Intention, Issues, Design, UML Diagrams and Development Process

1 Intention of Software

Intention was to create an animated agent-based model using 25 Drunks within a 300 by 300 raster grid environment. The 'drunkenviro.txt' file, shows the location of pubs and homes. This file was read in and forms the environment. Once leaving the Pub, Drunks were given a number that corresponds with their house number. Their movements over each square/point were tracked and a density map produced of this movement once all 25 Drunks reached their home. The Drunks were to be aided home by Police who would guide them to their respective homes. The density map was to be saved to a file as text.

2 Examples of Issues during Development & Overcoming them

- Current issues in Version 1 (most up-to-date model): Once the animation ran, Drunks appeared at their homes from the start. To rectify this within the agentframework.py under the Agent class, the x and y coordinates were specified so Drunks started off in the centre (at the Pub) (using *self.x = random.randint(128,148)* and *self.y = random.randint(138,159)*) and moved out randomly after the model was run.
    - The only issue following this was that Drunks moved out randomly but were not very successful at finding their homes in a relatively short space of time. The functions 'check_police', 'check_drunks' and 'move' were key in enabling the Police to gather a number of Drunks and guide them to their homes. However Drunks guided still struggle to reach home, as Police seem to guide them on the coordinates (0,0) to (300,300).
    - Once agents reach (300,300) an error (IndexError: list index out of range) shows. Despite multiple attempts this error has not be rectified. The model runs until this point. All other Drunks not guided move randomly in the direction of homes while this is going on.

- Current issue in Version 1 and 2: Despite changing the Excel document code, 1 is not added to the csv when an agent moves over an area. In Version 1, the Density can still be seen as the agents produce a purple-white trail showing agent movements in real time.

- Current minor issue: It is recommended to open the Model on a Windows Operating System. However, when doing so a pop up box entitled 'Figure' will appear alongside the 'Model' box. This is a pre-known issue that occurs with Windows. This Figure box is not used. To close the Model fully before restarting and rerunning the Model, both the Model and the Figure box need to be closed and the red square in the IPython console selected (if it is not already greyed out).

- Solved issue: When creating the animation, it ran automatically after pressing the green arrow. However at times, there needed to be a slight delay before the animation started. Therefore a dropdown Menu was added with the function to 'run' and 'close' the model at a time to suit the user.
- Solved issue: Police located at the pub from the start and moved randomly but did not move very far. Fixing this involved working in the agentframework2.py, increasing the number of iterations within the 'move' function (e.g. from self.x ± 1 to self.x ±5) so the Police moved faster

and covered greater distance. Also the random coordinates within the Police '__init__' function were changed from starting at the pub to starting anywhere in the grid (e.g. self.x = random.randint(128,148) to self.x = random.randint(0, 300)). Only the Drunks had to originate from the Pub.

3 EXAMPLES OF TESTING COMPLETED

- Throughout the Agent Framework and Model, specific text is printed or returned to show a certain action has occurred. For instance, from the start the agent specific number and whether they are at home ('True') or not ('False') is printed. When the check_drunks function is created in the agentframework2.py, the closest drunk is returned so help can be given by Police to guide nearby Drunks home.
- Statements with 'if' and 'else' are also used. If one outcome does not occur then another outcome will happen instead. For instance, if a Drunk reaches their home, they turn green. If the Drunk does not reach home, they turn red.
- All testing is shown by comments and 'Returns' in the text under defining a function.

4 THOUGHT PROCESS BEHIND THE SOFTWARE DESIGN

While designing the Graphical User Interface (GUI) and Website, the ideas behind Information Architecture creating clear, structured design to aid usability and functionality were followed (Rosenfeld, et al., 2015).

Within the GUI, clearly labelled and ordered buttons helped understanding of how to run the model (Rosenfeld, et al., 2015). Over 25 different colours and shades of colour were used to display Homes, Drunks and Police, but it was important to consider accessibility to all, particularly those colour blind. A colour blindness. A colour blindness simulator, Colour Oracle, was used. This applies a full screen filter showing common colour vision impairments (deuteranopia (common), protanopia (rare), tritanopia (very rare) and greyscale) (Jenny & Kelso, 2018).

Both the Website and GUI were tested using this application. All colours within the GUI were clear and easy to distinguish, particularly between the agents and homes. Improving accessibility was one of the reasons why the Website text, background colour and the sidebar and footer were different and distinguishable colours – white, black, grey and 2 different shades of blue (Benyon, 2019). Hyperlinks also changed to purple text after being clicked on once.

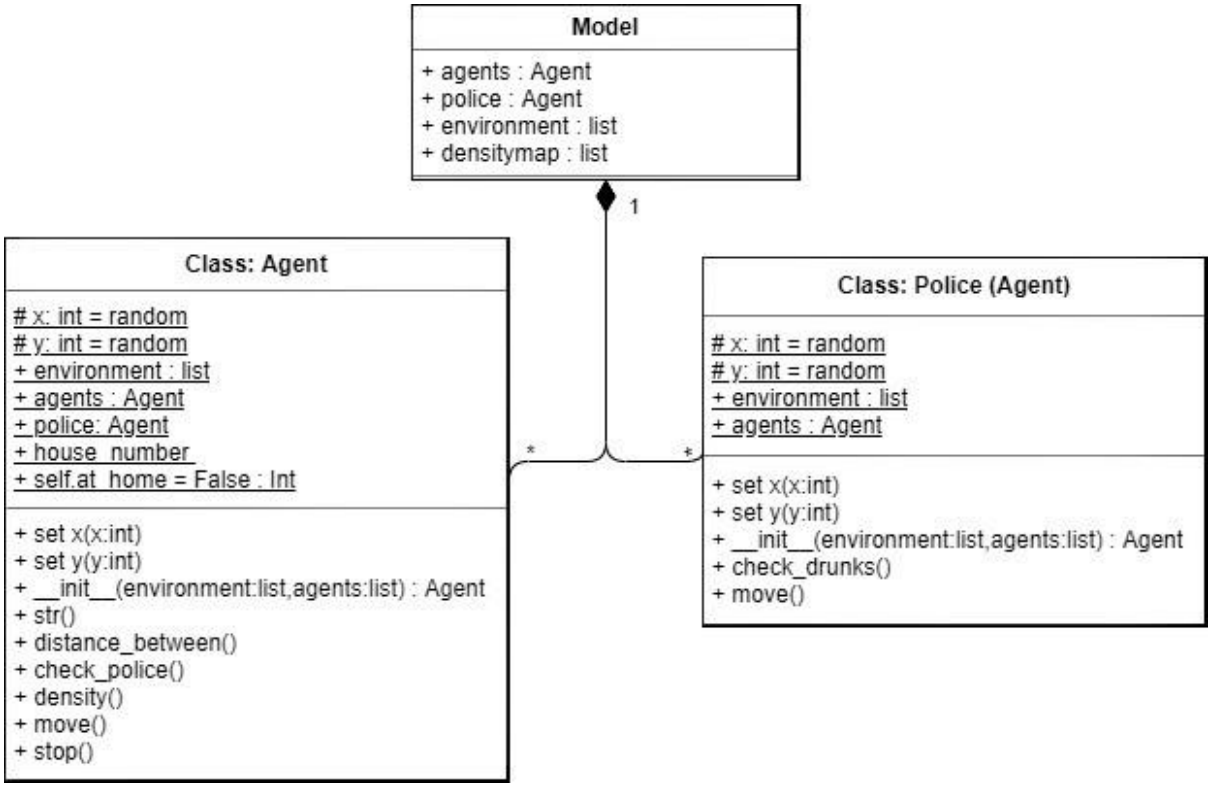
The Website layout was separated with headings, main text, photos, footer and sidebar. The sidebar and footer remained different colours and were located away from the main text to show a clear distinction between the website text and the most important hyperlinks (Edmonds, et al., 2007). Simplified sidebar text hyperlinks were ordered to show which link the user should select first if new to the site (Rosenfeld, et al., 2015). The top hyperlink is most important. The Home Page describes the main purpose of the website and which hyperlinks to click on next, showing a clear starting path for the user to navigate (Wurman, 1989). Sidebar hyperlinks change colour when the mouse hovers over them to show which one would be clicked by the user. This provides further clarification that it is the correct link to select (Wurman, 1989).

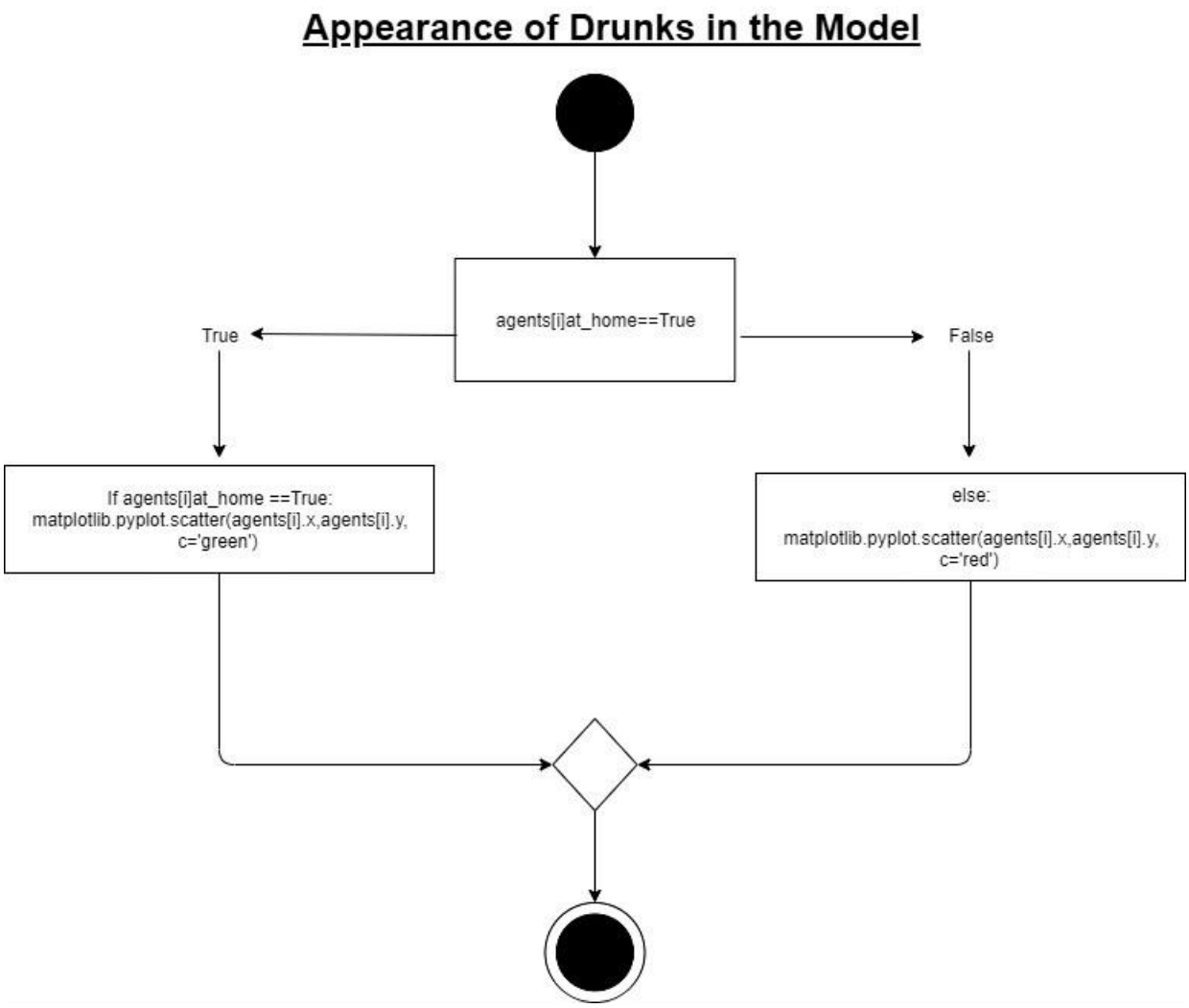
Throughout the website, clear simple wording was used and key information placed in bold or italics (Wurman, 1989). HTML coding used, particularly '@media' code, enabled those using technology other than a computer to have a slightly different styled website. In this instance, only the maximum height was specified, as the website was already sufficiently clear to be used on a range of devices.

The website was tested on a laptop, mobile and tablet to ensure usability, functionality and understanding (Rosenfeld, et al., 2015). Hyperlinks were also placed within the text rather than added to the sidebar as these hyperlinks would only be relevant on that particular webpage. Photos, screenshots and bullet points were included to help break up the text. On the Model webpages screenshots were used to aid understanding of the Model. Key information was placed closer to the top of the page, while further detail was described below.

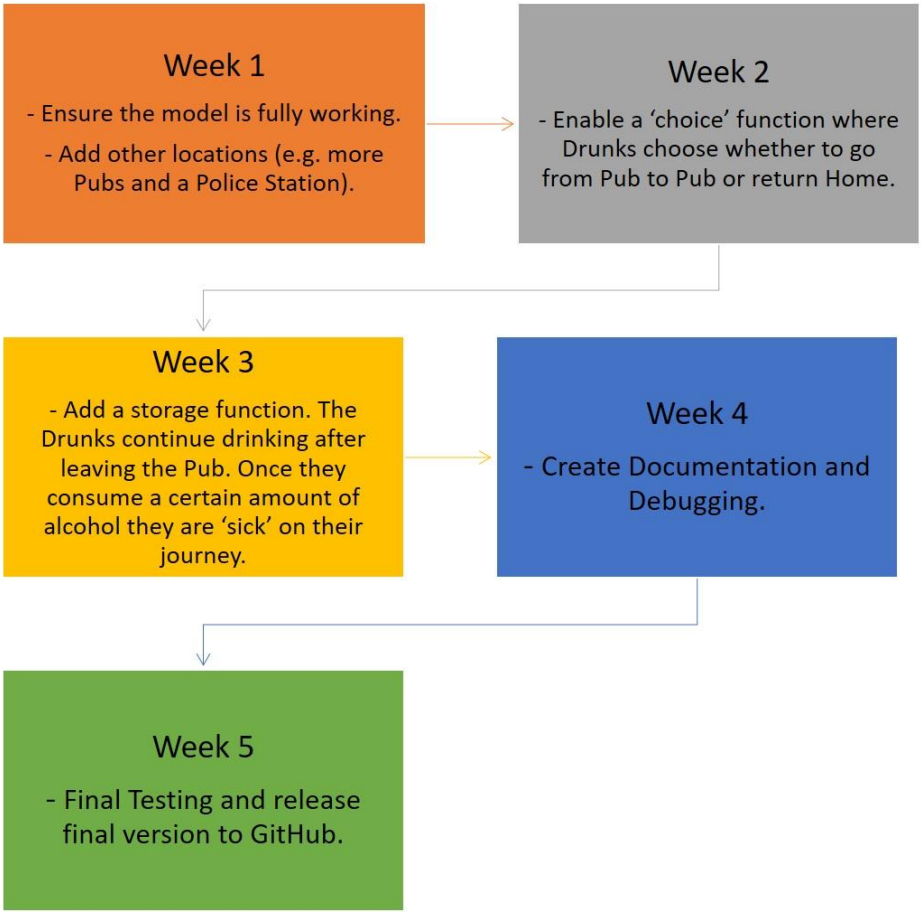
5 UNIFIED MODELLING LANGUAGE DIAGRAMS (UML)

A selection of UML Diagrams for the Model created using (Draw.io, 2020).




Appearance of Drunks in the Model
agents[i]at_home==True
True
False
If agents[i]at_home ==True:
matplotlib.pyplot.scatter(agents[i].x,agents[i].y,
c='green')
else:
matplotlib.pyplot.scatter(agents[i].x,agents[i].y,
c='red')

6 FUTURE DEVELOPMENT ROADMAP 2020



7 REFERENCE LIST

Benyon, D., 2019. *Designing User Experience: A Guide to HCI, UX and Interaction Design.* 4th ed. London: Pearson.

Draw.io, 2020. *Draw.io.* [Online]
Available at: https://www.draw.io/
[Accessed 22 January 2020].

Edmonds, A., White, R. W., Morris, D. & Drucker, S. M., 2007. Instrumenting the Dynamic Web. *Journal of Web Engineering,* 6(3), pp. 244-260.

Jenny, B. & Kelso, N. V., 2018. *Colour Oracle.* [Online]
Available at: https://github.com/nvkelso/color-oracle-java
[Accessed 20 January 2019].

Rosenfeld, L., Morville, P. & Arango, J., 2015. *Information Architecture: For the Web and Beyond.* 4th ed. Sebastopol: O'Reilly.

Wurman, R. S., 1989. Hats. *Design Quarterly,* 145(1), pp. 1-32.