



Bilkent University

Department of Computer  
Engineering

# Object-Oriented Software Engineering Course Project

*CS319 Course Project: Virion*

## Analysis Report (Parts 1-4, 5.1)

Mert INAN, Ulugbek IRMATOV, Irmak YILMAZ

Supervisor: Uğur DOĞRUSÖZ

Analysis Report (Parts 1-4, 5.1)

Oct 15, 2016

This report is submitted to GitHub in partial fulfillment of the requirements of the Object-Oriented Software Engineering course CS319.

# Contents

<b>1</b>	<b><i>Introduction</i></b>	<b>1</b>
1.1	Problem	1
1.2	Objectives	1
1.3	Target audience and environment	1
<b>2</b>	<b><i>Overview</i></b>	<b>2</b>
2.1	Virus Actions	2
2.2	Cell Actions	2
2.3	Organelles	2
2.4	Viruses	4
2.5	Proteins	5
2.6	Other structures	6
2.7	Scoring	6
2.8	Game Control	6
<b>3</b>	<b><i>Functional Requirements</i></b>	<b>7</b>
<b>4</b>	<b><i>Nonfunctional Requirements</i></b>	<b>7</b>
4.1	Performance	7
4.2	Usability	7
4.3	Pseudo Functional Requirements	7
<b>5</b>	<b><i>System Models</i></b>	<b>8</b>
5.1	Use Case Model	8
5.1.1	Visionary Real-Life Scenarios	9
5.1.2	Use Case Descriptions	10

# Analysis Report (Parts 1-4, 5.1)

*CS319 Course Project: Virion*

## 1 Introduction

Virion is an educational game where player tries to save a cell from virus attacks. Player can defend using proteins and other cell organelles such as mitochondria, vacuole, and cell wall. There are different kinds of viruses. Difficulty of the game is proportional to time. As time passes more dangerous viruses start coming. To get high score, player must keep cell alive as long as possible.

### 1.1 Problem

There are no educational games related to biology which specifically teaches the battle between a virus and a cell from both internal and external points of view. The quality of graphics of similar existing games is very low. They are not much fun to play either. Our team is planning to bring solution to this problem.

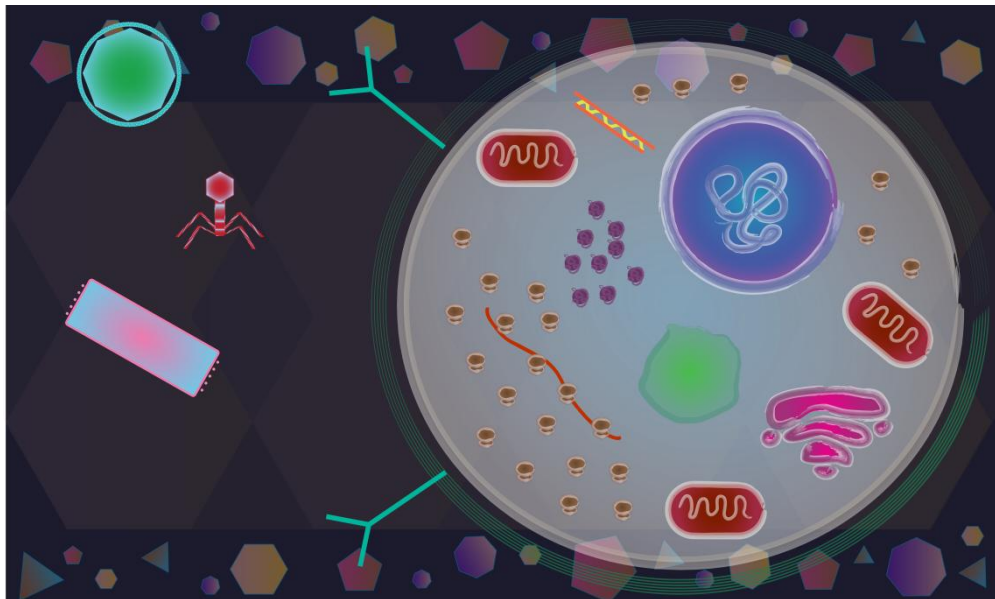
### 1.2 Objectives

We plan to create a game that teaches basic biological science in a fun way. We will make the graphics high quality as well to let the player retain the information learned.

### 1.3 Target audience and environment

- The target audience of the Virion game is students and individuals who are interested in learning about viruses.
- Virion game would be a jar file which can run on most operating systems such as Windows, macOS and Ubuntu.

This report contains an overview of the game. It also includes functional requirements, non-functional requirements, use case descriptions, and a use case diagram.



*Figure 1 This is a possible gameplay with the cell and the viruses*

## 2 Overview

Virion is a time-based strategic Biology game, in which the player attacks viruses of different strengths using an animal cell's defense mechanisms. The purpose of the game is to defeat as many viruses as possible before the eventual infection of the cell occurs. The game has no time limit, yet it is for player's own benefit to finish early. The player tries to produce specific proteins to attack at different stages of the infection using different organelles, while the virus tries to enter the cell and attach itself to the DNA of the cell and reproduce.

### 2.1 Virus Actions

Viruses have three main aims in the game:

1. Attach to the cell and inject its viral DNA into the cell
2. Reach the nucleus and attach itself to the host cell's DNA
3. Reproduce

All of these aims are consecutive phases. Viruses will move according to predetermined speeds and can enter to the game pane from any edge.

### 2.2 Cell Actions

There are three different types of defenses against the viruses:

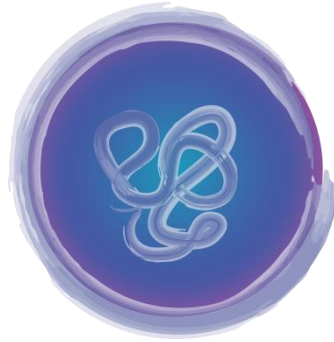
- preventing it from attaching,
- trying to destroy the viral DNA after the attachment,
- trying to halt the reproduction of more viruses after the infection.

Player can prevent the attachment of the virus by building a temporary wall around the cell, or change the structure of the attachment field so that the virus detaches. After the viral DNA is injected into the cell, the player can try to destroy it by using attacker proteins. After the infection occurs, there will be 5 minutes of "emergency state" in which the player can either try to save the cell by finding the viral DNA in the cell DNA and stop the production of more viruses to return to the normal state, or by sacrificing the cell itself and gaining bonus points. Cell moves very slowly inside the game pane.

### 2.3 Organelles

There are five different organelles each of which has important functionalities in protein production. Organelles move very slowly inside the cell.

- **Nucleus:** It is the main control center for protein production. Proteins to be produced are selected by clicking on nucleus. Nucleus communicates with the ribosome using an mRNA. The production time of mRNA is calculated by:  
 $0.1 * (\text{number of proteins}) * (\text{protein type coefficient})$  in seconds.



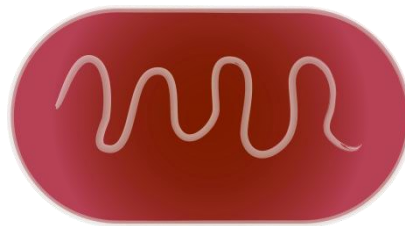
*Figure 2 Illustration of the nucleus*

- **Ribosome:** It is the production organelle of all the proteins. Ribosomes are selected by clicking and dragging. One ribosome can create one protein at a time. Production time of each protein is calculated by:  $10 * (\text{protein type coefficient})$  in seconds.



*Figure 3 Illustration of a ribosome*

- **Mitochondrion:** This organelle is the main producer of energy molecules called ATP. From the start of the game it produces 1000 ATPs per minute. Each production in different organelles requires ATPs according to their production time which is calculated by,  $10 * (\text{production time})$ .



*Figure 4 Illustration of a mitochondrion*

- **Golgi:** This organelle is responsible for turning proteins into Mega Proteins. The production time is calculated by,  $30 * (\text{protein type coefficients})$  in seconds.



*Figure 5 Illustration of Golgi*

- **Vacuole:** It is the storage organelle. Contains the Core Molecules to produce proteins, and mRNAs. Every production requires Core Molecules. At the start of the game it contains 10,000 molecules. Every minute, 1,000 molecules are added automatically. The required Core Molecules for a production is calculated by,  $100 * (\text{production time})$ .

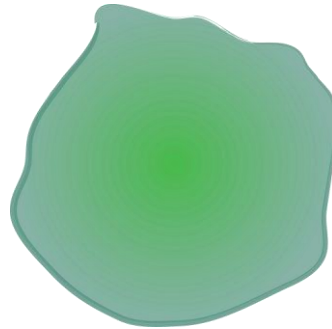


Figure 6 Illustration of vacuole

## 2.4 Viruses

Different virus types have different properties of speed and durability which affect the time to defeat the virus by the cell. After infecting the cell by attaching itself to the DNA of the cell, it starts using the resources of the cell itself. The durability of the virus is calculated by,  $5 * (\text{virus type coefficient})$  seconds of contact with proteins in the first phase,  $75 * (\text{virus type coefficient})$  in the second phase and  $20 * (\text{virus type coefficients})$  in the third phase. The core molecule usage is calculated by  $100 * (\text{virus type coefficient})$  per minute in the DNA attachment phase and double of that amount is used in the reproduction phase; and the ATP usage is calculated by  $200 * (\text{virus type coefficient})$  per minute in DNA attachment phase, and  $300 * (\text{virus type coefficient})$  per minute in the reproduction phase. There are five types of viruses:

- Helical Virus: Virus type coefficient is 2, has medium speed.
- Polyhedral Virus: Virus type coefficient is 4, has the second highest speed.
- Spherical Virus: Virus type coefficient is 3, has the highest speed.
- Complex Virus: Virus type coefficient is 1, has the second slowest speed.
- Virion: Virus type coefficient is 5. Has the slowest speed.

The frequency of viruses is determined by the function,  $f(t) = 16t^2 + t + 16$ . Hence, the number of viruses increases in quadratic form as time,  $t$ , increases, making the game harder. Apart from the frequency, the strength of each virus changes according to the function,  $g(n) = e^n$  where  $n$  is the number of viruses killed by the cell. This means that each virus' type coefficient increases by  $(\text{original virus type coefficient}) + e^n$ . The entry point of the virus to the game area is randomly found by the system.

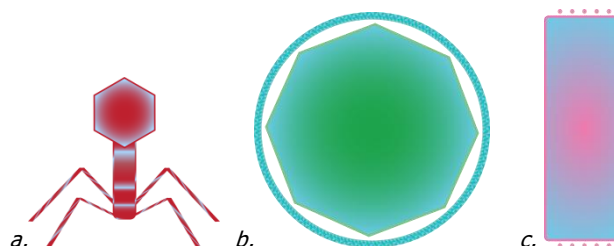


Figure 7 Illustrations of a. complex virus, b. polyhedral virus, and c. helical virus

## 2.5 Proteins

Proteins are the main weapons of the cell for protection. They are produced by ribosomes and modified by the Golgi. Proteins automatically denature and become unusable after a certain amount of time or effort. This denaturation time is calculated by,  $90 \times (\text{protein type coefficient})$  seconds. Its denaturation effort is calculated by  $20 \times (\text{protein type coefficient})$  seconds of contact with virus. Each protein has the shape as in Figure 8.a. Different types of proteins have different colors and Mega Proteins have a shape like in Figure 9.b. The types of proteins are:

- Wall Builder: Used to build a wall around the cell. Protein type coefficient is 5.
- Receptor: Used to detach the protein from its attachment point. Coefficient is 4. Requires 50 more Core Molecules on top of regular production.
- Viral DNA Attacker: Used to attack the viral DNA inside the cell. Coefficient is 1.
- Viral DNA Finder: Used to find and rip off the viral DNA inside the host DNA. Protein type coefficient is 6.
- Virus Reconstruction Stopper: Used to stop the reproduction of virus after infection. Protein type coefficient is 2.
- Antibody: Used to show the viral DNA after infection to gain bonus points. Protein type coefficient is 3. Requires 100 more Core Molecules on top of regular production costs.
- Mega Proteins: They are the enhanced versions of proteins. Mega Protein type coefficient is the double of the regular protein coefficient. Its denaturation time is 30 seconds more, and its denaturation effort is 10 contacts more in addition to the regular calculations.

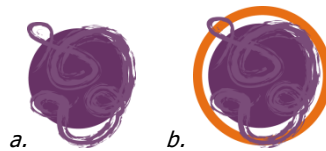


Figure 8 Illustrations of a. regular protein, and b. mega protein

## 2.6 Other structures

- mRNA: Provides the messaging between the Nucleus and Ribosome
- ATP: Is the main energy source for all procedures.
- Viral DNA: Part of the virus that gets injected into the cell.
- Core Molecules: Required building blocks for all proteins and mRNA.
- Cell wall: A temporary protection from attachment of viruses to the cell. If done by a regular wall builder protein, it lasts for 2 minutes, if a Mega Wall Builder is used, it lasts for 3 minutes. Requires 200 additional Core Molecules.

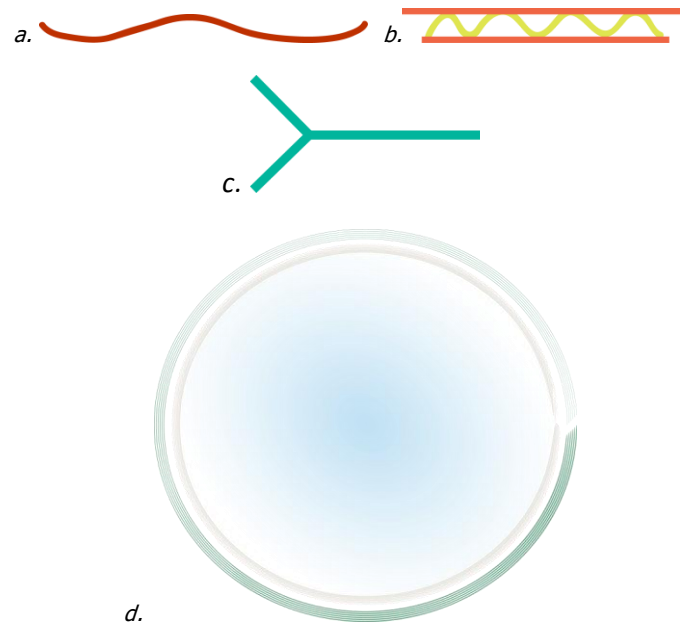


Figure 9 Illustrations of a. mRNA, b. viral DNA, c. antibody, and d. cell wall

## 2.7 Scoring

Scoring of the game is done by multiplying the number of viruses killed with bonus points according to virus type coefficients, then by subtracting the completion time and number of used resources and energy. Highest five scores are entered to the high score table.

## 2.8 Game Control

The user will be able to control the game just by using the mouse. Some functions that the game will provide are music toggling, scientific information pane, viewing high score tables, and pausing and continuing the game, which will be described in detail in functional requirements section.



### **3 Functional Requirements**

- The user will be able to interact with the cell and viruses by clicking and in some instances by clicking and dragging.
- The system will provide an information pane in which scientific information about viral infections and cell's protection methods will be presented to the user.
- The system will guide the user with a help functionality, which would show the essentials of the gameplay.
- The user can pause and resume the game.
- The user will be able to view the top five high score list.
- The user should be able to toggle the music on and off.
- The user should be able to see their score and time.

### **4 Nonfunctional Requirements**

#### **4.1 Performance**

Since the game consists of high quality graphics, in order not to affect the speed of the game, some measures should be taken such as:

- Animations should be kept to the minimum. If required in certain cases such as in destruction of viruses, clashes of viruses and organelles, and the movement of organelles, the animation either should be replaced with a static picture or should be very subtle.
- Vector formatting of the images should be used to reduce the picture size.
- The opening of the game should take at most 10 seconds. Switching to the pause game mode should take at most 1 second. Clicking on an organelle should take less than half a second.

#### **4.2 Usability**

- Provided information to explain the viral infections, functions of organelles, and aim of the game should be scientific.
- Since Virion is a strategic and educational game, it should catch the interest of the users. Therefore, lots of high quality graphical designs should be in the game.
- The organelles, cell and the viruses should be all designed by the game developers.
- The movement of the organelles and the viruses, the colors and the graphics should be smooth and good looking.
- The level of the expertise of the user should be beginner. The documentation about the programs used in implementation of the game should be provided to users.

#### **4.3 Pseudo Functional Requirements**

- It will be implemented in Java.
- The graphics will be designed in Adobe© Photoshop and Illustrator.

## 5 System Models

### 5.1 Use Case Model

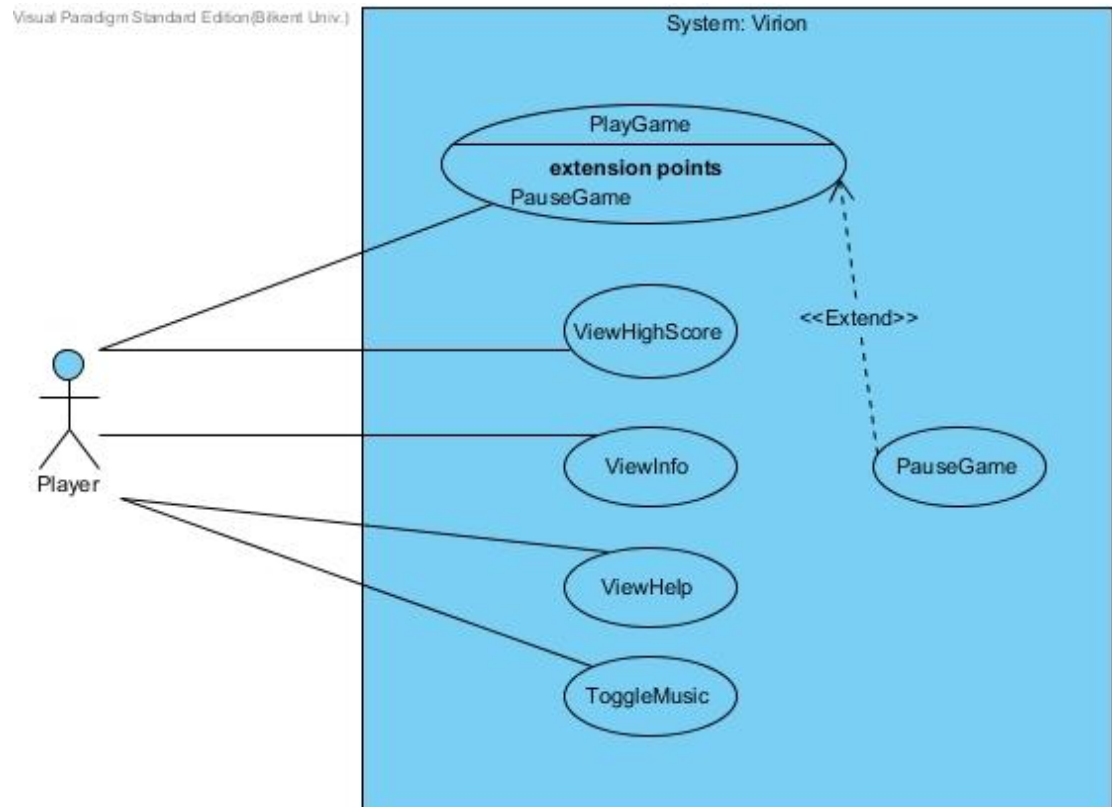


Figure 10 This figure shows the UML Use-Case Model of Virion

### 5.1.1 Visionary Real-Life Scenarios

<i>Scenario name</i>	understandViralInfections
<i>Participating actor instances</i>	alice: Player
<i>Flow of events</i>	<ol style="list-style-type: none"><li>1. While studying for her high school Biology examinations, Alice notices that she cannot totally grasp the concept of viral infections.</li><li>2. Alice then opens Virion to learn more about the interactions between a host cell and a virus.</li><li>3. Using the information pane, she learns different phases of viral infection.</li><li>4. Using this information, she plays the game and understands how cells fight against viruses.</li><li>5. She then applies her knowledge to her studies and examinations.</li></ol>

<i>Scenario name</i>	beginnerGameplay
<i>Participating actor instances</i>	bob: Player
<i>Flow of events</i>	<ol style="list-style-type: none"><li>1. Bob starts up Virion and as a beginner of the game, wants to learn the basics of how to play it.</li><li>2. Using the "View Help" option at the start of the game, he learns how to control the protein production inside the cell.</li><li>3. While playing the game, he forgets how many molecules are required to produce a certain protein. Then, pauses the game and opens the help screen to recall the number.</li><li>4. Bob continues to play the game until the strongest virus, "Virion", infects the cell. He tries to save the cell; however, he chooses to sacrifice the cell as the best strategy in his game.</li><li>5. He quickly tries to produce "Antibody", and the cell dies right after he manages to create it.</li><li>6. As this is Bob's first time of playing the game, his score is entered to the high score list.</li><li>7. Bob plays again to boost his high score.</li></ol>

### 5.1.2 Use Case Descriptions

<i>Use case name</i>	PlayGame
<i>Participating Actors</i>	Player
<i>Entry Condition</i>	Player has opened the game.
<i>Exit Condition</i>	Player's cell gets infected and dies and the surrounding cells cannot be protected by the Player.
<i>Main Flow of Events</i>	<ol style="list-style-type: none"> <li>1. The Player starts the game.</li> <li>2. The Virion System generates a cell with random amounts of organelles, then sends in a virus to attack the cell.</li> <li>3. The Player tries to defeat the virus by attacking its DNA using proteins.</li> <li>4. The Virion System sends in more viruses with increasing strength and frequency until the Player is no longer able to protect the cell. After a threshold of viruses is reached, the Virion System puts the cell into post-infection stage.</li> <li>5. The Player tries to protect the surrounding cells, until the Player is overcame by the viruses.</li> <li>6. The Virion System shows the time score.</li> </ol>
<i>Alternative Flow of Events</i>	<ul style="list-style-type: none"> <li>• The Player can get defeated before the post-infection stage. <ul style="list-style-type: none"> <li>• The Virion System will display the time score.</li> </ul> </li> <li>• The Player scores in top five. <ul style="list-style-type: none"> <li>• The Virion System asks for Player's credentials and adds them to the top five high score table.</li> </ul> </li> </ul>
<i>Quality Requirements</i>	<ul style="list-style-type: none"> <li>• The Virion System will send the viruses according to the quadratic function of <math>f(t) = 16t^2 + t + 16</math> where <math>t</math> represents time, and <math>f(t)</math> is the function of number of viruses.</li> <li>• The Virion System will increase the strength and durability of viruses according to the function, <math>g(n) = e^n</math> where <math>n</math> is the number of defeated viruses.</li> </ul>

<i>Use case name</i>	ViewHighScore
<i>Participating Actors</i>	Player
<i>Entry Condition</i>	Player has opened the game.
<i>Exit Condition</i>	Player returns to play the game.
<i>Main Flow of Events</i>	<ol style="list-style-type: none"> <li>1. Player chooses to view High Score option.</li> <li>2. The system displays high scores list on the screen.</li> <li>3. Player returns to play the game.</li> </ol>

<i>Use case name</i>	PauseGame
<i>Participating Actors</i>	Player
<i>Entry Condition</i>	Player is playing the game.
<i>Exit Condition</i>	Player resumes playing the game.
<i>Main Flow of Events</i>	<ol style="list-style-type: none"> <li>1. Player selects pause option during the game.</li> <li>2. Virion System pauses the game.</li> <li>3. Player continues playing the game.</li> </ol>
<i>Alternative Flow of Events</i>	<ul style="list-style-type: none"> <li>• Player chooses to exit the game after pausing.</li> <li>• Virion System closes the game.</li> </ul>

<i>Use case name</i>	ViewHelp
<i>Participating Actors</i>	Player
<i>Entry Condition</i>	Player wants to get help about the game.
<i>Exit Condition</i>	Player resumes playing the game
<i>Main Flow of Events</i>	<ol style="list-style-type: none"> <li>1. Player chooses to view help.</li> <li>2. The Virion System displays tutorials and necessary information to play the game.</li> <li>3. After reading, Player returns to the game.</li> </ol>
<i>Alternative Flow of Events</i>	<ul style="list-style-type: none"> <li>• Player chooses to view help after pausing the game. <ul style="list-style-type: none"> <li>• The Virion System displays information which includes the explanation of how game will be played.</li> </ul> </li> <li>• After reading, player returns to the game.</li> </ul>

<i>Use case name</i>	ViewInfo
<i>Participating Actors</i>	Player
<i>Entry Condition</i>	Player is playing the game.
<i>Exit Condition</i>	Player returns to the game.
<i>Main Flow of Events</i>	<ol style="list-style-type: none"> <li>1. Player chooses to view scientific information.</li> <li>2. The Virion System displays scientific information which includes explanations such as viral infections and types of viruses.</li> <li>3. After reading, player returns to the game.</li> </ol>

<i>Use case name</i>	ToggleMusic
<i>Participating Actors</i>	Player
<i>Entry Condition</i>	Player has started the game.
<i>Exit Condition</i>	Music is turned off OR Music is turned on.
<i>Main Flow of Events</i>	<ol style="list-style-type: none"> <li>1. The Player chooses to toggle the music.</li> <li>2. The Virion System turns on or off the default music.</li> </ol>