Initial Project Proposal

3D-Adventuring Game - The Lost Empire

Group 10: Daniel, Nianzu Ma, Yue Ben

# 1 Game Description

The Lost Empire is a 3D adventuring game. It will be a treasure hunting game amidst the unknown elements of oceans and seas. The game will be seen through a first person view, stressing the player’s comprehension of the 3D environment. The detailed graphics and character progression will provide a unique experience to our players. The focus of the game will be the Adventure Mode but the player can also explore random maps.

## 1.1 Scenario and Elements

**Role:** Player is a treasure hunter, attempting to explore well designed locations for personal gain.

**Locations:** Locations are underwater. They include but are not limited to castle, palace, pyramid, cave, dungeon, and so on. Each location is a map stored in the system.

**Scenarios:** For each location, the player will start out at an entrance. Their suit can only protect them from so much damage and may eventually break. They also only have so much oxygen. The completion of each location will be through a known task: to either find a story progressing item at the end, or to collect a number of items throughout the level. Not only does the player have to make it back out safely, but there are also other dangers along the way:

* Enemies can move and attack the character, and can be any imaginable water-based animal or another treasure hunter.
* Traps do not move but can still hurt or kill the player if sprung by the player.

The player can have more than a scuba suit and an oxygen tank to help them however. They can defend themselves with weapons against enemies. Weapons will either be blunt or sharp items. They can also have useful tools, including but not limited to a flashlight, a shovel, or a rope. In each location, the player can also find useful items to collect for themselves. This could include some weapons, some of the useful items, currency, or collectable items to sell. Weapons, tools, and equipment upgrades can be purchased with this currency.

## 1.2 Game Modes

Adventure Mode will be the focus of the game. The player will start out with one location or level to explore. The completion of levels will allow the exploration of more levels, but it will not be a linear path. Completing level 1 may open level 2, 3, 5, and 8, while completing level 3 and 5 may be required to access level 4. Completing each level will bring the player one step closer to finding Atlantis, which is expected to have the ultimate treasure. Outside of levels, the player may progress their character through weapon, tool or equipment purchases or improving their characters strength, stamina, or health.

To allow for a more wide ranging playing experience, the player will also be able to take their story character into randomly generated levels. The player can choose the size and theme of the level, and a unique level will be generated for them. Any collectable items found will stay with the character as if found through the story.

# 2 Domain

The Lost Empire will almost entirely take place underwater. Since the player is searching locations in the ocean, a realistic game would include proper scuba diving information. The player should feel like they are submerged in water while playing the game. They should experience ocean currents, buoyancy, and many natural obstacles including fish and coral reefs. The player is most certainly not immune to the dangers of the sea!

# 3 Hardware and Software Platform

The Lost Empire will be written in Java, and would ideally be compatible with any Desktop, Laptop, or Notebook with Java installed. Priority compatibility will be with the more recent versions of Windows: XP, Vista, 7, and 8. Java OpenGL will be necessary to create and display the underwater ruins.

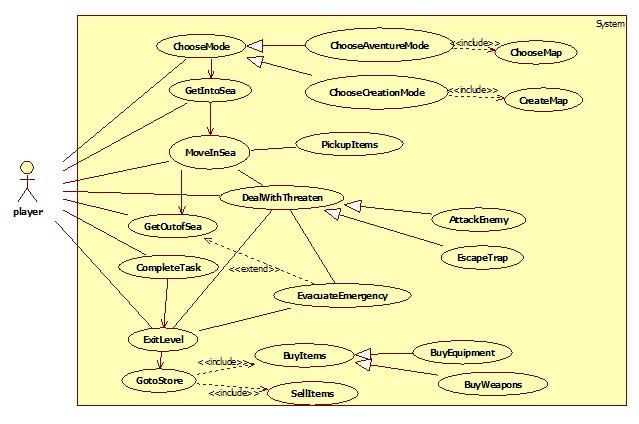
# 4 Client

Electronic Arts has expressed interest in a unique adventuring game with excellent graphics and is expected to publish the game. Aside from the domain restriction itself, the game will be most appealing to EA if it can reach out to a wide variety of players. This will be more likely based on the quality of the graphics, game play features, and character progression. The realistic diving elements will also be interesting to players but it can also become frustrating given the complexity of the diving itself, and EA will want a positive experience from a majority of their customers. EA may also want an online mode but that is currently not in our own plans.

# 5 End-User

The Lost Empire will reach a very large audience. Character progression and adventuring and exploration appeal to many pre-existing gamers. Since the game will also take place in oceans or seas, this game will also appeal to fans of scuba diving. The expected audiences include teenagers to adults in their thirties, video game players, and scuba diving fans.

# 6 Use Cases and scenarios of using the product



|  |  |
| --- | --- |
| Use case name | ChooseMode |
| Actors | Initiated by player |
| Flow of events | 1. The player chooses a kind of mode provided by system  2. The system enters the specific mode chosen by player |
| Entry condition | · The player has entered into the system |
| Exit condition | · The system switched to the mode chosen by player |

|  |  |
| --- | --- |
| Use case name | ChooseAdventureMode |
| Actors | **Inherited** from ChooseMode |
| Flow of events | 1. The player chooses the Adventure Mode provided by system.  2. The system switches to the Adventure Mode. |
| Entry condition | **Inherited** from ChooseMode |
| Exit condition | **Inherited** from ChooseMode |
| Quality requirements | At any point during the flow of events, this use case should **include** the ChooseMap use case. The ChooseMap use case is initiated when the player invokes the map function. When invoked within this use case, the system loads the map for the user. |

|  |  |
| --- | --- |
| Use case name | ChooseCreationMode |
| Actors | **Inherited** from ChooseMode |
| Flow of events | 1. The player chooses the Creation Mode provided by system.  2. The system switches to the Creation Mode. |
| Entry condition | **Inherited** from ChooseMode |
| Exit condition | **Inherited** from ChooseMode |
| Quality  requirements | At any point during the flow of events, this use case should **include** the CreateMap use case. The CreateMap use case is initiated when the player invokes the create function. When invoked within this use case, the system loads the environment for the player to create map. |

|  |  |
| --- | --- |
| Use case name | ChooseMap |
| Actors | Initiated by player |
| Flow of events | 1. The player chooses the available map listed by the System.  2. The System loads the data of the map. |
| Entry condition | · The player has chosen the Adventure Mode |
| Exit condition | · The chosen map is successfully loaded for player, OR  · The system prompts the reason why the map cannot be loaded |

|  |  |
| --- | --- |
| Use case name | CreateMap |
| Actors | Initiated by player |
| Flow of events | 1. The player requests to create a new map  2. The system creates a new map file for the player  3. The player chooses the parameters of the map given by system  4. The system creates the map according to the player’s requirement |
| Entry condition | · The player is in the Creation Mode |
| Exit condition | · The system saved the map file for the player, OR  · The player stops creating this map to create a new map |
| Quality requirement | · The map should satisfy the restriction of the relationship between parameters |

|  |  |
| --- | --- |
| Use case name | PickupItems |
| Actors | Initiated by the player |
| Flow of events | 1.The player chooses to pick up the item found in the environment  2. The system adds the item to the player |
| Entry condition | · The player has gotten into the sea. |
| Exit condition | · The player successfully added the items, OR  · The system prompts the reason why the player cannot collect the item |

|  |  |
| --- | --- |
| Use case name | DealWithThreaten |
| Actors | Initiated by the player |
| Flow of events | 1. Faced with threats in the sea, the player deals with it through escaping or attack.  2. The system calculates and shows the health status of both the player. |
| Entry condition | · The player’s health status is not zero |
| Exit condition | · The player’s health status drop to zero result in death and this level of the game is over, OR  · The player successfully gets rid of the threats |

|  |  |
| --- | --- |
| Use case name | AttactEnemy |
| Actors | Inherited from DealWithThreaten |
| Flow of events | 1. The player uses some weapon to attack the enemy  2. The system calculates and shows the health status of both the player and enemy |
| Entry condition | Inherited from DealWithThreaten |
| Exit condition | · The player’s health status drop to zero and the player is dead, OR  · The player’s scuba diving equipment is damaged, the player has to evacuate the level  · The enemy is killed by the player |

|  |  |
| --- | --- |
| Use case name | EscapeTrap |
| Actors | Inherited from DealWithThreaten |
| Flow of events | 1. The player is accidently stuck in the trap  2. The system adds some move restriction on the player  3. The player tries to escape the trap  4. The system records the activity made by the player and compares with the requirement to escape the trap. |
| Entry condition | · Inherited from DealWithThreaten |
| Exit condition | · The player successful escape from the trap, OR  · The health status of player drop to zero and the player is dead |

|  |  |
| --- | --- |
| Use case name | GetOutofSea |
| Actors | Initiated by the player |
| Flow of events | 1. The player chooses to get out of the sea  2. The system responds by making the player get out of the sea |
| Entry condition | · The player is in the sea |
| Exit condition | · The player is out of sea |

|  |  |
| --- | --- |
| Use case name | EvacuateEmergency |
| Actors | Initiated by the system |
| Flow of events | 1.The system prompt the player about his dangers, suggests the player discards all the equipment and rise rapidly to the surface of the sea  2.The player confirms to discard all the equipment  3. The system makes the player move to the surface of the sea as fast as possible, at the same time recording and calculating the player’s health status |
| Entry condition | This use case extends the GetOutofSea use case. When the scuba diving equipment is damaged or the volume of oxygen is low, this use case is initiated. |
| Exit condition | · The health status of the player drops to zero and the player is dead and loses his items and exits current level, OR  · The player gets out of sea and continues the game |

|  |  |
| --- | --- |
| Use case name | MoveInSea |
| Actors | Initiated by the player |
| Flow of events | 1. The player chooses the direction  2. The system gets the command and changes the position of the player |
| Entry condition | · The player is alive |
| Exit condition | · The position of the player is updated |

|  |  |
| --- | --- |
| Use case name | GetIntoSea |
| Actors | Initiated by the player |
| Flow of events | 1. The player chooses to get into the sea  2. The system responds by make the player into the sea |
| Entry condition | · The player is on the land |
| Exit condition | · The player is in the sea |

|  |  |
| --- | --- |
| Use case name | CompleteTask |
| Actors | Initiated by the player |
| Flow of events | 1. The player finds the clue to achieve the ultimate purpose in this level of the game  2. The system responds by prompting the player the option to continue to the next level |
| Entry condition | · The player finds the clue |
| Exit condition | · The system continues to the next level, OR  · The progression is saved and exits the game |

|  |  |
| --- | --- |
| Use case name | ExitLevel |
| Actors | Initiated by the player |
| Flow of events | 1. The player chooses to exit this level  2. The system prompt for confirmation  3.The player confirms to exit |
| Entry condition | · The player is dead, OR  · The player is alive |
| Exit condition | · The system saves the progression and the data, OR  · The system goes back to the current level |

|  |  |
| --- | --- |
| Use case name | GotoStore |
| Actors | Initiated by the player |
| Flow of events | 1. The player enters the store sell the items he or she owns in this level  2. The system adds money in the account of the player  3. The player choose the items he or she needs in the next level  4. The system reduces the money in the account of the player |
| Entry condition | · The player is alive and has got some items in the level |
| Exit condition | · The player got some items for the next level, OR  · The money in the account is not enough to afford the items needed in the next level; the player goes out of the store without items. |
| Quality Requirements | At any point during the flow of events, this use case should include SellItem and BuyItem. The SellItem is initiated when the player chooses the items he wants to sell. When invoked in this use case, the system adds money to the account of the player according to the value of the items. The BuyItem is initiated when the player choose the items he want to buy. When invoked in this use case, the system reduces the money in the account and equips the player with the item he bought. |

|  |  |
| --- | --- |
| Use case name | SellItem |
| Actors | Initiated by the player |
| Flow of events | 1. The player chooses the item to be sold  2. The system adds money to the player according to the value of the item |
| Entry condition | · The player has some item |
| Exit condition | · The increment of the players money |

|  |  |
| --- | --- |
| Use case name | BuyItem |
| Actors | Initiated by the player |
| Flow of events | 1. The player chooses the item to be bought  2. The system reduces the money in the account of the player |
| Entry condition | · The player’s money is enough for the item |
| Exit condition | · The decrement of the players money |

|  |  |
| --- | --- |
| Use case name | BuyEquipment |
| Actors | **Inherited** from BuyItem use case |
| Flow of events | 1. The player chooses the equipment to be bought  2. The system reduces the money in the account of the player |
| Entry condition | **Inherited** from BuyItem use case |
| Exit condition | **Inherited** from BuyItem use case |

|  |  |
| --- | --- |
| Use case name | BuyWeapon |
| Actors | **Inherited** from BuyItem use case |
| Flow of events | 1. The player chooses the weapon to be bought  2. The system reduces the money in the account of the player |
| Entry condition | **Inherited** from BuyItem use case |
| Exit condition | **Inherited** from BuyItem use case |

# 7 Responsibilities

A preliminary list of responsibilities will include:

* Graphics Team to produce the underwater locations, and the movement and display of the player and the enemies.
* Game play team to work on events between the character and other objects in the environment
* User Interface team to work on menus and player controls.
* Testing team to test functionality and game balance. Could also help debug.
* Story team to develop the story of the game.

# 8 Domain Expert

Dr. John Bell is the president of the Underwater Archaeological Society of Chicago, and can help us understand more information about scuba diving. He can be contacted at JBell@uic.edu and can provide more than enough information about scuba diving.



We plan on having everything ready for Development by May 3rd of this year.