Requirements and Analysis Document for the Vanaheim project (RAD)

1 Introduction

1.1 Purpose of application

The purpose of the game is to create a fun environment for the player. By using text to perform things, we hope to help people become better at typing.

1.2 General characteristics of application

The Vanaheim application is a single-player application with a graphical user interface (GUI) for Windows/Mac/Linux platforms.

The player will navigate through a 2-dimensional world. Navigation is made using the arrow keys. When navigating around the world, the player may be blocked by certain objects such as trees, water etc. Every interaction is made by typing commands. For example if the player wishes to equip his axe from inventory he type ”equip crude axe”.
The game won't have an end.

1.3 Scope of application

It is only possible to play the game alone. That is, there's no multiplayer option. There is no option to save a game and there is no saved statistics for player level, quests completed etc.

1.4 Objectives and success criteria of the project

- It should be possible to accept and complete a quest. A quest can for example be to kill five spiders.
- It should be possible to use commands to invoke actions in the game.

1.5 Definitions, acronyms and abbreviations

- GUI, graphical user interface
- FPS, frames per second
- Java, platform independent programming language.
- JRE, Java Runtime Environment. Additional software needed to run a Java application.
- Host, a computer where the game will run.
- NPC, non-player character
- Loot, pick up items from killed enemies.
- Quest, a player accepts a quest, performs it and then finishes it to gain a reward.

2 Requirements

2.1 Functional requirements

The player should be able to:

- Move around the world
- Attack monsters
- Kill monsters
- Loot monsters
- Start game from welcome screen
- End game to welcome screen and quit game
- Equip items
- Unequip items
- Use items
- Receive quest from NPC
- Complete quest
- Be able to change world, for example enter a house

2.2 Non-functional requirements

2.2.1 Usability

- Get response from NPC in a resonable amount of time
- Not too high latency that disturb the player

2.2.2 Reliability

NA

2.2.3 Performance

- FPS at 120. Not more not less. Too much FPS and the game will require too much from the users computer and too less the game will feel laggy. 120 FPS is recommended.

2.2.4 Supportability

NA

2.2.5 Implementation

To achieve platform independence, the application will use the Java environment. We use Slick2D library so we can focus on other implementation details.

2.2.6 Packaging and installation

- JAR-files

2.2.7 Legal

- Only use openSource sprites. Music are made by us.

2.3 Application models

2.3.1 Use case model

See APPENDIX for UML diagram and textual descriptions.

2.3.2 Use cases priority

1. Start game
2. Move around the map
3. Enter houses
4. Exit houses
5. Use simple command tasks
    - 3.1 Open inventory
    - 3.2 Close inventory
    - 3.3 Show questbook
    - 3.4 Hide questbook
    - 3.5 Equip an item
    - 3.6 Unequip item
    - 3.7 Use item
6. Fight monsters
7. Use more advanced command tasks
    - 4.1 Open console window
    - 4.2 Close console window
    - 4.3 Hit monster
    - 4.4 Talk to NPC
    - 4.5 Loot items
8. Win against monster
9. Loose against monster
10. Receive quest from NPC
11. Be rejected by NPC when not completed quest or completed quest
12. Finish quest
13. End game

2.3.3 Analysis Model

See APPENDIX

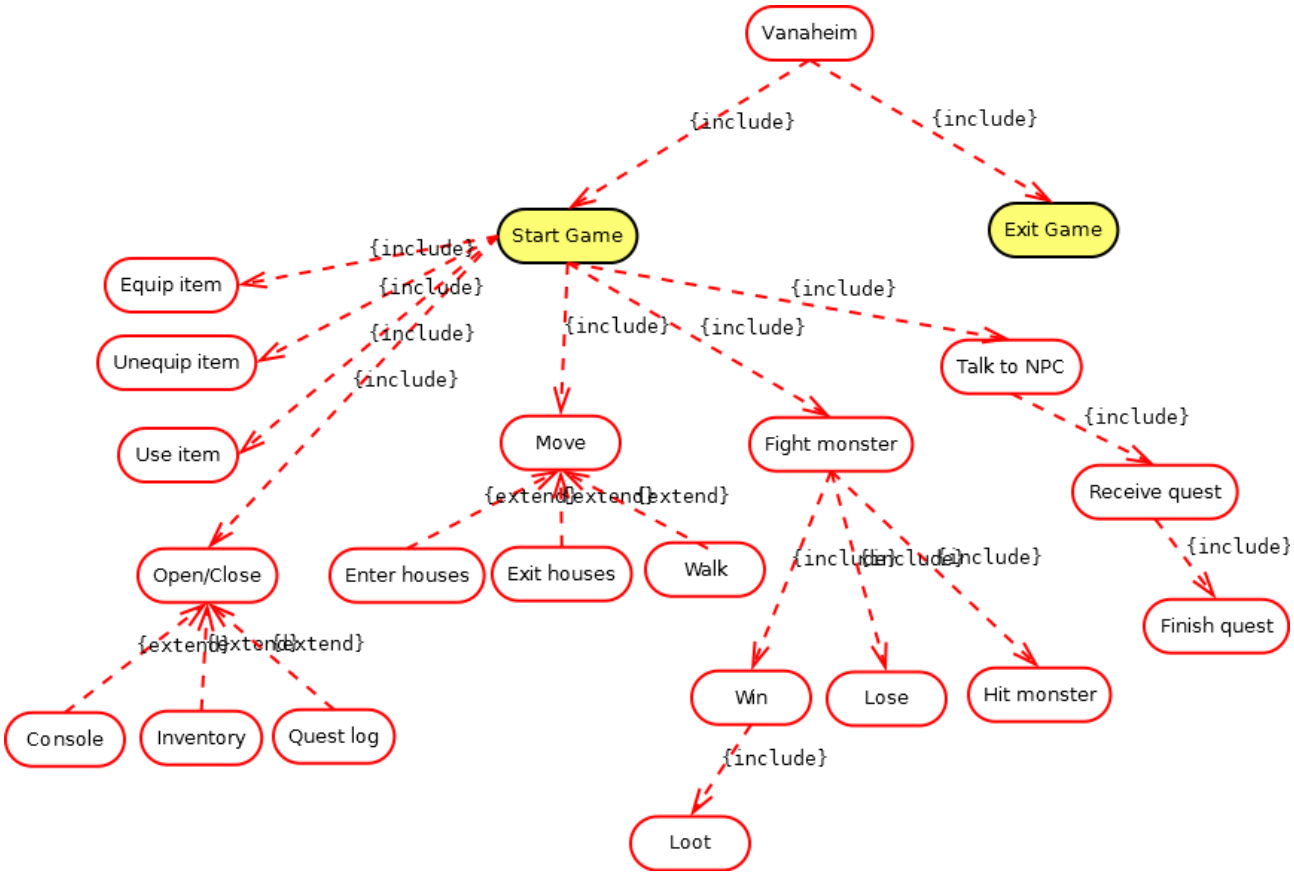
2.3.4 User inteface

The application will use a fixed GUI, always at fullscreen and with size 1024x768. This to ensure a good user experience when playing the game. If too small, the user can have problem to read certain things and therefor lack of game experience.

APPENDIX

Use cases

Overview



Use case texts

Use Case: (Battle)

Summary: Battle with a monster

Priority: mid

Extends: nothing

Includes: Start Game

Participators: Player, Monster

Normal flow of events

	Player	System
1	Enter a battle by walking on ”wrong” grass tile	
2		Start a battle with a monster generated on tile
3	See a battle sceen with player and monster on it	
4	Player need to attack the monster to win the game	Play as the monster and try kill the player by at the end of time hit the player
5		End the game. Loser is the one with no hp left. Drop loot if player win

Alternate flow

Flow 1.1 No monster on tile

	Player	System
1	Player walk on a tile with no monster on it	
2		no battle is generated
3	The player can still walk and remain in same world	

Exceptional flow

	Player	System
1	Player walk on a tile with monster on it	
2		Something goes wrong
3	The player can still walk and remain in same world	

Use Case: (Command)

Summary: Use a command to invoke actions in game

Priority: high

Extends: nothing

Includes: Start Game

Participators: Player

Normal flow of events

	Player	System
1	Type a command in command line	
2		Interpret command
3	Get the command executed	

Alternate flow

Flow 2.1 Given command is not found

	Player	System
1	Type a command in command line	
2		Don't find command
3	Nothing happens	

Flow 2.2 Command not okey at the time

	Player	System
1	Type a command in command line	
2		Command not okey at this point in time
3	Nothing happens	

Exceptional flow

	Player	System
1	Type a command in command line	
2		Something goes wrong
3	Nothing will happen	

Use Case: (Move)

Summary: Moves the character on the map

Priority: high

Extends: nothing

Includes: Start Game

Participators: Player

Normal flow of events

	Player	System
1	Press an arrow -key	
2		move character
3	See the char moving	

Alternate flow

Flow 3.1 Something block the way

	Player	System
1	Press an arrow -key	
2		road is blocked
3	The char stand still	

Flow 3.2 Try to walk through a door

	Player	System
1	Press an arrow -key	
2		try to move into house open a new world with char inside house
3	The char is now inside house	

Exceptional flow

	Player	System
1	Press a arrow -key	
2		something wrong
3	The char stand still	

Use Case: (Quest)

Summary: Receive and finish a quest from NPC

Priority: low

Extends: nothing

Includes: Start Game

Participators: Player, NPC

Normal flow of events

	Player	System
1	Talk to NPC	
2		Give quest to player
3	Player get quest in questbook	

Alternate flow

Flow 4.1 Quest is already taken

	Player	System
1	Talk to NPC	
2		Quest is already taken Repeat quest description for player
3	Player wont get a new quest. Get quest description again	

Flow 4.2 Quest requirements is completed

	Player	System
1	Talk to NPC	
2		Quest requirements is completed. Give reward
3	Player get reward	

Flow 4.3 Quest is completed

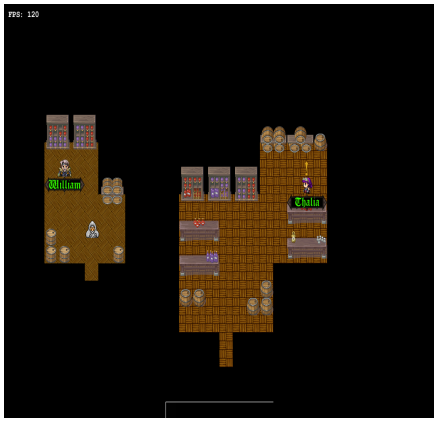
	Player	System
1	Talk to NPC	
2		Quest is completed Give message to player
3	Player get message	

GUI


FPS: 120
Vanaheim
Press 1 to start playing
Press ESC to quit the game


FPS: 120
Spider
Inventory


FPS: 120
Sam


FPS: 120
William
Thalia

Analysis model