Mobile Multiplayer Battleship-like Game

Project Proposal

San Jose State University
Computer Science Dept.
CS 161, Sec 01 - Software Project
Dr. Moazeni, Ramin

Team | DEV-T

Trinh Nguyen
Victor Fateh Firouz
David Navarro
Emmanuel Mendoza

Spring 2017

Description of Product:

A turn-based Battleship-like game where friends, family, coworkers, and strangers can connect. Players will connect via Bluetooth or Wifi or via a centralized server (we are still deciding which connection method to use). Players will be able to setup their own game board with a fleet of battleship-like game pieces. Players will then be immersed in a strategic battle between friends and foe in order to achieve victory. Victory is declared only once your opponent has been completely defeated.

We will initially start with a 1 vs 1 player game. However, our goal is to be able to support at least 4 players to create a more interactive game that supporting 2 vs 2 games or free-for-all games.

Note that Battleship may not be our final game. We are looking to make a product that distinguishes itself from other similar products. If we are unable to differentiate our game from others because there are already so many Battleship games available we may decide to change our game to something more original.

Overall, our main goal is to develop an interactive game that encourages customers to choose our game and continues to play it.

Need for Product:

Our everyday lives are filled with situations and moments that cause boredom, loneliness, depression, and anxiety; this happens whether you are alone or with friends, family, coworkers, and strangers. Developing a mobile game application that allows friends, family, coworkers, and strangers to connect together and interact socially while they play together can help pass time while alleviating boredom, loneliness, depression, and anxiety.

While several mobile game applications exist. There is a lack of game applications that allow people to connect and play together, while they are together. Our mobile game application aims to solve this problem and provide a therapeutic environment to help connect people; whether it’s during social gatherings, while waiting for your food to arrive at restaurants, during a long road trip, or on a trip abroad with no cellphone signal, our turn-based Battleship-like game will be there to help entertain you, your friends, family, coworkers, and strangers.

Potential Audience:

The potential audience would be friends, family, coworkers, and strangers looking to connect to help pass the time.

Discussion of Competing Products:

Other competing products includes Words with Friends, Hanging with Friends, and or other turn-based push notification games. It has similar ideas but with its core game being battleship.

High-Level Technical Design:

Game Design:

Since we chose to go with a turn-based battleship-like game. Our graphics don't have to be drawn continuously in real-time, so this may reduce some complexity since we are building a multiplayer game. For the game design, we would need to look into gaming APIs to render the 2D or 3D look of the game. We can either use Android standard APIs like Canvas API or OpenGL ES; or we can choose to use a third party framework/game-engine.

Some Android 2D Game Engines that we could explore are:

Cocos2D-x
Language: Lua
Orientation: 2D
Difficulty: Intermediate

Corona SDK
Language: Lua
Orientation: 2D/3D
Difficulty: Intermediate

EDGELIB
Language: C++
Orientation: 2D
Difficulty: Intermediate

GameMaker
Language: None (Graphical)
Orientation: 2D
Difficulty: Easy

GameSalad
Language: None (Graphical)

Orientation: 2D
Difficulty: Easy

HaxeFlixel
Language: Haxe
Orientation: 2D
Difficulty: Intermediate

libGDX
Language: Java
Orientation: 2D/3D
Difficulty: Intermediate

Marmalade
Language: C++
Orientation: 2D/3D
Difficulty: Intermediate

Stencyl
Language: ActionScript (optional), Objective-C (optional)
Orientation: 2D
Difficulty: Intermediate

Unity3D
Language: JavaScript (UnityScript actually), C# (Mono)
Orientation: 2D/3D
Difficulty: Intermediate

If we choose to connect players via bluetooth:

Strengths: No upkeep cost to run a centralized server. The close proximity restraint of bluetooth will create a more interactive/social experience amongst players. Players will be able to play in most environments where cell-signal is not necessary; such as an airplane, camping, or when traveling abroad. Bluetooth also uses less battery.

Weaknesses: Players will have to be in (very) close proximity to not become disconnected. Players will also be relatively forced to only play with others who simultaneously want to play and connect; this will make it more difficult to have interactivity with strangers since you have to be close proximity with known individuals who also connect.

Bluetooth High-Level Technical Design:

One player would need to act as a centralized server while other players connect as clients. The bluetooth connection would create a service listener that would allow us to create a communication API to relay player information between players.

If we choose to connect players via WiFi Direct or Mobile Hotspot:

Strengths: The close proximity restraint of Wi-Fi will create a more interactive/social experience amongst players. Players will be able to play in most environments where cell-signal is not necessary; such as an airplane, camping, or when traveling abroad. Connection via WiFi allows players to play interactively within a larger area when compared to Bluetooth.

Weaknesses: Players will have to be in proximity to not become disconnected. At least one player would need WiFi hotspot capabilities to host the game or all players would need support for WiFi direct capabilities on their mobile device.

WiFi High-Level Technical Design:

One player would need to act as a centralized server while other players connect as clients. The WiFi connection would create an action listener that would allow us to create a communication API to relay player information between players.

If we choose to connect players via a centralized server:

Strengths: Players will be able to play with other players around the world. Players can have more time to make their move.

Weaknesses: A recurring upkeep cost would have to be factored in to maintain a centralized server. Players would need access to data via WiFi or their cellphone provider. The game may take longer to play since players may take longer to respond to game moves. Players may be more prone to loose interest in the game.

Centralized Server High-Level Technical Design:

The game will need to be able to send and receive game states between friends. This could be either done by some kind of remote database or some kind of json trick that sends a string array of the game board state and player turns. If we use database, we might need to look into some kind of free/paid server that does this. One option that we may explore in this realm is using Firebase.

Resource Requirements:

We would need to research into game development.
We may need a database server that supports push notifications.

We may need to use a game API engine.
We will need to find free/attributed sprites/images to use for gameplay.

Potential Approaches:

To Connect Players:

Our first goal can be to connect two players. We can then try to create a simple mobile chat app that allows the players to send messages back and forth via text input. Once we succeed in this, then we can begin to design our gameplay API through that communication medium.

For Gameplay:

We start by designing the barebone game engine to work offline (single player). We then convert it to a multiplayer game. We can try coding it using no graphics at first, then add a graphics UI to the game itself.

Assessment of Risks:

There is a lot of research and learning to be done to implement this game. We also need to properly synchronize the game between players. We have one member who has taken a game development course and one member who is currently taking a game development course. We also have one member who has experience in database design/management. This knowledge base will be useful in overcoming obstacles faster. There will be a learning curve in that we have not used any of the mobile game engines before and we have not used firebase before.

The other complexity is supporting more than 2 players. Once we add 3+ players; we will have to account for team assignment / team balancing, team player disconnection issues, team player synchronization, and more complex game logic on which areas a player can see and which areas a player can attack.

Next Steps:

Build the barebone game.
Research and implement passing game state to other players.
Research and implement graphical solutions for the game.
With our current target goals being to implement our **Potential Approaches** (referenced above).