**Team A**

**High-level specification**

**Introduction**

Drone wars is an augmented reality first person shooter which uses a Parrot (Augmented Reality) AR Drone 2.0 Multi-Rotor (http://ardrone2.parrot.com) controlled by an Android Device to essentially dogfight other drones. It is going to be a free-to-download application which will be available on Google Play Store. The idea of Drone Wars App (DWA) is to entertain and provide a helicopter pilot like experience where the user can battle with other players. DWA will provide immersive Player vs. Player (PvP) gameplay with live video feed and a (Heads-Up-Display) HUD which will provide the user with all game and drone information. The main features of DWA include:

* Player vs. Player - PvP gameplay functionality: DWA will pair user vs user in real time to provide immersive and challenging gameplay.
* Limited and replenishing ammunition: DWA adds a level of strategy and challenge to allow the user to pick their shots.
* Limited Life: DWA allows the user some flexibility for mistakes and keeps the game going longer.
* Real-time Video Feed enhanced with Augmented Reality: DWA provides the user with an immersive gameplay experience.
* Limited Area: DWA limits the users to the same physical area for play requiring them to physically engage each other.
* Real-time tactile and auditory feedback: DWA app on the mobile device will cause the device to vibrate and make noise at appropriate times to include other senses in the AR immersive gameplay.

Drone Wars will be played on an Android Device running Android KitKat 4.4 as well we Android L. DWA’s interface will feature the AR Drone live video feed enhanced with an AR layer. The AR layer will contain the HUD display and virtual joysticks which will give the user precise control over the AR Drone. The HUD will include a dedicated target area, graphical readouts of virtual missiles remaining, a red outline around any enemy drones in sight, as well as remaining lives all overlaid on the video feed. The top action bar of the screen will display information such as Drone battery life, Wifi signal strength, the match clock, and phone battery. When an enemy comes into view, they will be highlighted with the aforementioned red outline.

There is also plans to add a global ranking system that will allow users to track their performance and compare it to other users. Every user would have to register and create an account on the game ranking server. Stats of the ranking system would include games played, wins, losses, hits, hits taken, a hit to hits taken ratio, missiles fired, missiles hit, and fired to hits ratio. All these stats would be calculated and updated at the end of every match. For this ranking server there would be three levels of user:

* **Guests** are users who choose just to play DWA without any game statistics. They will just play the game on a game to game basis. They will have no access to the ranking server.
* **Players** are the users who have created a login and are active in DWA. They are the ones who fly the drones and have read/write privileges to their username and password but read only privileges to game statistics.
* **Administrators** have control over all data in the server. Their duty is to moderate the server and make sure that all usernames are appropriate as well as editing any game data that may need it.

**Operating Environment**

1. **Overview**

Drone Wars will operate using a client-server architecture. The server will be implemented on a single computer running Ubuntu’s Server version. The server’s main task will be to provide communication between the two types of clients. Figure 1 depicts the complete, high-level, interaction between the server and the clients:

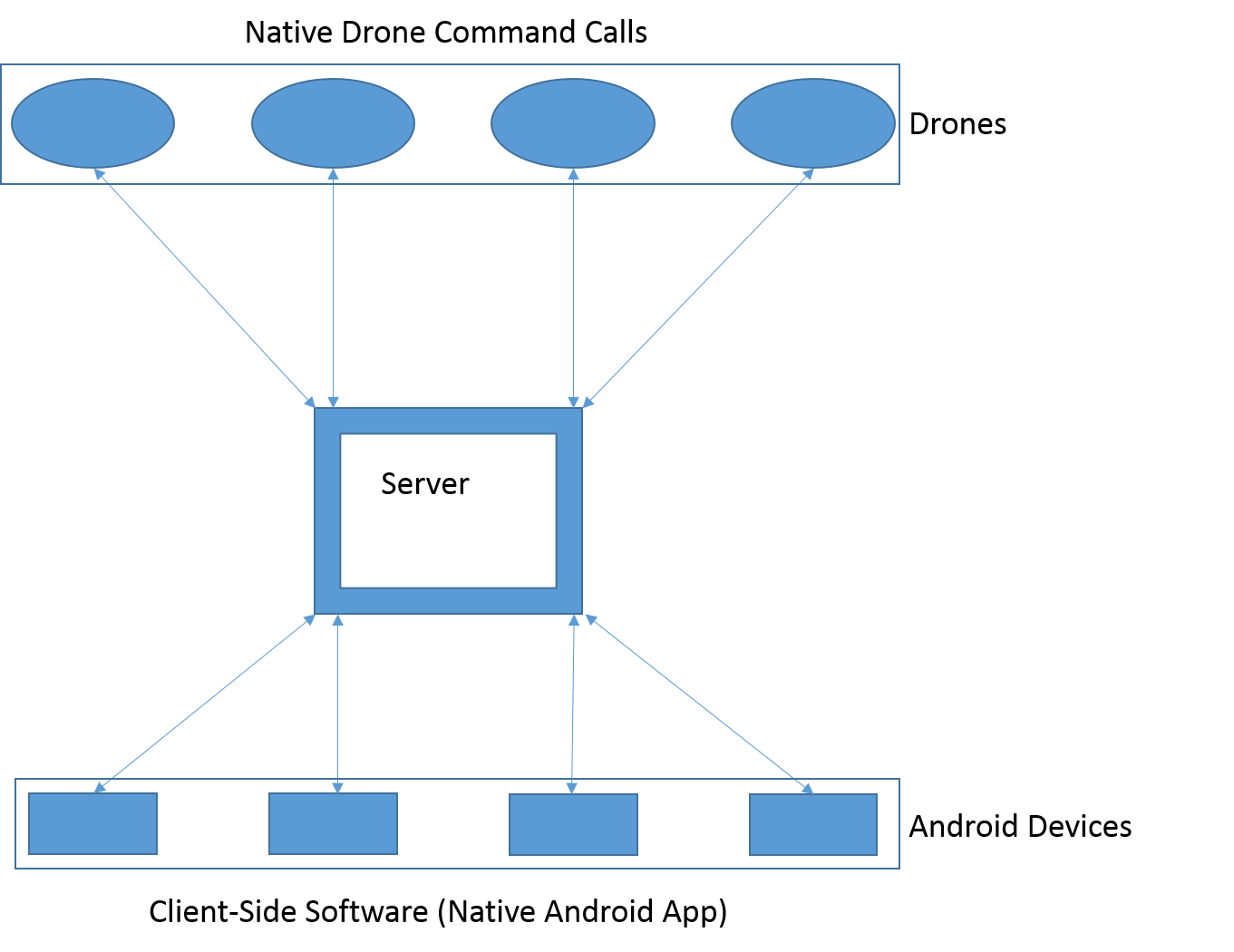


Figure 1. High-level client-server architecture

As shown in Figure 1, there will be two types of clients in this architecture. One client will be the physical flying drones, and the other will be the user’s personal mobile device. The drones will have a static IP address used to connect to the server. These static IP addresses will allow the server to distinguish the different drones so that the correct user commands can be sent to the appropriate drone.

In addition to relaying commands to the drones, the server will contain all game logic used in the DWA game. The server will be in charge of determining related game elements such as how much time is left, which drone still has life left and how much, how many bullets each user has and when they will have more. This function will be an interaction between the user’s client and the server.

The second type of client is the user’s Android mobile device. In order to interface with the server an Android application will be written. This application will be used for two functions; the first is a GUI and the second is to submit request. This application will display the video stream passed through the server from the drones. Additionally, it will have the HUD overlay placed over the video stream to provide additional context to the user. The second function is to actually control the drone. DWA will utilise a REST API to send JSON objects to the application server which will then convert the data into native ARDrone API calls and transmit them to the appropriate drone.

DWA is meant to be a versatile game which can be played either indoors or outdoors. In both cases, the server can be a simple laptop with the Ubuntu server installed. The only additional hardware needed in order to play DWA is a wireless router. The router will provide the Wireless LAN network for all Android devices and drones to connect to. The router will act as a connector between the android devices and physical drones.

1. **Server**

The server will utilize Node.js and node-ar-drone library to allow connecting multiple drones to the same network and provide the ability to map drones to the respective Android devices which will be used to control them. The server will begin by connecting to the AR Drone ad hoc network and execute a script which will force the AR Drone to connect to a specified router and assign it a static IP. The IP address will then be added to a list of all participating drones and when the user connects the Android device the same router and launches the Drone Wars application, he will have the ability to map himself to one of the AD Drones connected to the network. Once the AR Drone and Android device are mapped all the control calls from the Android device will be routed to the appropriate AR Drone using the node-ar-drone library.

The server will also utilize OpenCV to analyze the live video streams from all the drones and detect colored stickers which uniquely identify the participating drones. Once a drone is detected on the live video stream the user will be able to fire a virtual missile which will cause the targeted drone to lose a life. The node-ar-drone library will also allow the server to capture all navdata, animate movement or completely take control of any of the participating drones, so that if drone loses all its life server will force it to land on the ground. This library also provides a low level api UPD Control which will be used to communicate with the drones.

1. **Information**

The purpose of our project is to allow multiple users to control their drone to fight each other with augmented reality under a specified game concept. The generic Apps available on Apple App Store and Google Play market for drone controlling are mainly for single user who fly their drone individually. Our project will add complementary features to the drone and further multiply the enjoyment of owning a drone by allowing for realistic play and battling among multiple drones.

Player information and drone war statistics information

All the information will be stored on the server side. The players’ information will include the drone battery power, drone Wi-Fi strength, remaining amount of virtual ammunition, remaining amount of virtual life points. When any of the information goes into exception, the player side function will be disabled (for example, if there is no ammunition left, the player will not be able the fire). And the statistics of each player’s game won/lost will also be stored on the server. At the end of each game, the player will receive a pop-up window in the app, which asking if he/she want to upload the statistics to Game Center for iOS or Google Play for Android. If yes, the App will retrieve the latest game statistics of the player from the local server and then upload it to Game Center or Google Play. We will use MySQL as our database to store all the information.

The server will also keep transferring the video stream from drone’s camera to player’s device.

Information exchanged among the server, drone and Android mobile device

The information exchange mechanism are partly described in operating environment section. There will be only two types of information exchange in the game, one is between server and drones, the other is between server and device and all the information exchange will be on Wi-Fi. The server will collect all the move instructions from devices and transfer them to the drones. And the server will tell the device how many life points and ammunition left and theses information will be displayed graphically on the player’s device. The Android mobile device will not directly contact with the drone.

Client (Android mobile device) side information

When first time starting the App, the player can choose to login his/her Google Play or Game Center account such that he/she can upload the game statistic at the end of each game. The app will store the account information locally. When the game begins, the app will allow the user see the video captured by the drone’s camera at a real time.

1. **Performance**

The largest part of our project is the use of physical drones, to play an augmented reality combat game. The physical drones used in DWA, each have a camera that displays what is in front of the drone’s current position to the users via the Android UI. The video feed needs to be as close to real time as possible, so that latency issues do not hinder the user’s game playing experience. Ensuring the delay in the live video stream is minimized is vital to the success of the game and the integrity of the drone. The “combat” aspect of the game requires that each drone camera can identify other enemy drones color. Each player will be able to see the enemy drone in their Android device via the UI and hit the fire button. If there is any delay in the video feed, a user may think that they have registered a hit after an opponent has already cleared their sights.

Since Drone Wars is a multiplayer game, a server will be designed to first handshake with each drone. Once the handshake is completed each drone will be able to communicate  
with the server using its own static IP address. Each phone will also go through the “handshake” process, this will allow the server to take user input and push the instructions to the correct drone. The drones battery life, health, ammunition count, Wi-Fi strength and video feed have to all be sent from the drone to the server and pushed down to the correct players UI. The video stream from each drone is broadcast in 720p resolution at 30fps, the latency between the drone and server and the server and Android device has to be low.

To ensure that server’s communication with all devices has the lowest latency possible we are anticipating that TCP will be too slow, because of the real time video feed being so critical for game play. The video feed from each drone is so critical because there is no drone to drone communication, only visual via the two cameras mounted in the physical drone. The cameras will relay when an enemy drone appears in each player’s sights, and when the fire button is pressed the DWA will send the hit data to the server. The server will receive the fire request and remove one life from the corresponding drone that was “hit”. Instead of TCP we will be using the UDP protocol. Although UDP is less reliable with packet delivery than TCP, it does not wait for packet delivery confirmation from the server, it will just keep sending the data packets repeatedly. Since the delivery of data packets is much faster, any lost data packets will have minimal effect on the users gameplay. The physical drones and Android devices will be connected to the server via a dual band wireless router. If time permits we will also implement another server that will host a global ranking system for players. The global leader boards will be on another screen in the android app that allows the user to compare themselves to others and fight for the top spot. Since ranking is not as data intensive standard TCP connections should be sufficient enough to allow connection between the users android UI and the global ranking server.

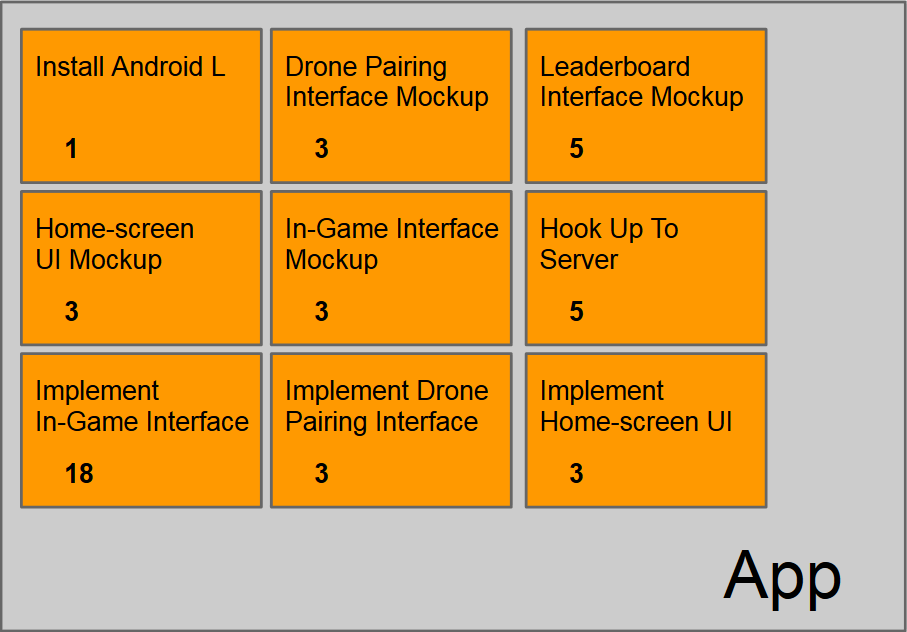
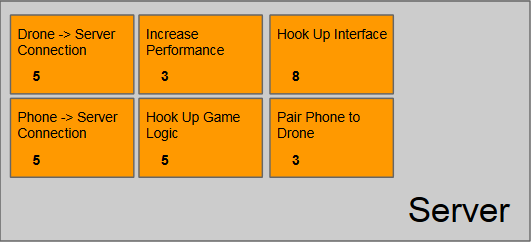
1. **Security**

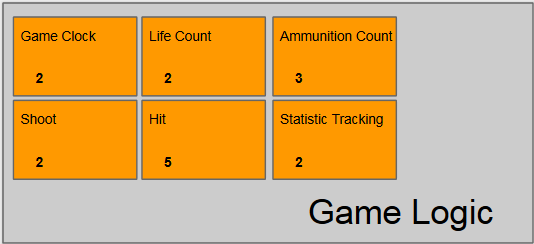
As a game in which each player controls a physical drone, the main security concern of DWA is to protect the control of the drones. Users don’t want their drones over-controlled by other people when they playing the game. The connection between an Android device and the drone is through a server which will provide an IP address. What we can do is to use a piece algorithm to encode the IP address and put algorithm in the Android application and server. By doing this, only person that can control the drone is one that uses the Android application and connects to a particular drone. Since we are using a server as an intermediate station, we don’t need to worry about any internet security, namely, the Android devices, the server and the drones will form a small isolated network.

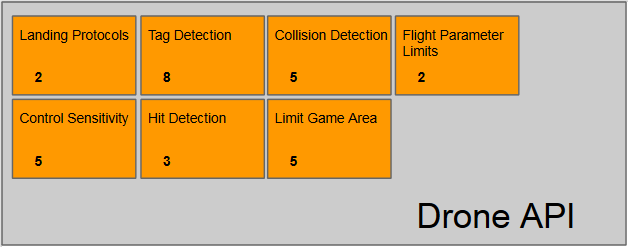
If time allows, we might implement a global ranking system. After each game, the server will send the statistics into the application. Then the application will be asked to upload the statistics when it can (has internet connection). Before the application upload the statistics, it will encrypt the data, so that the statistics will be sent securely to the Game Center for IOS devices and Play Games for Android devices. After this process, the data will be protected by the Game Center. Since when the device connects to the server, it cannot connect to the open internet, so the upload process has to be happen before or after playing the Drone Wars. What we want to do is to make it upload the statistics after playing.

Therefore, at that time, every time a user is done playing the game, the application will try to connect to some online server like Game Center in iOS devices. Only then we need to worry about that someone might try to take control of the drone through the internet.

**Product Backlog:**







**Glossary**:

* DWA - Drone Wars App
* AR - Augmented Reality - an overlay of graphics/information on top of the multi-rotors live video feed to provide the user enhanced gameplay experience
* Drone - an actual physical, multi-rotor unmanned aerial vehicle which can be controlled remotely.
* UDP - User Datagram Protocol, the internet protocol we will be using to transfer the data between server, drones and the Android device.
* Server - A computer which will be the central hub of communications for the clients in addition to processing all game logic.
* Handshake - An exchange between two components (server and client in our case) which creates a paired (bonded) relationship so that they can communicate.
* GUI - Graphical User Interface - The view layer of the application. It will contain the HUD and video stream from the drone.
* HUD - Heads-Up-Display - The overlaid graphics on the screen which provide readouts of information such as Game Statistics, Augmented Reality graphics, and virtual controller joysticks.
* Targeting Area - The area designated at the center of the video feed (on screen) which is highlighted with a green perimeter.
* Virtual Joystick Controls - The two joysticks on the User Interface which allow the user to control the drones throttle, yaw, pitch and roll.
* Fire Button - Allows user to fire virtual missiles.
* Action bar - will display game time, battery life, signal strength, remaining game time, drone life, ammunition count.
* PvP - Player vs. Player is a type of multiplayer conflict within a game between two or more players.