**Save the Scrum Master**

**COMP-231**

**Prepared by**

Aaron Gavendo

Kevin Polson

Guillermo Mansilla

Eduardo Carruyo

Sankar Deb Roy

**Contents at a glance**

Introduction

Game Overview

Description

Audience

Software and Hardware Requirements

Development Tools

Diagrams

Redundancy Elimination

Testing

Conclusion

**Introduction**

Our group chose to create a game for our project as three of our five members were in the gaming program. One of the non gaming group members became the scrum master and oversaw the process and the other learned to adapt to the technology.

We used XNA in C# to create the game as well as a tile map editor that was created in C# by one of our team members. All sprites and music were browed from other games and have all been properly referenced.

**Game Overview**

The plot of the game consists of a hero (***Lady Java****)* trying to save her scrum master and another three members that were kidnapped by some miscreant group. They are hidden in separate dangerous dungeons in a university campus. Lady Java has to rescue her three team members and scum master so they can hold a daily scrum. To rescue them she has to collect information from people by exploring the university campus. During the rescue she has to fight with her enemies as well as their boss. The Scrum Master will be rescued at last after the other three team members.

**Description**

***Plot***

The plot consists of a hero Java trying to rescue her three team members and her Scrum Master so they can hold a daily scrum. She will need to explore the university campus and talk to people to gain clues. She then must fight through the dungeons in order to save her team members. In the end the scrum master must be found to reveal an important lesson.

***Overworld***

The game starts with the player controlling the main character, who is called *Lady Java*. The over world is a university campus with several buildings. The player can enter the buildings to appear in rooms that contain things and people. The campus has people throughout who can be talked to.

***NPC***

Non Player Characters are walking through the campus and inside buildings. Some of them offer useful information other offer no help.

***Levels***

There are four dungeons in the game, one for each team member and one for the *Scrum Master*. The *Scrum Master* dungeon must be completed last. In the dungeons there are enemies that must be fought with a sword in order to pass to the boss area. The boss battles are hard and require many hits to kill. Once the boss is dead, the player has rescued a team member. Once saved, the team member will then be at a meeting location.

***Enemy***

Basic enemies die by a sword hit. Some enemies walk in a back and forth pattern and others move towards the player aggressively once the player is visible to them.

***Bosses***

The bosses are like the aggressive enemies but require many hits to kill.

***Music***

Different music is being played on the over world, in buildings and in dungeons.

**Audience**

The point of this game is to create a fun way to learn about the scrum process for younger people. The NPC’s that do not offer story information or clues will give out scrum facts. When a team member is rescued a key process to scum is revealed.

**Software and Hardware Requirements**

**Operating System** Windows XP with SP4, Windows 7/8

**Processor Speed** Pentium 4, 3.2 GHz or Power PC G5, 2.0 GHz or higher

**Minimum Memory** 512 MB

**Minimum Disk Space** 20 MB

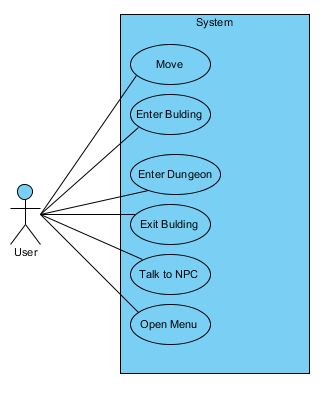
**Graphics Card** ATI Radeon 9800 w/ 256 MB video memory (Minimum)

**I/O Ports** VGA, DVI ports

**Development Tools**

|  |  |  |
| --- | --- | --- |
| **Purpose** | **Development Tools** | **Comments** |
| Design | MS Visio2010, MS Word 2010 |  |
| Application | MS Visual Studio 2010, XNA 4.0 |  |
| Presentation | MS Power Point 2010 |  |
| Art | MS Paint, Photoshop, Tile Map Editor | Tile Map Editor is an application developed by our team. |

# Overworld Navigation



Use Case Description:

|  |  |
| --- | --- |
| **Use Case Name** | Move |
| **Use Case ID** | UC-001 |
| **Super Use Case** | Overworld Navigation |
| **Actor(s)** | User |
| **Description** | The user controls the main character for it to navigate freely through the map. |
| **Post-Conditions** | The user must be able to see in real time the character moving in the screen accordingly to player’s input. |
| **Flow of Events** | 1. The game is running in “Overworld”. 2. The User presses the corresponding key to the movement he/she desires to perform. 3. The screen is updated and the position of the character changes according to the keys pressed. |
| **Exceptions** | The character gets blocked between collision of obstacles and the map edges. |
| **Priority** | High |
| **Assumptions** | The user is using a Keyboard to play the game. |
| **Input** | “Up”, “Down”, “Left”, “Right” Keys in the Keyboard for movement. |
| **Output** | Updated position of the character in the screen. |
| **Non Behavioral Requirements** | Keyboard connected to computer. |

|  |  |
| --- | --- |
| **Use Case Name** | Enter Building |
| **Use Case ID** | UC-002 |
| **Super Use Case** | Overworld Navigation |
| **Actor(s)** | User |
| **Description** | The user moves the character to a building door and the map changes from “Overworld” to inside the building. |
| **Pre-Conditions** | 1. User must be able to move. 2. The user must collide directly with the building’s door. 3. The user must be outside the building. |
| **Post-Conditions** | The screen stops showing the “Overworld” and displays the view from inside the building room. |
| **Flow of Events** | 1. The user moves the character to the entrance of a building. 2. Collision between the character and the door is detected. 3. The “Overworld” view disappears and the view of the room inside the building is displayed. |
| **Exceptions** | 1. The character gets blocked between collision of obstacles and the map edges. 2. Collision is not detected. |
| **Priority** | High |
| **Input** | “Up”, “Down”, “Left”, “Right” Keys in the Keyboard for movement. |
| **Output** | Updated position of the character in the screen, and new view form inside the building. |

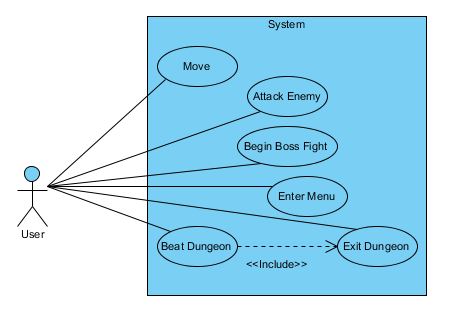
|  |  |
| --- | --- |
| **Use Case Name** | Enter Dungeon |
| **Use Case ID** | UC-003 |
| **Super Use Case** | Overworld Navigation |
| **Actor(s)** | User |
| **Description** | The user moves the character to a building door and the map changes from “Overworld” to inside the building. |
| **Pre-Conditions** | 1. User must be able to move. 2. The user must collide directly with the building’s door. |
| **Post-Conditions** | The screen stops showing the “Overworld” and displays the view from inside the building room. |
| **Flow of Events** | 1. The user moves the character to the entrance of a building. 2. Collision between the character and the door is detected. 3. The “Overworld” view disappears and the view of the room inside the building is displayed. |
| **Exceptions** | The character gets blocked between collision of obstacles and the map edges. |
| **Priority** | High |
| **Assumptions** | The user is using a Keyboard to play the game. |
| **Input** | “Up”, “Down”, “Left”, “Right” Keys in the Keyboard for movement. |
| **Output** | Updated position of the character in the screen. |
| **Non Behavioral Requirements** | Keyboard connected to computer. |

|  |  |
| --- | --- |
| **Use Case Name** | Exit Building |
| **Use Case ID** | UC-004 |
| **Super Use Case** | Overworld Navigation |
| **Actor(s)** | User |
| **Description** | The user moves the character to a building door and the map changes from inside the building view to the “Overworld” map. |
| **Pre-Conditions** | 1. User must be able to move. 2. The user must collide directly with the building’s door. 3. The User must be inside the building. |
| **Post-Conditions** | The screen stops showing the view from inside the building, and changes to the “Overworld” map view. |
| **Flow of Events** | 1. The user moves the character to the entrance of a building. 2. Collision between the character and the door is detected. 3. The inside the building view disappears and the view of the “Overworld” map is displayed. |
| **Exceptions** | 1. The character gets blocked between collision of obstacles and the map edges. 2. The collision is not detected. |
| **Priority** | High |
| **Input** | “Up”, “Down”, “Left”, “Right” Keys in the Keyboard for movement. |
| **Output** | Updated position of the character in the screen, and new view of the “Overworld” map. |

|  |  |
| --- | --- |
| **Use Case Name** | Talk to NPC |
| **Use Case ID** | UC-005 |
| **Super Use Case** | Overworld Navigation |
| **Actor(s)** | User |
| **Description** | The user interacts with NPC and a conversation is displayed at the bottom of the screen. |
| **Pre-Conditions** | 1. The Character must be right by the NPC. 2. The “Spacebar” key must be pressed first. |
| **Post-Conditions** | A conversation box appears in the bottom of the screen displaying messages to the player. |
| **Flow of Events** | 1. The user places the character close to the NPC. 2. The User presses the “spacebar” key to initiate the event to start the conversation. 3. The message from the NPC to the character is displayed in a conversation box at the bottom of the screen. |
| **Exceptions** | 1. The character is not close enough to the NPC 2. Collision is not detected. 3. No message is displayed. |
| **Priority** | Medium |
| **Input** | “Up”, “Down”, “Left”, “Right” Keys in the Keyboard for movement, “spacebar” to initiate conversation with NPC. |
| **Output** | Conversation box at the bottom of the screen. |

|  |  |
| --- | --- |
| **Use Case Name** | Open Menu |
| **Use Case ID** | UC-006 |
| **Super Use Case** | Overworld Navigation |
| **Actor(s)** | User |
| **Description** | The user is able to access the game menu from the game screen. |
| **Post-Conditions** | The menu screen is displayed for the user and the game is paused. |
| **Flow of Events** | 1. The user is in-game. 2. The User presses the key to enter the menu. 3. The game is paused. 4. The menu screen is displayed. |
| **Priority** | Low |
| **Input** | Menu key. |
| **Output** | Game menu screen. |

Dungeon Navigation



|  |  |
| --- | --- |
| **Use Case Name** | Move |
| **Use Case ID** | UC-007 |
| **Super Use Case** | Dungeon Navigation |
| **Actor(s)** | User |
| **Description** | The user controls the main character for it to navigate freely through the map. |
| **Pre-Condition** | Must be inside a Dungeon. |
| **Post-Conditions** | The user must be able to see in real time the character moving in the screen accordingly to player’s input. |
| **Flow of Events** | 1. The User is inside a Dungeon. 2. The User presses the corresponding key to the movement he/she desires to perform. 3. The screen is updated and the position of the character changes according to the keys pressed. |
| **Exceptions** | The character gets blocked between collision of obstacles and the map edges. |
| **Priority** | High |
| **Assumptions** | The user is using a Keyboard to play the game. |
| **Input** | “Left”, “Right” Keys in the Keyboard for movement. |
| **Output** | Updated position of the character in the screen. |
| **Non Behavioral Requirements** | Keyboard connected to computer. |

|  |  |
| --- | --- |
| **Use Case Name** | Attack Enemy |
| **Use Case ID** | UC-008 |
| **Super Use Case** | Dungeon Navigation |
| **Actor(s)** | User |
| **Description** | The user is able to attack enemies in the screen and defeat them. |
| **Pre-Condition** | Must be inside a Dungeon. |
| **Post-Conditions** | The defeated enemy disappears from the map. |
| **Flow of Events** | 1. The User moves close enough to the enemy to perform an attack. 2. The User presses the attack key. 3. Collision is detected and if the attack was within range the enemy is defeated and disappears. |
| **Exceptions** | The attack is out of range to be effective. |
| **Priority** | High |
| **Assumptions** | The user is using a Keyboard to play the game. |
| **Input** | “Left”, “Right” Keys in the Keyboard for movement and attack key. |
| **Output** | Attack animation and enemy is removed from the screen. |
| **Non Behavioral Requirements** | Keyboard connected to computer. |

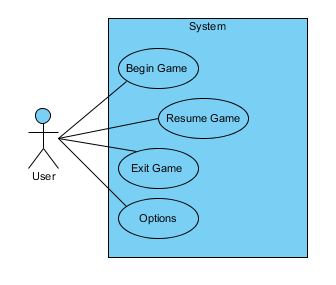
|  |  |
| --- | --- |
| **Use Case Name** | Begin Boss Fight |
| **Use Case ID** | UC-009 |
| **Super Use Case** | Dungeon Navigation |
| **Actor(s)** | User |
| **Description** | The user starts a fight with a final enemy at the end of the Dungeon. |
| **Pre-Condition** | 1. Must be inside a Dungeon. 2. The player must reach the final stage of the Dungeon. |
| **Post-Conditions** | The fight begins and the User is not allowed to exit the room where the fight is taking place. |
| **Flow of Events** | 1. The User reaches the Dungeon’s Boss room. 2. Boss appears and the battle begins. |
| **Exceptions** | The Dungeon has already been cleared. |
| **Priority** | High |
| **Input** | “Left”, “Right” Keys in the Keyboard for movement and attack key. |
| **Output** | Dungeon Boss appears in the screen. |

|  |  |
| --- | --- |
| **Use Case Name** | Enter Menu |
| **Use Case ID** | UC-010 |
| **Super Use Case** | Dungeon Navigation |
| **Actor(s)** | User |
| **Description** | The user is able to access the game menu from the game screen. |
| **Post-Conditions** | The menu screen is displayed for the user and the game is paused. |
| **Flow of Events** | 1. The user is in-game. 2. The User presses the key to enter the menu. 3. The game is paused. 4. The menu screen is displayed. |
| **Priority** | Low |
| **Input** | Menu key. |
| **Output** | Game menu screen. |

|  |  |
| --- | --- |
| **Use Case Name** | Begin Dungeon |
| **Use Case ID** | UC-011 |
| **Super Use Case** | Dungeon Navigation |
| **Actor(s)** | User |
| **Description** | The User defeats the Dungeon Boss. |
| **Pre-Condition** | 1. User HP above 0 (zero). 2. Boss HP reaches 0 (zero). |
| **Post-Conditions** | 1. A member of the “Scrum Team” is released. 2. The state of the Dungeon changes to cleared. 3. New information about other dungeons becomes available to the user by the NPCs |
| **Flow of Events** | 1. The User decreases the HP of the Boss to 0 (zero). 2. Clear message is displayed. 3. Team member is rescued. 4. The character exits the Dungeon. |
| **Exceptions** | The User is defeated by the Boss. |
| **Priority** | High |
| **Output** | Dungeon Boss appears in the screen. |

|  |  |
| --- | --- |
| **Use Case Name** | Exit Dungeon |
| **Use Case ID** | UC-012 |
| **Super Use Case** | Dungeon Navigation |
| **Actor(s)** | User |
| **Description** | The User walks out of the dungeon or clears it. |
| **Pre-Condition** | 1. User must be inside a Dungeon. 2. User must be able to move. 3. The User collides with the Dungeon’s entrance. |
| **Post-Conditions** | The Dungeon side view changes to the up view inside a building room. |
| **Flow of Events** | 1. The User moves to the entrance of the Dungeon. 2. Collision is detected. 3. The side view of the Dungeon disappears and the view from inside the building room is displayed. |
| **Exceptions** | The Dungeon has already been cleared. |
| **Priority** | Medium |
| **Input** | “Left”, “Right” Keys in the Keyboard for movement. |
| **Output** | View from inside the building is displayed. |

Menu Navigation



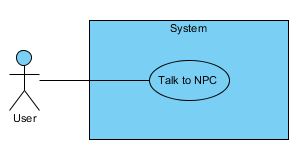
|  |  |
| --- | --- |
| **Use Case Name** | Begin Game |
| **Use Case ID** | UC-013 |
| **Super Use Case** | Menu Navigation |
| **Actor(s)** | User |
| **Description** | The game starts. |
| **Pre-Condition** | Game is being started for the first time. |
| **Post-Conditions** | The “Overworld” is displayed and the player is able to start playing. |
| **Flow of Events** | 1. The user starts the game program. 2. The menu is displayed in the screen. 3. The user clicks on “Start Game”. 4. The game starts. |
| **Exceptions** | The game has been started before. |
| **Priority** | Medium |
| **Input** | Accept key. |
| **Output** | Game “Overworld” screen view. |

|  |  |
| --- | --- |
| **Use Case Name** | Resume Game |
| **Use Case ID** | UC-014 |
| **Super Use Case** | Menu Navigation |
| **Actor(s)** | User |
| **Description** | The game resumes at the same state in which it was before the menu was opened from the game. |
| **Pre-Condition** | The game is paused and the menu is open. |
| **Post-Conditions** | Game resumes, and the player is able to control the character again and continue with the game. |
| **Flow of Events** | 1. The user selects the “Resume” option from the menu. 2. The menu closes. 3. The game screen is displayed. 4. The game resumes. |
| **Exceptions** | The game has been started before. |
| **Priority** | Medium |
| **Input** | Accept key. |
| **Output** | Game screen view. |

|  |  |
| --- | --- |
| **Use Case Name** | Exit Game |
| **Use Case ID** | UC-015 |
| **Super Use Case** | Menu Navigation |
| **Actor(s)** | User |
| **Description** | The game program closes. |
| **Flow of Events** | 1. The user selects the “Exit” option from the menu. 2. The game program closes. |
| **Priority** | Low |
| **Input** | Accept key. |

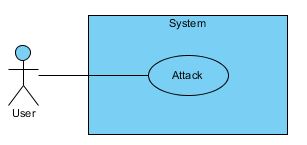
|  |  |
| --- | --- |
| **Use Case Name** | Options |
| **Use Case ID** | UC-016 |
| **Super Use Case** | Menu Navigation |
| **Actor(s)** | User |
| **Description** | The menu displays another menu to handle the options for the player regarding the game. |
| **Post-Conditions** | All changes save and are applied to the game. |
| **Flow of Events** | 1. The user goes into options in the game menu. 2. The menu is displayed in the screen. 3. The user performs changes to the settings. 4. Changes are saved and applied in the background. 5. The user can keep doing changes or go back to the game menu. |
| **Priority** | High |
| **Input** | Accept key. |

NPC Interaction



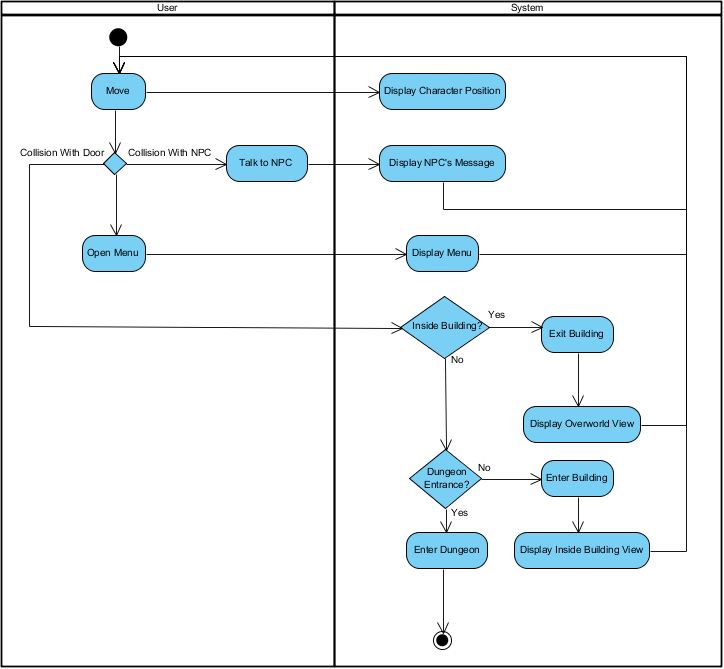
|  |  |
| --- | --- |
| **Use Case Name** | Talk to NPC |
| **Use Case ID** | UC-017 |
| **Super Use Case** | NPC Interaction |
| **Actor(s)** | User |
| **Description** | A message from the NPC is displayed in the bottom of the screen. |
| **Pre-Condition** | User must be in collision with the NPC. |
| **Flow of Events** | 1. The user starts the game program. 2. The menu is displayed in the screen. 3. The user clicks on “Start Game”. 4. The game starts. |
| **Exceptions** | The game has been started before. |
| **Priority** | Medium |
| **Input** | “Spacebar” key. |
| **Output** | Message from NPC |

Enemy Interaction

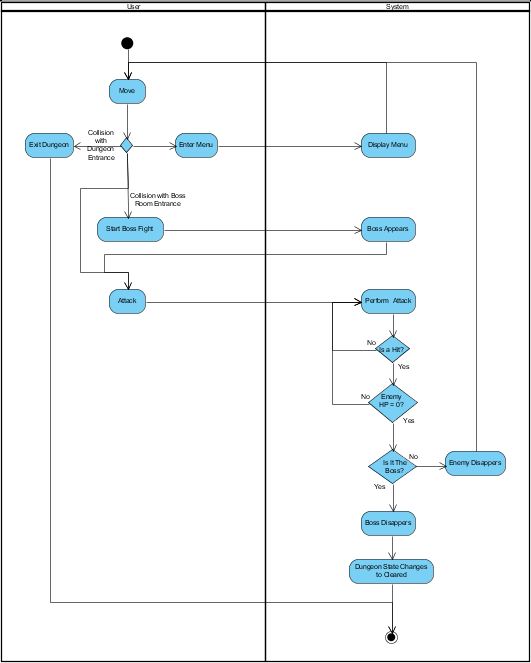


|  |  |
| --- | --- |
| **Use Case Name** | Attack |
| **Use Case ID** | UC-018 |
| **Super Use Case** | Enemy Interaction |
| **Actor(s)** | User |
| **Description** | Enemies try to attack the user. If the enemies are hit by the user they lose HP, and when the HP reaches 0 (zero) the enemy disappears. |
| **Pre-Condition** | User must be in a Dungeon that hasn’t been clear. |
| **Flow of Events** | 1. Enemy is in range of attack from the user. 2. Damage to both user and enemy are applied. 3. Enemy HP reaches 0 (zero) and disappears. |
| **Exceptions** | The user HP reaches 0 (zero) |
| **Priority** | High |
| **Input** | Attack key |
| **Output** | Enemy and attacks in the screen |

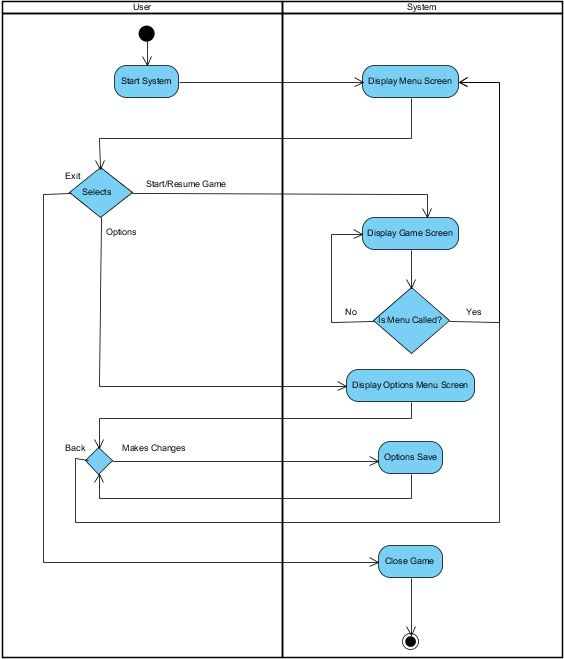
Overworld Navigation Activity Diagram



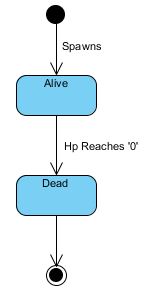
Dungeon Navigation Activity Diagram



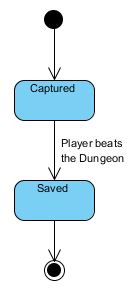
Menu Navigation Activity Diagram



Enemy State Diagram



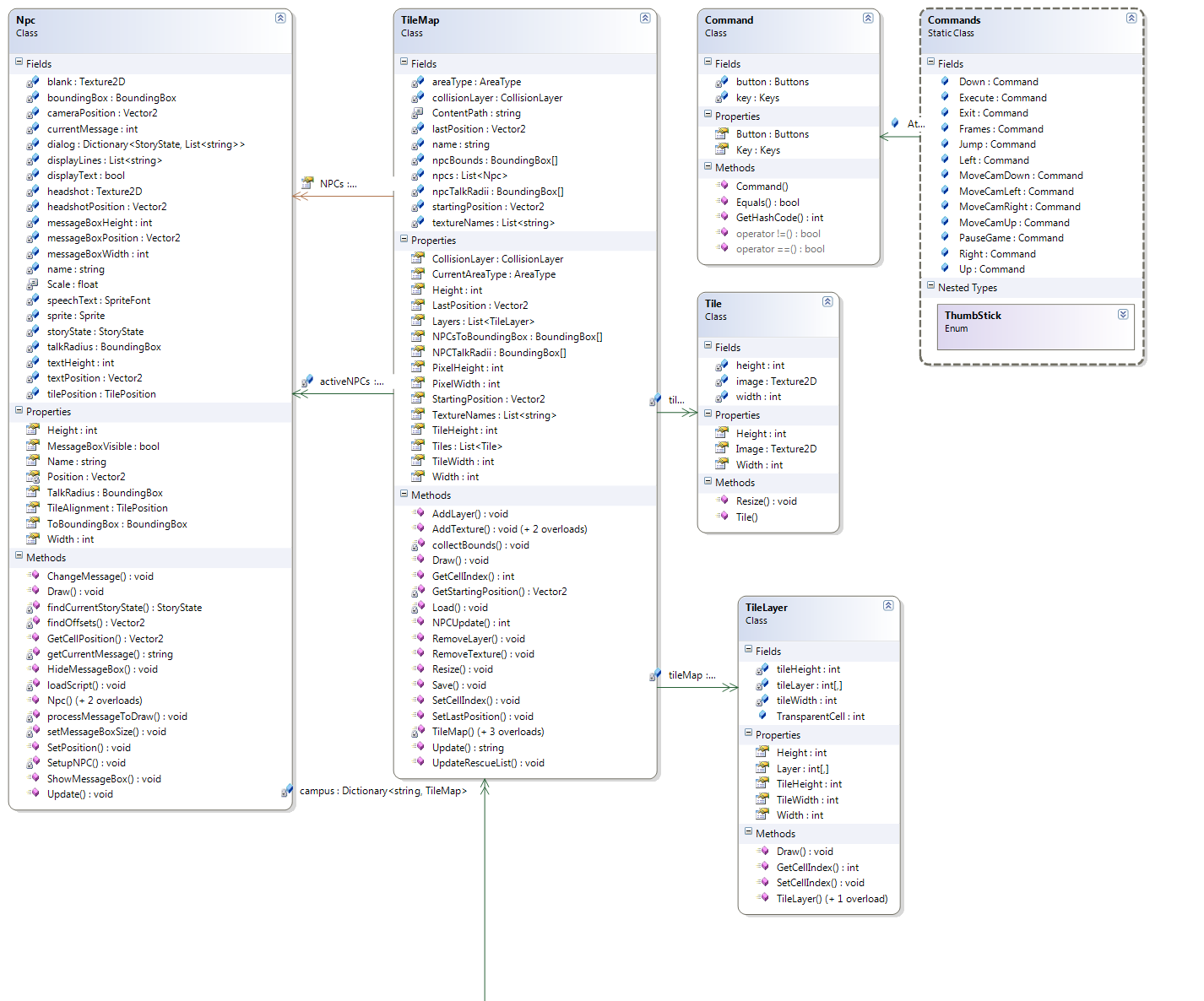
Team Members State Diagram



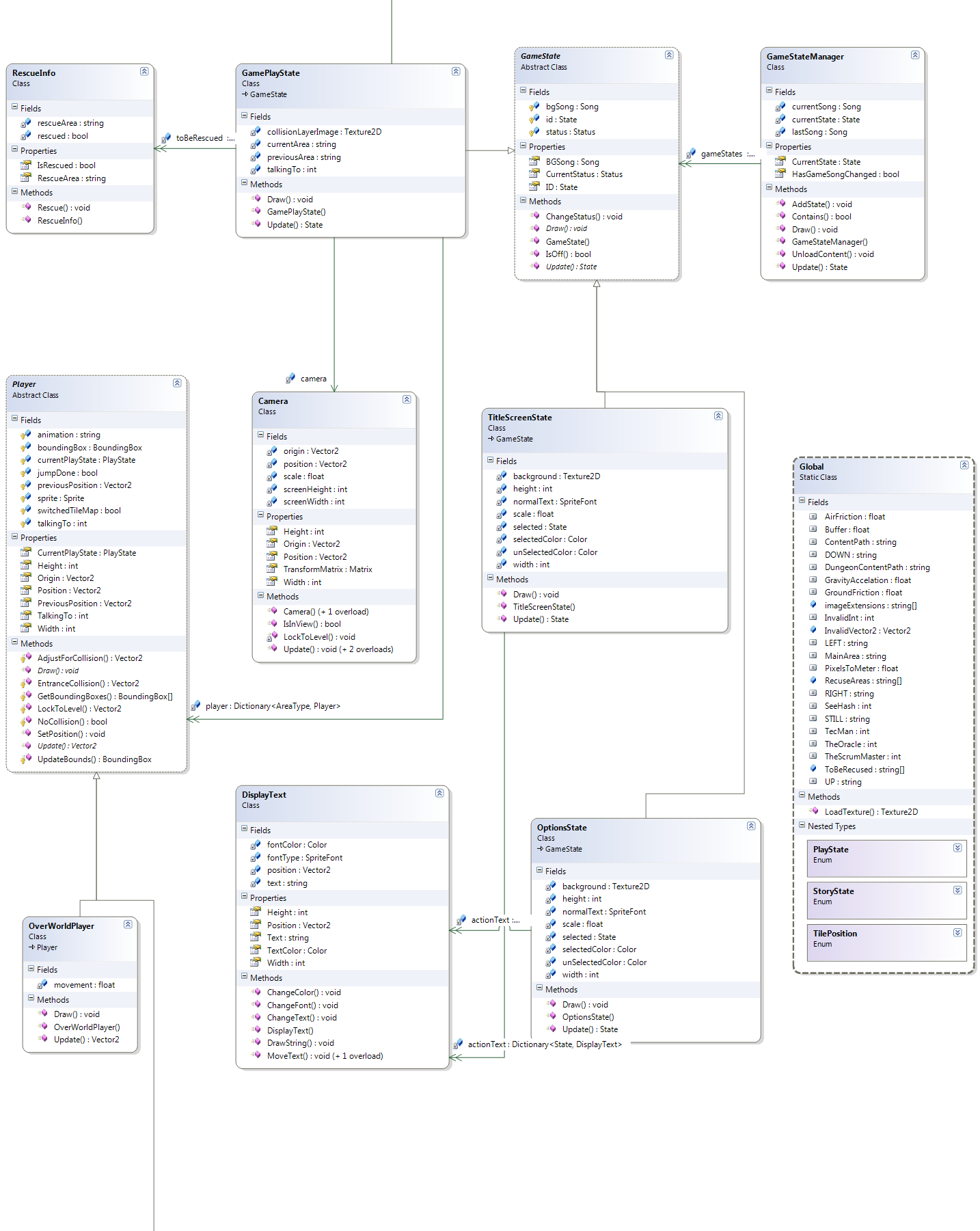
Dungeon State Diagram



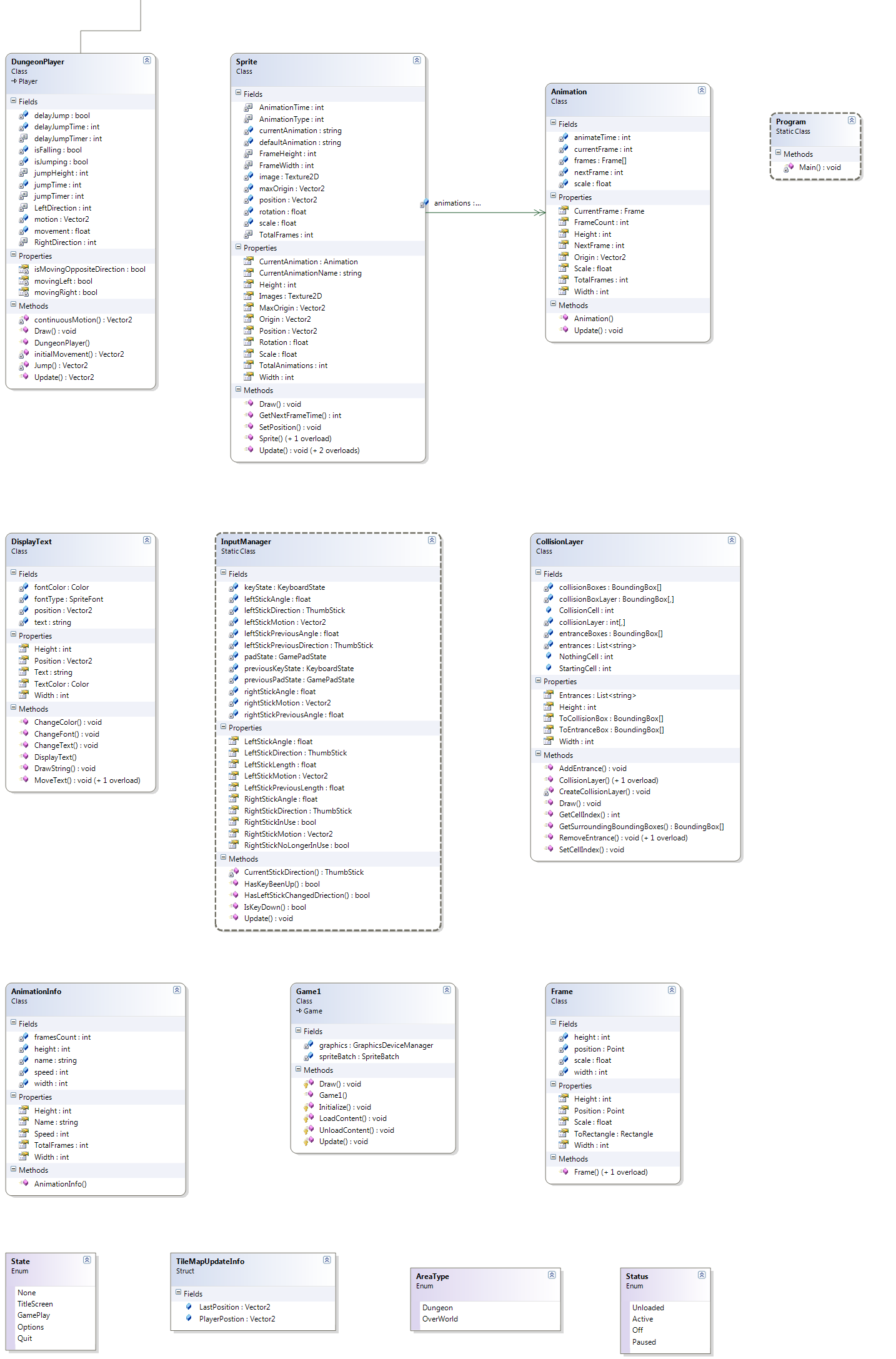
**Class Diagrams**

****

**Class Diagrams (Cont.)**

****

**Class Diagrams (Cont.)**

****

**Redundancy Elimination**

|  |
| --- |
| Maps of dungeons and campus scenes were moved into text files so that they would not take up space in the games code. |
| A Tile Map Editor was created so that instead of coding each section of the map it could be laid out with convenient and visual user interface to instantly gain access to the imagery the end user would be seeing. |

**Testing**

Quality assurance is a big challenge in game development; we tried to assure the quality of the project the best we can. All code that was produced was carefully tested to make sure it would work in regular circumstances and in exceptional circumstances as well.

**Game testing**: The primary function of game testing is to discover bugs and fix them. Interactive entertainment software testing is a highly technical field requiring expertise, analytic competence, critical evaluation skills, and endurance.

In **self-testing** we tried to test whether it meet the requirements that is listed in the game design documents. If it does not meet expectations then it will be reworked until ready for submission.

In **group testing** one of us plays as real target user and the others watch how it works and see if it replicates what we want the end user to see.

**Conclusion**

Our game is an application that people can use to entertain themselves and learn about the daily scrum process. It was designed, produced and tested to assure the best quality possible.