**THE DESIGN METHOD IN ENGINEERING**

**Problematic Context**

A group of software engineers faces a challenging project: they want to create a maze game in which the main character or hero must find the exit while being chased by an enemy, this enemy will always know the shortest path to the hero. If the player finds the game too difficult, there will be a hint option, which will offer a help in determining the path or the way to the exit. The goal is to offer an immersive and suspenseful gaming experience. However, the engineers are faced with a dilemma: how to design a dynamic and challenging maze that offers interesting and fair routes for the player and at the same time allows the enemy to pursue them intelligently?

**PHASE 1. IDENTIFYING THE PROBLEM**

**Identification of needs and symptoms:**

**-**For the hint option in the game, the engineers must find valid routes from the player's current position to the exit.

**-**For the game to be interesting, the engineers must plan the enemy's movements, allowing them to pursue the player strategically while avoiding obstacles.

-It is important to determine the shortest routes from the player's current position to various points of interest, such as the exit.

-The group must create different mazes in order for the game to be interesting (not always using the same scenery).

**Problem identification:**

In order to create the videogame, the group of engineers need to solve how to determine shortest paths, distances and travel times within their game.

| **Customer** | Group of programmers |
| --- | --- |
| **User** | Clients of the group |
| **Functional requirements** | · FR 1. Create a maze.  · FR 2. Generate the player.  · FR 3. Generate the enemy.  · FR 4. Show the way to the exit.  · FR 5. Allow the player to move using the keyboard.  · FR 6. Create the graphical user interface. |
| **Context of the problem** | A group wants to create a video game that allows the users to try to escape from a maze while they are being chased |
| **Non-functional requirements** | NFR1: Scalable Software |

| **Name or identifier** | FR 1. Create a maze | | | |
| --- | --- | --- | --- | --- |
| **Summary** | The system must allow the creation of the object/screen maze that can be generated randomly. | | | |
| **Inputs** | **input name** | **Data type** | | **Selection or repetition condition** |
| isDirected | boolean | | Determines if the paths of the created maze could be traveled in a single direction or not |
| start | | Object | It is the point at which the player appears |
| exit | | Object | It is the point that the player must reach without being eliminated by the enemy |
| **General activities needed to obtain the results** | · Receive the attributes of the maze. | | | |
| **Result or postcondition** | The maze was successfully created | | | |
| **Outputs** | **Output name** | | **Data type** | **Selection or repetition condition** |
| errorMssg | String | | Show was an error during the creation of the maze. |

| **Name or identifier** | FR 2. Generate player | | | |
| --- | --- | --- | --- | --- |
| **Summary** | The system must allow the automatic generation of the player once the user clicked the “play” button | | | |
| **Inputs** | **input name** | **Data type** | | **Selection or repetition condition** |
| name | String | | Since there is only one player, this is added with their name |
| **General activities needed to obtain the results** | · Verify that the maze has been previously created.  · Create the player and place them at the starting point of the maze. | | | |
| **Result or postcondition** | The player was successfully generating | | | |
| **Outputs** | **Output name** | | **Data type** | **Selection or repetition condition** |
| errorMssg | String | | Show it was an error during the creation of the player. |

| **Name or identifier** | FR 3. Generate an enemy | | | |
| --- | --- | --- | --- | --- |
| **Summary** | The system must allow the automatic generation of the enemy once the user clicked the “play” button | | | |
| **Inputs** | **input name** | **Data type** | | **Selection or repetition condition** |
| position | String | | Since there is only one player, this is added with their name |
| velocity | | double | This attribute determines how fast the enemy will move. |
| **General activities needed to obtain the results** | · Verify that the maze has been previously created.  · Create the enemy and place them at some point of the maze. | | | |
| **Result or postcondition** | The enemy was successfully generating | | | |
| **Outputs** | **Output name** | | **Data type** | **Selection or repetition condition** |
| errorMssg | String | | Show it was an error during the creation of the enemy. |

| **Name or identifier** | FR 4. Show the way to the exit | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| **Summary** | The system must allow the view of the path that leads from the player's position to the exit of the maze | | | | | |
| **Inputs** | **input name** | **Data type** | | | **Selection or repetition condition** | |
| positionPlayer | Object | | | It’s needed to know the player's position to show the path to the exit. | |
| exit | Object | | | It’s needed to know the exit point of the maze to be able to show a path from the player to this point | |
| **General activities needed to obtain the results** | · Verify that there is a path from where the player is to the exit of the maze.  · Increase the speed of movement of the enemy | | | | | |
| **Result or postcondition** | The path to the exit is show | | | | | |
| **Outputs** | **Output name** | | | **Data type** | | **Selection or repetition condition** |
| msg | | String | | | Show the steps from the player to the exit |

| **Name or identifier** | FR 5. Allow the player to move using the keyboard | | | |
| --- | --- | --- | --- | --- |
| **Summary** | This feature enables the player to control their in-game character's movement using the keyboard. | | | |
| **Inputs** | **input name** | **Data type** | | **Selection or repetition condition** |
| keyPress | String | | Whenever the player presses a key |
| **General activities needed to obtain the results** | * Continuously monitor keyboard input from the user. * Validate which key was selected by the user to know in which direction their character will move | | | |
| **Result or postcondition** | The player moves in the corresponding direction | | | |
| **Outputs** | **Output name** | | **Data type** | **Selection or repetition condition** |
|  |  | |  |

| **Name or identifier** | FR 6. Create the graphical user interface | | | |
| --- | --- | --- | --- | --- |
| **Summary** | The video game must be playable using a graphical user interface, it will be the way the player will interact with the system. This way, the user will be able to click buttons, introduce data in data fields, select elements from a list, and other kind of behaviors related to RPG games. | | | |
| **Inputs** | **input name** | **Data type** | | **Selection or repetition condition** |
| User input | Action | | Whenever the player make a action like use the keyboard or click |
| **General activities needed to obtain the results** | * Monitor the user information extracted from the graphical user interface. | | | |
| **Result or postcondition** | The system will receive the user input information through the graphical user interface. | | | |
| **Outputs** | **Output name** | | **Data type** | **Selection or repetition condition** |
| screen |  | |  |
| messages |  | |  |