**Project Overview:**

The project structure will rely primarily on four classes: MazeCell, Maze, Robot, and Hub. Each MazeCell will contain its grid row and column, as well as an enumerated type Marker which will be used to identify caverns, as well as cells already visited by travelling robots. The Maze will read in the structure of the maze from a text file and create a collection of MazeCells to represent it. It will also contain the queue of robots that will need to explore the Maze. Finally, the Maze class will contain the PaintComponent function for visual rendering of the project. Robots will have member variables storing their location and destination on the maze. The destination will be updated by a receiveCommand function which will set the destination to a user-chosen cavern. Additionally, Robots will have functions allowing them to explore the maze by identifying valid target MazeCells. While a robot is exploring, it will leave behind markers so that it knows where it has already been. It will also update a map member variable, allowing the robot to remember the parts of the maze it has already accessed. Lastly, the robot will have a function allowing it to communicate with the robot behind it and update its own map accordingly. The Hub class creates the GUI and handles the overall functionality of the program. It will allow the user to select a target cavern, and then it will issue the next robot the command of finding it. The Hub will also have the main function, which will create the Hub, and Maze.

**Development Strategy:**

Development of this project will exemplify Agile Programming tenets such as frequent deliveries and constant refactoring. Project advancement will rely heavily on incremental test driven development. Furthermore, the project will be split into subprojects, each of which are assigned a number of story points. In this case, finalizing the MazeCell and Maze classes are allocated five and ten story points respectively. The Robot class, which requires the most functionality, is allocated twenty story points. Finally, the Hub earns fifteen story points because of its all-important user interfacing for a project total of fifty story points. Each class will be further broken down into even smaller subprojects which will become clearer after implementation has begun. Given that the project is due to be completed in about twenty days, a group velocity of three story points per day (or an individual velocity of five story points per week) is preferable to leave extra time for final modifications.

Dividing the project into smaller sections is useful in determining the major functionality that will be completed by each of the due dates.

Deliverables:

The first deliverable is due on Tuesday, November 18. Since this allows one week for development, approximately 20 story points should be completed by that day. The MazeCell and Maze classes are important to finish early since the rest of the program depends on them. Since they are worth fifteen points, five more story points should be earned prior to the first due date. The next logical thing to implement is exploration for the Robot class. A Robot should be able to identify target locations and explore the maze using an algorithmic approach similar to a modified depth-first traversal. Completing the implementation of this functionality will earn the group the extra five story points needed for the first deliverable.

The second deliverable allows another week for development. By that time, the Robot class should be finalized. Robots should be able to adjust their exploration patterns by referencing their maps. They will also be able to communicate with the robots following them in the queue and download their map information. The remaining five story points in the second due date come from simple interfacing. In this case, it would be most useful to visualize the robot behavior, so the maze and maze components should be painted.

The final program is due another week after the second deliverable. Finalizing the Hub earns the final ten story points. Hopefully, extra time will allow debugging and refactoring, as well as aesthetic improvements to the GUI.

**Testing Strategy:**

It is useful to discuss the testing strategy with respect to what functionality will be implemented for each deliverable.

For the first deliverable, it is necessary to test the loading of the Maze from a text file. This will be done by selecting a few choice MazeCells and ensuring that they have the values for their row and column member variables. The number of rows and columns in the Maze will also be checked. Additionally, cells that should be the cavern type will be tested to see if they contain the correct marker. A quick check will also ensure that the four Robots are created successfully. Robot position and destination can be tested to make sure that the Robot knows which cell it is currently occupying, and which cavern marker it needs to find. The only Robot functionality that needs tested for the first deliverable is selection of valid targets for exploration. A target is valid if it is adjacent to the robots current cell, and if the two cells are not separated by a wall.

The second deliverable requires more sophisticated tests. It will be necessary to test that a Robot will go directly to a cavern if a valid route exists in its map. This can be done by testing the number of movements the Robot makes before it finds the correct cell. Testing whether robots add cells to their maps correctly can be tested by artificially directing a Robot’s motion pattern and then making sure that the map contains the cells the Robot occupied. Similarly, communication between Robots can be checked by making sure that one robot’s map contains cells only known by the Robot behind it after communication. The rest of the project relies on visualization, so JUnit testing will become less important than simple visual verification that the GUI is working correctly.

**Conclusion:**

The Maze Runner project requires that 50 story points be completed in a three week period. The project will be divided into subunits, each of which is important to complete by a certain stage in project development. Test driven development and other Agile techniques will be employed in order to ensure a quality project.