Design Document

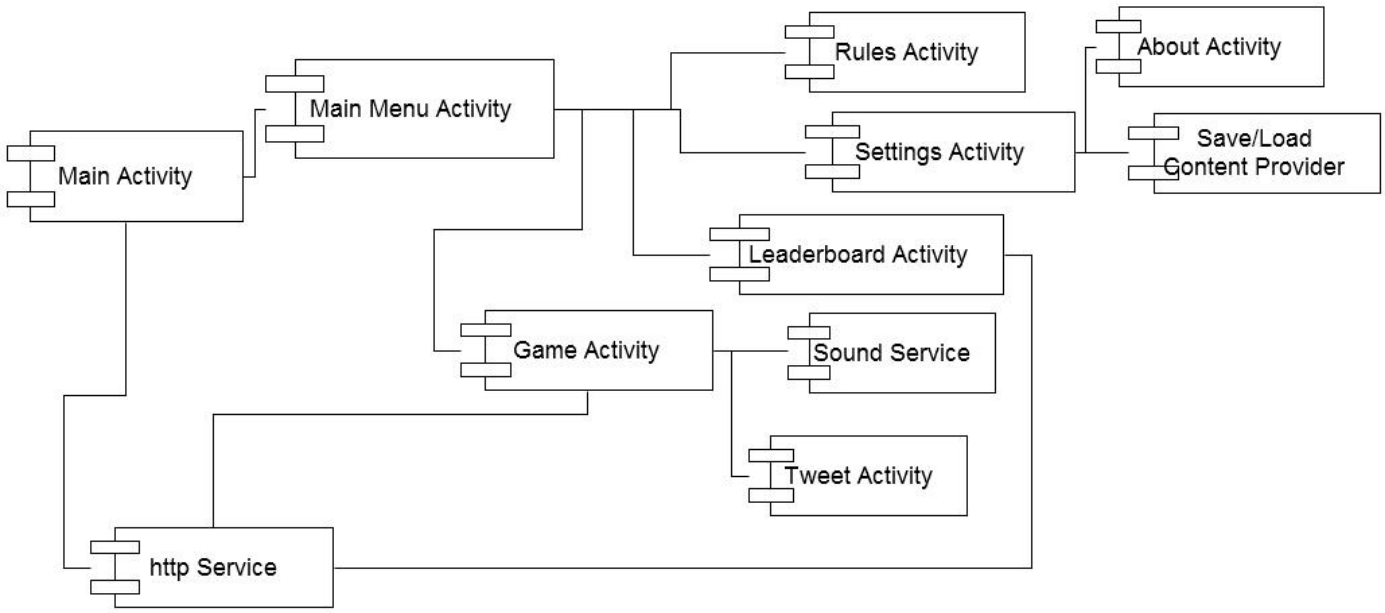
Vi-Char Mobile Development Division

Michael **DuBois**
Nathan **Pastor**
Robert **Shapiro**
Davis **Shurbert**
Kirah **Taylor**

Introduction

This document describes the design specifications for the mobile component of the Vi-Char game. Android owners who wish to participate in the game will be able to load and install the application onto any mobile phone running Android version 2.2 or higher. Through the application, users will be able to view and interact with the game's augmented reality world. In the main use context, users will open the application on their Android device. Upon doing so, the user will be prompted to create a player profile or login via Twitter. Once the user logs in successfully, the application will display the main game view, through which the user can view the game world by directing their device's camera toward augmented reality tags in the real world. Once the user has successfully located these tags, he or she will be able to interact with the game world by touching the device's screen -- these touch interactions may be simple button presses or more complicated multi-touch gestures, depending on the gameplay requirements. In addition to viewing and interacting with the gameworld, the user will also be able to access leaderboards and game information through a context menu. Moreover, when the webserver deems it is appropriate, the user will also be given the option to tweet a message (through our graphical user interface) in response to game-related "twitter challenges".

Architecture Design



The overall system architecture will include a motion capture module which will interface directly with the game engine. Users of the motion capture module will control the main game character. The game engine will in turn interface with the server. The server back-end is responsible for coordinating communication between the game engine and the mobile devices. The front end will consist of a website where players can view the leaderboard, and will also provide a way for users to sign in and post tweets which may affect the gameplay.

The architecture of the mobile application will be composed wholly of Android application components, including activities, services, intent objects, and content providers. All these components will reside in one Java package (edu.pugetsound.vichar). Other Android design elements, such as fragments, will be implemented as needed. AR will directly interface only with the mobile application, so game engine information intended for that component will be routed through ours.

Required interfaces include

An augmented reality view

Game state information related to UI elements

Gameworld information/coordinates

Unique user identification
Leaderboard information
Tweet results

Provided interfaces include
Gameworld information/coordinates
User registration request
User touch events
User tweets

Module Design

Main Activity

Description: The first screen of the game which accepts login information and supports creation of a player profile.

Purpose Statement: This module allows the user to log in or register, either via Twitter or through the game's own credentials. Login information must be authenticated through the game server or through the Twitter API, and will be stored locally for future use.

Provided Interface:

Credentials for Twitter

Credentials for game server

Required Interface:

Username authentication from server

Twitter authentication

Main menu Activity

Description: The game's main menu, which includes options such as launching the game, viewing the rules, and changing the application settings.

Purpose Statement: To allow the user access to more information and options, such as the rules, documentation, and optional settings.

Provided Interface:

Graphical User interface

Sound Service

Description: This service will play sound effects and/or music from the phone speakers depending on cues from the game engine.

Purpose Statement: To further engage the user with the game world and make it a more immersive environment in general. The user will get an audio response from their touch screen commands, as well as the visual response offered by other services. The addition of audio will make our game world more engaging and alert other players to the user's actions.

Provided Interface:

Plays audio

Required Interface:

Prompts from the game world. This may not necessarily be "play audio" commands from other modules, but possibly a "player has jumped" indicator, for example.

Http Service

Description: This service handles all http functionality our application will need to support, including contacting the game server and contacting the Twitter server.

Purpose Statement: To connect the mobile application to the other components of the game

Provided Interface:

Posts to server

Sends requests to server

Required Interface:

Post/request prompts (Service must know what to post/request)

Tweet Activity

Description: Allows users to tweet during gameplay

Purpose Statement: To implement our Twitter gameplay functionality, encouraging users to post Tweets which will have some effect on the game world.

Provided Interface:

Sends Tweet

Required Interface:

Twitter credentials

Settings Activity

Description: An activity page that allows the user to alter or disable non-essential features, such as sound.

Purpose Statement: It exists to allow the user to tailor their gameplay experience according to their own preferences.

Provided Interface:

Settings changes

User interface to change settings

Rules Activity

Description: A simple activity page listing the rules of the game and instructions for playing.

Purpose Statement: It exists to help the user quickly learn how to play the game properly.

Provided Interface:

User interface to view rules and instructions

Leaderboard Activity

Description: Displays current leaderboard information, personal score, and other game relevant scoring information that a user may find interesting.

Purpose Statement: It exists to give the user further information about the game and to make the playing experience more interesting..

Provided Interface:

User interface to view leaderboard/scores

Required Interface:

Leaderboard/score data from the server

Game Activity

Description: The central game activity, where the actual gameplay occurs and where the user is able to interact with the the game world.

Purpose Statement: It exists to allow the user to view and change the game world

Provided Interface:

Transmits screen gestures (touches)

Send game information to AR (from server via HttpService)

Required Interface:

Game information from server (via HttpService)

Game world fragment from AR division

Current Twitter challenge information from game engine/server (via HttpService)

Save/Load Content Provider

Description: Allows application to save and load information from permanent memory

Purpose Statement: To allow the application to store information permanently, including login credentials.

Provided Interface:

Saves information to permanent memory

Retrieves information from permanent memory

Required Interface:

Save requests

Load requests

Other Design Views

Per the Android best practices guide, all of our application assets will reside in the res folder. This includes strings for the GUI, background images for the menus, and gameplay sound effects.

Design Rationale

In most cases, the Android architecture and associated best practices guide our design decisions. The Android API provides a base Activity class to hold and control main views and their associated actions. Likewise, the API provides a Service class for invisible, background operations, and a ContentProvider class for local data storage and retrieval. In cases such as these, we see no reason to deviate from Android development standards. Of particular note, however, we have determined that the Android Fragment class represents the best way to display the Augmented Reality view, since it allows us to "black box" the internal operations of the view and overlay our own GUI Fragments without much trouble. In order to use this class and maintain support for Android 2.x, we have chosen to include the Android support package for Android APIv4. While this increases the size of our application, it allows for a clean implementation without compromising our compatibility requirements.

The Vi-Char team has decided that all mobile interactions destined for the game engine should be first passed to the web server. Likewise, we expect all game-display information to be passed to our application via the web server -- save game assets like graphics and sound files, which will be stored on the phone with the application's other resources.

Implementation Notes

The mobile application described here will be implemented in Android Java using standard Android packages. All division members will code in the Eclipse IDE. With the Android Eclipse add-on, this IDE will provide us with a powerful and intuitive development environment. Our division will use the Git version control system to track changes, and more specifically we will implement the Git Flow branching model. This model requires the gitflow extension which can be used with the msysgit or github clients. Our main repository will be hosted in a private account at GitHub.com.

Our documentation and API will reside alongside our repository on Github.com, in the form of a wiki. Internal division issues as well as inter-division issues will be tracked via the issues feature on the Github website.

Glossary and References

Activity

An Android application component which consists of a single screen that the user can interface with.

Android

A mobile operating system developed by Google. Currently the most popular mobile OS.

Android Java

A modification of the Java object oriented programming language for use with Google's Android mobile operating system.

API (Application Programming Interface)

Documentation describing how outside services can interface and interact with our component.

Broadcast Receiver

An Android application component which sends and receives intent objects.

Git Flow

A branching model for use with the Git version control system. More information can be found at http://nvie.com/posts/a-successful-git-branching-model/.

Git Version Control

A distributed version control system for keeping track of software development. More information can be found at http://git-scm.com/.

IDE (Integrated Development Environment)

A program designed to facilitate software development, which allows users to write and compile code and provides tools for debugging.

Intent Object

Intent objects are sent and received by broadcast receivers in order to launch new activities. This is how different components of an Android application are called and launched.

Service

An Android application component which performs long running procedures in the background of the application.