**Plan of Attack Watopoly**

**Class Descriptions**

**Controller**

Controller

For MVC architecture, this is the controller. As such it is the class that contains the main function and handles the input from the user. The input is then relayed to the Board class the be interpreted.

**View**

BoardDisplay

The BoardDisplay is the class that outputs the state of the Board to the user. To begin this class will output text to the console, and later may be implemented to use graphics.

**Model**

Board

This is the central class of the model, containing the list of players currently in the game, as well as each of the individual cells that compose the Watopoly board.

Cell

Each possible space on the board is represented by a Cell, with its different subclasses have different actions to be executed by the polymorphic action() method.

Property

Subclass of Cell that represents Cell that can be bought by a Player

ImprovableProperty

Subclass of Property representing properties that can be improved with houses and hotels

Gym, Residences, SLC, Needles, TimsLine, GoToTims, GooseNesting, CoopFee, Tuition

All are subclasses of Cell that have the actions as defined in the Watopoly specifications

**Schedule of Progress**

**Completed by August 5th**

Work collectively on the UML diagramming, that must be completed by deadline 1.

Have set up a git server that is accessed by all members of the group where all the code will stored. We will make sure to always pull from the server before starting coding, and save frequently.

**August 7th**

We have divvied the classes of the model between the group members into three groups based on which classes can be implemented rather independently. Alex will be responsible for implementing the methods of the Player class, James responsible for Cell, and its subclasses, and Jay the Board and BoardDisplay classes.

Focus will be on creating well documented and followable code that is coherent for all the group members to understand regardless of authorship.

This step is to get each class to a basic state, the specifics will the worked on later by all members.

**August 8th**

Upon completion of the last step each member will thoroughly review each others code for possible logic errors. This will also allow each member to become acquainted with the code base.

Test the code as a single unit and verify its functionality.

**August 9th**

Start the implementations of the Controller class, first focusing on player movements, making sure that the position on the display is updating accordingly, based on their dice roll.

Another group member will focus on the starting, saving and loading games such that the game will save the player’s position.

**August 10th**

Work on perfecting the act of Players buying properties and paying rent to other players.

Update the saving methodology to include saving player’s money and properties owned.

**August 11th**

Begin working on implementing the rules for bankruptcy

**August 12th**

Work on implementing the SLC and Needles Hall Card

As well as buying improvements for owned properties, as well as making sure that the given players owns a full group, and is allowed to perform such improvements.

**August 13th**

Complete the saving and loading method to encompass all aspects that need to be saved. The save should work such that there is no difference in the game between when the game is saved and loaded.

**August 14th**

Work on trading properties and cash between players.

The following steps are guidelines for the latest dates that we are aiming to have each respective task completed by. The following tasks are only to be completed if we have extra time.

Work on the adding of House Rules to the game, this is most likely done through the Decorator design pattern

The implementation of graphics instead of the textual display, since we are adhering to MVC principles, this would only require re-implementing the BoardDisplay class.

The other members could work on the addition of Computer Players to play as opponents.