**CS 490 – Homework Assignment – Due 10/6/14**

Based on the project that your team has selected in 490L, collaborate with your team and prepare a brief plan (between 2 and 3 pages in Word or PDF) containing at least the following information:

**Vision Statement**

The goal of our project is to create a two player android mobile application game where the two players use strategy and the rules of the game to destroy their opponent.

**Team Members:**

Matt Hamersky

Matt Newbill

Max Sadovskiy

**Project Technical/Functional Description – provide enough concise detail that I will understand the top level functionality and requirements of the product**

The application will launch into a main menu where the user can select settings, exit, or start game. On start game click, the user will be linked over the internet to a competing player where they specify the game options. Then they will start the game which will allow each player to continue taking turns until one player is defeated. A player is defeated once all of his units run out of health points.

**Development methodology for requirements, design, implementation, and validation**

We are using the scrum methodology with 2 week long sprints. The main requirements for the application are creating a mobile platform application for android. The final result should be a turn-based strategy game with specific terrain and unique units. Units should be able to perform different actions such as movement, attack, and possibly some sort of special actions. Should create a framework to connect players over the internet in which they will communicate throughout the game. Create a backend apache server to host connection between the two players.

We started by creating a UML diagram using Microsoft Visio. The first UML diagram we created shows the objects and their relations (attached below). We had one person create it and the rest of us reviewed it in a code review session in lab. The framework for our game engine was written from scratch, making use of Android’s SurfaceView to make direct draw calls to the operating system. The engine handles touch events, collision, lighting, and updating and rendering states. The engine is designed as a finite state machine, propagating all calls down through the state hierarchy to the underlying game objects.

Our main approach for the project is to use the android SDK and eclipse’s android plugin to help auto generate and override methods within the SDK. This allows us to add the logic and functionality our application needs.

**Development tools that will be used (or are being evaluated for use) and how they fit into the methodology**

*Project development process description--do not describe generic Scrum, but specifically how your team will execute this project e.g. team organization/roles, what/when planning will be done, description of the product backlog, releases, sprints, and retrospectives (reviews), artifacts/documentation prepared, etc.*

The team has split its skills by specialty, Matt Hamersky is working on the graphical user interface, Matt Newbill is working on the system architecture, and Max Sadovskiy will be working on developing complex algorithms. Our sprint lengths are two weeks and in each sprint we assign each member tasks based on their specialty. We are using Eclipse as our main integrated development environment and Github as our source control. We will be utilizing the android SDK and using Paint.NET as our image editor.

**Description of how your Project Asset Library (PAL – see syllabus) is organized and accessed**

**PAL Location:** <https://github.com/MattNewbill/android_combat_game/tree/master/Project%20Specs>

The Project Asset Library can be accessed using the link above. It contains our sprint schedules, product backlog and additional informative documents we use for our project.

**Description of project risks – technical, schedule, resources etc.**

The does not see many technical risks in the future. The only risk we foresee is not finishing our application according to schedule. The core application functionalities should take four sprints and after finishing our first sprint we are confident we can stick to this schedule.