CA326 Technical Specifications

SMART FIRE SAFTEY SIMULATOR

Conor Reilly & Eoin Clayton

Table of Contents

1.1 Overview

The Smart Fire Safety Simulator is designed to help assist people who are trying to evacuate a building in a safe and sufficient way. Currently, in buildings across the world, fire evacuation systems are outdated. This simulator will give an insight into how modern technology can help change the way we look at building evacuation to potentially save thousands of lives.

By using search algorithms, this simulator will help find the safest route out of a building, rather than the nearest exit as current systems use. This system if implemented correctly in a building can direct hundreds of people to safety. The system currently uses an A* Search algorithm, this allows the user to find the best route with the lowest cost based on certain heuristics. The fire runs off a boundary-fill type algorithm that allows it to spread as natural and realistic as possible. In a real-world scenario there would be variables such as fire alarms or open/closed doors that would change this algorithm.

The simulator will work off either a predetermined layout represented by a text file or as a customisable function allowing users to directly implement the layout or floor plan of the building they are in. The options available for customisation are; where to place a wall, where to place a human, where to place an exit and where the origin of the fire is. From here, the user can select how much time has passed since the fire has begun and the simulator will then calculate the best route for the user to take.

The main features of our project are:

- An A* Star Search algorithm that directs a user to the nearest exit.
- A fire algorithm the fills the layouts in a natural way.
- A customisable layout that allows users to select specifications for the simulation.
- A readable text file to represent the layout of a building.
- An option to scale the length of time the fire has been under way.
- A GUI based on TkInter to represent all the data and algorithms.

1.2 Glossary

Safest route
The route of passage that avoids the fire while also directing the user to the exit

A* Search
Computer algorithm that is widely used in pathfinding and graph traversal, which is the process of finding a path between multiple points while avoiding obstacles.

Nodes
A point in a network or diagram at which lines or pathways intersect or branch.

Boundary Fill
A fill algorithm that chooses a point and fills all surrounding points until entire layout is filled with selected nodes.

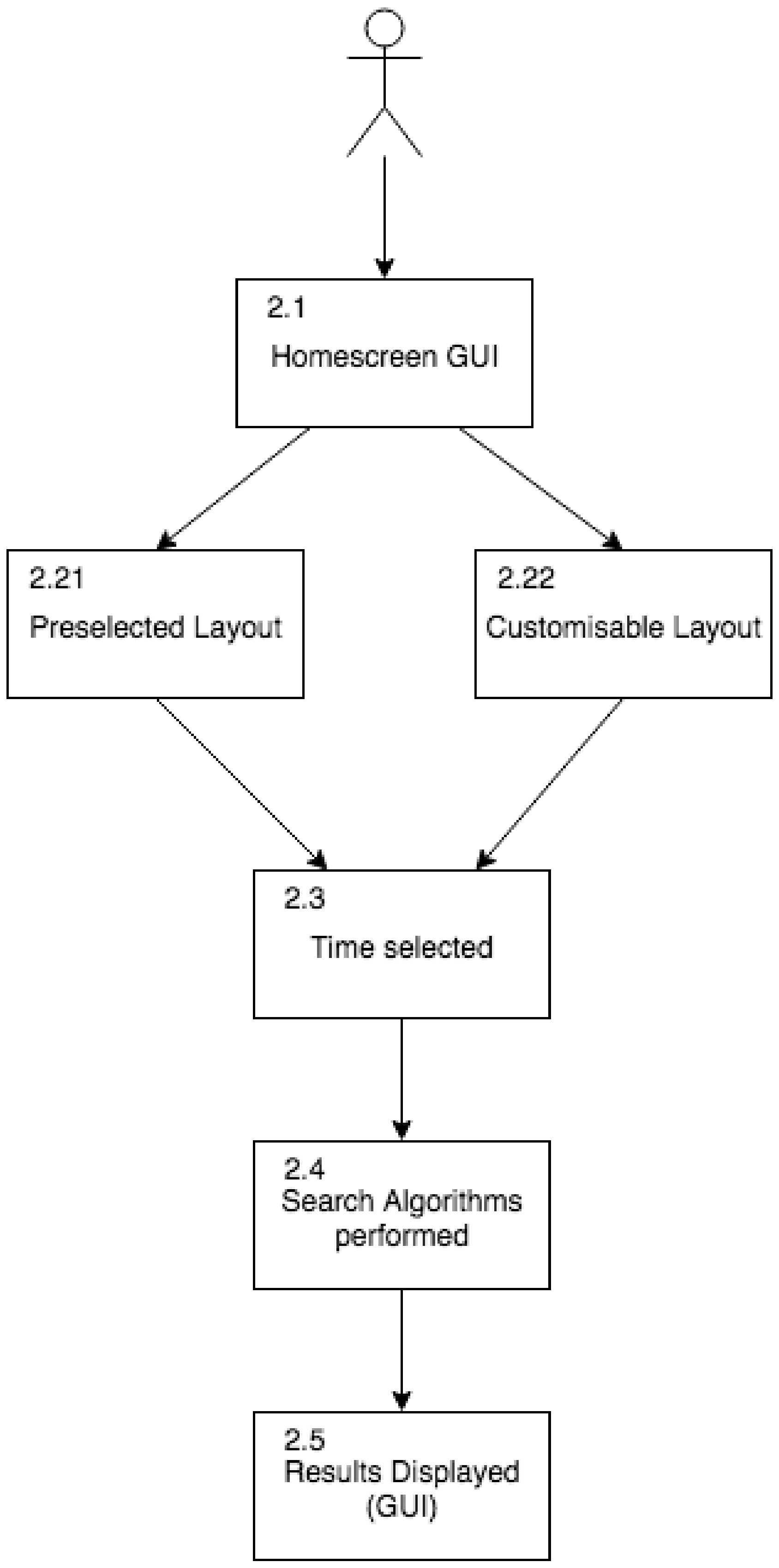
GUI
Graphic User Interface, form of user interface that allows users to interact with electronic devices through graphical icons and visual indicators.

TkInter
Python's standard GUI (Graphical User Interface) package.

Python
Python is an interpreted, high-level, general-purpose programming language.

Text File
A simple unformatted text file that can be read from a python file.

2. System Architecture



2.1 Homescreen GUI
The initial GUI is read and loaded for the user to see. Here they will have an option to select a preselected layout, customisable layout or exit the application.

2.21 Preselected Layout
Here a user can choose a preselected or saved layout of their choice, this will then immediately send them to the layout screen.

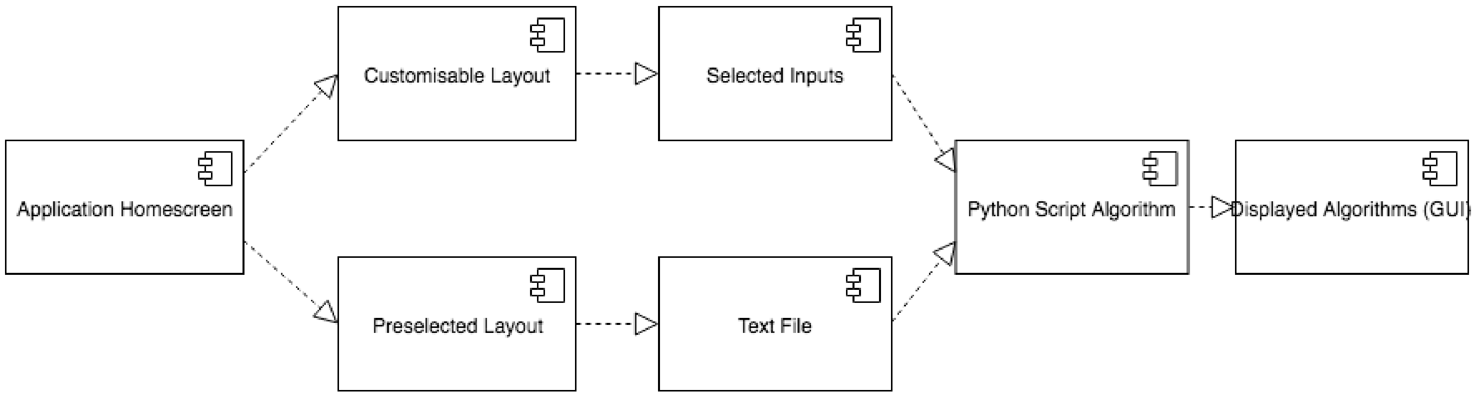
2.22 Customisable Layout
Here a user can choose their own layout, by adding a starting point, exit point and origin of fire.

2.3 Time selected
The user must select a time for how long the fire has been spreading, this is measured in seconds as accurately as a real fire might spread.

2.4 Search Algorithms performed
The system will handle both the fire boundary-fill algorithm followed by the A* search algorithm. Both will be calculated by means of python scripts each with individual features.

2.5 Results Displayed (GUI)
Both search algorithms are displayed for the user whether through customisation or preselected layout. This will display the fire spreading and the best route to leave the building.

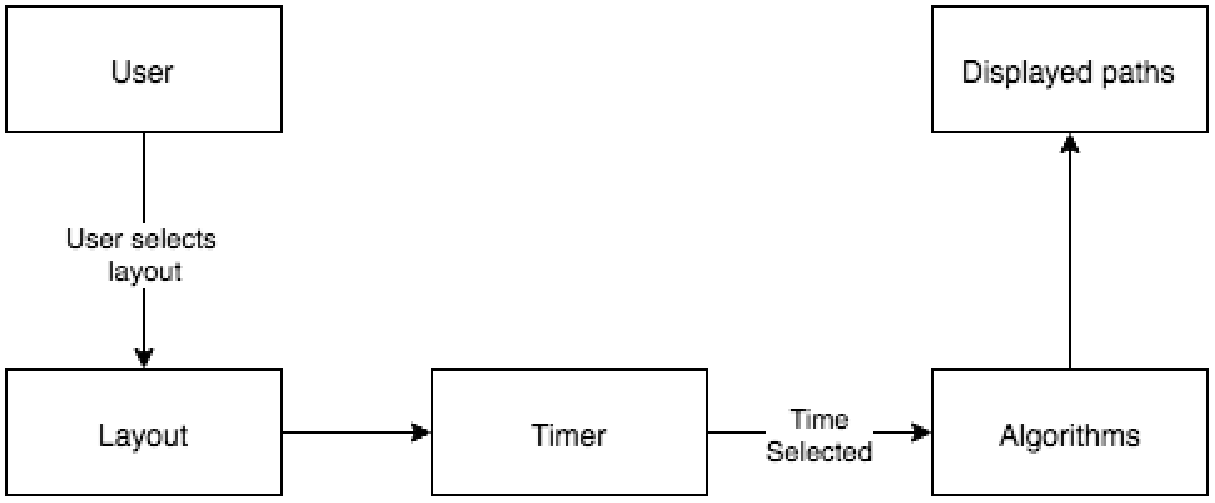
3.1 Component Diagram



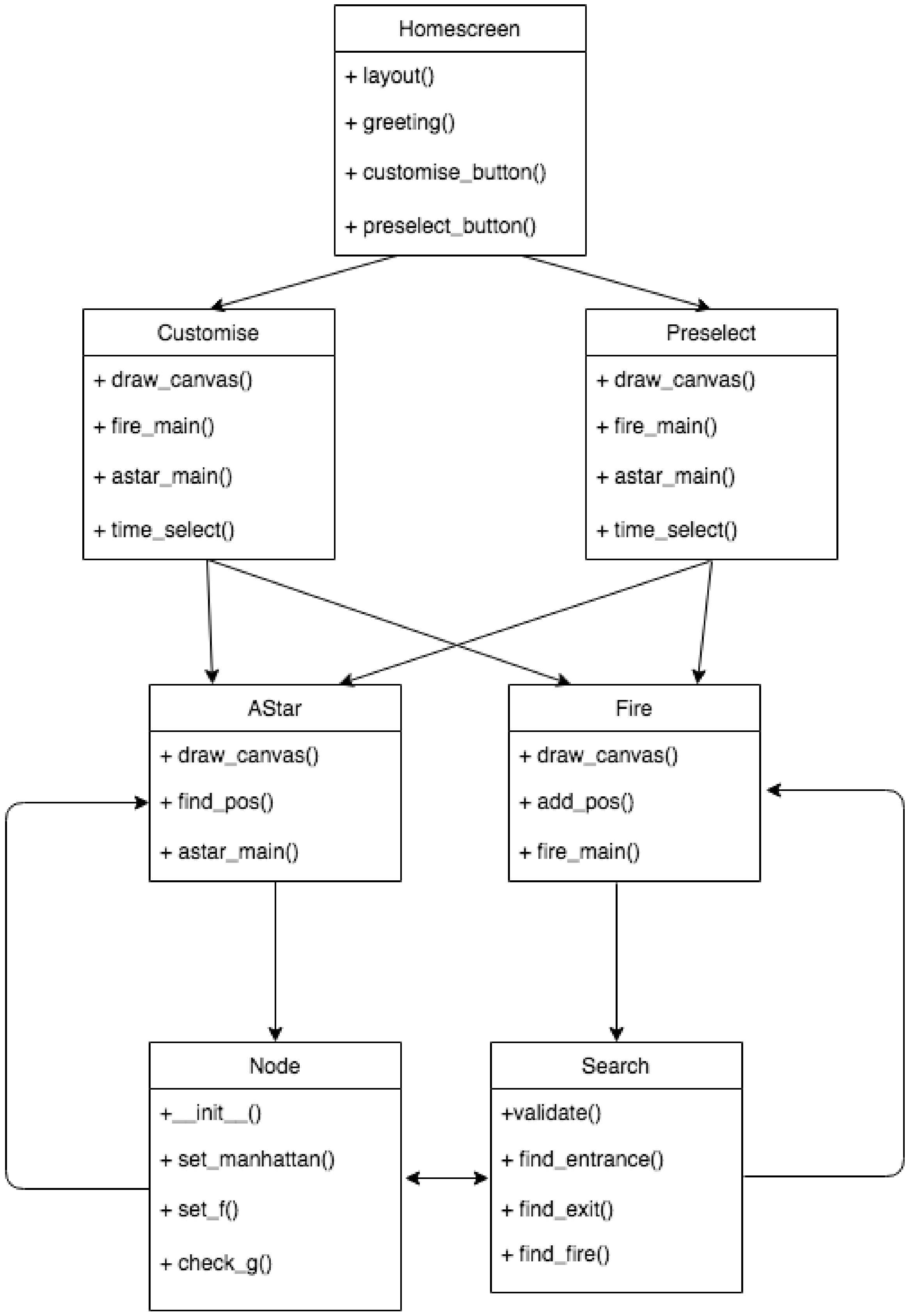
A component diagram depicts how components are wired together to form big and complex software systems. They are used to illustrate the structure of the entire systems.

Here we have our complete system from the Application Homescreen which the user is then given the option of either customisable layout or preselected layout. Each of these has unique features. Both are passed to the python scripts where the algorithms are run. These are then represented by the GUI for the user to see.

3.2 Data Flow Diagram

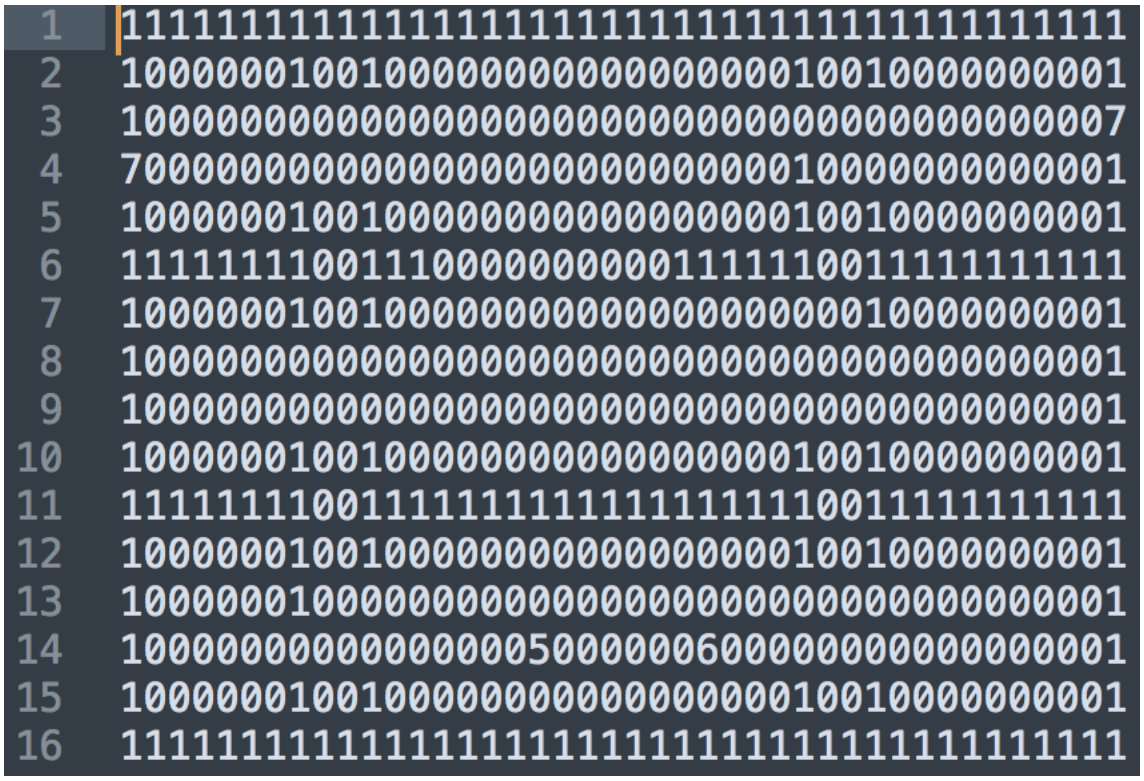


3.2 Class Diagram



4 Problems and Resolutions.

Grid creation problem.
Our initial approach to create a grid was to immediately create a fully customisable layout scenario. Here we would create an array of values, each representing a different possibility. For example, 1 being a wall and 0 being a passage way. We found this very difficult to build immediately so instead, we read our plan in from a text file and simply where a 0 was, put a blank white space. From this we were able to build our customisable option. Here is an example of our original layout we used.



```
111111111111111111111111111111111111111111
100000010010000000000000000010010000000001
100000000000000000000000000000000000000007
700000000000000000000000000010000000000001
100000010010000000000000000010010000000001
111111110011100000000001111110011111111111
100000010010000000000000000000010000000001
100000000000000000000000000000000000000001
100000000000000000000000000000000000000001
100000010010000000000000000010010000000001
111111110011111111111111111110011111111111
100000010010000000000000000010010000000001
100000010000000000000000000000000000000001
100000000000000005000006000000000000000001
100000010010000000000000000010010000000001
111111111111111111111111111111111111111111
```

A* Search
Although we had previous knowledge of how A* search works from our Artificial Intelligence and Algorithms module, we originally struggled with our own implementation. After consulting our mentor about this issue he gave us several pieces of advice and resources that may guide us to a solution.

Event Handling in TkInter

This was a new topic that neither of us had dealt with in much detail before.

The implementation of button clicks and how they would represent squares on our customisable layout or the implementation of a slider to represent time were additions we had adapted later on in our project to make the system more adaptable. We originally found these tasks quite difficult but after several tutorials on YouTube and several python forums such as StackOverflow, we managed to implement everything correctly.

Another issue we found was that event handling was unique to what operating system you were using. For example, Linux event handling and Windows are slightly different. This became an issue when transferring and comparing code with one another. Eventually we decided to each keep our own models of code with only slightly different event handlings. This allowed us to work individually and collaboratively on the code without changing major areas due to different operating systems.

5. Installation and Configuration

Installation guide

To successfully run the simulator ensure you have the latest version of Python and tkinter installed on your PC

Install Python for windows:

Go to https://www.python.org/downloads/windows/

Hover over the downloads tab on the nav bar and then click the download box
Open the file once it is downloaded and follow the instructions on screen

Install Python for mac:

Go to https://www.python.org/downloads/mac-osx/

Hover over the downloads tab on the nav bar and then click the download box
Open the file once it is downloaded and follow the instructions on screen

Installing and running Smart Fire Safety System

Go to https://gitlab.computing.dcu.ie/claytoe2/2019-ca326-claytoe2/tree/master/src

Download the file by pressing the cloud icon in the top right corner

Open your Command prompt

Change into the files directory by typing cd followed by the name of the file

Change into the 'src' folder directory by typing 'cd src'

When in the src directory type 'py main.py' on window or 'python3 main.py' on mac

Tkinter will open and the program will start running