Project Report

ISYS20182: Practical Project Management & Professional Development

Group Li:

Florian Held N0774540

Sagar Joshi N0774756

Nathan King N0733894

Steffan Walker N0756079

**Abstract**

The assignment for Practical Project Management & Professional Development was to design and implement a piece of software as a team. As a collective we decided to produce an educational simulator which would demonstrate the spread and movement of an epidemic.

The decision to make an epidemic simulator was made as there are very few online recourses that show the spread of an epidemic. There is a low level of awareness surrounding this topic and it is something that people should be made aware of. The primary audience for this project is biology students and epidemiologists, but it is usable by all people due to the simple and uncomplicated user interface.

The simulator uses a graphical user interface and displays an accurate representation of the United Kingdom, the land is colored green and the see blue. The pixels on the land of the UK represent people, when the color of the land changes that means the people have changed state. The GUI allows the user to input two values for variables which will affect the simulation. The variables are the recovery rate and the probability of being infected. The user can then choose the speed of the simulation. Once the user has made their selections, they can start the simulation and then either pause or stop it through buttons.

The simulator makes use of the SIR model to ensure the spread is accurate and realistic. The SIR model looks at 3 variables, susceptible people, infected people and recovered people. These 3 different states are represented in the simulation by changing the color of the pixel when its state changes.

**Contents**

Introduction 1

Survey of Existing Solutions 2

New Ideas 5

Design and Development 7

Evaluation of Product 14

Discussion/Conclusion 17

Professional, social, ethical and legal issues 18

References 19

Appendices 20

**Introduction**

1.0 Background

An epidemic is the widespread occurrence of an infectious disease in community at a point in time, these diseases can be spread in a variety of ways such as; through the air, from direct or indirect contact with another person, saliva, urine, blood, through sexual contact, and through contaminated food and water. In a densely populated area such as a city the spread of an infectious disease can happen extremely quickly, but with modern medicine and science the rate of spread has been decreased a lot, and in some cases can be stopped completely before an outbreak.

Although science and medicine have made huge strides in preventing epidemics there is still a possibility that one will take place.

To be able to see the spread of an epidemic, simulations are used often. Most simulations show how the disease will spread in a closed area with a small population of people. Although this does show how a disease spreads, it is not very realistic.

The group has decided to produce an accurate, realistic epidemic simulator which will show the spread of an epidemic across the United Kingdom. The simulator will use a map of the UK and have the epidemic start in a random location. The disease will then spread across the country. The SIR model will be used to provide accuracy.

1.1 Aims:

The main aim of this project is to educate people about the spread of an epidemic and allow the user to see how a disease can spread across a country. This will be achieved by producing an accurate and realistic simulator using a map of the United Kingdom.

1.2 Objectives

As this is a complex project there are 4 fundamental objectives:

**Map** – The simulation should be on the countries map, this will allow it to be more realistic, showing how the disease would react to water, densely populated areas and distances between islands.

**Clear –** The simulation should be clear. The user should be able to see the different states of the people using different colors.

**Realistic –** The simulation should be realistic, it should use real logic and biology to simulate how diseases would spread in real life.

**Educational –** It should give the user an insight into how diseases work in the real world. There should be pre-sets already in the simulation that allows the user to see how different types of bacteria and virus would spread.

**Survey of Existing Solutions**

2.0 Current programs

Once the decision to produce an epidemic simulator was made, the team began to investigate other existing products which are available on the web. To produce a useful and intuitive piece of software the team must first see what pieces of software are already available and look at their features as well as identifying their strengths and weaknesses.

First visualization of the SIR model

|  |  |  |  |
| --- | --- | --- | --- |
| **Name of solution:** | **Features:** | **Strengths:** | **Weaknesses:** |
| The Gleamviz program ( <http://www.gleamviz.org/> ) | Gleamviz uses 3 models to represent the infection with improved accuracy. These are the epidemic model to mathematically simulate the infection from person to person, the mobility model to represent how easy it is for people to travel from one location to another, and the population model representing data like urban density. Gleamviz also has the option of representing the outbreak in a 3D environment. | -Well designed and implemented.  -Uses real time data to show the spread of the epidemic. | -Overly complicated, only usable by experts. |
| Shodor Disease model( <https://www.shodor.org/featured/DiseaseModel/JavaScript> ) | This model represents the spread of a disease in a population. It is meant to illustrate the various factors that can affect how quickly and how far a disease can spread. The model only shows people, they can have one of three states; susceptible, infected or immune. There are also doctors who can cure the infected. | -Accurately shows how disease spreads.  -Allows the user to change factors that effect the disease spread.  -Allows the use of doctors, this relates closer to real life disease spread.  -Shows the number of heathy, infected, immune and killed people. | -It does not use a map, rather 3 areas separated by a wall with small spaces for people to pass through.  -Only shows a maximum of 2000 people. |
| FRED US measles simulator ( <https://fred.publichealth.pitt.edu/measles> ) | This program shows only the spread of measles in given US cities. Rather than a simulation the website offers videos of how the disease spreads. This website is useful compare to others as it shows how city density can effect the spread of a disease. | -Relevant to real life, real time disease spread.  -Shows how diseases spread in a city.  -The user can change the city. | -Pre recorded video not a simulation.  -Same spread each time. |

2.1 Product Demand

After searching the web looking for other programs which shares the same idea as our teams, it has been concluded that there is a demand for a free to use, accurate epidemic simulator which is based on a real-life map. The demand mainly lies within education, for a student of epidemiology or any science looking at disease spread it can be hard to picture how a disease can spread over a given area.

Most simulations on the market look at the spread of a disease between people as objects, so there will be x number of people in given space and the disease will spread between them. This only shows how disease spreads between people, but to be able to see how a disease spreads over a large geographical area while still using people is something completely different.

2.2 Research About The Topic

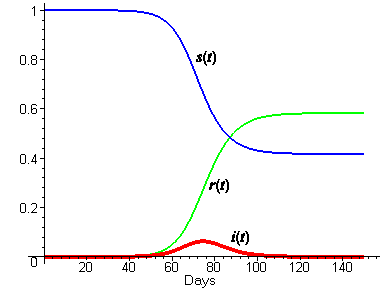
An epidemic is the widespread occurrence of an infectious disease in community at a point in time. Numerous epidemics have been recorded in the human history. The more humans evolved the less common epidemics became. With modern science and medicine people can now predict the spread of an epidemic and in some cases, prevent it. To be able to produce an accurate program the team must investigate epidemics which have already taken place, how they were stopped and how they spread. Millions of people have been killed by epidemics in the past, some of the most famous epidemics include:

*Tuberculosis* – This disease is responsible for1.3 million deaths annually, it remains one of the leading causes of mortality worldwide. (Kozlov, 2019) Strains of this disease have been traced back to the DNA of ancient Egyptian mummies, revealing that this epidemic has been wreaking havoc on humanity for thousands of years. Caused by a bacteria that spreads through the air. In the 19th century, tuberculosis killed an estimated one-quarter of the adult population of Europe, and by 1918 one in six deaths in France were still caused by Tuberculosis. (Healthcarebusinesstech.com, 2019)

*Polio* - This disease is suspected to have been around for thousands of years. It usually spreads through contaminated food and water, and targets the human nervous system, which causes symptoms that usually affect the legs and lead to paralysis. In the United States, outbreaks peaked in 1952, when 57,628 cases were reported throughout the nation. Though there is no cure for this disease, a vaccine was developed in the 1950s that has proved effective for decades. Since this is a disease only carried by humans, widespread vaccinations are expected to remove any traces of the disease in future generations. (Healthcarebusinesstech.com, 2019)

*Yellow Fever* - This disease is related to malaria in that it spreads from person to person through infected mosquitoes. Responsible for taking millions of lives and wiping out entire colonies and villages during the time of Napoleon’s reign, yellow fever still persists in areas of Africa and South America. While some infections are less severe than others, it is a life-threatening disease that can cause severe internal bleeding and liver failure, which is marked by the yellowing of the skin – prompting this disease’s name. (Healthcarebusinesstech.com, 2019) The most deadly case of yellow fever was during 1793 in the US where it killed over 4,00 people in the span of one month.

These are only three of a list of deadly epidemics, many epidemics have similar characteristics. A lot of epidemics also share the way the disease is spread. Through the use of history and scientific studies made into epidemics scientists are able to produce models to predict the spread of a contagious disease. There are many models to predict an epidemics spread but the most efficient is the SIR model. The SIR model is an epidemiological model that calculates a theoretical estimation to the number of people infected with a contagious disease in a closed population over a period of time. The sir model makes use of 3 dependent variables, the number of susceptible people ***S(t)***, number of people infected ***I(t)***, and number of people who have recovered ***R(t)***. The independent variable is time **t**, it is measured in days.



SIR Model diagram (Smith and Moore, 2019)

This diagram above shows the SIR model I action, as the number of infected people increase the number of susceptible people drop and the number of people who have recovered rises, the s(t) and r(t) variables both plateau when the number of people infected decreases to none.

This model could be used by the team to make the simulator accurate. This will be possible as the simulator will be set in a closed population over a period of time. The model also makes use of several equations, these could be implemented in the code.

**New Ideas**

3.0 Program Introduction

As a collective the group decided to produce an epidemic simulator to show the spread of a disease across the UK using the SIR model. Although there are some epidemic simulators on the market, the group has yet to come across one which uses the SIR model and display the spread of a disease on a rea-life map. The group is looking to spread awareness and educate people about epidemics and their spread.

### 3.1 Functional Requirements

### After seeing what is available in the market, the group thought of new idea for the functions of the system. The simulator will need to include features which are already expected of a simulator like this, but to make it stand out it must include features which have not been seen before.

A functional requirement defines a function of a software system or its component. A function is described as a set of inputs, the behavior and outputs. Functional requirements are specific functionality that define what a system is supposed to accomplish. The following are the functional requirements of the proposed epidemic simulator:

FR1: Produce an educational simulator

The system should be an accurate epidemic simulator based on the SIR model.

FR2: Display A Map

The system must display the map of the United Kingdom, the simulation will be run on this.

FR3: Use Different Colors

The system must make use of at least 3 different colors to show the pixels state. There should be a different color if the pixel is susceptible, infected or recovered.

FR4: Accuracy

The system must be accurate, to calculate the rate of spread the system should use the SIR model.

FR5: GUI

The system will use a GUI so that the user has an easy experience when using the system.

FR6: User Input

Through a GUI the user will be able to enter the recovery time and probability of spread.

FR7: Change speed

The system should allow the user to change the speed of the spread. The user will be able to choose between half, normal and double speed.

FR8: Display Useful Information

The system should display the number of people infected, recovered and dead.

Before the group develops and expands on the current ideas the group focused on ensuring the system would deliver on a set of key features of the simulator. The key features included having a educational simulator (FR1) which was based on a map of the United Kingdom (FR2). The system would also be expected to show if a person was infected by changing the color of a pixel, the pixel should have a color for being susceptible, infected and recovered (FR3). Once these core features were implemented into the simulator the group began to expand the functionality to make it more realistic and user friendly.

The first new feature to be added after the core features was improving the accuracy. The group researched ways in which epidemics were measured, predicted and modeled. The SIR model was the most accurate and implementable model available. The SIR model makes use of a set of equations which can be used to calculate several things but most importantly the rate in which disease spreads, the recovery time and the probability of spread. The SIR model and its equations would be what the simulation works off to ensure its accuracy (FR4). Once the SIR model was implemented the group looked to make the system more usable by users. For this the group decided to implement a GUI (FR5). Through the GUI the user will be able to change a number of variables including the recovery rate, probability of spread and the speed of the simulation (FR6&7). Finally, when the simulation is running some important information is presented to the user such as the number of people infected and the number of people who have recovered (FR8).

3.2 Non-Functional Requirements

Non functional requirements don’t look at the behavior of the system but rather the operation of the system.

NFR1: Maintainability

The system should be easily maintained.

NFR2: Efficiency

The system should make use of libraries to allow quick and smooth movement in the simulation.

NFR3: Operating Constraints

The simulation must run within the constraints set by the SIR model.

NFR4: Scalability

The system must be scalable.

NFR5: Reliability

The system should run from start to finish one of every thousand times its run, in absence of hardware failure.

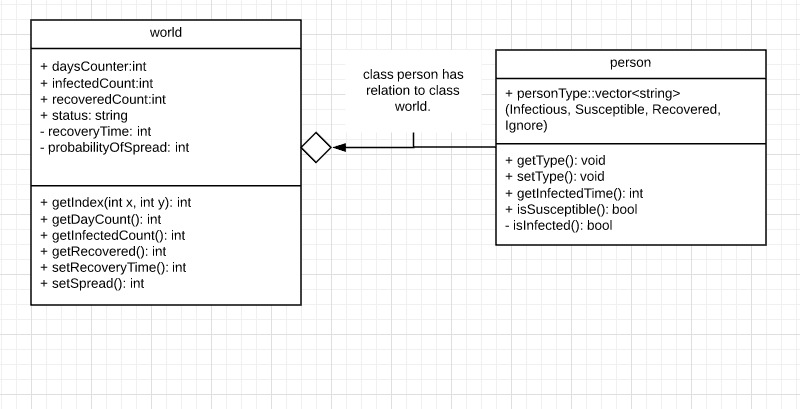
**Design and Development**

4.0 Class Diagram

This project is planned to be programmed using object-oriented programming. We will have classes and objects to create an efficient and effective solution for the problem. A class is a blueprint or a template which describes and defines its own data members and member functions, these are then used and are accessible by creating an instance of that class. An object is an instance of a class, it has set states and behaviors. In this part of the project, only class diagrams will be created as it will give a better idea of how the simulator should be programmed.

For this product two main classes were considered, they can be seen in the screenshot below. The class world uses the person class. Each person is a collection of pixels. In the class world, there are a few important attributes. The attribute daysCounter this will just be a counter which shows the days passed. Followed by the method getDayCount() will count and display the days to the user. This works similarly for all the counts such as infectedCount and recoveredCount. Moreover, the attribute recoveryTime will show how long it takes a person to recover after being in an interaction with the disease. The attribute probabilyOfSpread will take in account how much the disease is going to spread. Both of these attributes will be adjustable by the user in the GUI, making for an interactive experience.

The class person will set the collections of pixels to a personType which can be infectious, susceptible and or recovered. These methods and attributes will then be used by the class world to make a change in the system.

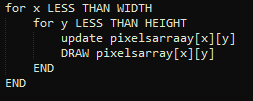


4.1 Code design and development

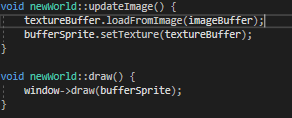
This section is written to provide a good overview of the process from designing the code to then writing the code based of the initial design. The section will also highlight any issues during the development of the code therefore causing the actual code to differ from the initial design of the code.

One of the issues during the development of the simulation was the efficiency of the program, the issue was that the program would really slow when actually running the simulation after hours of testing the team found that it was running slow due to every single pixel being drawn individually every single frame, this was hugely inefficient and required a rework from our initial code design. To overcome this problem the team had to change the program to update an image in our program, this is then drawn to

the screen once every frame. This is called an image buffer; this is much more efficient as instead of having thousands of draw function calls every frame it has one draw function call.

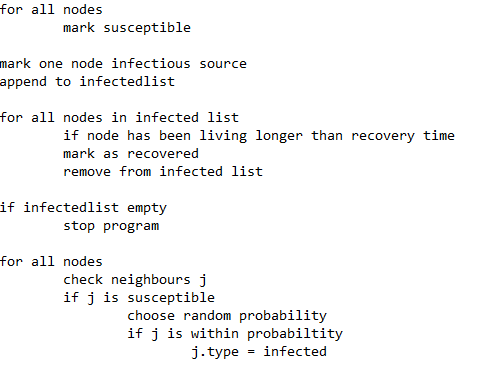


The initial design for drawing pixels to the screen:

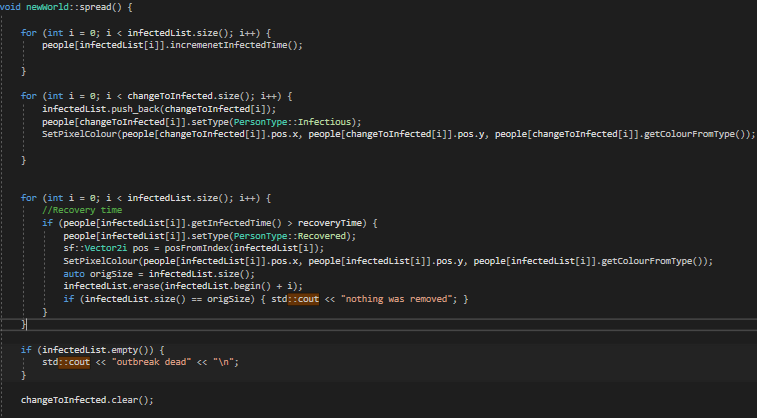


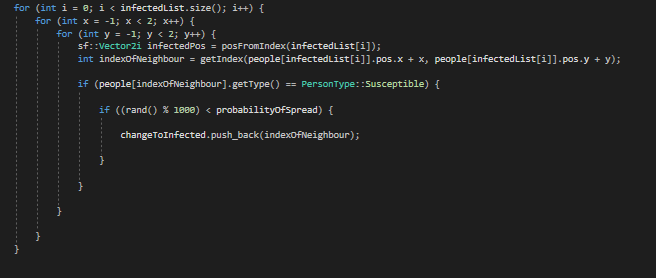
Our actual code that we ended up with in our program:

Next, the whole point of the program is to visualise a disease outbreak, to do this we decided to use the SIR model (Susceptible, Infected and Recovered), we decided to use a type of sir model that has a fixed recovery time from infected to recovered, also it is a random probability that a neighbour of an infected person will be infected as well. How we designed it was to loop through every person that was infected and check their neighbours and calculate the probability of them becoming infected and then updating the people array

.

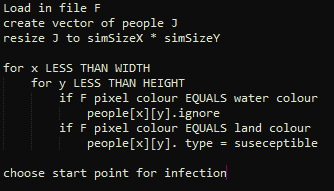
This is the code that was designed for the spread of the disease through people.



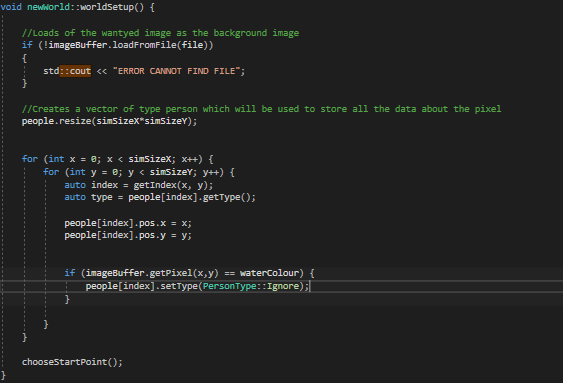


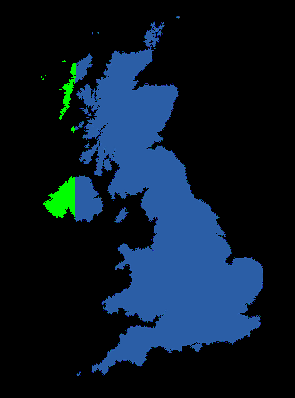
Furthermore, another main piece of the program is the map and simulation setup, this code is relatively simple using some in-built SFML functions that come with the library. As you can see in the pseudo code the code should load in a designated image. And split the pixels into type for example if the pixel is water coloured it will tell the program to ignore those pixels however if the pixel colour is land coloured then it will turn the type into susceptible.

This is the pseudocode that was developed for setting up the map



This is the actual code that was written in the program.

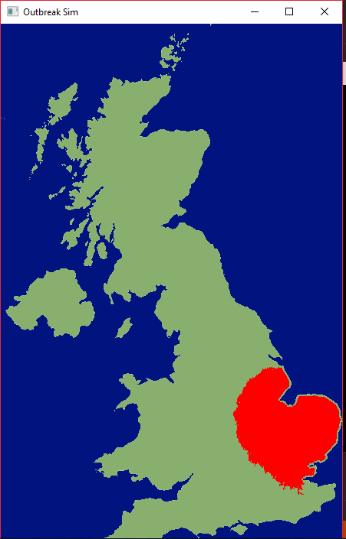


4.2 Code Iterations

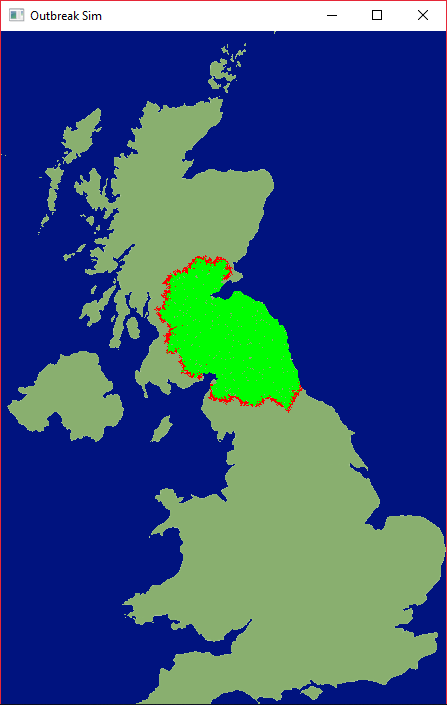
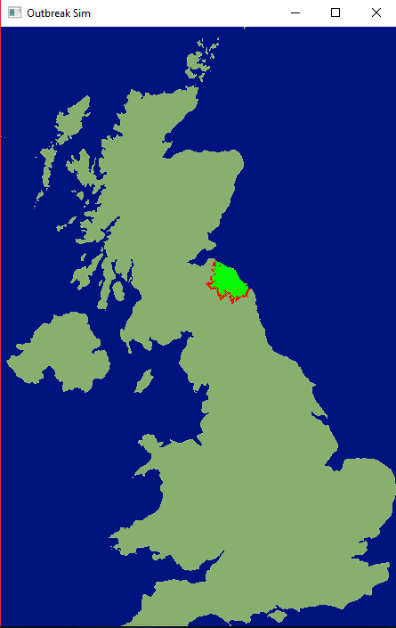
Iteration 1

This is the first step of the program that the group have made so far, what it does is that it loads in a map of the UK, onto the screen using SFML functions. Then what it does is that it changes the pixels of the UK to a different colour. This is relevant and useful as this what will happen in the simulation however with more logic. The colours in the simulation will represent different states for the people that will live there, for example, red will mean infected, blue susceptible to the disease and green immune to the disease. Therefore getting the ability to change the colour of the pixels early on is key. However, the next step for the simulation is to build in some random simulation spread without putting in the actual disease rules.

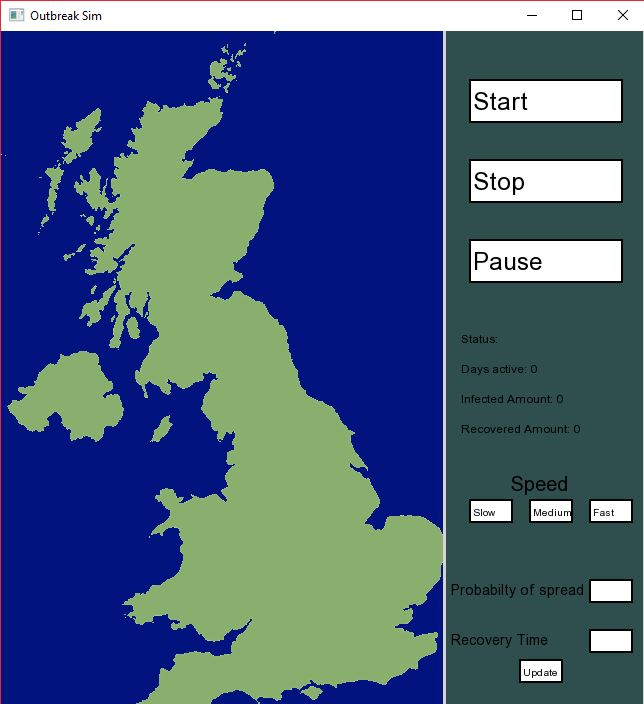
Iteration 2



This is the next iteration of the simulation as you can see here there is now a starting point for a disease that is randomly generated and the disease actually spreads, however the spread is not actually based of any rules, the computer just picks a random direction for the disease to spread into, however this is putting the program in a much better position rather than the simulation just iterating over land and changing the colour of it, as you can see for with the two images the disease can start in different locations and spread to cove the land.

Iteration 3

This is iteration 3 of the development of the simulation this iteration is where the sir model was actually implemented into the simulation, as you can see in the images, the red pixels means that the person is infected, and the green pixels mean is recovered from the infection. This is screenshot is produced from using recovery rate as 50, and the probability of spread is 1%. The next iteration is to implement a gui to allow the user to change the recovery rate and the probability of spreading.



Iteration 4

This is iteration 4 of the project, this iteration is to do with the GUI and giving the user some input into the simulation, as you can see the GUI allows the user to Start, stop and pause the simulation. Also, the GUI allows the user to change the two variables for use in the SIR model; the recovery time and the probability of infection spread. All the GUI elements were made by the team using SFML base features such as rectangles and text.

**Evaluation of Product**

The development of the product has finally been completed. This section will explain how the product has met against the functional/non-functional requirements as well as the objectives. For this to be possible various testing methods will be considered such as code testing and usability assessments. If the group was able to meet the requirements and objectives successfully it will then ensure that the disease epidemic simulator is working correctly and efficiently. The group has made a disease epidemic simulator that shows the spread of a disease epidemic across the UK. Next, the objectives of the product will be looked at.

5.1 Objectives

**Map**

One of the objectives was that the simulation should be on a map, this will allow it to be more realistic, showing how the disease would react to water, densely populated areas and distances between islands. All these factors would lead to the probability of the disease spreading. The group was able to meet this objective at some extent as the final prototype was running on the map of the UK. However, complex factors such as the population density can be included later in the development timeline. Factors like these were considered but not implemented due to a shortage of time. Therefore, it was an overstated factor that can be implemented in the next update.

**Clear**

The simulation should be clear; the user should be able to see the different states of the people using different colors. The group successfully met this objective as there were three colors describing the three states of the people’s condition. Red considering the infected, green as in the immune and the dots were the people that weren’t infected by the disease.

**Realistic**

The simulation should be realistic, it should use real logic and biology to simulate how diseases would spread in real life. This objective was clearly met as the SIR model was used during the development of the product. This is an epidemiological model that computes the theoretical number of people infected with a contagious illness in a closed population over time.

**Educational**

It should give the user an insight in how disease work in the real world, there should be pre-sets already in the simulation that allows the user to see how different types of bacteria and virus would spread. The product has been deemed as educational, as users from students to a person of any age interested in how a disease spreads can learn exactly that from the product. Although, interactivity is at the minimum as different bacteria and viruses cannot be entered manually or be selected by the user. However, factors such as the recovery rate of the disease as well as the probability of it spreading can be changed by the user.

5.2 Evaluation against the functional requirements

During the planning phase of the project, many functional as well as non-functional requirements were created. These were the basic targets set by the group together. To ensure that the final product was successfully created it is important that it met most, if not all the requirements.

**FR1: Produce an educational simulator**

The system should be an accurate epidemic simulator based on the SIR model. The final version of the disease epidemic simulator was accurately made where the SIR model was implemented. In future versions of the product, additional functionalities are worth analyzing and implementing. For example, the selection of diseases and their characteristics.   
  
**FR2: Display A Map**

The system must display the map of the United Kingdom, the simulation will be run on this. This was successfully met as a map was clearly included in the final iteration of the product. In future iterations, it can be ideal to have attributes to the specific location where the disease is spreading. For example, when choosing UK, the recovery rate might be higher and probably of the disease spreading might be lower, than in other countries which are less developed.

**FR3: Use Different Colors**

The system must make use of at least 3 different colors to show the pixels state. There should be a different color if the pixel is susceptible, infected or recovered. Three colored pixels were implemented in the product where the red pixels mean that the person is infected, and the green pixels mean the person has recovered from the infection. On the other hand, it can be better to have additional pixels and or graphical representation of other factors such as airports. Airports can be implemented and through this, diseases can be transferred from one island to another or from one country to another.   
  
**FR4: Accuracy**

The system must be accurate, to calculate the rate of spread the system should use the SIR model. The system is using the SIR model in the implementation. The product is using the SIR model as this is the most accurate model to represent the spread of diseases. However, the SIR model, may not be ideal to use in anything other than epidemic disease spread representation. For example, if the problem was to make a simulation of how a virus spreads on a computer, then another model for example the SIS model maybe more relevant and meaningful to use.   
  
**FR5: GUI**

The system will use a GUI so that the user has an easy experience when using the system. The group has met this requirement as the GUI is simple and interactive, with functionality for the user to change the recovery rate and the probability of spreading. As mentioned before, further functionality can be added to change the attributes or the type of disease in future iterations of the product.

**FR6: User Input**

Through a GUI the user will be able to enter the recovery time and probability of spread. In the final iteration of the product the user can change the recovery rate and the probability of the disease spreading.   
  
**FR7: Change speed**

The system should allow the user to change the speed of the spread. The user will be able to choose between half, normal and double speed. In other words, this can also be described as the intensity of the disease. If the disease is of a higher intensity, the speed of it spreading would also be higher. The group was successful in adding this functionality to the system.

**FR8: Display Useful Information**

The system should display the number of people infected, recovered and dead. The group has made it so that the user is also able to see the number of people who are infected, cured and dead.

5.3 Evaluation against non-functional requirements

**NFR1: Maintainability**

The system should be easily maintained. The group has made it so that the system is easily maintainable. Other functionality other than those which already have been implemented can also be added easily.

**NFR2: Efficiency**

The system should make use of libraries to allow quick and smooth movement in the simulation. In the beginning, the group was using multithreading for smooth movement in the simulation. However, in the later iterations it has been proven that image buffer is much efficient in comparison to multithreading. This is because this is a simple product and having techniques like multithreading may not be suitable for a product of this scale.

**NFR3: Operating Constraints**

The simulation runs within the constraints set by the SIR model.

**NFR4: Scalability**

The system must be scalable. The system is scalable as you can add many other functions.

**NFR5: Reliability**

The system should run from start to finish one of every thousand times its run, in absence of hardware failure. After testing the product multiple times, it is proven to the group that it is reliable.

**Discussion/Conclusion**

The program, that the team has worked together to develop, is a pandemic simulator capable of simulating the spread of a disease throughout the United Kingdom. The program uses the SIR model which is a mathematical model used for simulating epidemics with great accuracy. The program graphically represents the UK’s residents using a different colored pixel for individuals with a different relation to the disease such as infected, immune etc.

Through the team’s development of this program, a lot has been learned about producing a program to simulate an event such as an epidemic as well as the necessity to produce efficient code that will run smoothly on common computers. In order to simulate an epidemic, there is a wide variety of variables, such as person type which covers susceptible, immune infectious and ignored, that must be taken into consideration if it is to be accurate. These variables must be applied to around 120,000entities that make up the UK’s simulated population. For instance, each person has a unique probability chance that the disease will infect them as well as a chance for them to become immune. These probabilities multiplied by many simulated entities leads to an exponential number of calculations being computed every second, which brings us to what was learnt about hardware.

Due to the high number of calculations needed to simulate a disease the program had to be designed extremely efficiently so the CPU could calculate and display the epidemic in real time to the user. During the early development stages the program was less efficient than the final version that exists now, in that it spread across the map slowly and updated in stages rather than real time. After some changes to the compute, draw cycle and smart methods of only performing calculations that were necessary, the program runs much smoother and is much more efficient. An image buffer was implemented to stop the program drawing for every pixel spread and instead draw once every pixel has had a chance to spread. An if statement to check if individuals in the array are susceptible to the disease, allowed the program to only calculate the spread probability of infected pixels rather than every pixel in the array. These changes made the program more efficient which in turn created a smoother rendering of the spread for the user.

Beyond the current state of the program, further improvements are possible such as a population density model to alter the rate of infection based upon geographical location. The implementation of a transport network that would allow infected cells to interact with cells across long distances as opposed to a nearest neighbour spread. These improvements would add to the program's accuracy however, they would also require many hours of work and planning to include. Unfortunately, there was not enough time available on this development cycle which is why these changes have not been implemented. In addition to working on the epidemic simulator, each team member had various other course modules to manage and work on which reduced the total amount of time each team member could work on the project.

The projects planning, management and development was carried out with relative ease due to the team’s consistent meetings and frequent communication. This allowed team members to raise any issues they had with the project and solve them quickly due to the assistance of other team members. In terms of the team’s access to facilities to work on the project, each team member possessed a home computer, a laptop and 24/7 access to the university library computers which made hardware for development easily accessible.

The program was developed using MS Visual Studio which team members had access to through the university's free software site or by freely downloading it from the VS website. The documentation and diagrams for the project were created using MS Word which is installed on the library computers and is available to the team members through Microsoft’s student office package.

Ultimately the project was successful in its aim to produce a piece of software that simulates an epidemic to a reasonable degree of accuracy and represent the infection visually to the user. The project could have been improved upon further if time was in greater supply to the team members however, with the resources available they consistently maintained communication to discuss the project and managed their time effectively to find a balance between this and other projects.

**Professional, social, ethical and legal issues**

The idea of making a disease epidemic simulation program comes with many different issues, such as professional, social, ethical as well as legal issues. This program is a lot more than just a regular application or a game but is a study of infectious disease outbreaks and the program can be used as a tool to teach about today’s growing public health issues and practices. This simulator can be useful for scientists and can also be used by anyone who may be interested in learning how quickly disease outbreaks can take place. Moreover, in the past there have been occasions where a disease outbreak took place for example, the Ebola Virus and simulation programs were made while this was taking place. It is important to make sure that this does not happen, as it may raise emotional social and ethical issues. This program is only to promote education to those who wish to learn about how a disease outbreak takes place. Therefore, no advantage is taken of any real-world situations and any social and or ethical morals are not challenged.  
   
Advancements in technology, brings a lot of legal and ethical questions and because of this many area of computer science are considered. Privacy is an area that cause many legal and ethical issues. In the case of a program or application, a user may have to log In or sign up to achieve rewards or track their progress, it is important to consider that private information like this is not disclosed. Another example of an issue can be hacking, where a user might want to break the program. This may or may not lead to a user accessing private information such as I.P. addresses and or rewards they may not have achieved yet in a wrong way. To prevent this, there is the Data Protection Act, which sets out principles and makes a statement that data needs to be protected and used in the correct way.

**References**

**Research into topic:**

Kozlov, R. (2019). *Tuberculosis Case Study*. [online] Testtargettreat.com. Available at: https://www.testtargettreat.com/en/home/educational-resources/case-studies/tuberculosis-case-study.html [Accessed 1 May 2019].

Healthcarebusinesstech.com. (2019). *The 10 deadliest epidemics in history*. [online] Available at: http://www.healthcarebusinesstech.com/the-10-deadliest-epidemics-in-history/ [Accessed 1 May 2019].

Smith, D. and Moore, L. (2019). *The SIR Model for Spread of Disease - The Differential Equation Model | Mathematical Association of America*. [online] Maa.org. Available at: https://www.maa.org/press/periodicals/loci/joma/the-sir-model-for-spread-of-disease-the-differential-equation-model [Accessed 2 May 2019].

<https://www.shodor.org/featured/DiseaseModel/JavaScript>

<http://www.gleamviz.org/>

[https://fred.publichealth.pitt.edu/measle](https://fred.publichealth.pitt.edu/measles)s

**Appendices**

**Appendix A**

**Meeting Minutes**

****

****

****

****

****

****







