GRIFFITH COLLEGE

# Assignment Cover Sheet

| | | | |
|---|---|---|---|
| **Student name:** | **Myo Thet Tun** | | |
| **Student number:** | **2918551** | | |
| **Faculty:** | **Computing Science** | | |
| **Course:** | **BSCH** | **Stage/year:** | **2** |
| **Subject:** | **System Analysis and Design (SAD)** | | |
| **Study Mode:** | Full time ✓ | Part-time | |
| **Lecturer Name:** | **Dapeng Dong** | | |
| **Assignment Title:** | **File System Visualization Group Project** | | |
| **No. of pages:** | **21 pages including Title page** | | |
| **Disk included?** | Yes | No | ✓ |
| **Additional Information:** | (ie. number of pieces submitted, size of assignment, A2, A3 etc) | | |
| | **21 pages of A4 including Title page** | | |
| **Date due:** | **18/12/2017** | | |
| **Date submitted:** | **18/12/2017** | | |

**Plagiarism disclaimer:**

*I understand that plagiarism is a serious offence and have read and understood the college policy on plagiarism. I also understand that I may receive a mark of zero if I have not identified and properly attributed sources which have been used, referred to, or have in any way influenced the preparation of this assignment, or if I have knowingly allowed others to plagiarise my work in this way.*

*I hereby certify that this assignment is my own work, based on my personal study and/or research, and that I have acknowledged all material and sources used in its preparation. I also certify that the assignment has not previously been submitted for assessment and that I have not copied in part or whole or otherwise plagiarised the work of anyone else, including other students.*

**Signed:**  **Date: 18/12/2017**

**Please note: Students MUST retain a hard / soft copy of ALL assignments as well as a receipt issued and signed by a member of Faculty as proof of submission.**

# Group Report

## 1. Executive Summary

As we can see that the rise of advanced technology in the computing world, there are many new system visualisation tools emerged in the recent years. However, the main goal of the system visualisation tools is the same, to display all the folders and files in structured view and to use graphical tools to make users more visualise how much size of the disk is already used and what kind of files use the most on the specified disk drive.

Our system visualiser is based on java and it is platform independent. So, the system is designed to run on all operating system. It can read all the folders and files on the specified disk drive and display them in a structured tree view and display as various sizes of graphical squares depending on the size of the files on the disk.

For the marketing point of view, the main consumer groups would be Windows and Linux users. Our main goal is to help windows users to visualize their data. Due to the design of Linux systems it is easier to locate data in the drives but our program also offers information on general statistics of the system. Like the total amount of files, the path for the heaviest folders etc... This can help users track unwanted files and located them much faster.

The product will be delivered in a single java binary that can run on multiple systems. This property makes it very light weight and it doesn't require a difficult installation. The program performs reasonably well, it is stable, efficient and robust. The performance may vary between systems since the limitation is the random-access time of the storage unit.

In this document we reflect the process of the development of the program and also an overview of the different solutions on the market. Everything related to the design, usability, performance and feasibility will be discussed deeply on the following sections.

The project can be produced in a short period of time and will require low maintenance due to the nature of java. Our objective is to make a fully platform independent software running on java capable of satisfying the needs of our target customers.

## 2. Introduction

File visualisation is one way of data visualisation. Data visualisation is a representation of pictorial or graphical way to describe the actual data. So, why data visualisation? It enables decision makers or

people who are in the managerial level to view all the data visually as a whole process so that they can analyse, grasp difficult concepts, and make the important decisions effectively. Now with the help of technology, using interactive visualisation of data makes an advantage to view a large bunch of data in various graphical formats.

The use of graphical formats in describing data is not a new discovery. Maps and charts had been using to show data since 17th century and the invention of pie chart came in the early 1800s.

Why it is important to use visualisation technique? It highly depends on how human brain works. Because human brain tends to understand easily and quickly by analysing graphs or charts rather than looking at spreadsheets or reports especially when dealing with a huge amount of data. There are various ways to represent data in visualisation such as 3D, blocks, bars, dots, and so much more depending on individual needs.

### 2.1 Motivations

Computers are machines which perform functions systematically. A computer has many built-in functions stored in it. One of the most important features among these functions is file system. What exactly does file system mean? It is a kind of system which can control and categorise all the files on the disk drives and it can turn out to be a single file and let the user view a specific location on the computer. For example, data related to the operating system is categorised and stored in many different folders because of the file system.

File visualisation is one of the ways to explore the size of all the files on the specified disk visually in a graphical mode. There are many file visualisation tools available on the internet which representing the file system visually in different methods and different graphical formats. Among these, some of the example system visualising tools we found are WinDirstat and fsv.

### 2.2 Nature of the software

The main feature of the software is that it is platform independent and it uses some components introduced in Java 7 that makes the work much easier. For example, after doing some research, the solution that we implemented is the current most robust and efficient provided by the Java JDK. It can traverse 10,000 directories including the files on them in an average time of 600 milliseconds on an HDD drive.

The software implements multithreading components for the UI, so it can interrupt the path walker and close if necessary. The performance of the program is heavily dependent on the speed of the hard

drive. SSD's will have a much better performance specially since they have an even much higher reading speed compare to writing speed.

### 2.3 Current solutions

One of the most famous current solutions found online is Windirstat visualiser. It is a very nice system visualiser solution to use as a system visualiser. And it has versions for Windows, Mac and Linux.

Another solution that impressed us is fsv (File System Visualiser) which is a 3 dimensions system visualiser.

### 2.4 Challenges

The biggest challenge of this graphical representation of the files in the UI. Since the number of directories on a System can be very large it is difficult to find the best fit approach for it. For example, if the sizes and folders are represented in squares, how can they be distributed? In order to fit the squares in the screen without wasting space is a big design challenge. Another way to overcome this problem is instead of using squares lines can be used. As if the folders consisted on graph lines with an x axis as the number of the folder and a x axis representing the size in the y axis. A disadvantage of this approach is that if the files system is too big the output is unreadable. The last challenge to overcome is how to read the size of the files. In order to get fast performance, the files are being compressed before they are loaded in memory. So, the size that is accessible with the methods of the previous interface is not the current size of the files.

### 2.5 Brief introduction to the solution

Files Library in Java contains a static method called walkFileTree that takes a path and a class. This method will travers recursively the whole tree of the computer without crashing or throwing exceptions when a given directory can't be opened. The class used in the method must implement a FileVisitor interface to defines 4 methods used during the traversal. As the function visits each file those methods are called performing the action desired on the given directory. This is accomplished due to polymorphism. In order to create the UI, the swing framework is used. Java is a multi-threaded programming language that means that UI rendering can be separated from the flow of the program. To accomplish that the Thread class is used which works similar to event listeners in JavaScript when an event is triggered the Thread assigned to it will execute a function to perform an action in the UI.

If instead a synchronous single-threaded approach is taken the UI will be unresponsive because actions like closing the window will have to wait until other processes finish in order to stop the program.

### 2.6 Unique features

The unique features of the system are to view all the files and folders of the specified disk drive as a system tree structure and use java swing to visualise the type and size of folders and files in this disk drive.

### 2.7 Consumer groups and marketing potential

The main consumer for this product are Windows users. The reason for that assumption is that *NIX Systems treat storage differently to Windows. Unix was originally developed as a server OS, one of the problems that servers might need to overcome is the storage representation. Unix treats every drive as one, abstracting partitions and all elements of the system. This allows for example un Linux pc's to have a separate partition called Home that only contains user configurations and programs separated from System files this has multiple advantages one of them is that the user can easily check his files and manage them.

Windows treats disks and partitions as different drives. That makes it harder to use different partitions to run a windows system. The disadvantage of that is the difficulty to distinguish from system files and program files even if installations in windows are done in a folder locate in C:/prgram files reality is that those programs then save important information like configuration or registry keys in other places of the computer making it harder to track and uninstall without an uninstaller produced by the developers of the program. Eventually drives might end up have multiple "junk" files. A software like a file system visualizer that runs in java, is packaged in one single jar file and it will allow the user to track and locate unwanted files.

## 3. Background Research

### 3.1 Introduction to current solutions

Some of the interesting current solutions we found on the internet and took as reference solutions to the project are as follow.

1. WinDirStat

It is a disk cleaning tool with an easy-to-understand interface. It analyses disk space usage and presents different file extensions in many different colours. It also shows the sizes of all the files in the specific disk drive with the biggest size first. You can also do the clean-up while viewing the colourful graphic visualiser.

2. fsv (File System Visualizer)

It is a Unix/Linux based file system visualizer which shows all the files and directories in two kinds of three dimensions view, namely, MapV view and TreeV view.

### 3.2 Introduction to technologies used in the system

For programming environment, Eclipse Java IDE is used to write the program. Overview of the system is to read file specifications such as file names and sizes containing in the specific folder and display the statistics of the files on the form together with graphical bar formats.

For GUI, Eclipse WindowBuilder is used in our system. WindowBuilder is a combination of SWT Designer and Swing Designer which helps a lot for Eclipse users to create Java GUI applications without bothering of writing of complex coding. It uses the WYSIWYG visual designer and layout tools which enables us to create simple windows applications to the complex ones.

## 4. Analysis

### 4.1 Feasibility Analysis

The objective of the system is to implement a java program that must support at least one file system and to visualise file system in structured view.

According to MoSCoW analysis, the program must have the ability to read all the folders and files from a specific disk drive. It must show all the folders and files in tree directory view. If it is possible to include, it should visualise the specific file types in different graphical structures or in different colours.

#### 4.1.1 Technical Feasibility

According to the technical point of view, one feature available in java called FileVisitor interface which can visit recursively all the files in a file directory tree. Also, there are many technical reviews and video clips of showing how to use this technology.

#### 4.1.2 Operational Feasibility

In terms of performance, our system should respond in a few seconds. Only if it is needed to read the whole disk drive with many folders and subfolders, it will take about one to two minutes to be completed and to show the result. The system provided users with detail information of the disk drives and the sizes and types of the files on the specified disk with a minimal cost available. The system is

expected to give the user's requirements with a reliable service or with minimal bugs and using as minimal resources of the operating system as possible.

### 4.1.3 Economic Feasibility

The resources we need are mostly from the internet and almost all of them are free of charge. The only thing we need is to get our computers connected with Wi-Fi connection. From this project, the user will get many benefit of viewing the system in detail such as comparing or managing what kind of files take the most space and how much space is left on the disk drives.

The development cost is very less amount and operational cost is just to have a PC with Windows or Linux on it.

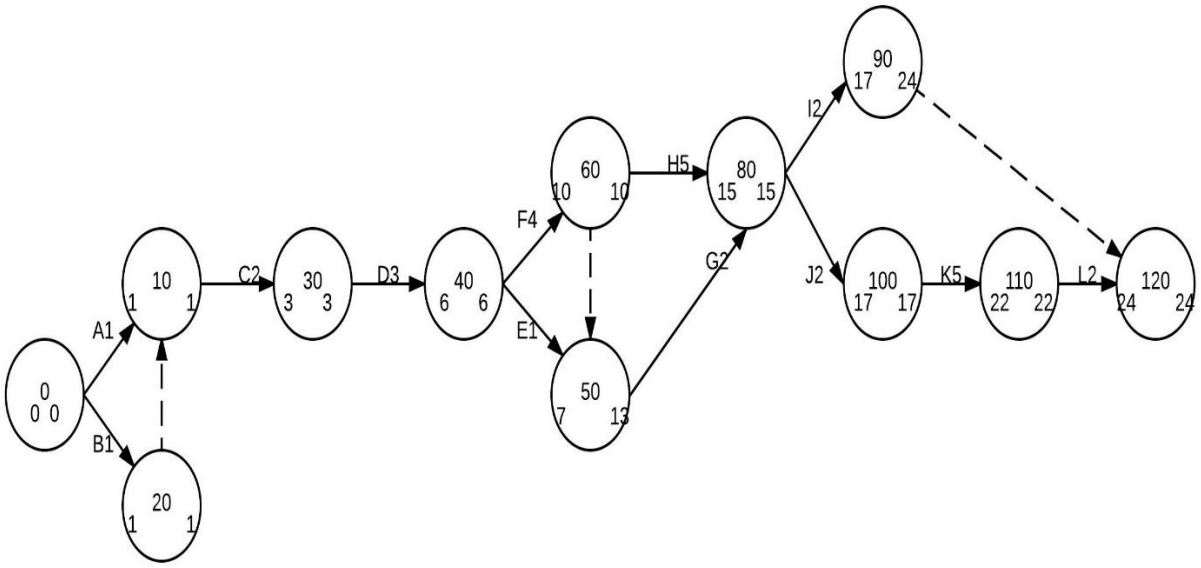
### 4.1.4 Scheduling Feasibility

We have the project duration of about one month. According to the requirements report, it is not a very big and complicated process. However, we considered that it may be difficult to show all the folders and files using GUI format. So, we decided to work according to the tasks and durations identified in the Critical Path Analysis. It can be considered that the project is possible to be finished on time with all the user's requirements fulfilled.

## 4.2 Critical Path Analysis

| Tasks | Description | Duration (Days) | Preceding Activity |
|---|---|---|---|
| A | research current available solutions | 1 | - |
| B | draw CPA diagram | 1 | - |
| C | gather requirements | 2 | A, B |
| D | perform feasibility | 3 | C |
| E | write documentation for introduction and background research | 1 | D |
| F | draw use cases, sequence diagrams and state diagram | 4 | D |

| | | | |
|---|---|---|---|
| G | write documentation for Feasibility analysis and UML diagrams | 2 | E, F |
| H | draw class diagrams, deployment diagrams, component diagrams and write specifications | 5 | F |
| I | write documentation for class, deployment, component diagrams and specifications | 2 | G, H |
| J | design software interface | 2 | G, H |
| K | implement coding | 5 | J |
| L | write documentation for implementation and evaluation | 2 | I, K |

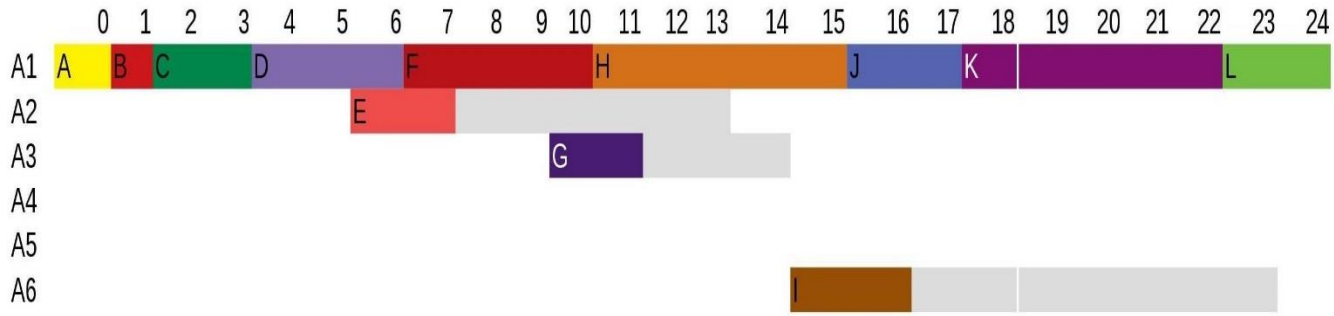



| Tasks | Total Float | Free Float |
|---|---|---|
| A | 1 – 0 - 1 = 0 | 1 – 0 - 1 = 0 |
| B | 1 – 0 - 1 = 0 | 1 – 0 - 1 = 0 |
| C | 3 - 1 - 2 = 0 | 3 - 1 - 2 = 0 |

| | | |
|---|---|---|
| D | 6 – 3 – 3 = 0 | 6 – 3 - 3 = 0 |
| E | 10 – 6 – 1 = 3 | 7 – 6 - 1 = 0 |
| F | 10 – 6 – 4 = 0 | 10 - 6 - 4 = 0 |
| G | 15 – 10 – 2 = 3 | 12 – 10 – 2 = 0 |
| H | 15 – 10 – 5 = 0 | 15 – 10 – 5 = 0 |
| I | 22 – 15 – 2 = 5 | 17 – 15 – 2 = 0 |
| J | 17 – 15 – 2 = 0 | 17 – 15 – 2 = 0 |
| K | 22 - 17 – 5 = 0 | 22 – 17 – 5 = 0 |
| L | 24 – 22 – 2 = 0 | 24 – 22 – 2 = 0 |

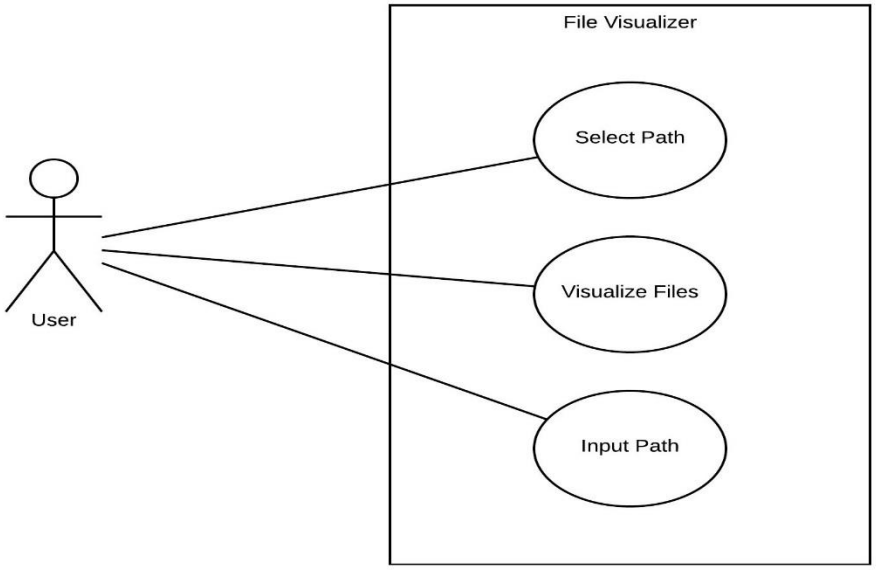
**Critical Path: A, B, C, D, F, H, J, K, L**

## Gantt Chart

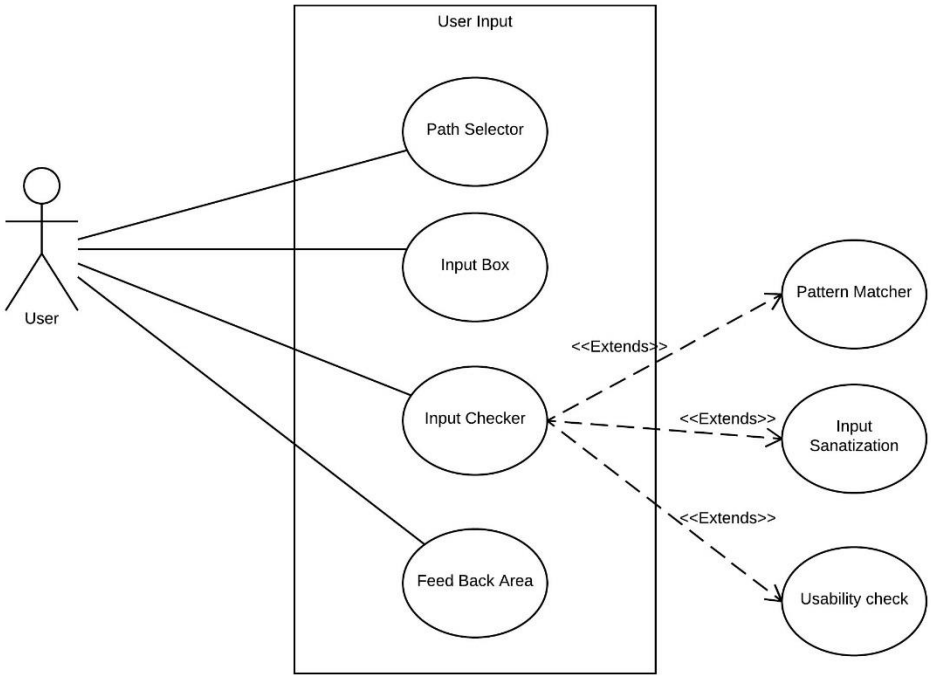



## Use Case Diagrams

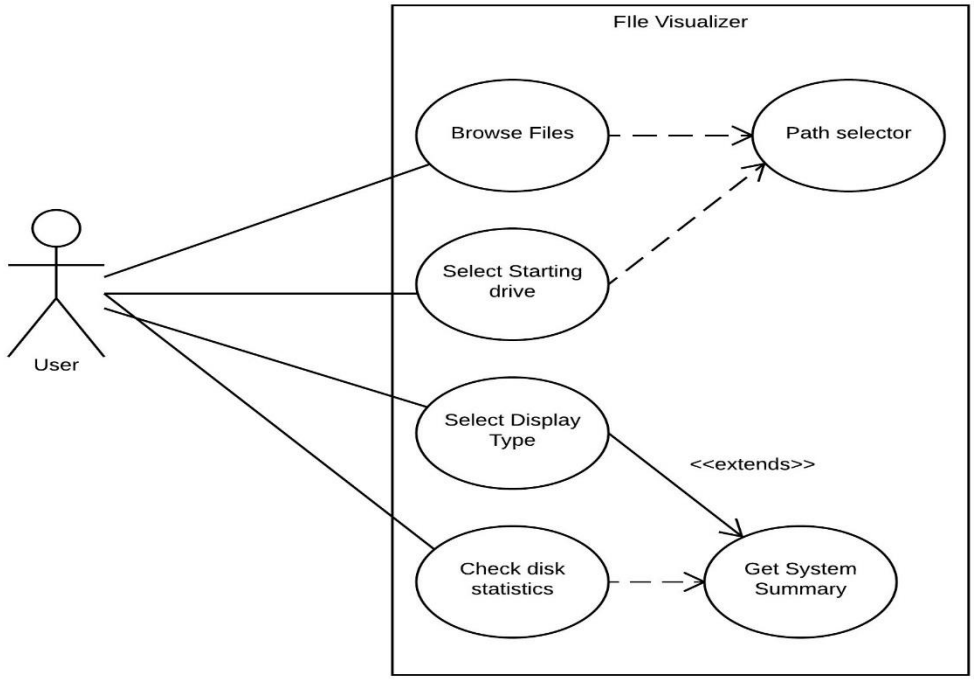
Case 1


File Visualizer
Select Path
Visualize Files
Input Path
User

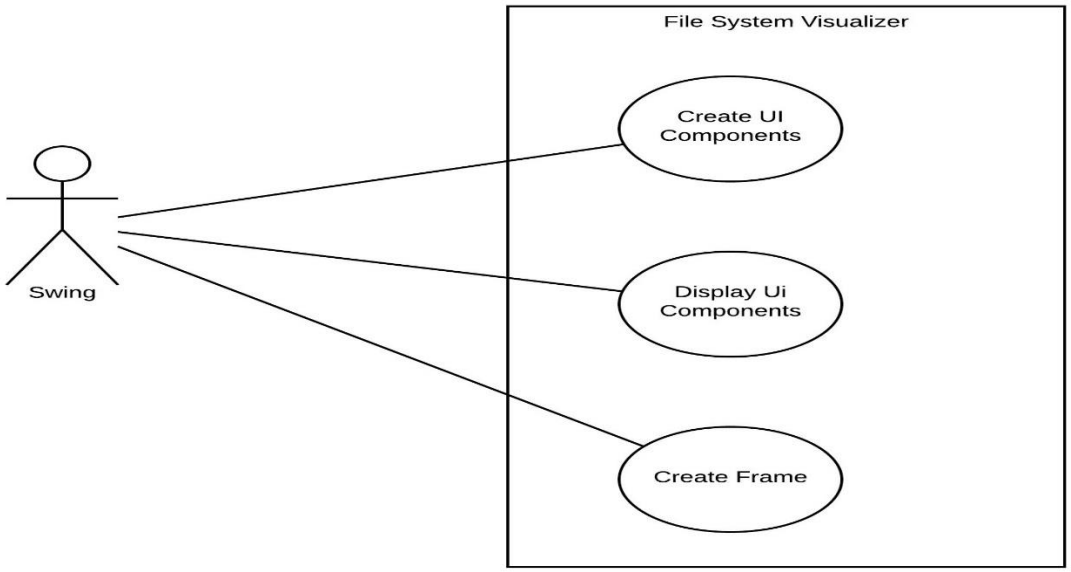
Case 2


User Input
Path Selector
Input Box
Input Checker
Feed Back Area
User
<<Extends>>
<<Extends>>
<<Extends>>
Pattern Matcher
Input Sanatization
Usability check

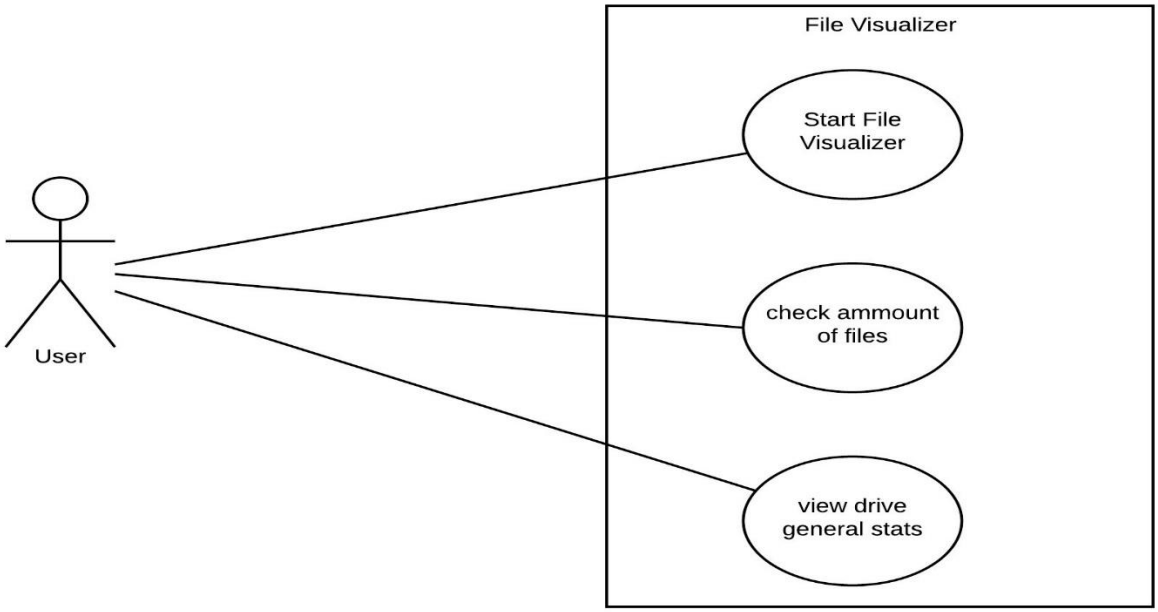
Case 3


FIle Visualizer
Browse Files
Path selector
Select Starting drive
User
Select Display Type
<<extends>>
Check disk statistics
Get System Summary

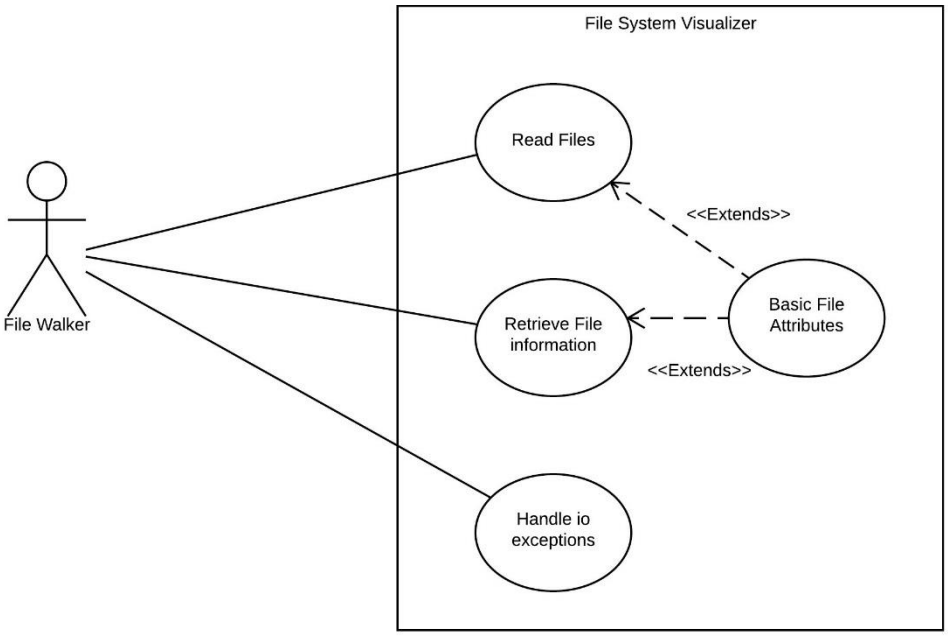
Case 4


File System Visualizer
Create UI Components
Swing
Display Ui Components
Create Frame

Case 5

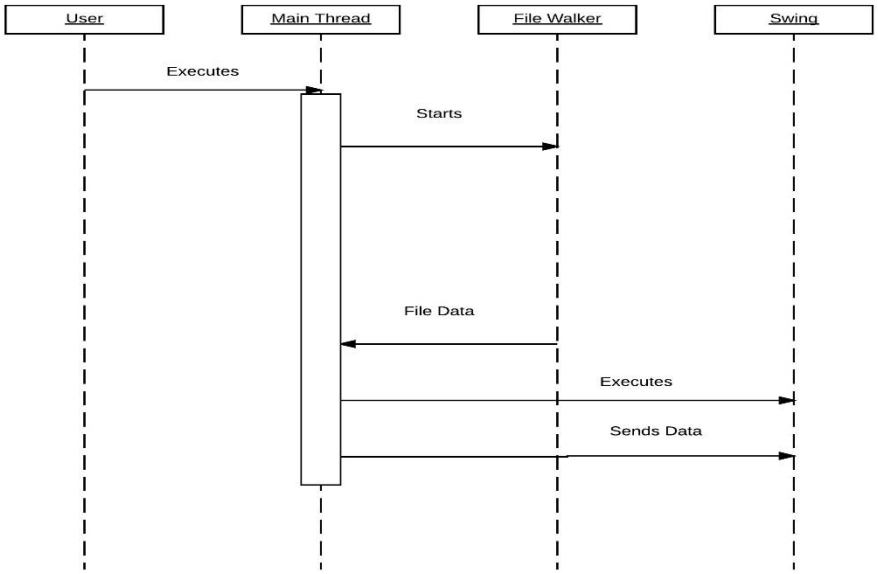

File Visualizer
Start File Visualizer
check ammount of files
view drive general stats
User

Case 6

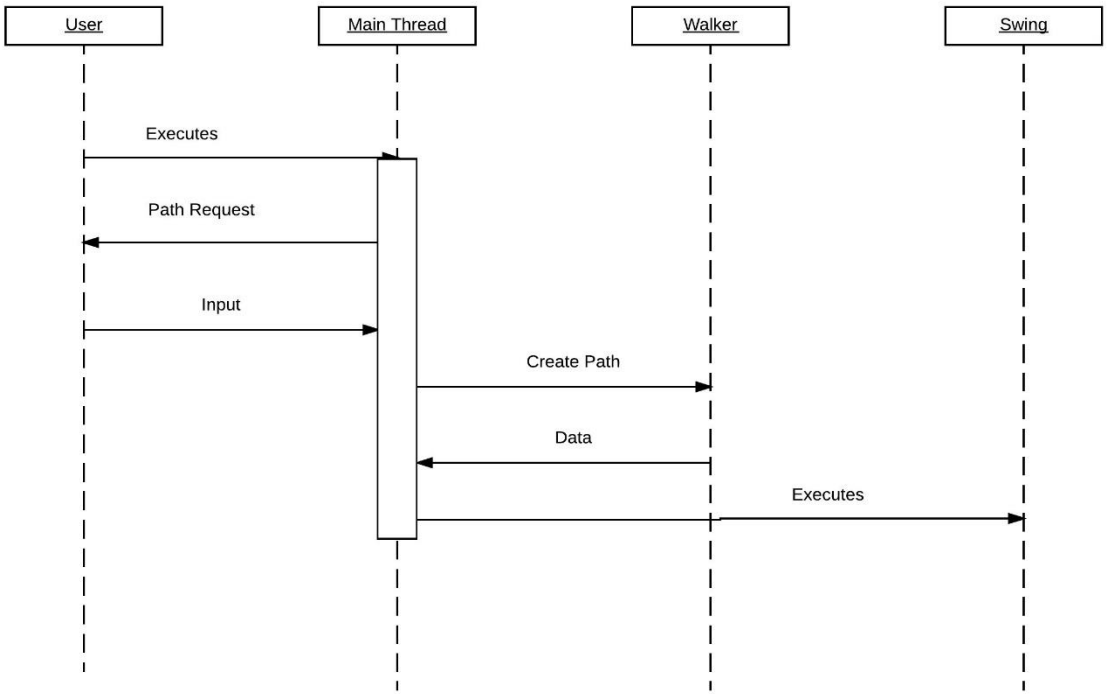

File System Visualizer
Read Files
<<Extends>>
Basic File Attributes
Retrieve File information
<<Extends>>
Handle io exceptions
File Walker

## Sequence Diagrams

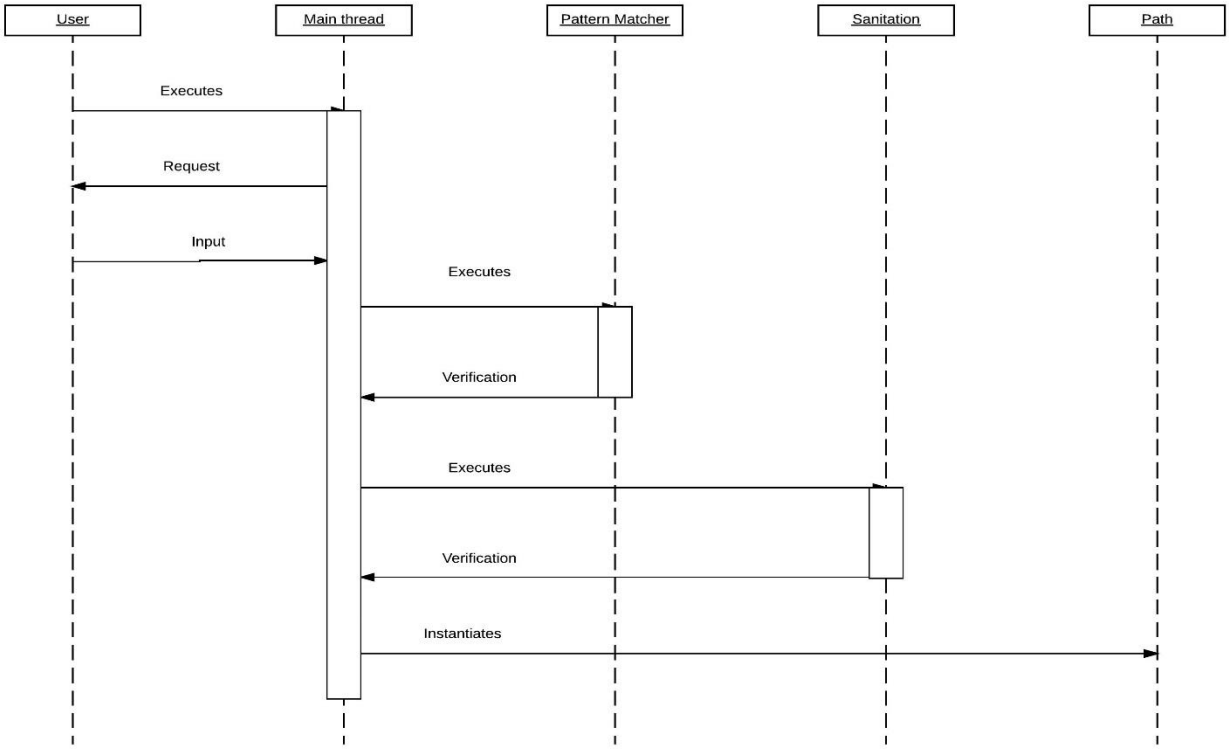
Sequence 1



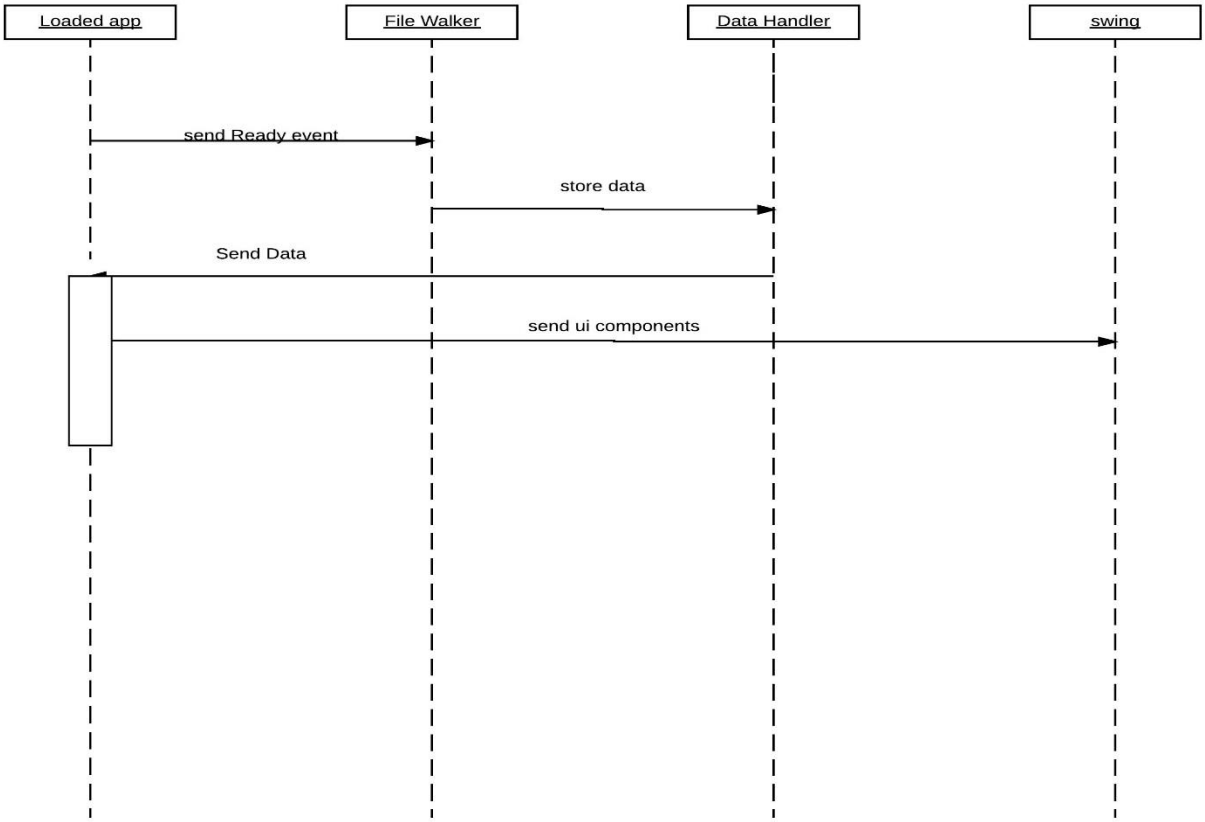

Sequence 2



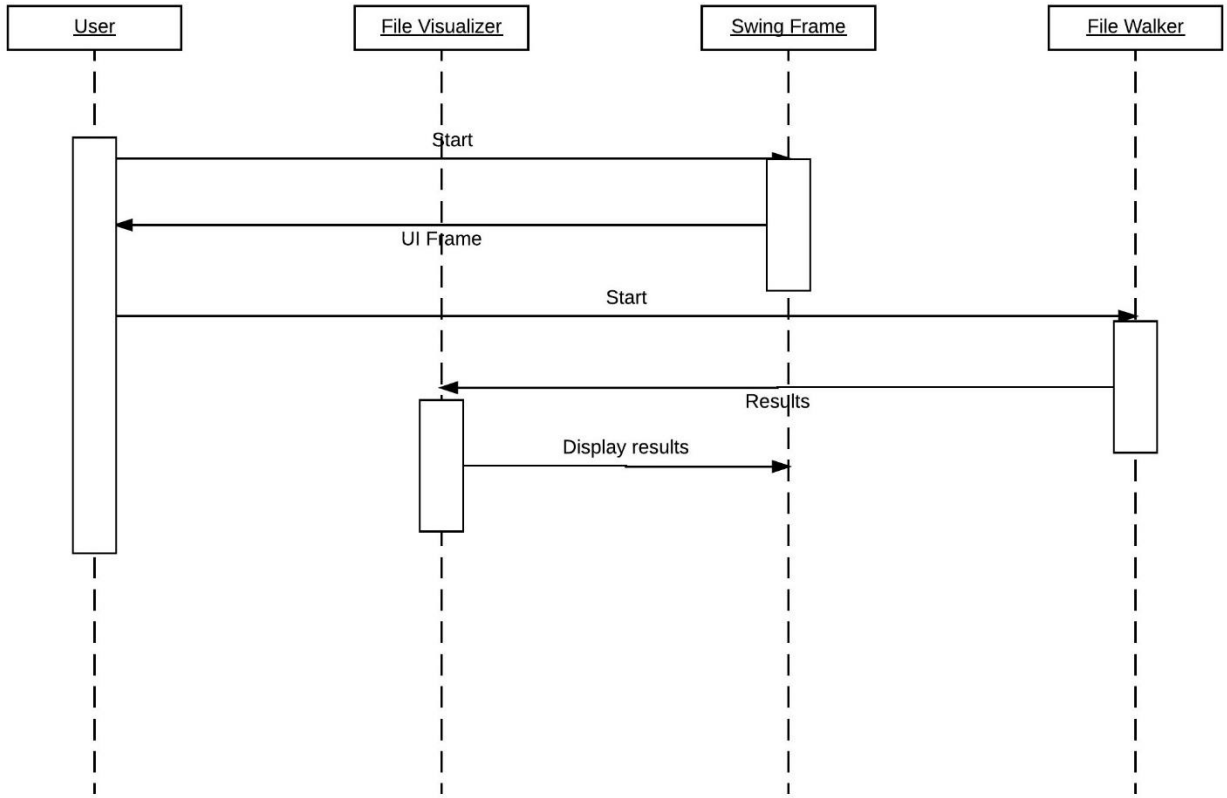

Sequence 3


User
Main thread
Pattern Matcher
Sanitation
Path
Executes
Request
Input
Executes
Verification
Executes
Verification
Instantiates

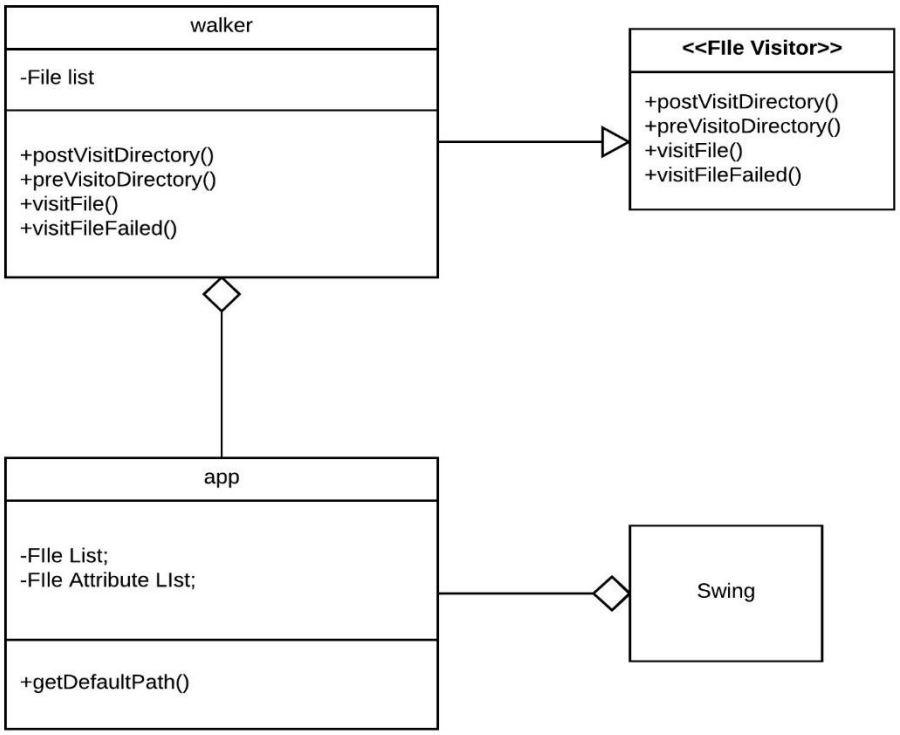
Sequence 4


Loaded app
File Walker
Data Handler
swing
send Ready event
store data
Send Data
send ui components

Sequence 5

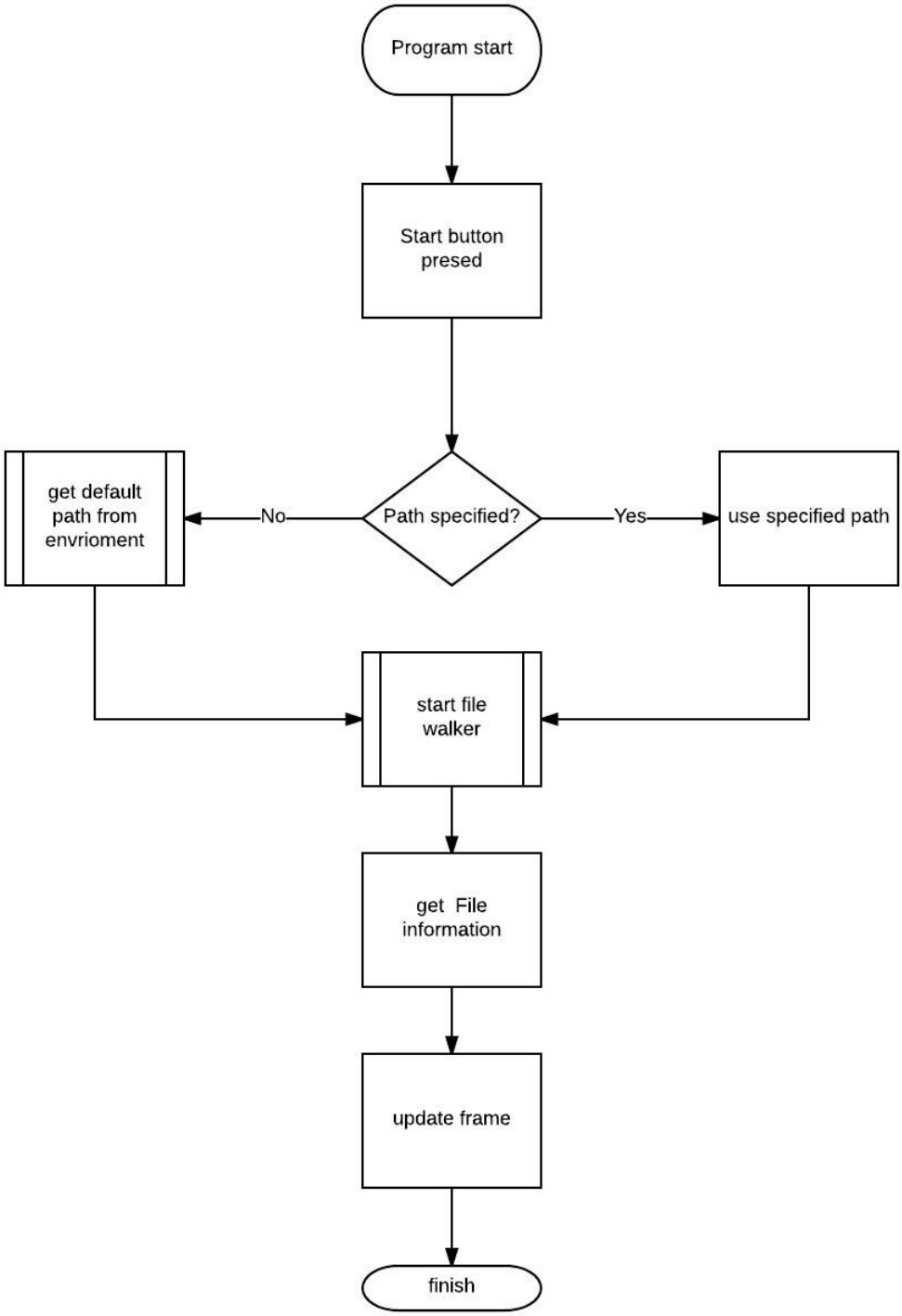


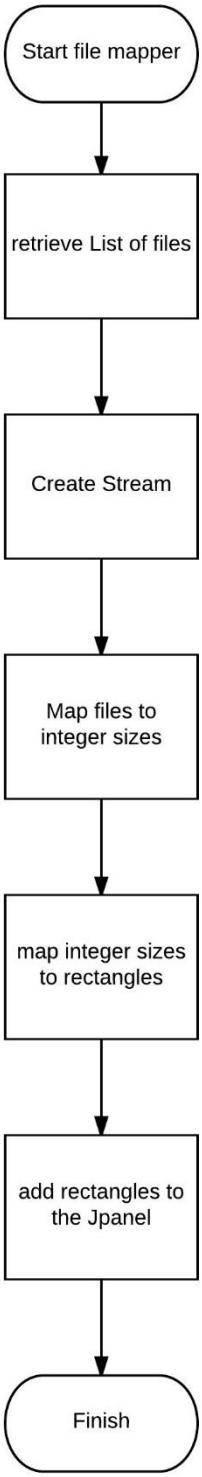

## Class Diagram

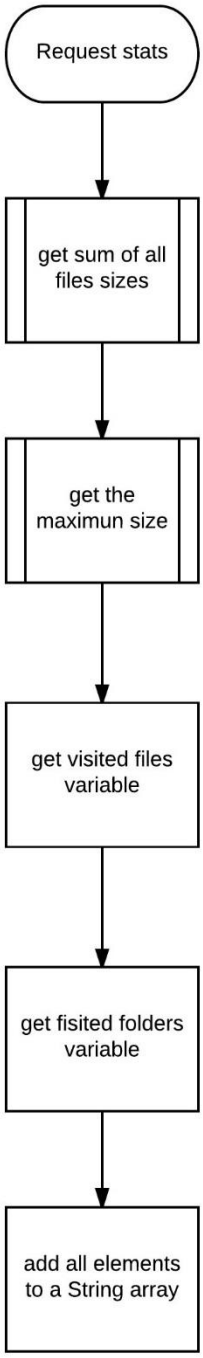


# 5. Implementation

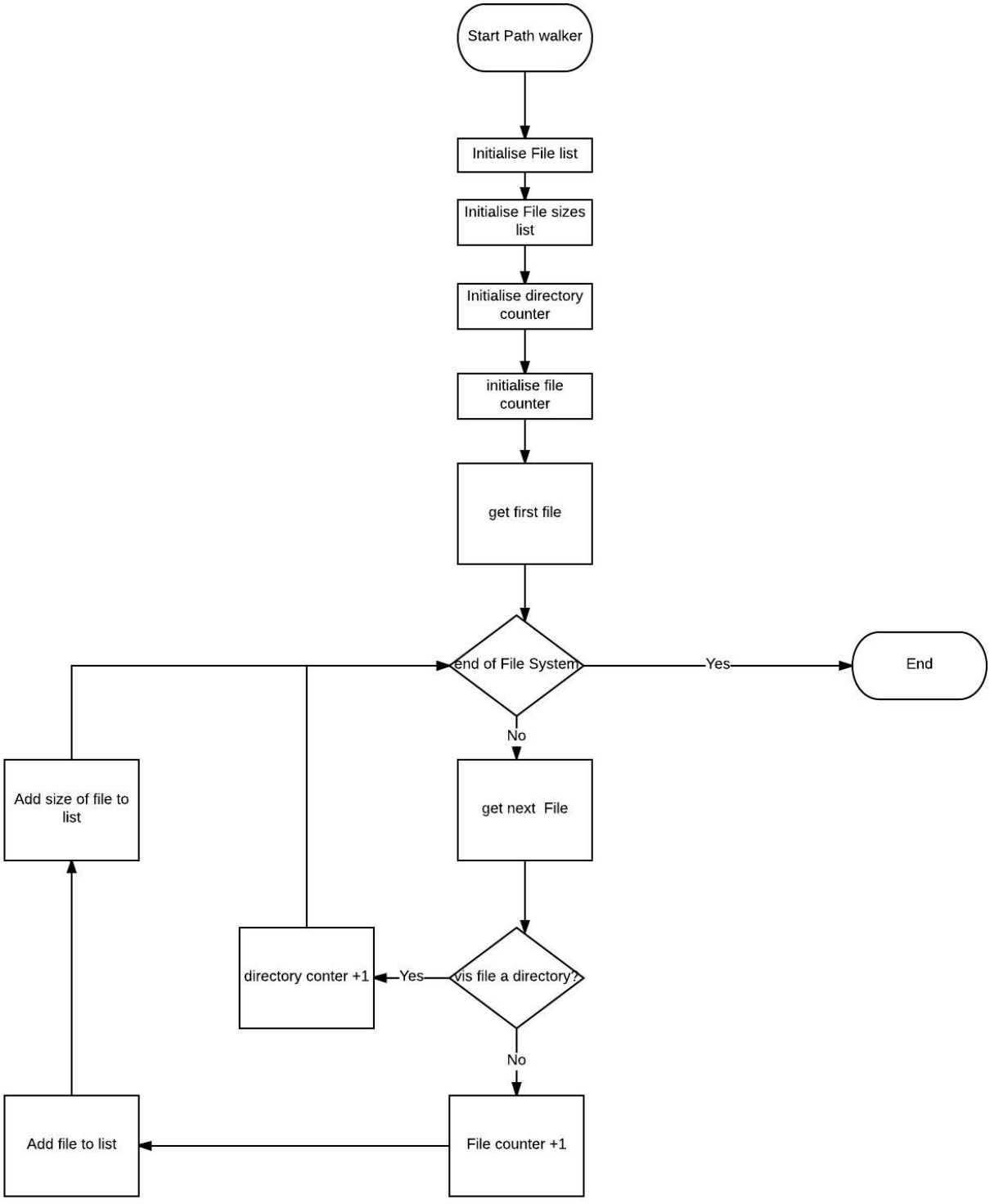
## 5.1 Algorithms (Flowchart)






This is accomplished in java by using aggregate functions and higher order functions




# 6. Evaluation

## 6.1 System Resource Requirements

**Minimum Requirements**

| | |
|---|---|
| **Operation System** | Windows 7 |
| **CPU** | Intel Core 2 Duo 2.4GHz |
| **Memory** | 2GB |
| **Hard Disk** | 2GB |
| **Media** | ROM drive or USB flash drive |
| **Software** | Java Platform SE Binary |

**Recommended Requirements**

| | |
|---|---|
| **Operation System** | Windows 8 |
| **CPU** | Intel Core i5-3230M Processor 2.60 GHz |
| **Memory** | 4GB |
| **Hard Disk** | 2GB |
| **Media** | ROM drive or USB flash drive |
| **Software** | Java Platform SE Binary |

## 7. Conclusion

In conclusion, we have tried our best of time and effort to accomplish this project. During this nearly one-month period, we have done many research about the file visualisation process, GUI, Java Swing technology and much more. But due to limited time of project duration, we have omitted some parts in the documentation and some functions in the program. However, we hope that this File Visualisation System will perform really well and to satisfy all the user requirements and the attached documentation would also fulfil the requirements of the user.

### 8. Bibliography

https://www.sas.com/en_ie/insights/big-data/data-visualization.html#

https://blogs.systweak.com/2016/10/top-10-best-disk-space-analyzer-software/

http://fsv.sourceforge.net/

https://windirstat.net/index.html

# 9. Appendix

## 9.1 Operation Manual

### User Guide Lines

This software has been tested in Linux and windows systems so it is expected to function properly on those systems. Assuming the user have the required System Requirements the program is expected to run well on his system.

The default language of the system is not supposed to affect the program, however when tested in Portuguese Windows the program had difficulties when loading system files.

### Operational Manual

The program comes with a built-in file browser that allows the user to locate graphically the starting path to visualize the files. In order to open this file browser the Browse button beside the input panel must be pressed.

Once the path has been specified press the start button which will start the traversing process. If no path has been specified the program will retrieve the default path from the environmental variables. If the user is running a windows operation system the default path will be the start of his home drive. If the operating system is Linux-based then the default path will be the home folder.

### Execution Environment

The environments where the software has been tested are Windows 10, Ubuntu Linux and Fedora. The default language on those systems was English. The version of the java Virtual Machine where it has been developed and executed was the JDK 7, the program won't work on any previous versions of java.

## 9.2 System Requirements

| | |
|---|---|
| **Operation System** | Windows 7 |
| **CPU** | Intel Core 2 Duo 2.4GHz |
| **Memory** | 2GB |
| **Hard Disk** | 2GB |
| **Media** | ROM drive or USB flash drive |
| **Software** | Java Platform SE Binary |

**********