**Module « Windows development »**

**-**

**Project « EpiBubble »**

**Software Architecture Specifications**

**(SAS)**

Table of contents

[Introduction 3](#_Toc483407758)

[1. Project context 3](#_Toc483407759)

[2. Global architecture 3](#_Toc483407760)

[3. Component description 3](#_Toc483407761)

[a. Component 1 3](#_Toc483407762)

[b. Component 2 3](#_Toc483407763)

[c. Component 3 3](#_Toc483407764)

[4. Traceability matrix 3](#_Toc483407765)

# Introduction

The aim of this software architecture specification (SAS) is to present the technical elements necessary for the EpiBubble project.

# Project context

The project consists of creating a bubbler shooter game. The goal of the game is to clear the playing field by forming groups of three or more like-coloured bubbles. The game ends when the balls reach the bottom line of the screen. The more balls destroyed in one shot, the more points scored. A player wins when there are no balls remaining on the playing field.

# Global architecture

Program

Input

Board

# Component description

## Program

The class Program only has the Main() function in it. It will create an Input class, to catch all user’s inputs, and a Board class where the game is hosted.

## Input

The class Input only has the Read() function in it. It catches all user’s inputs to throw the information in the Board class. It will simulate a bubble to each input before the user presses Enter to visualize the shot.

## Board

The class Board host the game with the board, the bubbles and the shooting and destroying part.

The Build() function is called at the creation of a Board to initiate all variables and values.

Copy() copies the current Board into another one and send it as a return value.

Loose() will be call after a shoot if the player has lost.

Shoot() is the function called by the Input class when a user has pressed enter. An angle is sent to it and start calculating the endpoint of the bubble.

Destroy() is called by Shoot() to destroy any bubbles if needed.

AddLine() put down all the other bubble by x-1 and then add a new line in top with some random bubbles.

And DisplayBoard() displays the board in the console.

## Config

# Traceability matrix

This matrix makes the correspondence between components, classes, functions, and requirements developed in the request for proposal:

|  |  |  |  |
| --- | --- | --- | --- |
| **Id requirement** | **Requirement description** | **Component** | **Function / action** |
| *REQ\_FUNC\_010* | There are by default 6 different bubble colours. | *Program* | *Enum Color* |
| *REQ\_FUNC\_020* | The screen contains by default 15 rows. | Board | Variable YSIZE |
| *REQ\_FUNC\_030* | Each row can contain by default at most 17 bubbles. | Board | Variable XSIZE |
| *REQ\_FUNC\_040* | Rows are indexed from 0 to (row\_count – 1) where row\_count is the row count in the game. | Board | Variable board[x, y] |
| *REQ\_FUNC\_050* | Row 0 is the row located on the top of the grid. | Board | Variable board[x, y] |
| *REQ\_FUNC\_060* | Bubbles on rows R and R+1 are disposed in staggered rows | Board | Variable board[x, y] |
| *REQ\_FUNC\_070* | When user groups three same colour bubbles, or more, those bubbles are destroyed. | Board | Function Destroy() |
| *REQ\_FUNC\_075* | When launch, the bubble can have attached other’s | Board | Function Shoot() |
| *REQ\_FUNC\_080* | If a bubble is attached to only one group and this group is destroyed, this bubble is destroyed too. | Board | Function Destroy() |
| *REQ\_FUNC\_085* | When bubble cannot attached other’s, the bubble is attached on row 0 | Board | Function Shoot() |
| *REQ\_FUNC\_090* | By default, each 6 shots, a new row is inserted in row 0. | Board | Function Shoot()  Variable Shoots + MaxShoot |
| *REQ\_FUNC\_100* | When a row is inserted, the row is fully filled by bubbles with random colour | Board | Function AddLine() |
| *REQ\_FUNC\_110* | When a row is inserted, if three bubbles or more have the same colour and are grouped, they must not be destroyed until the user shot them. | Board | Destroy() is called after Shoot() |
| *REQ\_FUNC\_120* | For each bubble destroyed with another group, the score is increased with 15 points. |  |  |
| *REQ\_FUNC\_130* | Destroying a group of N same coloured bubbles increases the score of (15.(N1)) points |  |  |
| *REQ\_FUNC\_140* | The gamer can see what is the colour of its next bubble. | Board | Function DisplayBoard()  Variable NextBubble |
| *REQ\_FUNC\_150* | An arrow must indicate the user the direction of his shot |  |  |
| *REQ\_FUNC\_160* | Bubble can bounce on left and right sides of the screen context |  |  |
| *REQ\_FUNC\_170* | When the application is started, a window is displayed to allow user to start a new game or to continue its previous game if available | Input | Function Menu() |
| *REQ\_FUNC\_180* | If a new game is selected, a level window is displayed to allow user to select the level of the game | Input | Function Menu() |
| *REQ\_FUNC\_190* | Three levels are available for user: easy, classic, hard. | Input | Function Menu() |
| *REQ\_FUNC\_200* | On the game screen, three buttons are available: Save, Quit and Setup. | Input | Function Read() |
| *REQ\_FUNC\_210* | When save is selected, a window is displayed to ask user name of this game. |  |  |
| *REQ\_FUNC\_220* | If the game is saved by the user, a file save.txt must be stored in a save directory in the game directory installation. | Input | Function Save() |
| *REQ\_FUNC\_230* | When a game is saved, parameters of the current game must be saved. | Input | Function Save() |
| *REQ\_FUNC\_240* | When a new game is selected, default parameters must be restored. | Input | Function OldGame() |
| *REQ\_FUNC\_250* | When quit is selected, a window is displayed to ask user if he really wants to quit the game. | Input | Function Read() |
| *REQ\_FUNC\_260* | When setup is selected, a window is displayed to modify user parameters. |  |  |
| *REQ\_FUNC\_270* | In the setup window, shot number between two row insertions of each levels can be modified. |  |  |
| *REQ\_FUNC\_280* | In the setup window, bubble and arrow colours can be modified. |  |  |
| *REQ\_FUNC\_290* | In the setup window, speed shot bubble can be modified to low, normal and fast. |  |  |
| *REQ\_FUNC\_300* | By default, speed shot bubble is:  • 80 pixels/second for low speed  • 150 pixels/second for normal speed  • 300 pixels/second for fast speed |  |  |
| *REQ\_FUNC\_310* | By default, shot number between two row insertions is:  • 10 for easy level  • 6 for classic level  • 3 for hard level | Config | In file |
| *REQ\_FUNC\_320* | The bubble colours must be selected among a selection of those colours:  • Black #000000  • Silver #C0C0C0  • Grey #808080  • White #FFFFFF  • Maroon #800000  • Red #FF0000  • Purple #800080  • Fuchsia #FF00FF  • Green #008000  • Lime #00FF00  • Yellow #FFFF00  • Blue #0000F0  • Cyan #00FFFF |  |  |
| *REQ\_FUNC\_330* | The six colours which are selected by default are:  • Blue  • Red  • Cyan  • Yellow  • Fuchsia  • Lime | Board | Enum Color  Function DisplayBoard() |
| *REQ\_FUNC\_340* | The arrow colour must be selected among a selection of those colours, which can be modified in a configuration file:  • Astral #327DA0  • Rose #F64A8A  • Olive #808000  • Grey #808080  • Maroon #800000  • Gold #FFD700 |  |  |
| *REQ\_FUNC\_350* | The arrow colour selected by default is Astral |  |  |
| *REQ\_FUNC\_360* | On PC, the arrow can be managed with the mouse or the arrows on the keyboard. |  |  |
| *REQ\_FUNC\_370* | On PC, if the arrow is managed by the mouse, the bubble is shot when the user clicks |  |  |
| *REQ\_FUNC\_380* | On PC, if the arrow is managed by the arrows on the keyboard, the bubble is shot when the user activates the space key on the keyboard. |  |  |
| *REQ\_FUNC\_390* | On Xbox, the arrow must be managed with left and right buttons on the joystick. |  |  |
| *REQ\_FUNC\_400* | On Xbox, the bubble is shot when user activates the A button (green button) on the joystick. |  |  |
| *REQ\_FUNC\_410* | On Xbox, user can access to buttons by activating the Y button (yellow button) on the joystick and navigate between them with left and right buttons |  |  |
| *REQ\_FUNC\_420* | On Xbox, to get back to the game, user must activate the Y button (yellow button). |  |  |
| *REQ\_FUNC\_430* | On a touch-screen device, the arrow must be managed by touch: touching the arrow and positioning it to the appropriate position. |  |  |
| *REQ\_FUNC\_440* | On a touch-screen device, the bubble is shot when user press a “shot” button |  |  |
| *REQ\_FUNC\_450* | The game can be over when no bubble remains |  |  |
| *REQ\_FUNC\_460* | The game can be over when the bubble is attach over the map. | Board | Function Shoot() |

|  |  |  |  |
| --- | --- | --- | --- |
| REQ\_DATA\_010 | Default count of bubble colour must be modifiable in a configuration file. |  |  |
| REQ\_DATA\_020 | Default row count must be modifiable in a configuration file |  |  |
| REQ\_DATA\_030 | Default maximum count of bubbles in a row must be modifiable in a configuration file. |  |  |
| REQ\_DATA\_040 | The number of same colour bubbles required to destroy a group must be modifiable in a configuration file. |  |  |
| REQ\_DATA\_050 | The number of shots required to insert a line must be modifiable in a configuration file |  |  |
| REQ\_DATA\_060 | Bubble colours available must be modifiable in a configuration file |  |  |
| REQ\_DATA\_070 | Arrow colours available must be modifiable in a configuration file |  |  |
| REQ\_DATA\_080 | Default shot number between two row insertions for easy, classic and hard levels must be modifiable in a configuration file |  |  |
| REQ\_DATA\_090 | Default speed shot bubble must be modifiable in a configuration file |  |  |