>
</tr>
</tbody>
</table>

Table IV. Network Profiling Summary for Wireless LAN Technology (4 Devices)

<table>
<thead>
<tr>
<th>Wireless LAN</th>
<th>Received</th>
<th>Processed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client1</td>
<td>0.7202 secs</td>
<td>0.1101 secs</td>
</tr>
<tr>
<td>Client2</td>
<td>0.7252 secs</td>
<td>0.0898 secs</td>
</tr>
<tr>
<td>Client3</td>
<td>0.1128 secs</td>
<td>0.5509 secs</td>
</tr>
</tbody>
</table>

The same test was conducted for the IP-Based for the Wireless Technology. Tables II, III, and IV show the results for two, three and four devices. The data tabulated are the amount of time the message is received upon sending and the total processing time. It can be seen that the transmission time of the IP-technology is faster compared to Bluetooth. The processing time however is almost the same.

**6. CONCLUSION**

This IDE focused more on code generation. The proponents did not concern themselves on creating an engine where games will revolve. Generation of the code based on the game design is done during the compilation of the game.

Network technologies such as Bluetooth and GPRS were studied to know their limitations and issues if certain multiplayer mobile games will run under them and how the games can be implemented using these technologies. Network architectures and algorithms for message passing were designed for connectivity and to promote consistency of game elements during game play. Games created with the IDE can be incorporated with multiplayer capabilities. Games can run under Bluetooth, cellular and wireless technology. This means that games can run under the client-server architecture or through peer-to-peer architecture.

User acceptance testing was done by the proponents based on the criteria they have established. The people then evaluated the system based on the criteria given and were asked to give their comments, recommendations and suggestions on the system. The results of the user acceptance testing appeared to be satisfactory as the average answer of each respondents falls under the rate of 4 – 5. Although they found the system very simple to use in creating games, they also gave suggestions to improve the GMobile system and its implementation.

**7. RECOMMENDATIONS**

GMobile is an IDE that allows users to create multiplayer mobile games. The IDE includes basic functionalities such as adding, deleting, and associating events of different objects that will be included in the game. Things such as scripting, editing of code templates are not possible for this current version of IDE. Thus, it is recommended, if this study will be extended, to add scripting for users to add codes for the implementation of other features such as Artificial Intelligence (AI), events and actions that the user wants to include in his game.

GMobile was constructed under C# and .Net Framework, which made the IDE code templates to be limited under this technology. Computers cannot run the IDE if they don’t have the .Net Framework. Hence, it is recommended to have other programming languages be studied (e.g J2ME and C++) so that various code templates will be compatible in the compilation and building of the game.

**8. REFERENCES**


