Software Engineering Project Report

X.1 Introduction

X.1.1 Intentions of this report

Most part of Role Playing Game Soul have been developing. Writing this report is to give a efficient way to develop. By this report, we would develop project with more materials to standard the development process and rise the development efficiency which lead to a better release.

X.1.2 Background

Project Name: **Soul of Hero**
Team Name: **Soul**
Team Member: Yao Chenzhen(15205941)
Chen Jin(15205898)
Wang Maozhun(15205930)
Dong Yuehui(15206084)

X.1.3 Project development documents

System Requirements: systemRequiremetns.docx
Functional Requirements: functionRequirements.docx
User Interface Specification: userInterfaceSpecification.docx
Class diagram: Soul1.0.2.3.asta
Bug sets: buglist.docx

X.2 Project Requirements Analysis

X.2.1 Project Idea

Role Playing Game is the most ancient and the most types game and we thought the RPG have a special status in the world of game. Everyone have a hero dream or something they cannot realize in reality, and we want to come true them by hands.

The main ideas of our game is that the player plays role in a virtual world and there are many monster who damage fair of world, the player should beat them and save the world. Otherwise, many people exited could be communicated with and they would lead the development of story.

X.2.2 System Requirements

X.2.2.1 Characters, maps items and skills are most necessary in the game.

X.2.2.1.1 Characters should be divided to player and npc.

X.2.2.1.2 Map should hold locations. Location is guiding by direction. Exit is to link each location.

X.2.2.1.3 Item should be divided to common item, food, weapon, armor.

X.2.2.1.4 Skill need a skill base to store it.

X.2.2.2 Archive to manage, save and load game data.

X.2.2.3 IO controller to handle input and output.

X.2.2.4 A graphical user interface is needed.

X.2.2.4.1 Log in.

X.2.2.4.2 Help.

X.2.2.4.3 Description.

X.2.2.4.4 Main game.

X.2.2.4.5 Archive.

X.2.2.4.6 Inventory.

X.2.2.4.7 Character's properties.

X.2.2.5 A database to store data.

(More details in work package.)

X.2.3 Functional Requirements Specification

X.2.3.1 Player hold data about name, description, properties, location, inventory and can operate them.

X.2.3.2 Inventory hold items and can operate them like get, remove them.

X.2.3.3 Map hold locations and can operate them like get, remove them.

X.2.3.4 Skill Base hold skills and can operate them like get, remove them.

X.2.4 User Interface Specification

X.2.4.1 Log in interface.

X.2.4.2 Archive interface.

X.2.4.3 Dialog box for handling communication.

X.2.4.4 Inventory interface.

X.2.4.5 Item interface.

X.2.4.6 Character interface.

X.2.4.7 Map interface.

X.2.4.8 Skill base interface.

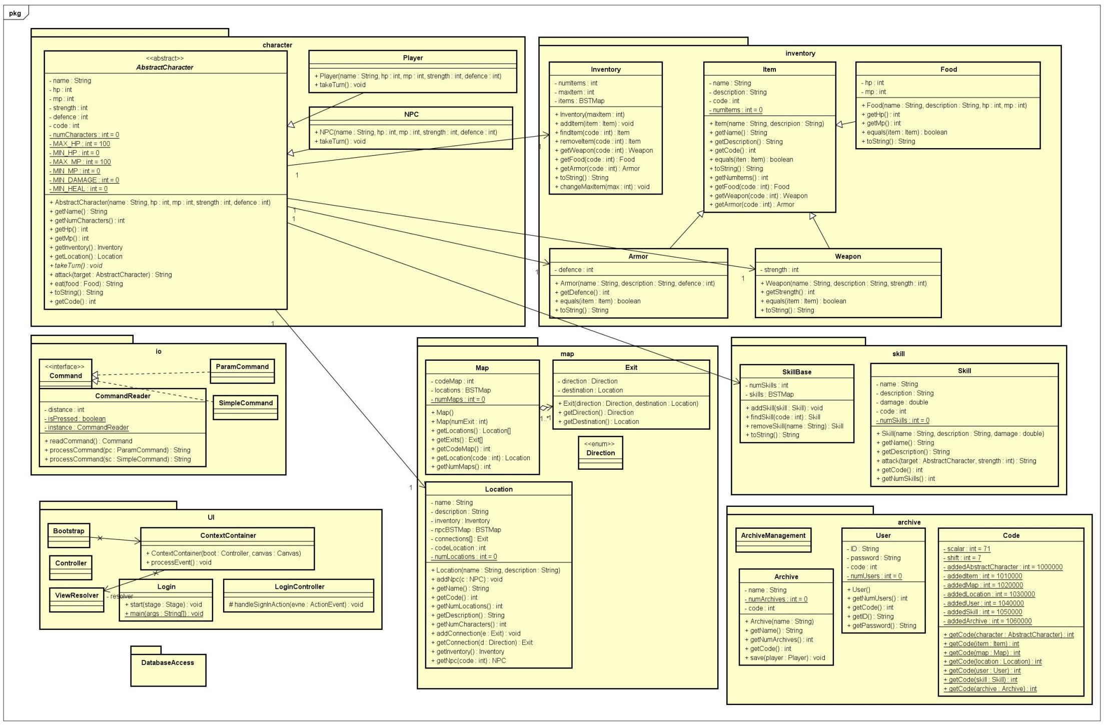
X.2.4.9 Skill interface.

X.2.4.10 Description interface.

X.2.5.11 User interface.

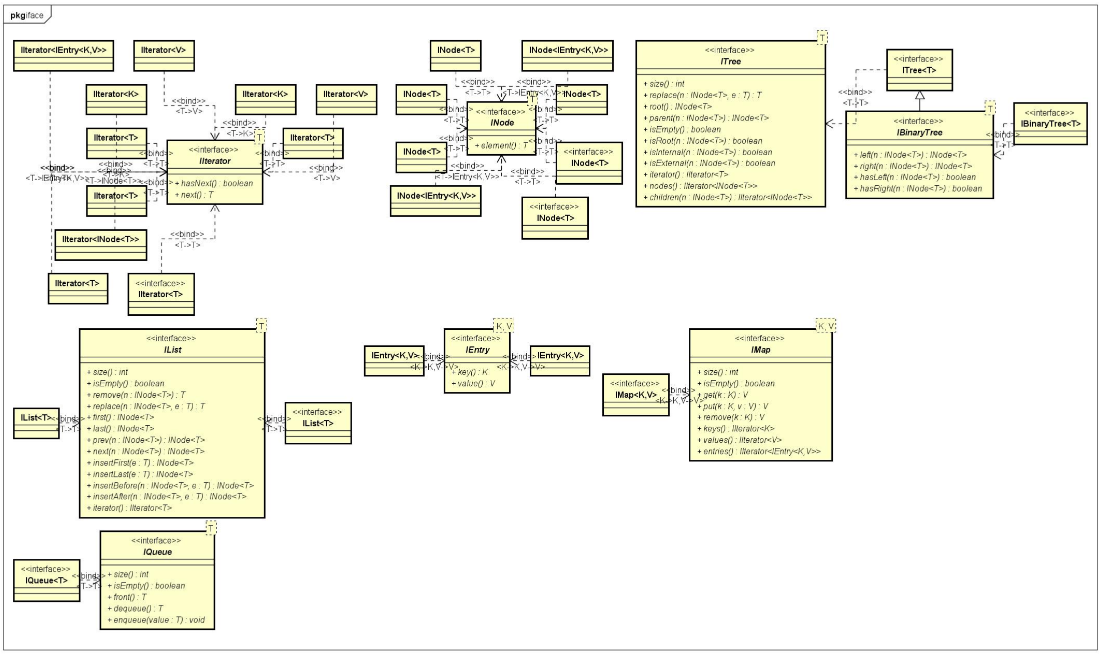
X.2.4 Class Diagram

X.2.4.1 Overall

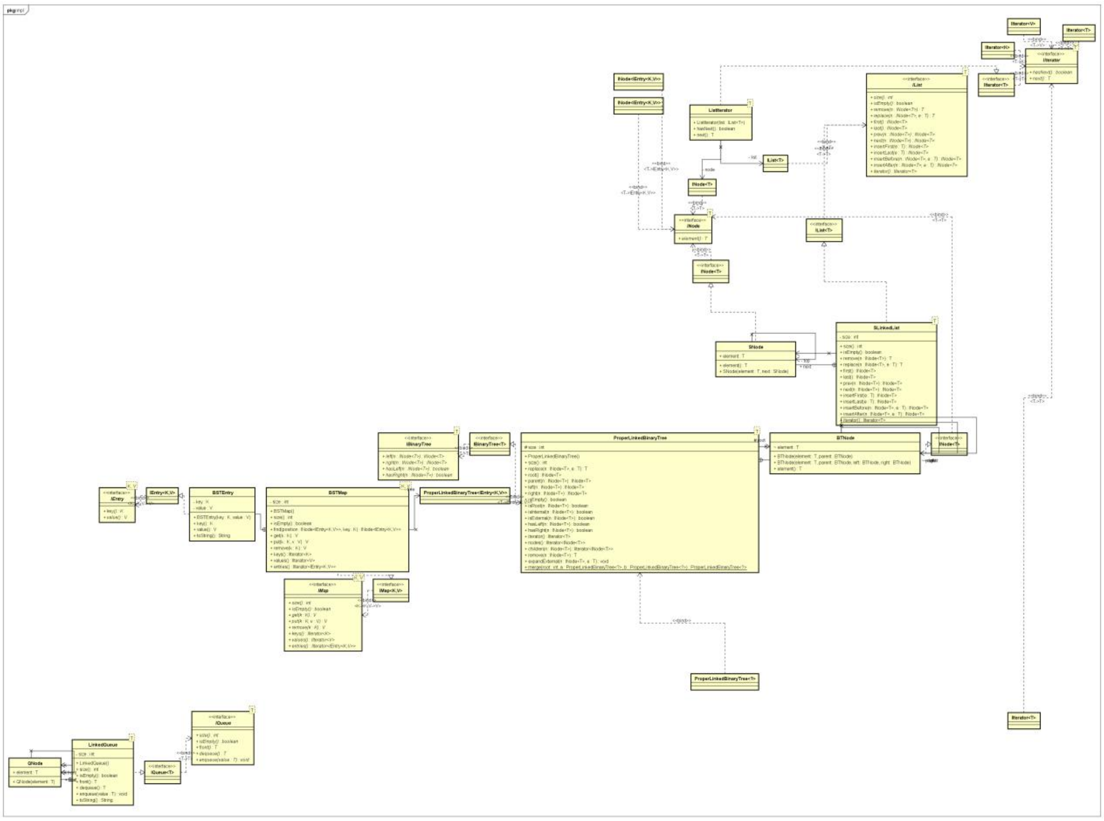


pkg
character
<<abstract>>
AbstractCharacter
- name : String
- hp : int
- mp : int
- strength : int
- defence : int
- code : int
- numCharacters : int = 0
- MAX_HP : int = 100
- MIN_HP : int = 0
- MAX_MP : int = 100
- MIN_MP : int = 0
- MIN_DAMAGE : int = 0
- MIN_HEAL : int = 0
+ AbstractCharacter(name : String, hp : int, mp : int, strength : int, defence : int)
+ getName() : String
+ getNumCharacters() : int
+ getHp() : int
+ getMp() : int
+ getInventory() : Inventory
+ getLocation() : Location
+ takeTurn() : void
+ attack(target : AbstractCharacter) : String
+ eat(food : Food) : String
+ toString() : String
+ getCode() : int
Player
+ Player(name : String, hp : int, mp : int, strength : int, defence : int)
+ takeTurn() : void
NPC
+ NPC(name : String, hp : int, mp : int, strength : int, defence : int)
+ takeTurn() : void
1
inventory
Inventory
- numItems : int
- maxItem : int
- items : BSTMap
+ Inventory(maxItem : int)
+ addItem(item : Item) : void
+ findItem(code : int) : Item
+ removeItem(code : int) : Item
+ getWeapon(code : int) : Weapon
+ getFood(code : int) : Food
+ getArmor(code : int) : Armor
+ toString() : String
+ changeMaxItem(max : int) : void
Item
- name : String
- description : String
- code : int
- numItems : int = 0
+ Item(name : String, descripion : String)
+ getName() : String
+ getDescription() : String
+ getCode() : int
+ equals(iten : Item) : boolean
+ toString() : String
+ getNumItems() : int
+ getFood(code : int) : Food
+ getWeapon(code : int) : Weapon
+ getArmor(code : int) : Armor
Food
- hp : int
- mp : int
+ Food(name : String, description : String, hp : int, mp : int)
+ getHp() : int
+ getMp() : int
+ equals(item : Item) : boolean
+ toString() : String
Armor
- defence : int
+ Armor(name : String, description : String, defence : int)
+ getDefence() : int
+ equals(item : Item) : boolean
+ toString() : String
Weapon
- strength : int
+ Weapon(name : String, description : String, strength : int)
+ getStrength() : int
+ equals(item : Item) : boolean
+ toString() : String
io
<<interface>>
Command
ParamCommand
SimpleCommand
CommandReader
- distance : int
- isPressed : boolean
- instance : CommandReader
+ readCommand() : Command
+ processCommand(pc : ParamCommand) : String
+ processCommand(sc : SimpleCommand) : String
map
Map
- codeMap : int
- locations : BSTMap
- numMaps : int = 0
+ Map()
+ Map(numExit : int)
+ getLocations() : Location[]
+ getExits() : Exit[]
+ getCodeMap() : int
+ getLocation(code : int) : Location
+ getNumMaps() : int
1..*1
Exit
- direction : Direction
- destination : Location
+ Exit(direction : Direction, destination : Location)
+ getDirection() : Direction
+ getDestination() : Location
<<enum>>
Direction
Location
- name : String
- description : String
- inventory : Inventory
- npcBSTMap : BSTMap
- connections[] : Exit
- codeLocation : int
- numLocations : int = 0
+ Location(name : String, description : String)
+ addNpc(c : NPC) : void
+ getName() : String
+ getCode() : int
+ getNumLocations() : int
+ getDescription() : String
+ getNumCharacters() : int
+ addConnection(e : Exit) : void
+ getConnection(d : Direction) : Exit
+ getInventory() : Inventory
+ getNpc(code : int) : NPC
skill
SkillBase
- numSkills : int
- skills : BSTMap
+ addSkill(skill : Skill) : void
+ findSkill(code : int) : Skill
+ removeSkill(name : String) : Skill
+ toString() : String
Skill
- name : String
- description : String
- damage : double
- code : int
- numSkills : int = 0
+ Skill(name : String, description : String, damage : double)
+ getName() : String
+ getDescription() : String
+ attack(target : AbstractCharacter, strength : int) : String
+ getCode() : int
+ getNumSkills() : int
UI
Bootstrap
Controller
ViewResolver
- resolver
ContextContainer
+ ContextContainer(boot : Controller, canvas : Canvas)
+ processEvent() : void
Login
+ start(stage : Stage) : void
+ main(args : String[]) : void
LoginController
# handleSignInAction(evne : ActionEvent) : void
DatabaseAccess
archive
ArchiveManagement
Archive
- name : String
- numArchives : int = 0
- code : int
+ Archive(name : String)
+ getName() : String
+ getNumArchives() : int
+ getCode() : int
+ save(player : Player) : void
User
- ID : String
- password : String
- code : int
- numUsers : int = 0
+ User()
+ getNumUsers() : int
+ getCode() : int
+ getID() : String
+ getPassword() : String
Code
- scalar : int = 71
- shift : int = 7
- addedAbstractCharacter : int = 1000000
- addedItem : int = 1010000
- addedMap : int = 1020000
- addedLocation : int = 1030000
- addedUser : int = 1040000
- addedSkill : int = 1050000
- addedArchive : int = 1060000
+ getCode(character : AbstractCharacter) : int
+ getCode(item : Item) : int
+ getCode(map : Map) : int
+ getCode(location : Location) : int
+ getCode(user : User) : int
+ getCode(skill : Skill) : int
+ getCode(archive : Archive) : int

X.2.4.2 dsa interface



X.2.4.2 dsa implementation



The class diagram would be updated with development of project.

X.3 Work Package

X.3.1 Work Package of Yao Chenzhen(15205941)

PROJECT NAME	**Work package inventory, map, IO, UI, archive, dsa**		
Start Date	Semester 1 Week 2	**Finish Date**	Semester 2 Week 14
Aim / Objective	Completing basic structure of project and detailed code of package inventory, map, IO, UI, archive, dsa, code. **In the final, giving a nice RPG!**		
Work package Leader	Yao Chenzhen(15205941)		

Contributors to this package	Yao Chenzhen(15205941) Chen Jin(15205898), Wang Maozhun(15205930) Dong Yuehui(15206084)
Description / Activities	Task X.1 **Completing UML class diagram due to previous system requirements, so that each member can easily understand every class and what their package should do.** • X.1.1 the overall class diagram is one of the most important part in our project, which could be specified with specific classes, methods and their relations. So I firstly decide which class I need to add and complete them. • X.1.2 entering attributes and methods of these classes and decide the parameters. Task X.2 **Completing data structure and algorithms (dsa) and class code which is to give a code that represents the ID to user, player, NPC, item, map, location, etc.** •X.2.1 the interface include node, entry, list, queue, map, tree, binary tree, iterator. Firstly, complete them with suited way to the project. • X.2.2 implements these algorithms such as list iterator, single linked list, linked queue, proper linked binary tree and give the final: binary search tree by map. The BSTMap is to store object in efficient way. • X.2.3 completing code. The ID plays role in searching. For example, giving the ID of an item, then, searching it in a BSTMap type attribute in inventory and the item would be returned efficiently. Task X.3 **Completing code of package inventory which including class Inventory, Item, Food, Weapon, Armor and package map which include class Map, Location, enum Direction, Exit.** • X.3.1 Completing package inventory. Class Inventory is to store items and is responsible for methods like get item. Class item is to determine basic and same properties of all item and would be extends by Food, Weapon and Armor. • X.3.2 Completing package map. Class Map is to store locations and is responsible for methods like travel. Class location is to store location of object, npcs to fight and scripts. Otherwise, enum direction and class exit can limit bond of locations. Task X.4 **Completing code of package archive which including class Archive, Code, User and User Management which is to manage archives of game.**

	• X.4.1 Completing all of package archive. Class Archive implements the function that going on previous time line so that player can suspend the schedule. Class User is to store information of each user like ID, password and Class User Management is to manage users. Task X.5 **Completing package UI by using JavaFX which providing graphical user interface that including Class Bootstrap, Login, Controller and many css, fxml and jpg files.** • X.5.1 completing class Bootstrap which is to lead all over the game. • X.5.2 completing class Login which is to deal with user's register and taking access to game. • X.5.3 completing classes varieties of controller for handling events. • X.5.4 writing code on fxml and css to give the graphical interface with a favorable version. Task X.6 **Finally, testing games using blend of traditional test and JUnit. As many as possible doing JUnit on each method if we have enough time.**		
Milestones		**Semester**	**Week**
	M X.1 **Alpha**: Completing most tasks and giving a version that could be executed which may be have some bugs but it could be play with fundamental functions.	**2**	**10**
	M X.1 **Beta**: Linking to database like MySql to give a better experience on saving and loading. Optimizing the details of GUI.	**2**	**12**
	M X.1 **Final Release**: Completing tasks on debugging and adding functions needed. **Finally, making game reliable, available and robust.**	**2**	**14**
Deliverables		**Semester**	**Week**
	D X.1 Final Release	**2**	**14**
	D X.1 We used Astah to generate an UML class diagram and developing with it. The asta file and our projects could be found on **https://github.com/CaptainSoul/Soul**	**2**	**14**

X.3.2 Work Package of Chen Jin(15205898)

PROJECT NAME	**Work package Character&Skill**		
Start Date	Semester 2 Week 8	**Finish Date**	Semester 2 Week 16
Aim / Objective	Character Class, including abstract class , NPC class and player class Skill Class, connecting with Characher Class		
Work package Leader	Chen Jin(15205898)		
Contributors to this package	Chen jin (15206898) Wang Maozun (15205930) Dong Yuehui (15206084) Yao Chenzhen (1520541)		
Description / Activities	Task 1.1 The first task is Character Class Implementation in alpha version. The Character part means that both of player characters and enemy characters. The full attributes of these characters will be implemented, and we can experience it just like any other RPG games. This attributes will be changed by player's action during the game. Besides, other actions relied on Character class will be implemented, too. That is, some methods related to other classes done by teammates. But the core of RPG game is character, so in another word, Character part is the most important part in our game. The Skill part means that both of player Task 1.2 The second task is Skill Class Implementation in alpha version. The Character part means that kinds of skill that both of players and enemies can use. This class is based on the character class, so I do it after task 1.1. This class may be used in battle mode, or other events. Task 1.3 The third task is Character Class Testing in alpha version. After all methods have been done in this class, I will focus on its testing code. I will use unit testing to test instance variables and methods of character class. And I will retain the test code for the purpose of further programming. After testing, I will consider whether add or delete some variables and methods or not, and communicate with teammates. Task 1.4 The fourth task is Skill Class Testing in alpha version. After all methods have been done in this class, I will focus on its testing code. I will use unit testing to test instance variables and methods of skill class. And I will retain the test code for the purpose of further programming. After testing, I will consider whether add or delete some variables and methods or not, and communicate with teammates. Task 1.5 The fifth task is Character Class Implementation in beta version.		

	Based on the alpha version, we will consider to add or delete some attributes and methods in this class. Task 1.6 The sixth task is Skill Class Implementation in beta version. Based on the alpha version, we will consider to add or delete some attributes and methods in this class. Task 1.7 The seventh task is Other Classes Implementation in beta version. Other parts just like GUI or SAVE/LOAD system will be implemented in beta version, and can increase the interest of our RPG games. Task 1.8 The eighth task is Character Class Testing in beta version. After all methods have been done in this class, I will focus on its testing code. I will use unit testing to test instance variables and methods of character class. And I will retain the test code for the purpose of further programming. After testing, I will consider whether add or delete some variables and methods or not, and communicate with teammates. Task 1.9 The ninth task is Skill Class Testing in beta version. After all methods have been done in this class, I will focus on its testing code. I will use unit testing to test instance variables and methods of skill class. And I will retain the test code for the purpose of further programming. After testing, I will consider whether add or delete some variables and methods or not, and communicate with teammates. Task 1.10 The final task is Other Classes Testing in beta version. After all methods have been done in these classes, I will focus on its testing code. I will use unit testing to test instance variables and methods of each class. And I will retain the test code for the purpose of further programming. After testing, I will consider whether add or delete some variables and methods or not, and communicate with teammates.		
Milestones		**Semester**	**Week**
	M X.1 **Alpha**: Completing most tasks and giving a version that could be executed which may be have some bugs but it could be play with fundamental functions.	**2**	**10**
	M X.1 **Beta**: Linking to database like MySql to give a better experience on saving and loading. Optimizing the details of GUI.	**2**	**12**
	M X.1 **Final Release**: Completing tasks on debugging and adding functions needed. **Finally, making game reliable, available and robust.**	**2**	**14**
Deliverables		**Semester**	**Week**

	D X.1 Final Release D X.1 We used Astah to generate an UML class diagram and developing with it. The asta file and our projects could be found on **https://github.com/CaptainSoul/Soul**	**2** **2**	**14** **14**

X.3.3 Work Package of Dong YueHui(15206084)

PROJECT NAME	**Work package Inventory&Archive about database**		
Start Date	Semester 2 Week 8	**Finish Date**	Semester 2 Week 16
Aim / Objective	Inventory system, treasure in ground, get and drop items, Save and download players' information		
Work package Leader	Dong Yuehui(15206084)		
Contributors to this package	Dong Yuehui (15206084) Yao Chenzhen (1520541) Chen Jin (15206898) Wang Maozun (15205930)		
Description / Activities	Task X.1 completing array to save the items, completing method to get and drop items (I will finish this task work together with Yao Chenzhen). · X 1.1 We finish the inventory part, this part is about items in players' bag and treasures in the ground. We built an array to save all this tings, players' bag and treasures will have some connect, players will get something from treasures, and drop some items in the ground. That means I will build an array to save items in the ground (I will finish this task work together with Yao Chenzhen). · X 1.2 After save these things, we will make a method to get the name of items and connect to them by their name. By use this method, we can choose things to drop, we call it remove things, also, we need a method called add things. Use this method, we can finish all the function we want in inventory module. Task X.2 completing the connecting between java and MySQL to use the save and load system. · X 2.1About the save and load module, we want to use the knowledge from Database and Information System. We have learned the MySQL connect with java. We will connect them together to save and load the information of players, when this system finish, players will can use it to make sure he can play next time in the same development in the story, instead of start it again from the beginning.		

	Task X.3 completing test to make sure it works normal. When we finish all this job, we will test it by some simple get and drop items, save and load a player information. If we make sure all this module could work successfully, our programming will have base function. This is our Alfa version.		
Milestones		**Semester**	**Week**
	M X.1 **Alpha**: Completing most tasks and giving a version that could be executed which may be have some bugs but it could be play with fundamental functions. M X.1 **Beta**: Linking to database like MySql to give a better experience on saving and loading. Optimizing the details of GUI. M X.1 **Final Release**: Completing tasks on debugging and adding functions needed. **Finally, making game reliable, available and robust.**	**2** **2** **2**	**10** **12** **14**
Deliverables		**Semester**	**Week**
	D X.1 Final Release D X.1 We used Astah to generate an UML class diagram and developing with it. The asta file and our projects could be found on **https://github.com/CaptainSoul/Soul**	**2** **2**	**14** **14**

X.3.4 Work Package of Wang Maozhun(15205930)

PROJECT NAME	**Work package Material**		
Start Date	Semester 1 Week 2	**Finish Date**	Semester 2 Week 14
Aim / Objective	Learn Photoshop, and design the all kinds of pictures in our games.		
Work package Leader	Wang Maozun (15205930)		

Contributors to this package	Wang Maozun (15205930) Yao Chenzhen (1520541) Chen Jin (15206898) Dong Yuehui (15206084)		
Description / Activities	Task 1:I will spend time studying basic photoshop. First, for it's my first time make picture by myself, I need to know how to make a picture. I will learn some basic knowledge by myself through book or internet.I will learn how to use the software and how to make picture through. Then I will learn how to make a picture better through it. Task 2:I will first write a story for our game. Our team will talk about the story toghther, I will write a story record to what they like and what they want. Maybe design some roles named by ourselves , and they will become what we like. The story will contain a main role, it's the most important in the game. All the story will happened with him. And player will control him and play game. We will let the player have lots of choices, they can choose many professions, to increase their playing experience. But whichever profession they choose the story will be only. So in the story the role will not have typical profession feature. Task 3:I will make maps according to the story. First, I need to design a whole map and it include lots of small maps. The small maps are connected, some only connect with another one, some main maps will connect with some maps. In my opinion the big maps will contain several parts, main maps will connect with parts. The maps in parts will be the same. Then I will set NPC in the main maps and also some small maps according the story. Some NPC will not need in the story but it's indispensability, just like shop NPC. Some NPCs is in the story, so what they look like need conform to their character. When I design them, I will think about it. Task 4:The most important is what is the main role like. It must be want it look like in the story. We will make him be a hero in the story, but the character of him will not only is justice, but also some necessary characters just like sometimes he will be angry, he will also entangled. In all he will not be a simple hero, we want him become more abundant. So I will think all about these, and design a image for our main role. Then I will also make him move and do simple active in the game. Also when he change weapons, his action will not change but the weapon will change in his hand, according to his weapon, when he is fighting, he will active different. Task 5:According to the story, I will increase music in maps, also some fighting sound.		
		Semester	**Week**
	M X.1 **Alpha**: Completing most tasks and giving a version that could be executed which may be have some bugs but it could be play with fundamental functions.	**2**	**10**

	M X.1 **Beta**: Linking to database like MySql to give a better experience on saving and loading. Optimizing the details of GUI.	**2**	**12**
	M X.1 **Final Release**: Completing tasks on debugging and adding functions needed. **Finally, making game reliable, available and robust.**	**2**	**14**
Deliverables		**Semester**	**Week**
	D X.1 Final Release	**2**	**14**
	D X.1 We used Astah to draw an UML class diagram and developing with it. The asta file and our projects could be found on **https://github.com/CaptainSoul/Soul**	**2**	**14**

X.4 Current Progress & Lessons

X.4.1 Current Progress

We have completing most package of project: character, map, inventory, skill, dsa.iface, dsa.impl.

We still develop on archive, UI and IO.

X.4.2 Lessons

X.4.2.1 We should familiar with tools like astah. Developing would be more efficient with them.

X.4.2.2 Project Requirements Analysis is very important to do well. If not, we would cost added and more time on it later.

X.4.2.3 After forming a team, we should select fixed time in every week for discussing.

X.4.2.4 Some unsubstantiated requirements might be the most terrible thing rather than that substantiated.

X.4.2.5 Good Communication can facilitate project progress.