The Lost Empire

Requirements Specification

Version 1.0

Mar 15, 2013

Group 10

By Daniel, Yue Ben, Nianzu Ma

Table of Contents

I. Project Drivers

1. The Purpose of the Project

1a. the User Business or Background of the Project Effort

We will build a 3D thinking game related to domain of scuba diving named The Lost Empire.   It will be a treasure hunting game in the deep unknown dangers of the sea. The Player will start out with nothing but a scuba suit and the hopes of finding the ultimate treasure.
The game will include Adventure Mode, Online Mode and Creation Mode. In the Adventure Mode, user chooses existing maps in the system to take adventure. In Online Mode, Players can either team up or compete against one another. In creation Mode, users can create maps themselves and share it with others in order to build their own story.

1b. Goals of the Project

We want to develop a 3D-thinking featured scuba diving game on the PC, eliminating the real environment of scuba, as well as additional gaming elements such as treasure collecting and selling, and equipment upgrading.

2. The Client, the Customer, and Other Stakeholders

2a. the Client

EA is expected to publish the game.

2b. the Customer

Target customers include PC game players and scuba diving fans.

2c. Other Stakeholders

- System designers, developers, testers

Every team member in our group.

- Domain experts

Domain experts will introduce technical details of scuba diving, including the underwater environment, proper equipment, threats and solutions, and so on. Based on his suggestions, system designers design game settings such as available treasure in level and equipment in store, health constraints within level, game-over conditions, and so on and so forth. Professor Bell and Michael Angelo Gagliardi have provided detailed information about scuba diving.

3. Users of the Product

The Lost Empire is aimed at several different types of Players, according to their gaming and diving experience.

Table 1 Users of the Product

User Name	***User Role***	***Subject Matter Experience***	***Technological Experience***	***Characteristics***
Experienced Scuba Diver	Playing the game	Master	Master	Well-educated, at least mid-twenties
Recreational Scuba Diver	Playing the game	Master / Journeyman	Master / Journeyman	Well-educated, at least early twenties
New Scuba Diver	Playing the game	Novice	Journeyman / Novice	At least early twenties
Experienced Gamer	Playing the game	Novice	Master	Teenager or twenties, probably Male
Recreational Gamer	Playing the game	Novice	Master / Journeyman	Teenager to thirties,
New Gamer	Playing the game	Novice	Journeyman / Novice	Teenager to thirties

II. Project Constraints

4. Mandated Constraints

The Lost Empire is a 3D adventuring game, and it's mandated to be 3D thinking. To this point, we allow the character to see and move 3-dimensionally, under the constraints of how real scuba divers see and move.

5. Naming Conventions and Definitions

Player: the one who is playing the game.

Character: the scuba diver in the game, controlled by the Player.

Level or **Game**: one level implies scuba diving once. Different Levels match different difficulties (with respect to depth, map area, and so on) and featured environment (such as sunk ship, deep sea. shallow water, and so on). For every play, the Player choose one Level and adjust equipment for the character to meet the minimum requirement for the Level, and then starts the game. After successfully diving and collecting Items within Level, the Character exits Level with Items collected.

Breathing Rate: Breathing Rate is calculated within Level. The Character has initial Gas volume and Breathing Rate values while starting the Level. As time goes by, also if the Character meets with Threats, the Breathing Rate values will change. The Breathing Rate affects the speed of consuming Gas. If no Gas left, the Player fails this Level.

Item: treasure or anything that the Character collects within Levels, such as Fish or Pearls in the sea, Gems found in a sunken ship, and so on.

Threat: any threat for Character within Levels, such as big Fish that will attack Character, Traps such as Silt and so on. Character can use Tools to escape Threats, but their Health or Equipment might be affected or damaged.

Store: a Store is provided outside Levels, where player can sell Items for money, and also spend money in buying or upgrading Equipment.

Equipment or **Tool**: equipment for scuba, which includes Scuba Suit, Gas, Knife, Flashlight, Rope, and so on. Player can buy new equipment or upgrade existing equipment in the Store outside Level.

Oxygen Toxicity: the condition in which the partial pressure of Oxygen becomes toxic to breath, causing seizures. This occurs at 1.6 ATA Oxygen (O2). A diver may have a chance to survive if diving with a group of people, but will more often than not die if diving by themselves because the seizure causes them to lose their breathing regulator and get water into their lungs.

Nitrogen Narcosis: the condition where Nitrogen has a similar effect to consuming alcohol.

Decompression Sickness: the condition where the body is breathing gas at a different pressure than the outside environment, and the gas forces itself out of the body similar to opening a plastic bottle of soda.

6. Relevant Facts and Assumptions

6a. Relevant Facts

The game is designed to be virtual reality, and we will eliminate the real scuba diving sports as much as we can. Therefore, there are some basic facts we shall follow:

- The scuba diver can only dive for a limited time in the water, under constraints of their equipment.
- According to the equipment, the diver can dive in different depth in the sea.
- The diver can only carry certain equipment and certain amount of gas with him.
- The diver can only carry certain amount of items with him.

6b. Assumptions

- For each Level, there is a fixed time value (i.e. the initial Breathing Rate of the character over the initial Gas volume) for the Character.

- The Gas volume can only decrease within Level, negative correlated to Breathing Rate, which can increase while coping with Threats. If the Character exit game with Gas greater than 0, he succeeds the Level and keep all Items collected within Level; else he fails, and cannot take Items out of the Level.
- The Character can buy certain amount of certain Gas before Level starts in Store, according to the Level he will choose.

III. Functional Requirements

7. The Scope of the Work

- 3D graphics display
- Geography analysis

  Design scuba diving environment in the sea, including silt, rock, and so on.
- Scuba diver movement

  The diver can move forward, or push / pull themselves up, down, left, right, forward-left-up, and so on. He can also turn his head to see different directions.
- Equipment and Tool design

Table 2 Equipment design

Equipment	***Property*** ***(for upgrading)***
Suit	Weight, Strength
Gas	Content, Volume
Knife	Sharpness, Hardness, Weight
Rope	Length, Toughness, Weight
Flashlight	Weight, Brightness, duration

- Threat and escaping design

Table 3 Threat and escaping Tool design

Threat	***Effects***	***Escaping Tool***
Stuck in rocks	Breathing Rate, Suit (may be damaged)	Rope, Knife
Silt	Breathing Rate	Rope
Fish attack	Breathing Rate, Suit, Items (may be dropped)	Knife

Any threat can be deadly if the Character has no proper Tool to escape. As long as the Character chooses right Tool accordingly, the system will automatically use the Tool and save the Character from Threat.

A small development team should be able to implement the simulation of character movement in first person view. They should be able to implement real time calculations based on the statistics needed for diving (as outlined in the preliminary first person UI). They should be able to implement inventory maintenance inside and out of levels, tool selection and use inside of levels, and the store and character progression. The adventure level selection could be implemented, but unless someone amongst the team is a graphical programmer, only 1 very simple level is possible to create. Limits could be placed on the complexity of tools and items to simplify development.

8. The scope of the Product

Our Functional Requirements, as seen below, came mostly from a number of use cases that were created for the initial project proposal. Not all of them translated to Functional Requirements, and these use cases were not updated to reflect the Functional Requirements, but below are the use cases that most of our Functional Requirements came from. We kept them based on the previous, outdated step to talk about how and why we changed them. The other Requirements came from an increased knowledge of scuba diving or from collaboration on the vision of the game.

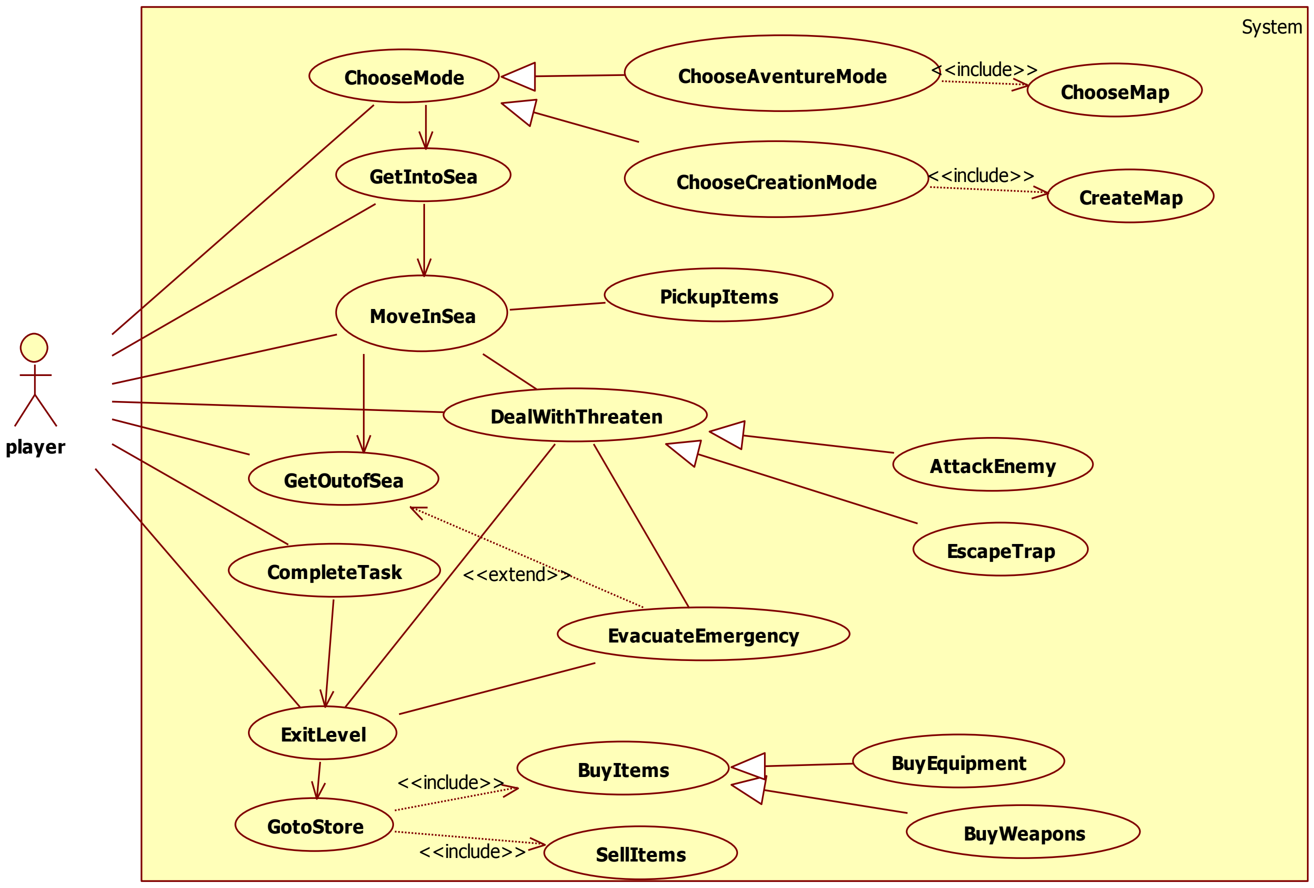

Figure 1 Use Case diagram of the Lost Empire

Use case name	ChooseMap
Participating actors	Initiated by Player
Flow of events	1. The Player chooses from the available maps 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 for fail loading map.

This was intended for our Adventure mode map selection, and has been expanded upon for #4.

Use case name	CreateMap
Participating 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 level 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 gave up this map to create a new map
Quality requirement	• The map should satisfy the restriction of the relationship between parameters

This was intended as a stand-alone game mode, but we decided to allow Adventure. Mode to directly include a random map generator along with the adventure levels, and translated to #8.

Use case name	PickUpItems
Participating 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 Character
Entry condition	• The Character has got into sea.
Exit condition	• The Character successfully added the items, OR • The System prompts the reason why the Character cannot collect the item

PickUpItems became #21, and was expanded upon to include specific inventory constraints.

Use case name	DealWithThreats
Participating actors	Initiated by the Player
Flow of events	1. Faced with threats in the sea, the Character deals with it through escaping or attack. 2. The System calculates and shows the health status of both the Character and the Threat if it's some entity in the sea.
Entry condition	• The Character's health status is not zero
Exit condition	• The Character's health status drops to zero which results in his death and this level of the game is over, OR • The Character successfully gets rid of the threats while possibly losing some non-fatal health value

DealWithThreats was an abstract use case containing similar elements amongst two use cases. AttackEnemy and EscapeTrap are also referenced below. DealWithThreats is relevant because the concepts of being damaged and/or escaping are still in the game, but increasing the Character's breathing rate due to stress from injury was a more realistic option for our domain instead of health bars and underwater sea battles. DealWithThreats outlined the need for requirements #16 to #19, #22, and #24.

Use case name	AttackEnemy
Participating actors	Inherited from DealWithThreats
Flow of events	1. The Character uses some weapon to attack the enemy 2. The System calculates and shows the health status of both the Character and enemy
Entry condition	Inherited from DealWithThreats
Exit condition	• The Character's health status drops to zero and the Character is dead, OR • The Character's scuba diving equipment is damaged, the Character has to evacuate the emergency • The enemy is killed by the Character

AttackEnemy was deemphasized after gaining more information about the domain of scuba diving. The focus of the game will be about exploration. The user will not have a health bar and instead experience increases to breathing rate. But the user's character will have a number of knives and some blunt objects, and they do have the option of attacking the computer entities if they choose. #7, #18, #19, #22, #22, and #24 are related to AttackEnemy.

Use case name	EscapeTrap
Participating actors	Inherited from DealWithThreats
Flow of events	1. The Character is stuck in the trap 2. The System adds some move restriction on the Character 3. The Character tries to escape the trap 4. The System records the activity made by the Character and compares with the requirement to escape the trap.
Entry condition	• Inherited from DealWithThreats
Exit condition	• The Character successfully escaped from the trap, OR • The health status of Character drops to zero and he is dead

EscapeTrap maintained a similar functionality at this stage, except health bars were removed and the breathing rate concept was used again. Since the diving experience cannot entirely be simulated due to the lack of real physical contact, we decided to include a 3rd person view in addition to the 1st person view. This is a choice; however; 3rd person view does not allow any actions. Rather, it is intended to be used when the user is stuck, so they can see how they are stuck and how to remove themselves. EscapeTrap translated to #13, #16, #17, and #19.

Use case name	MoveInSea
Participating 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 Character
Entry condition	• The Character is alive
Exit condition	• The position of the Character is updated

Our original idea for MoveInSea does not match a realistic scuba diving scenario and has been changed. The basic concept is the same, but we greatly expanded on it. #14 and #15 were created to satisfy the new intent of underwater navigation.

Use case name	GoToStore
Participating actors	Initiated by the Player
Flow of events	1. The Player enters the store sells the items he or she owns in this level 2. The System adds money in the account of the Player 3. The Player chooses the items he needs in the next level 4. The System reduces the money in the account of the Player
Entry condition	• The Character is alive and has collected some items in the level
Exit condition	• The Player bought some items for the next level, OR • The Player sold some items and increased their money, OR • The money in the account is not enough to afford the items needed in the next level; the Character 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 wants to buy. When invoked in this use case, the System reduces the money in the account and equips the Character with the item he bought.

GoToStore was a use case that was related to several other use cases, but we collapsed it to one requirement. We realized that we needed a few extra things to make the store more ideal at this stage. The user doesn’t necessarily have to sell his collected items, they will have a storage outside of levels. And the user must have some way to gain money without spending money, otherwise they may be too scared to buy anything from the store if all of their money is needed for entering levels. GoToStore is related to #6, #7, #8, #20, and #23.

Use case name	ExitLevel
Participating actors	Initiated by the Player
Flow of events	1. The Player chooses to exit this level 2. The System prompts for confirmation 3.The Player confirms to exit
Entry condition	• The Character is dead, OR • The Character is alive
Exit condition	• The System save the progression and save the data in the database, OR • The System goes back to current level

ExitLevel has been changed slightly for clarity and depth. If the user dies in a level, it is not a proper exit and it is instead a game over. Also, decompression stops change where the exit is. If the user decides to perform decompression stops, then the exit is at the surface instead of a cave entrance. #3, #12, #23, and #24 are related to ExitLevel.

9. Functional and Data Requirements

1) From the starting menu, the system will provide access to load previously saved adventure modes and allow the creation of a new adventure mode.
2) The system will continuously save the states of five adventure modes, and the creation of a sixth adventure mode must overwrite one of the five previously saved states.
3) The system will save the state of a Player's adventure mode before beginning and upon the Player’s successful exit of each adventure level.

4) The system will allow the Player to select a level to explore from any available adventure level, but there will be a base cost to explore that level from equipment and gas supply.
5) The system will unlock additional adventure levels for the Player to explore upon completing the required objective(s) to unlock a particular adventure level.
6) The system will contain some method to receive money at no original monetary cost.
7) The system will provide a store outside of levels for the Player to receive money from selling previously collected items and spend money on equipment or character training.
8) The system will allow the Player to search through their bag, and to select or un-select equipment to carry with them into the next level while they are outside of levels.
9) The system will be able to randomly create levels, subject to optional Player selected constraints, with collectible items, which can be explored with the Player's adventure mode character.
10) The system allows the Player's selection of time to explore each level based on the size of that level, but this selection has a direct relationship to a monetary cost.
11) The system will allow the Player’s selection of maximum depth to dive in each level; however, this selection will alter the cost, an entrance must still be accessible at the selected depth, and the selection cannot exceed the shallowest or deepest points in that level.
12) The system will allow the Player to choose to avoid decompression stops before they begin each level, but the system will reward the completion of any level with decompression stops.
13) The system will display either a first person view of each level the Character is currently diving, and will also display statistics about the Character’s current dive (breathing rate, oxygen partial pressure, remaining gas supply or time remaining) and the objective(s) as a projection on their mask, or a third person view of the

Character, with no statistics displayed and no allowed actions except to return to first person view.

14) The system will allow the Character to swim forward, push or pull themselves, reposition their body, control their buoyancy, follow a reel line, or rotate their head subject to normal human being movement capabilities, which will also change the user's location inside and the first person view of the level.
15) The system will not prevent the Character from navigating too shallowly or too deeply and the system will make the Player’s character feel any negative effects of diving outside the ranges of the at depth gas.
16) The system will have collision detection between the Character and traps, which themselves are either intentional traps or unstable objects in the levels and can damage or restrict the movement of the user.
17) The system will allow the Character to free themselves when the Character becomes stuck or buried, possibly displaying the actions freeing the Character in third person view, but this extra use of force will temporarily increase the Character's breathing rate.
18) The system will control non playable entities that may or may not be able to move around, damage the Character, or be damaged by the Character.
19) The system will translate Character damage from entities or traps as minor cuts or bruises, and automatically increase the Character's breathing rate proportional to the damage as an increase in stress from the injury.
20) The system will have collectible items in each level; however, the system will only save those items outside of that level if the Character successfully exits that level. The system will limit the volume of items the Character can carry. The volume of items impacts buoyancy and breathing rate. The Character can drop items, and the system will remove items from level if the Character exits this level with the items.
21) The system will allow the use of tools that the Character brought into the level, including but not limited to secondary light sources, reels, and knives; all which may temporarily increase the Character’s breathing rate depending on the tool and how it is used.

22) The system will force the death of non-playable entities if their health reaches 0.
23) The system will force the death of the Character if the Character runs out of gas or if they suffer from Oxygen Toxicity or Decompression Sickness.
24) If the system has forced the death of the Character, the system will stop the Player's current playing session, display the game over screen, and will eventually display the starting menu unless the Player acknowledges the game over themselves.

IV. Nonfunctional Requirements

10. Look and Feel Requirements

10a. Appearance Requirement

1) The appearance of the operation interface should be sports-oriented, dynamic and shows movement. The user interface could integrate the dynamic lines, pictures and shape for every bar, button and background. The dominant hue should be blue, which is the main color the divers would see when they are diving.
2) As for this game is named “the Lost Empire”, the appearance should show some mysterious atmosphere.
3) The product should comply with corporate branding standards.

Fit Criterion:

1) Because most user are teenage, young adult or scuba diving enthusiast, they love sports and have enthusiasm. The appearance should cater to their spirit and give them enticement.
2) For most people, ocean gives people the mysterious feeling. The theme of our game is Lost Empire discovery, the mysterious atmosphere given by the appearance could evoke the user’s desire to discovery in the game.
3) The office of branding should certify the product complies with the current standards.

10b. Style Requirement

- The style of the software should be authentic, professional and accurate. It should not have too much cartoon or animation element in the design.

Fit Criterion:

This game is not only for entertainment, but also aim at giving user the most authentic feeling when they are playing the game. So the style should give user the feeling that that's what we are experiencing in the game is exactly the same as realistic diving experience.

11. Usability and Humanity Requirements

1) Operation conventions of this game for users are compatible with the popular PC games in the market.
2) For users who do not know diving skills could learn diving knowledge in the tutorial provide by the game or learn by themselves through entry level suggested by the game.

11a. Ease of Use Requirement

1) The Operation convention of this game for user is compatible with the popular PC games in the market.
2) The game should be easy for teenagers older than 11 years old or adults to start after a few minutes' video tutorial.
3) The system shall provide a scuba diving video tutorial and an operation instructions document, so as to help any user start the game in a few minutes.
4) The system shall provide a naive tutorial/entry level.
5) The game should not frustrate the users who do not know diving skills before playing, it should attract them to learn in the skills in the game.
6) The game should make the users in different background (teenager, young adult, with diving knowledge, without diving knowledge) want to usually play it.

Fit Criterion:

1) Any users that have some experience playing PC games before should have the instinct and successfully be able to do the basic operations (basic operations take 70% of the total operation) of the game without seeing any instruction or tutorial.

2) 70 percent of test panel of 11-year-old children should be able to successfully complete the first entry level within 20 minutes.
3) 80% percent of a test panel of the users without any diving experience should want to learn it from game and usually play the game to practice.
4) An anonymous survey should show that 75% of the intended users usually play the game after 2 hours familiarization.

11b. Learning requirements

The learning process to play the game should be divided into two part and discussed in several situation. First is the process to learn how to operation the character in the game, this should be easy for most game players. Second, because this game is designed in the scuba diving domain, it will require some basic domain knowledge for user to learn, so it takes a little time to learn this part. For different users, the requirement should be specified as below:
1) The game should be quite easily for users who have basic domain knowledge to use.
2) For user who do not know basic domain knowledge, there are entry level and video tutorials for the user to learn. Learning process through the help facility provided by the game should be interesting, clear and heuristic.

Fit criterion:

1) For users who know the basic domain knowledge of scuba diving, they do not need the second learning process and can easily operate the game within 10 minutes. If they usually play PC games, they should have the instinct to know the basic operations of the game, and by a glance at the simple instruction they know all the operation disciplines of the game.
2) 85% of the user in the test panel who do not have any basic knowledge of scuba diving, they shall have the interest to finish the interactive help with patience because the learning process is quite interesting and attractive for them.

11c. Personalization and Internalization Requirement

1)The user interface of the software should support multiple language, spelling preferences and language idioms to facilitate the users from different countries.

2) The game should supply several kinds of interface with different styles of icons and buttons or other elements to fit the preference of the people of different backgrounds.

Fit Criterion:

1) The language, spelling preferences and language of the game should support all the official languages around the world especially for the users in the intended countries.
2) Make the users from different countries have the feeling that the game is exactly designed for them, not just roughly translated from other languages.
2) For people of different gender, character, they can always find the style they like in the kinds of interface provided. However, all kinds of interfaces should comply with the criteria mention in 10. Look and Feel Requirement.

11d. Understandability and Politeness Requirement

1) The game should use both symbols and words if necessary to express items and concepts so that they could be easily understood by scuba divers and also users who do not have basic domain knowledge of scuba diving.
2) For items and concepts illustrated by symbols or pictures (such as different kinds of scuba), users should know the meaning of difference and know how to choose without seeing the detailed instruction.

Fit criterion:

There should not be too much content in the game which would frustrate users who do not have much domain knowledge of scuba diving. Most concepts and items should be friendly to all kinds of users, which means they should be either illustrated by symbol and picture or given detained explanation or both.

12. Performance Requirements

1) The system could automatically adjust the graphic quality according to the hardware of the computer to optimize both image quality and fluency.
2) The response time should be consistent during the game.

12a. Speed and Latency Requirements

1) Any interface between a user and the system of the game should have maximum response time of 0.5 second.
2) The response time should be consistent during the game.
3) The response shall be fast enough to avoid making users feel latency of the game and user should have the feeling of the fluency of the game.
4) When making diving plan for users, the time taken should within 5 seconds, and should display progress bar for the user.

Fit criterion:

The response time during playing should give user the feeling that the system of the game is an accurate, fast-responded, and reliable system.

12b. Safe-Critical Requirements

1) For this game is plan to be sold internationally, the product should follow the safety standards of the intended market different country.
2) For China market, the game shall embed the avoid-addiction-system into the system.

12c. Precision or Accuracy Requirement.

All values will be calculated to two decimals places. This includes time (likely to hundredths of seconds), depth (likely to hundredths of feet or meters depending on country), partial pressure of Oxygen (likely to hundredths of ATA), and breathing rate (likely to hundredths of meters or feet cubed per second). The narcotic effect of Nitrogen will also be recorded to two decimal places, but its value is equivalent to feeling under the effects of alcoholic beverages, so it will likely just be represented as a number with a fractional component and no units. All values should also be accurate to within ±.05 of the real time value.

12d. Reliability and Availability Requirement

1) The game should be available for use 24 hours per day, 365 days per year.
2) The game's possibility of failure should be very low and almost can be ignored.

12e. Robustness or Fault-Tolerance Requirement

1)When comes with a crashes caused by software bugs, the user could recover the software by restarting and all the information in the last progress save point could be completely reloaded in the system.
2) If the software is forced to be closed or improperly close, the data and progress information of user should well kept by the system.
3) The cause of crash should be collected by the software and send to maintenance department of this game for future improvement.

Fit Criterion:

The system should be stable even when crash happed it could also give user the feeling that system is reliable for all the data and record is kept properly anytime.

12f. Capacity Requirements

So far our game is stand-alone game, so support a single player is fine in this game.

12g. Scalability or Extensibility Requirements

1) The development company of the game should be able to add new series of adventure mode of different theme/story to the game by adding new package of maps into the system. Such additions do not require modification of existing system.
2) The development company of the game should be able to add new mode to the game such as online mode and creation mode. Such additions do not require modification of existing system.

13. Operational and Environmental Requirements

13a. Production Requirement

1) The software shall be distributed as compressed file (as for the format, use the most popular compression format in windows, Unix-like systems).
2) The software shall be able to be installed by an untrained user without the help of other instructions.

3) The product should be of the size that can fit into DVD.

4) The product should also be downloaded at the website to facilitate the users who do not have CD-driver. And the software can be activated by using the activation key bought be user.

5) The install process may be various in different computer, but the most time taken should within 20 minutes under the minimum requirement of computer hardware.

13b. Release Requirement

1) Because the development department collect data from user each time the software have abnormal behavior, the development department will give maintenance releases every two month if necessary.

2) The users could easily install the fix package of the game with the help of instruction and it would not affect or change the customization or progress records of user in the system.

14. Maintainability and Support Requirements

14a. Maintenance Requirement

The maintenance process of the software should be done by both user and developers. When the software does not work well (crash happened or other bugs), developer remotely collect the error report and develop a fix package and give users announcement. Then user choose to install the released package to strengthen the robustness of the game.

Fit criterion:

Users of difference background even 11 years old teenage could maintain the software without any help.

14b. Implementation Environment and Adaptability Requirements

1) The Lost Empire should be written in Java and compatible with any desktop, laptop, or Notebook with Java runtime environment.

2) The operation systems environment should be any popular version of windows

operating system (windows XP, windows vista, windows 7, windows 8) and any UNIX operating systems (Mac OS X, Linux, Solaris)

3) Java OpenGL is required for the graphic display.

Below are Minimum requirement and recommended specifications, equivalent hardware are acceptable.

Table 4 Supported OS - Windows

Windows		
	MINIMUM REQUIREMENTS	RECOMMENDED SPECIFICATIONS
Operating System	Windows® XP/Windows Vista®/Windows® 7/Windows® 8 (Updated with the latest Service Packs)	Windows 7/ Windows 8 64-bit with latest service pack
Processor	Intel® Pentium® D or AMD Athlon™ 64 X2	Intel Core 2 Duo 2.2 GHz, AMD Athlon 64 X2 2.6GHz or better
Video	NVIDIA® GeForce® 6800 or ATI™ Radeon™ X1600 Pro (256 MB)	NVIDIA GeForce 8800 GT, ATI Radeon HD 4830 (512 MB) or better
Memory	2 GB RAM (1 GB Windows XP)	4 GB RAM
Storage	25 GB available hard drive space	
Internet	Broadband internet connection	
Media	None for the recommended digital installation	
Input	Keyboard and mouse required. Other input devices are not supported.	Multi-button mouse with scroll wheel
Resolution	1024 x 768 minimum display resolution	

Table 5 Supported OS - Mac

Mac		
	MINIMUM REQUIREMENTS	RECOMMENDED SPECIFICATIONS
Operating System	Mac OS® X 10.7.x (latest version)	Mac OS X 10.8.x (latest version)
Processor	Intel Core™ 2 Duo	Intel Core i3 or better
Video	NVIDIA GeForce 8600M GT or ATI Radeon HD 2600	ATI Radeon HD 5670 or better
Memory	2 GB RAM (1 GB Windows XP)	4 GB RAM
Storage	25 GB available hard drive space	
Internet	Broadband internet connection	

Media	None for the recommended digital installation	
Input	Keyboard and mouse required. Other input devices are not supported.	Multi-button mouse with scroll wheel
Resolution	1024 x 768 minimum display resolution	

15. Legal Requirement

Scuba diving like many other active sports, has inherent risks which might lead to injure or in extreme cases, cause to death. The game is developed for scuba divers and other users for entertainment and also have the diving experience with basic diving knowledge and skills. The software should not be used as a training tool or sole source of reference. It cannot replace the real training provided by accredited instructors and diving professionals.

1) User should agree with the liability contract when register the new player in the game. Agree that this game should not be used as the sole source of reference nor should replace the real life training provided by accredited instructors and diving professionals.

2) Users should agree the Released Parties should not be responsible or liable for any death, injury or other damages caused as a result of using the software.

V. System Models

16. Use Case model

Please refer to 8 for the Use Case Model of the game.

17. User Interface Design

- To choose Level:

There are a pool of Levels in the system, with a recommended routine for the surfing order among the Levels, basically according to the difficulties. The Player can click and view the description for each Level and select any one to begin.

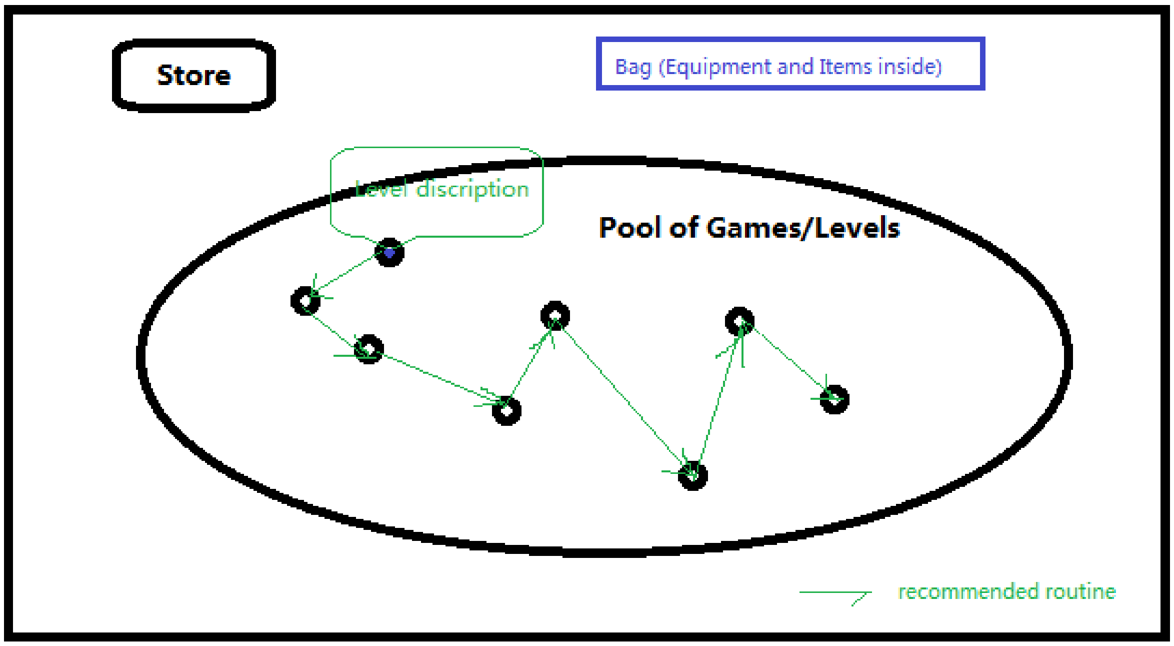

Figure 2 UI - Select Level

- The Bag view:

At any time inside or outside the Level, the Player can click Bag and see Equipment and Items inside. The basic contents shown in the Bag are the Equipment the Character has (including Equipment in use within Level or selected for next Level, and Equipment he owns but is not using for current Level), the Money the Character has, and the Items the Character is carrying. For all the items the Character has collected, he has to carry them with him all the time, even across Levels. A price is attached for each Item, at which the Character sells the Item. A “Sell Item” button is shown if the Player open his Bag in store, and he can click to select an Item and sell it for money. A “Drop Item” button is shown at any time, and the Player can click to select an Item and drop it for no money.

There are space or number constraints for Equipment in use, Equipment in bag and the Items.

Equipment				Money	$90.00		
Equipment In use	Suite 1	Knife 1	Knife 2	**Items**	Item 1 $5.00	Item 2 $14.00	Item 3 $3.50
	Rope 1	Flashlight 1					
Equipment In bag	Suite 2	Knife 2					
						Sell Item	Drop Item

Figure 3 UI - sample Bag view

There is also constraint on the equipment in use: at most 1 gas capsule, 1 suit, 3 Flashlights, and 5 knives can be carried (i.e. in use) at the same time.

- The Store view:

At any time outside the Level, the Player can go to Store to sell Items and buy or upgrade Equipment. The Bag view is embedded in the Store view.

Equipment for sale	Suite 4 $35.00	Knife 3 $20.00	Gas 1 $10.00			**The Bag view**
Upgrade Equipment	Suit weight -1 $15.00		Knife sharp +1 $10.00			
	Rope length +2 $5.00		Gas +50 $25.00			
		Buy Equipment	Upgrade Equipment			

Figure 4 UI - sample Store view

After the Player select some Equipment for sale, he can Buy it. The Player can also select one equipment from bag and one corresponding upgrade in store to upgrade his own equipment. All operations under bag space constraints and money constraints.

- Within Game/Level:

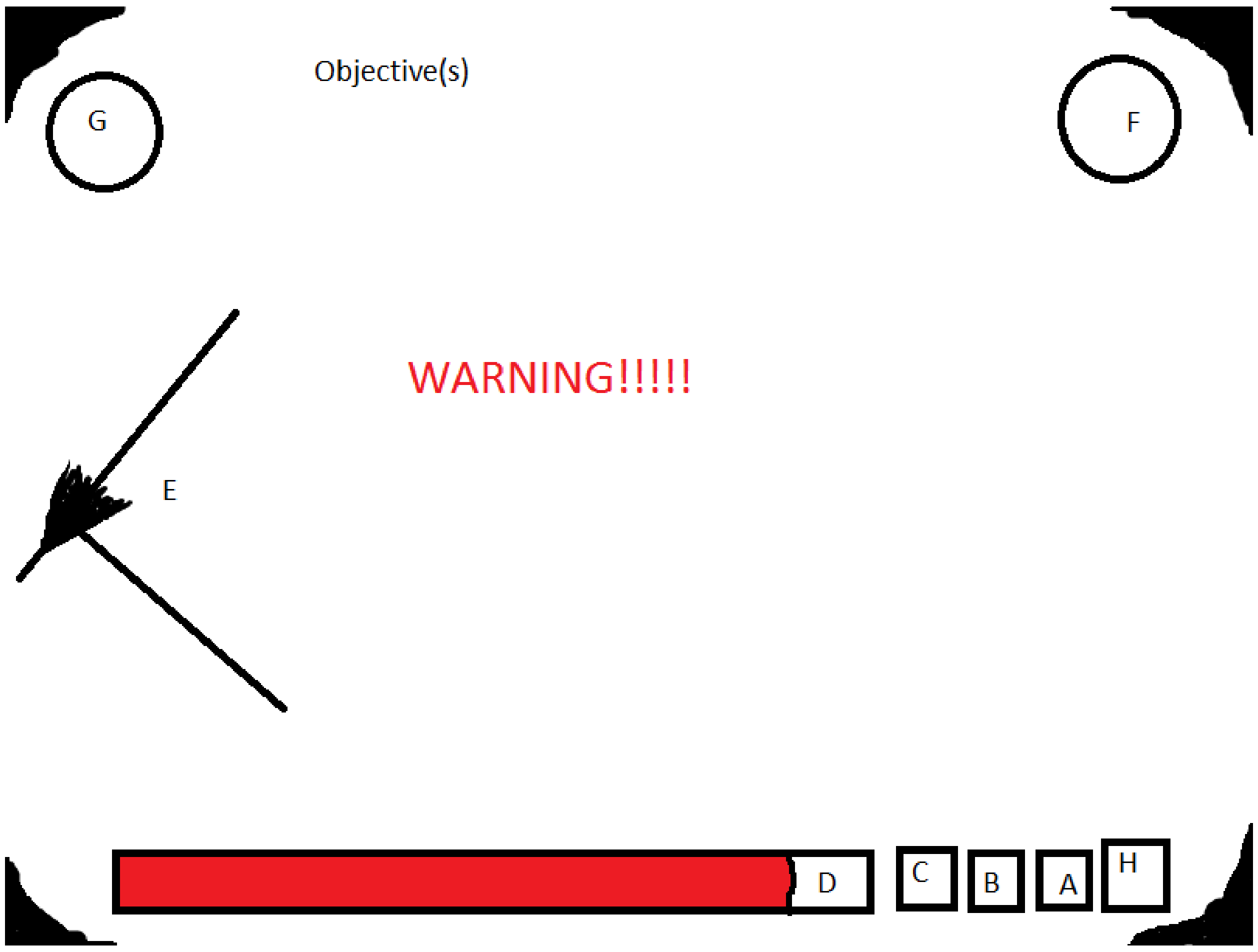

Figure 5 UI - first person view in level

A few things to mention not containing letters include the Objective(s) at the top of the screen, any potential Warning Messages related to the information at the bottom, and the blacked out portion on the corners representing a scuba mask.

A: Effect of Nitrogen Narcosis. More knowledge is needed for custom gases, but if Player is breathing air, every 33 feet = 1 Gin, if player is at 66 feet, A would show 2.

B: Oxygen ATA: Player must keep their Oxygen ATA below 1.6 ATA or they will die. Example: If player is at 200 feet depth, 200 / 33 feet = 6.66, + 1 for sea level = 7.66.

1.2 / 7.66 = .157. Player could be breathing 15.7% Oxygen gas mixture at this depth to be safe, and this may be showing 1.2 ATA. However, (1.6 / .157) - 1 = 9.19, * 33 = 303.3 feet. Player would die if deeper than 303.3 feet and breathing 15.7% Oxygen because Oxygen ATA would be at 1.6 ATA and this would cause Oxygen Toxicity.

C: Breathing Rate: More information is needed for specific calculations, but this would be in some volume of gas consumption per second.

D: Time remaining: This would be represented as a bar and with a time in minutes/seconds on the bar. The time remaining would also be based on the time it would take for the gas volume to be consumed at the current breathing rate, and the bar may actually increase if the Player can decrease their breathing rate.

E: Reel line: If the reel line is in sight, the Player will be able to see it in their first person view. At intersections or maybe randomly by the Player’s placement, there will be arrows pointing to the exit that they may also be able to see if in their view.

F: Tool in Right Hand: This will be a selectable item based on the user, implementation to select pending, but could be anything including knife, container of gas, reel line, extra flashlight, or their bare hand. Bare hand would be required to follow a reel line on the right side of their body.

G: Tool in Left Hand: This is the same concept as the right hand; however, they may not have access to the same tools with their left as their right depending on the placement of tools on their body.

H: Current depth.

Appendix

Table of Tables

Table of Figures