Augmented Reality and Image Recognition Based Framework for Treasure Hunt Games

Zsolt Bálint, Botond Kiss, Beáta Magyari and Károly Simon
Babeș-Bolyai University, Cluj-Napoca, Romania,
Faculty of Mathematics and Computer Science,
Computer Science Department
bzsozi@preferansz.ro
kssboti@gmail.com
beata.magyari.89@gmail.com
ksimon@cs.ubbcluj.ro

Abstract— The scavenger hunt and treasure hunt game categories are relatively well-known. The main idea of these games is to reach different checkpoints and execute specific tasks at these locations. There are many successful games based on this idea, like Geocaching. There are also some frameworks for building games, like Scvngr, but generally these frameworks have some disadvantages. The placement of real physical objects ("treasures") could be expensive, GPS-based localization cannot be used inside buildings etc. The lack of generality could be a problem, as well.

A general framework for the development of augmented reality-based, multi-user online treasure and scavenger hunt games for mobile devices. The "treasures" can be virtual 3D objects, appearing as a part of the augmented reality. The integrated image recognition support offers new possibilities in game creation and customization. GPS-based localization is combined with image recognition based localization, in this way the created applications can be used inside buildings, as well.

Keywords- treasure hunt game, scavenger hunt game; augmented reality; image recognition; Android; iOS; Vuforia framework

I. INTRODUCTION

The realization of a framework supporting the development of augmented reality-based, multi-user online treasure and scavenger hunt games on smartphones is presented. These game categories are relatively well-known. The main idea is to reach different checkpoints and eventually execute specific tasks at these locations, like acquiring information or finding a specific object etc. The proposed ARTHAS (Augmented Reality for Treasure Hunt Applications) framework supports the development of these types of games, and provides exciting new possibilities offered by virtuality. At the checkpoints the "treasures" can be found as a part of the augmented reality.

The development of the framework was motivated by the fact that the popularity of treasure hunt games is rapidly growing, but currently there are only a few games using augmented reality. During the creation of a regular treasure hunt game, the placement of real physical objects (treasures) is time consuming and expensive. By using virtual objects these costs can be significantly reduced. Furthermore, augmented reality can serve with many other interesting solutions, allowing the transformation of scenes.

ARTHAS also provides image recognition support, offering new possibilities in game creation and customization. For achieving a better accuracy, GPS-based localization is combined with image recognition based localization. Using this technique, the created applications can be used inside buildings, as well.

The framework is based on client-server architecture and its complexity is relatively high, being composed by several subsystems. When developing such a framework the compliance with appropriate development principles and the optimal selection of design patterns is very important. It is essential to select an efficient method for dealing with concurrency. A central server is responsible for controlling the entire game, which means that many clients have to be served at the same time (for example, in the case of a competition for teams). The architecture of the framework and the method used for communication and concurrency management are briefly discussed in the paper.

Client applications have been developed for the iOS and the Android mobile platforms. While playing, the checkpoints provided by the server are displayed on a map, and a proposed route is marked to help the user finding and reaching them. The Google Maps API is used for providing this service.

Reaching a checkpoint, the exact location of the hidden object is determined using image recognition. When the exact location is found, the "treasure" can be seen in the augmented reality. The Vuforia framework is used for providing these services.

The requirement specification of the project, the domain and use case analysis is also presented in the paper, and some implementation details are briefly described. Some utilization and further development possibilities are also discussed.

The first public game created by the framework is currently in the testing phase. Some scenarios from this game are also presented.

Creating treasure hunt games is not the only possibility for using ARTHAS. The framework could also be used for creating touristic, educational and other useful applications.
II. TECHNOLOGIES, DESIGN PATTERNS, DEVELOPMENT TOOLS

ARTHAS is a relatively complex framework. It is based on client-server architecture and uses several other frameworks for providing some specific services.

A. Server-side technologies

The technologies used for developing the server are listed in Table I.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MySQL Server 5.5.8</td>
<td>Database server</td>
</tr>
<tr>
<td>JDK 1.6 [8]</td>
<td>Java Development Kit</td>
</tr>
<tr>
<td>Eclipse 3.7.2 (Helios)</td>
<td>Java Development Environment</td>
</tr>
</tbody>
</table>

For realizing the client-server communication the Apache Mina framework [4] is used. The Android client also uses the Mina framework for establishing the connection with the server. The communication between the server and the mobile client applications is based on the TCP/IP protocol, which is realized by the Mina framework using the Java NIO API.

The main advantage of Mina is efficient concurrency handling. Thousands of client requests can be served at the same time, using a single execution thread on server-side. It is not necessary to perform expensive thread creation operations when new clients are connected. With the help of Mina the server can process messages on multiple levels using certain filters.

The communication between the iOS client and the server is realized using the CocoaAsyncSocket framework [10], which provides powerful asynchronous socket libraries on the iOS platform.

B. Client-side technologies

The client applications are implemented for the iOS and the Android mobile platforms. The technologies used by these client applications are listed in Table II. and Table III.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android Development Tools</td>
<td>Development environment for Java</td>
</tr>
<tr>
<td>Android SDK Tools</td>
<td>Android software development kit</td>
</tr>
<tr>
<td>Android NDK Tools</td>
<td>Tool for integrating native code in Android</td>
</tr>
<tr>
<td>Apache MINA 2.0.2</td>
<td>Framework for client-server Communication</td>
</tr>
<tr>
<td>Google Maps API</td>
<td>API for using the Google Maps service</td>
</tr>
<tr>
<td>OpenGL ES</td>
<td>Framework used for 2D and 3D Graphics</td>
</tr>
<tr>
<td>Vuforia SDK</td>
<td>Augmented reality software development kit, providing image recognition support</td>
</tr>
</tbody>
</table>

Android [8] is a Linux-based operating system mainly used by smartphones and tablets. The main language for implementing Android applications is Java. The Java runtime environment used by the platform is provided by the “DalvikVM” virtual machine. The Linux kernel is responsible for handling the integrated hardware of the mobile devices, for ex. touch screen, Wi-Fi, etc.

The architecture of the platform is multi-layered. On the lowest level there are the Linux drivers. On the second level we can find system libraries. The core element of the system, the Android Runtime, is also part of this layer. It contains the Dalvik virtual machine and the main Java packages. The third level is the application development framework, with Java packages. These packages are used by the applications running on the platform.

Android development is supported by ADT (Android Development Tools), an Eclipse plug-in, which facilitates efficient project construction by providing a corresponding development environment. The Android SDK (Software Development Kit) provides many useful tools and libraries, and it contains the emulator.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective-C [5] [6]</td>
<td>Programming language used for iOS</td>
</tr>
<tr>
<td>iOS SDK 5.1 [7]</td>
<td>Software development kit for iOS</td>
</tr>
<tr>
<td>Xcode 4.3 [7]</td>
<td>Objective-C development environment</td>
</tr>
<tr>
<td>Vuforia SDK v1.5 [2]</td>
<td>Augmented reality software development kit, providing image recognition support</td>
</tr>
</tbody>
</table>

iOS is the operating system of iPhone, iPad and iPod touch devices, managing the hardware of these devices and providing technologies required to implement native applications. iOS consists of four layers: Core OS, Core Services, Media and Cocoa Touch. Objective-C is the primary programming language in Mac software development, so in iOS development, too. It is an object-oriented language derived from C and Smalltalk.

The iOS SDK is a software development kit containing all the necessary interfaces and tools for developing, installing and running native applications on iOS devices. The SDK packages are grouped according to iOS layers, the development is realized using these packages.

For the iOS client application the Google Maps service integration has been realized using the Map Kit framework. Map Kit provides interfaces for embedding maps into application views, for adding annotations and overlays to the map, and for performing reverse-geocoding lookups.
C. Frameworks for Augmented Reality and Image Recognition

For image recognition the Vuforia [2] technology is used. Vuforia is a framework developed by Qualcomm. It uses the mobile device as a "magic lens" or a looking glass into an augmented world where the reality and virtuality appear to coexist. In the application the live camera preview image is rendered on the display to represent a view of the physical world. Virtual 3D or 2D objects are placed over the live camera preview and they appear to be tightly coupled with the real world.

The image recognition support provided by the Vuforia framework can be used for achieving better localization accuracy. Based on the camera image, different algorithms are applied for detecting targets, and after detection the augmented reality views are rendered. The Target Management System of the http://ar.qualcomm.at site produces data and xml files from images. By assigning these files to the project the appropriate data for image recognition is provided.

The OpenGL ES (OpenGL for Embedded Systems) is a platform-independent API [1] developed by Khronos Group. It allows the management of the graphics card. Using this API, three-dimensional models can be visualized on mobile phones, PDAs and game consoles. It creates a powerful and flexible interface between software and graphics acceleration. In the case of the mobile client applications it is used for rendering the objects visualized in the augmented reality.

D. Design Patterns

The server has a multitier architecture. Several basic design patterns [3] are also used on server-side.

Database connection is realized using the Singleton pattern. The data access layer is based on the Abstract DAO Factory model. The instruction (command) processing classes are based on the Command and Singleton design patterns. Using the Command design pattern the server can be easily extended. No already implemented parts have to be modified for adding a new functionality.

The client applications follow the MVC (Model-View-Controller) design pattern. Command and Singleton design patterns are used on client-side, as well. The Delegate design pattern has also an important role in the case of the iOS client.

III. THE ARTHAS FRAMEWORK

The ARTHAS framework aims to provide opportunities for developing online treasure or scavenger hunt games for smartphones, in which the searched "treasures" are virtual objects, parts of the augmented reality. Checkpoint localization accuracy is improved using image recognition. The games can be developed by customizing specific framework elements.

A. Requirement specification

Functionality provided by the server:
- Database management
- Information services
- Game control

Functionality provided by the client applications:
- Registration
- Login and logout
- Accessing available games
- Browsing available games
- Providing map and route planning service
- Retrieving tasks and checkpoints
- Camera control
- Image recognition
- Representing the augmented reality
- Sending notifications to the server

B. Main entities and use cases

The main system entities are: users, games, checkpoints, tasks and messages. The corresponding model classes are grouped in a separate package, which is shared between the subsystems.

The application can be used by registered users. After a successful user login an available game can be selected and started. A game can be suspended at any time, and it can be resumed later. After starting a game the first checkpoint or task is retrieved from the server. Checkpoints are marked on the map and routes are proposed for reaching them. If the user finds the checkpoint using the GPS and the camera, or he resolves a task, he reports back to the server. The game continues with the next step, a new task or checkpoint.

C. Software architecture

The software consists of three subsystems: server, Android client and iOS client.

The application follows the principles of multitier software architecture. Data accessing is based on the Abstract DAO Factory design pattern. The main entities are stored in the ARLib package, this package being shared between the subsystems.

The server contains four more packages: message, command, game and database. The messages are received by the server's message handler class. These messages are processed by different classes, using the Factory design pattern. The command classes are implemented based on the Singleton pattern. The game package contains classes for controlling the game. The data access layer is composed by classes and interfaces from the database package.

1) The Android client

The main packages of the Android client:
- core: login, registration and game choosing views
- connection: classes responsible for the communication between the client and the server
- map: classes responsible for map rendering, checkpoint marking and route planning
- imagetargets: classes responsible for rendering images captured by the camera, image recognition and visualization of the augmented reality
After starting the application a connection handler object checks the internet connection and creates the connection with the server. This object is responsible for message sending, too. The received messages are handled by a client handler object.

The core package contains five views: account creation, login, game selection, submenu and a tabbed view. In the submenu a new game can be started or a suspended game can be resumed. The tabbed view holds three tabs: map, camera and database.

The map package contains one view. The corresponding activity renders the map provided by the Google Maps API. There are two classes for creating overlay objects holding checkpoints and the user's position. Another class is responsible for providing the route from the user’s position to the checkpoint. A separate class is responsible for drawing the checkpoint, the user’s position and the route on the map’s surface.

The database activity contained by the database package is the third tab within the tabbed view. It displays information about the user, the checkpoints and the game. The Vuforia framework is integrated by the imagetargets package.

2) The iOS client

In the case of the iOS client the most important groups are listed below:

- **ARConnections**: classes for client-server communication
- **ARMessages**: classes creating messages for the client-server communication protocol
- **ARMainViewControllers**: controllers for the main views
- **ARGameViewControllers**: controllers for game views
- **ARViews**: graphical user interfaces
- **ARVuforia**: classes for the integration of the Vuforia framework and classes responsible for displaying three-dimensional models
- **ARCommands**: command classes
- **ARMapAnnotations**: classes for map management

The AsyncSocket framework is responsible for network communication. According to the Delegate design pattern, the client is informed when the connection with the server is established, interrupted or data is read. After the decoding process the message is transmitted to the protocol handler class. Here the message is transformed into command using the Factory design pattern and this command is executed. The command classes are based on the Singleton design pattern. While executing a command an appropriate controller class receives a notification.

The controller class for the map view implements the appropriate protocol from the Map Kit. This controller receives checkpoints from the map annotation handler object, and displays these points on the map. A separate class is responsible for supplying the checkpoints to the map. It sends a request to the server for acquiring the points, and then passes the results to the annotation handler, which notifies the map controller.

The Vuforia framework is initialized by the camera controller. This controller is responsible for image management, placing the recognizable images into the proper image holders. It is also responsible for the instantiation of the controller class, which contains the view representing the augmented reality. This view class is derived from the corresponding class of the Vuforia framework, and it is responsible for the augmented reality view, performing drawing operations on the camera image. While looking for checkpoints with the camera, the image recognition algorithms provided by the Vuforia framework are working in the background. The view class creates model arrays containing the C and Objective-C data exported from three-dimensional models. The number of models displayed at the same time has to be specified. Vuforia continuously calls a method using a background thread for rendering the current view. The models are rendered using OpenGL functions.

D. Functionalities provided by the client applications

After starting the client application, the first view is the login page. After a successful login on the next view the user can choose between three options: start a new game, resume an already started game, and logout.

![Figure 1 Map view on the iOS client application. The current location of the user is marked with a blue icon. The next checkpoint is in the center of Cluj-Napoca. A green marker is displayed at the checkpoint's location and a possible route is proposed.](image)
During the "treasure hunt" the user receives tasks and checkpoints from the server. The targets are visible on the map, together with proposed paths. When the player reaches a checkpoint, he can find the "treasure" using the camera. After a successful "discovery" notification can be sent to the server.

IV. UTILIZATION POSSIBILITIES

The Augmented Reality framework provides possibility to develop online treasure and scavenger hunt games on smartphones, in which the "treasure hunt" happens in the augmented reality.

The development of a new game requires the customization of some framework elements. This customization process is very simple, due to the well-defined interfaces. Once the game idea is ready, and the rules are specified (for example, based on a storyline), the appropriate interfaces should be implemented accordingly on the server side. The definition of the checkpoints is the next step, for fixing the game scenes. In addition, it is necessary to specify the exact location of the searched object. This operation can be done by providing an image from the exact location. This image is added to the application, and it will be identified by the framework using the image recognition algorithms. The next step is the specification of the objects, which have to be shown in the augmented reality. These objects can be specified by two- or three-dimensional models. The three-dimensional objects can be edited with any 3D modeling tool, and then they have to be converted into C or Objective-C data to integrate them into the project. For the games different modes can be specified: team or individual competitions, or non-competitive games. Finally, when the new game is ready, it can be published for starting the "adventure".

V. THE FIRST ARTHAS GAME

The first treasure hunt game created by the ARTHAS framework is currently in the testing phase and it will be published in 2012. The location of the adventure is Cluj-Napoca. The game has some educational aspects, providing historical information about the city. The checkpoints are located in famous places, in this way the game also serves as a tourist guide. Old city scenarios, already missing buildings and attractions can be seen through the "magic lens", famous persons can be met in the augmented reality.

The game flow is specified by the following steps:

- Starting the game the user receives a task (e.g. a question) from the server
- By solving the task (e.g. correctly answering the question) the next checkpoint is provided
- The checkpoint can be reached using the map
- At the checkpoint the camera can be used for finding the hidden treasure. Image recognition is used for exact localization. The treasure is a part of the augmented reality.
- After finding the treasure a notification can be sent to the server and a new task (e.g. question) is received, for acquiring the next checkpoint.

For a better illustration here is a scenario from the game:

- **Checkpoint**: the courtyard of the Toldalaki-Korda Palace
- **Treasure**: an old picture (from 1928) is displayed in the augmented reality. Kós Károly and Bánffy Miklós are talking about the Erdélyi Helikon journal. They are going to meet with a poet at his favorite cafe.
- **Task/question**: Who is the poet? (Answer: Áprily Lajos)
- **Next checkpoint**: The coffee-room of the Continental hotel (the old New York hotel, Áprily's favorite place)
- **Next treasure**: an architect tool (a 3D object in the augmented reality)
- **Next question**: whose tool is this? (Answer: Pákey Lajos the architect of the building)

The game is composed by several similar scenarios.

VI. CONCLUSIONS AND FURTHER DEVELOPMENT

A framework for the development of augmented reality-based adventure games for Android and iOS platforms has been proposed. Due to the software architecture, a high number of simultaneously connected clients can be served; the system can be easily maintained and upgraded.

By customizing specific elements from the framework online treasure and scavenger hunt games can be easily created and published. The searched "treasures" are part of the augmented reality, in this way the cost of game creation is significantly reduced in contrast to games with real objects. Furthermore, augmented reality can serve with many other interesting solutions, allowing the transformation of scenes.
For "treasure" localization image recognition methods are used. In this way objects can be localized more precisely, and the application can be used inside buildings.

Currently the creation of a new game can be done by implementing specific interfaces from the framework. A further possibility is the development of a web-based user interface for game creation. More storyline specification options can be supported; the graphical views can be improved. Client applications for other mobile platforms (for ex. Windows Phone) can also be implemented.

REFERENCES

[3] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Design Patterns