1. Analyze requirements to determine appropriate testing strategies [default 20%]

*1.1. Range of requirements, functional requirements, measurable quality attributes, qualitative requirements, …*

A wide range of requirements was identified in the Requirements Document. The following combination of attributes in the Requirements Document was identified to be the most diverse and important:

* Safety Requirement 1.1.4: The drone must not fly into pre-defined no-fly areas on the Rest server where people are crowded together to minimize the consequences of impact in case of hardware of software failure.
* Correctness Requirement 1.2.4: The orders retrieved from the Rest server for the required day should be validated for correctness (details in the document).
* Suitability Requirement ????: The system shall aim to deliver as many orders as possible for any given day. (decide which part it belongs to)
* Efficiency Requirement 2.1: The system should aim to have a runtime of 60 seconds or less before terminating. This runtime should be achieved on a machine that has its system specifications similar to or better than the machine student.compute.inf.ed.ac.uk when it is lightly loaded (i.e., when the who command lists fewer than ten users using the machine).

1.2. Level of requirements, system, integration, unit.

The requirement 1.1.4. shall be tested primarily using unit tests, to make sure the drone never traverses into a no-fly area in a wide range of different circumstances. Similarly, the algorithm to satisfy requirement 1.2.4. can be tested using unit tests, by creating different individual orders with valid/invalid details and validating them, before testing integration with the Rest server for data retrieval. The system tests shall be performed by providing the system with a wide range of orders, validating them, computing flightpaths, and observing the outcome files are consistent with the expected result.

1.3. Identifying test approach for chosen attributes

Performance testing for requirement 2.1. mostly focuses on the drone algorithm, as it is likely to take most of the time, compared to the validation mechanism. Thus, a wide range of different orders shall be created, to restaurants that are located close and far from the drone starting point and the computation time will be recorded for any given day. The orders for testing all requirements shall be created both manually and using a randomisation tool.

1.4. Assess the appropriateness of your chosen testing approach

The performance tests are difficult to perform due to the lack of order data are when generating realistic loads for testing. In addition, the Rest server access time may vary and thus influence the performance tests. Furthermore, the Rest server data is not currently modifiable, as it has been created by a different team working on this project. This makes it more difficult to test the correctness requirement 1.2.4. using integration tests with the Rest server for data retrieval.

2. Design and implement comprehensive test plans with instrumented code [default 20%]

2.1. Construction of the test plan

2.2. Evaluation of the quality of the test plan

2.3. Instrumentation of the code

The Test Plan Document outlines instrumentation and scaffolding necessary for the testing. The proposed scaffolding ensures a variety of tests to meet the requirements. Some of the scaffolding is necessary to ensure the individual parts of the system work correctly without reliance on others (for example to make sure objects can be deserialized correctly without server connection, scaffolding is required to imitate the connection), while other is required when generating data for performance testing or ensuring the correctness requirements (like no-fly zone requirement) hold.

2.4. Evaluation of the instrumentation

The Test Plan Document currently contains extensive plans for scaffolding to generate synthetic data. While the randomly generated data together with manual testing can provide more thorough testing, it might require more time and effort than it has been allocated for this project, which might decrease the quality of the project. The possible improvement could be generating data on the server – currently the data on the server is not modifiable due to the restrictions by the Informatics Large Project course organisers. The data could be simulated by creating another server, but this requires additional finances.

3. Apply a wide variety of testing techniques and compute test coverage and yield according to a variety of criteria [default 20%]

3.1. Range of techniques

3.2. Evaluation criteria for the adequacy of the testing

3.3. Results of testing

3.4. Evaluation of the results

4. Evaluate the limitations of a given testing process, using statistical methods where appropriate, and summarise outcomes. [default 20%]

4.1. Identifying gaps and omissions in the testing process

4.2. Identifying target coverage/performance levels for the different testing procedures

4.3. Discussing how the testing carried out compares with the target levels

4.4. Discussion of what would be necessary to achieve the target levels.

5. Conduct reviews, inspections, and design and implement automated testing processes. [default 20%]

5.1. Identify and apply review criteria to selected parts of the code and identify issues in the code. [default 20%]

5.2. Construct an appropriate CI pipeline for the software

5.3. Automate some aspects of the testing

5.4. Demonstrate the CI pipeline functions as expected.