Requirements Specification

Vi-Char Mobile Development Division

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

Introduction

This document is the requirements specification for the mobile device development division of Vi-Char and has been composed by all team members. It details the essential tasks and goals that the team needs to accomplish in order for the system as a whole to integrate together properly and be ready on time. The strengths and qualifications of each individual member working on this part of the overall project are detailed as follows.

Michael DuBois: UI/UX Design & Front End Development for webapps. Experience writing in Java. Experience with git versioning. Capable with Adobe Photoshop/Illustrator.

Nathan Pastor: Experience writing in Java, and extensive knowledge of QA test routines. Well versed in audio processing software and sound design.

Robert Shapiro: Experience with Java and some experience with C++ and Python. Capable with Adobe Creative Suite and passingly familiar with the Android SDK. Industry outsider with a passion for learning new skills and mobile software.

Davis Shurbert: Experience writing in Java, mild familiarity with Python. Extensive background in Mathematics. Hard worker, devoted student.

Kirah Taylor: Extensive previous experience with Java programming and previous knowledge of testing methodologies.

Executive Summary

The end goal of this project is a multiplayer platformer game that takes place in an augmented reality environment. The main character's avatar is controlled by a motion capture device whilst Android-toting onlookers manipulate the game environment in real-time. Users, here defined as anyone involved in affecting the environment with their mobile phone (specifically not controlling the main character), will be able to view the augmented reality world on their mobile devices and interact with the game. The game engine will be responsible for the physics of the game as well as generating images. All components of the game system (mobile devices, game engine, augmented reality, etc) will be synchronized through a dedicated server which will also keep track of past games and maintain a leaderboard.

It will be necessary for the application to run smoothly and quickly so that it can keep up with the real time augmented reality environment and remain synchronized with the other users of the application. The GUI must be reasonably simple and immediately intuitive so that gameplay will not be inhibited by poor performance, synchronicity errors, or user confusion.

Most all requirements, functional and nonfunctional, relate back to the core need of being able to view and interact with the game environment through the mobile device quickly and easily.

Application Context

The mobile development team's main objective will be to develop an Android application that can interact with the overall Vi-Char game system. Users of the application will be able to view the game's augmented environment using their phone's camera. They will also be able to interact with the game world through our application with an easy-to-use GUI and common touch gestures. Game-relevant information generated by this interaction will be sent to a dedicated server so that updated environmental information can be propagated to all participating users. This way, multiple users will be able to simultaneously use our application to view and interact with a single instance of the game from different mobile devices.

In order to facilitate this objective, our application must be able to take video feed from the mobile device's camera and pass it to the augmented reality module for processing. The application should then utilize the returned analysis to overlay images -- provided by either the game engine or the AR module -- at predetermined coordinates specified by the AR module.

When the user touches the screen during gameplay the gesture type and location will be sent to the game engine and/or the augmented reality module, which will determine and actuate the effect on gameplay. This could include the introduction of obstacles for the motion controlled character, or the clearing or rearrangement of obstacles. Through this mechanism the users will be active contributors to the game experience.

Sounds will be generated when the user performs certain actions through use of the GUI and when certain events occur in the virtual environment, meaning the application will be able to take input from the game engine on when and what sounds need to be created and produce audio accordingly.

Users of the application will be the general public. They may have prior experience with Android devices and the Android OS, but this will not necessarily be the case for all, hence the requirement of reasonable simplicity and immediate intuitiveness in the GUI.

The application, with its view, touch and sound capabilities, is intended specifically for use with the aforementioned game and will be integrated directly with the other game modules. The application therefore is only intended to be useful within that context: When it is able to communicate with the game engine, the augmented reality module, the web server and the motion capture module.

Functional Requirements

Allow the user to download and install the app

Actors: User, Web Server

Preconditions:

1. The user possesses device

Scenario:

1. The user downloads the application from web server
2. The user installs the application

Postconditions

1. The application is installed with full functionality

Allow user to start the game and potentially create a player profile

Actors: User, Web Server, Game Engine

Preconditions:

1. The user has installed the application on their android device

Scenario:

3. The user opens the application, sees menu
4. The user selects “Enter Game” or equivalent
5. The user is prompted to create or select a player profile
    a. Here we ask for any information/permissions that will be required for potential social networking integration
    b. The user should be asked to select a name for scoreboards and player-to-player interactions

Postconditions:

1. The user’s device and profile is registered with the game server

Receive and display augmented reality-enhanced view

Actors: Augmented reality module, phone

Preconditions:

1. The camera is facing the game environment
2. The network connection is strong
3. The application is open

Scenario:

1. A feed has been sent to augmented reality previously

2. A feed is received back from the augmented reality module and displayed to the user.

Postconditions:

1. The user has the augmented reality feed on the screen

Allow touch input (i.e. tapping, dragging, and gestures)

Actors: User, phone, augmented reality module

Preconditions:

1. Game has been launched and is running on device

Scenario:

1. The user touches screen in particular location
2. The gesture type and the touch location sent to the augmented reality module

Postconditions:

1. The information on the touch has been sent to AR module
2. The game continues to run

Play audio

Actors: Users, phone speakers, game engine module

Preconditions:

1. The user has begun the game

Scenario:

1. The user activates the camera and faces it toward the game environment
2. The user establishes a feed for the game world display
3. The user makes a valid tap for input on the touchscreen or an audio-triggering event happens in the game world
4. The phone speakers play audio

Postconditions:

1. The user exits the game

Nonfunctional Requirements

Compatible with Android 2.0, and more recent. More than 99% of Android users do not use the most recent distribution.[1]

Intuitive and quick-to-use GUI. Ease of use for the user is indispensable, and having a simpler cleaner interface serves the secondary purpose of keeping the application small and clean.

Must display AR visuals in near real time. This is essential for the game to be able to be played without delays.

Debugging capability. This will allow more productive testing, and will allow us to track down problems more quickly. Debugging will be particularly important once the different modules are joined.

Use Git versioning. A versioning system is necessary in any project of this scale, and Git is the one that has been selected for use on this project.

Installable from the web. The application needs to be able to be quickly installable even by a complete newcomer to the Android operating system.

Consistent and ample documentation. This is necessary for any project despite being frequently neglected. Documentation is important for integration with other modules and for future changes and use.

Extendibility. It will likely be necessary to add or remove features from the project.

Clean, efficient code. Something that is good for any program, but important here due to significant space and speed requirements, and the need for other teams to efficiently familiarize themselves with the code.

Seamless integrations with other Vi-Char modules. This will allow the application to integrate with other modules properly and easily so that the game is playable, reliable, and enjoyable.

Potential Challenges

Phone hardware limitations, such as battery drain, slow hardware, slow network connection etc. This will be dealt with the best way it can be, by keeping the application as small and fast as possible.

Compatibility errors. This potential problem will be dealt with the only way it can be: through forethought and testing. It will be attempted to be sure it doesn't happen and if it does, a workaround will be found.

Issues with WiFi network. There are limited remedies to work around or fix WiFi issues from the mobile application's end. Consequently, it is imperative to make sure that WiFi capability within the application is designed as efficiently as possible. In addition, the project architecture will keep WiFi use to a minimum to reduce the number of instances of WiFi issues.

Coordination between teams. This involves deciding where components of the game are supposed to be located, such as the game engine, and extends to being able work together to fully integrate all of the modules. If teams do not coordinate properly, the whole project may fall apart and the mobile device application will be useless. Sure avoidance of this situation requires ample communication and coordination.
Deadline-related constraints. It may occur that there simply isn't enough time to get the application finished properly, partially, or at all. This will be combated by working as efficiently and quickly as possible to get it finished, within reason.
Tracking down the causes of bugs. There are many independently developed components to the game, meaning the root causes of issues may be difficult to trace. There will need to be extensive and regular testing to be sure this doesn't become too large of a problem, alongside effective debugging routines.

Timeline

September 14th: Finish requirements specification document
September 17th: Create an application to build off of (i.e. the "Hello world" step) including non-Market installer.
September 24th: Create basic main screen/main activity UI
September 28th: Finish design document
October 8th: Handle touchscreen inputs
October 15th: Send video feed to AR team's module or to server
October 25th: Finish intermediate report and finish intermediate presentation
November 5th: Display images at coordinates given by AR team
November 16th: Finish test document
November: Full integration with game engine, web server and AR module. The application will be complete and ready for use with the rest of the game.
December 10th: Finish final report and present final demonstration

References

1. Agarwal, Amit. (8/01/2012). "The Latest Distribution of Android Versions," *Digital Inspiration*. http://www.labnol.org/gadgets/android-versions/24555/

Glossary

1. Touchscreen: The electrical visual display of the phone that can detect the presence and location of a touch within the display area.

2. Camera: An object within the phone that can record images that can be stored directly or transmitted to another location and display the images on the touchscreen.

3. User: People involved in affecting the gameplay environment other than through motion capture.

4. Player: A person using motion capture to control the main character of the platformer

5. GUI: Graphical User Interface. It allows users to interact with the application through the use of visible elements including images, buttons and text.