Software Requirements

This document contains the software requirements for the software being tested, i.e., the Informatics Large Practical (ILP) also known as the PizzaDronz app. This document covers *the stakeholders, the functional requirements, and measurable attributes of the software*.

Stakeholders

We define stakeholders as someone who would be directly impacted by the software. This section outlines the stakeholders for the PizzaDronz software (software being tested) as well as why said stakeholders are relevant to the software. The stakeholders are:

1. The students (Customers): The students are the primary target customers of the application. They are relevant as stakeholders because they would be the end-users of the system and they are the customers are whom the entire system is based.
2. The Pizza Vendors: The pizza vendors are the service providers for the system. So, these pizza providers represent all the restaurants as well as the menus each pizzeria provides. They also allow the orders to be placed and consequently collected by the drones. They are a relevant stakeholder since they would be proving the service the system is based on top of.
3. The Drone Providers: The drone providers would provide the basis for the service provided by the system making them another service provider. Since the software is only half of the system that we are working on, having the physical drones would be needed to fulfil the orders placed on the system.
4. The Engineers: The engineers represent the Software as well as the Hardware engineers that would work towards making the service operational. Since, both the Software and the Hardware engineers, are working towards making the service operational (each in their own way, working on the necessary software or implementing the drone’s functionalities) the engineers are therefore a relevant stakeholder.
5. The Government: The Government would act as a relevant stakeholder in multiple ways, they would need to regulate the service since the service would need to abide by the relevant air traffic and not cause any disruption to the residents. The government could also grant funds to make the service profitable.
6. The public in the operational area (The Environment): Since the drones will operate on public areas, we need to make sure that they don’t risk the safety of the public/environment. We also need to ensure that the drones are not threatening to the public/environment. The above stated reason makes the public in the area a relevant stakeholder.

Functional Requirements

The ‘functional requirement’ of the system is defined as “*what the system should do*”. Along with specifying the functional requirements for the system, we also categorize them into functionality, correctness, reliability, recoverability, maintainability, usability, efficiency, fairness and performance. The functional requirements are:

1. The system should be able to portray important locations like the drone’s position, pizzerias and the delivery point (Appleton Tower) in terms of latitude and longitude. This requirement could be categorized under functionality.
2. The system should consider two locations that have a Euclidean distance of 0.00015 between to be ‘close’ (so if the drone is ‘close’ to a point, we consider the drone to be at the point). This requirement could be categorized under functionality.
3. The system should not let the drone exceed 2000 flying moves in a day. Since we consider the drone to be able to make a total of 2000 moves before it runs out of charge, having a drone make more than 2000 moves (running out of charge) while in operation, would cause the loss of the drone and potentially cause harm to civilians or the environment. This requirement can be categorized under functionality and reliability.
4. Every flying move that the drone makes should result in the drone moving a distance of 0.00015 degrees. This requirement could be categorized under functionality and correctness.
5. Each flying move that the drone makes the drone move along a straight line in one of 16 directions (primary, secondary, or tertiary compass directions). This requirement can be categorized under correctness and functionality.
6. The drone has two ‘move’ capabilities: moving or hovering. The hovering should take place when the drone is supposed to pick up or deliver an order. This requirement could be categorized under functionality and correctness.
7. The system should start and end at Appleton Tower every day it is in operation. Another additional requirement would be that the drone is close to Appleton tower when it is about to run out of charge (close to running out of 2000 flying moves). The requirement could be categorized under functional, reliability and correctness.
8. The system ensures that the drone moves are not what could be resulting in an ‘illegal’ move. An illegal move would be if a drone flew into a ‘no-fly-zone’ or if once the drone enters the ‘central area’, it does not leave until it makes it delivery in Appleton tower. Another variation of an illegal move would be if the drone leaves the ‘central area’ it should not enter the central area again until it picks up the pizza. Enforcing the concept of a ‘no-fly-zone’ or a ‘central area’ was done to minimize the obstruction to the public and more specifically to the students in the vicinity of the drone’s operational areas. This function could be categorized under correctness, functionality, reliability, safety, and recoverability.
9. The system should be able to retrieve all relevant data (restaurant locations, No-fly-zone details, Central area details, restaurant menu and order details) from a REST-server. The retrieval of the data would be done from the URL passed to it as a command line argument. This requirement can be categorized under liveness, maintainability, and functionality.
10. The system should validate each order for the given date of operation. In case the order is not valid, due to any reason, label it with the appropriate order outcome depending on which information was invalid. The orders can have one of 10 outcomes (2 of which depict a valid order). This could be categorized under correctness and functionality.
11. The system should be able to generate the flightpath that the drone takes in a single day (abiding by all the constraints) in 60 seconds or less. This requirement can be categorized under efficiency and safety.
12. The system should also aim to maximize the number of orders fulfilled in a day. This requirement can be categorized under correctness, fairness and performance.

Measurable Attributes

The measurable attributes in the system are:

1. Efficiency: We measure the efficiency of software by determining the amount of computational resources required to complete a task, and how quickly it is performed.
2. Recoverability: We measure the recoverability of software by evaluating its ability to restore normal function after a failure, and the time it takes to do so.
3. Performance: We measure the performance of software by determining how well it performs its intended tasks and how quickly it completes them.
4. Availability: We measure the availability of software by evaluating the proportion of time it is accessible and usable for its intended purpose.
5. Usability: We measure the usability of software by determining how easily it can be used by its intended audience, and how well it meets their needs.
6. Reliability: We measure the reliability of software by evaluating its ability to perform its intended tasks accurately and consistently over time.
7. Security: We measure the security of software by evaluating the measures in place to prevent unauthorized access, use, disclosure, or destruction of data.
8. Maintainability: We measure the maintainability of software by evaluating the ease of making changes and fixing defects, and the cost of doing so.
9. Scalability: We measure the scalability of software by evaluating its ability to handle increased loads and continue to perform well as demand grows.