**Third Year Project Questions**

The aim is to create an **outstanding** body of work demonstrating a **very deep** insight into the problem and to present it as such.

**Introduction**

|  |
| --- |
| What is the real world problem you are looking at? |
| IP traffic simulators with more realistic flow and simpler interface than currently used systems. In addition to this, a program that will support whatever protocol the developers wish to test against using a plugin system. |

|  |
| --- |
| Why is it important? |
| It is important as the current software out there does not have particularly configurable flow, have complex interfaces, cannot be automated or have a very small subset of protocols with which they can work |

|  |
| --- |
| Are there any legal or ethical issues? |
| Regarding content of the project, any licensing issues will be avoided by using as few third-party libraries as possible. Ethically, the project could help to trivialise Denial-Of-Services attacks by providing a platform by which unsavoury people could launch high traffic attacks against networks, even communicating using the networks preferred protocol method. |

Research

|  |
| --- |
| What don’t I know that I need to find out? (Your research questions) |
| I need to establish what software already exists and what features they may or may not have. I also need to establish how much load IP sockets can handle through a given interface. The best way to model threads in a C/C++ program along with proper threading techniques.  A well-rounded plugin system will also need to be researched. Along with any good practices involved here. The goal here is to make sure that it is as simple as possible and re-useable as initial review of how it works shows that it relatively complicated, and limiting class creation would be preferable. |

|  |
| --- |
| How am I going to carry out a search for information? |
| The search for this information will take place on The UWE Library for academic sources. Finding software that matches my descriptions will be done via Duck-Duck-Go search. Good C++ practice and help will come from StackOverflow.com and Scoot Meyers’s effective C++. |

|  |
| --- |
| What is the current state-of-the-art (technology and algorithms)? |
| The current state-of-the-art software used for IP traffic modelling is a program called IPSL. And is based off the Modelica library. While it isn’t directly related to IP traffic this would give the user/tester Fine grain control over how the traffic is sent. However, it is an incredibly complicated piece of software to use; both in language and interface. |

|  |
| --- |
| How have other people tried to solve it (products)? |
| Apache have JMeter™ which can be used to test a wide variety of protocols (HTTP/S, SOAP/REST, FTP, LDAP, etc). it features full multithreading and a highly extensible core. However, these extensions don’t allow a user to add more protocols. The software itself is written in pure Java and can also produce visualizations of the traffic/data.  IP Traffic – Test & Measure is another variation and offers advanced network and packet analysis, and offers the use of different protocols. However, the data being sent cannot be easily customised or created by the user and is exclusive to specific hardware. |

|  |
| --- |
| What am I claiming about this project? (Claim/Hypothesis/Assertion) |
| An extensible multi-threaded traffic simulator with user written testcases and plugins should be achievable. While it may lack the nicer features that other pieces of software have, a full featured and open API would allow the program to be expanded, at will, by anyone who would be interested. |

|  |
| --- |
| What are my project goals and objectives? |
| By the end of this project I plan on having a fully realised multi-threaded traffic simulator with plugin facilities to extend the operational use of the program and a comfortable API that can be used to improve user readability. |

|  |
| --- |
| What will be my success criteria? |
| Success for the project will be measured by its ability to load different plugins and send traffic using multiple threads. In addition to this, the fullness of its API will also be an important factor. Success will be based on the program’s extensibility and ease of use within that.  This will be measured against the acceptance tests as laid out in the next section in conjuction with its unit-tests |

**Requirements**

|  |
| --- |
| Describe the context of the system |
| The System will be running in a testing/development environment for a company running high-traffic systems potentially using obscure or complicated protocols that might not be supported generally. |

|  |
| --- |
| What is the scope of the system (what will it do and what will it not do)? |
| The system will perform load testing using one to many threads, with data built from XML testcases and user made loadable plugins. It’ll also offer some chaos feature to make traffic less uniform. |

|  |
| --- |
| How were the requirements identified? |
| The requirements were identified based on the designs and the interfaces and functions of area of the program. |

|  |
| --- |
| Did you gather any requirements from potential users (or others) and if yes, how? |
| I gained some requirements from my own observations while working on placement using your in-house software “iRate”. |

|  |
| --- |
| What requirements were found during research? |
| During research I found that having an outward API would very important and stick to the software’s mantra of extensibility more fully. |

|  |
| --- |
| How are your requirements going to be formally expressed (list, use case model, etc)? |
| Requirements will be formally expressed as a list |

|  |
| --- |
| How will you prioritise your requirements? |
| Priority of requirements will be calculated based on basic programmatic function up to libraries that aren’t required for the program to function but are more for a more well-rounded experience.  BASICS: Network communication, threading library, plugin infrastructure, logging, basic configuration, basic CLI, basic control API  VANITY: API for graphs, increased configuration capability, Increased CLI capabilities, API for automation |

Note: The reason for each requirement will need to be explained.

Here’s where you might list your requirements....

|  |
| --- |
| How will the requirements be tested (functional and non-functional)? |
| Most of the testing will be performed via unit tests per class and module. However, for full acceptance tests, carefully controlled steps will be taken to follow different Users’ journeys through the program to complete a variety of tasks. |

|  |
| --- |
| Do you have test data? |
| Specific test cases will be written to test different aspects of the interpreter. Specific servers operating on certain protocols will also be created as a way to test plugin aspects |

|  |
| --- |
| Do the requirements break down to produce sub-projects, if so how? |
| Due to my intention of having the program run as a service in Linux, a series of scripts or separate executables will need to be used to communicate with the service. These scripts, therefore, will need to be encompassing and robust that they can effectively interact with and enable the service to be controlled by a user. |

**Methodology**

|  |
| --- |
| What methodology will be used during your development? |
| During development I will be employing a Spiral/TDD methodology. Completing a single requirement and making sure that it is fully tested and realised before moving on. When most of the libraries are completed I’ll go back over each component and check to make sure they’re still fulfilling their role.  If changes are made to make a test work and the program is edited. All tests will be ran again, as per TDD. |

Design

|  |
| --- |
| What alternative designs have you considered? |
| I have considered designing my software as both a launchable app and a service. |

|  |
| --- |
| Why have you chosen the design that you have? |
| I chose the design I have (service/daemon) so that the software can run multiple test cases, either using different protocols, against different servers, the same server or even ac ombination between of three. Thus allowing the software to run multiple “sessions” and be managed by “PID 0 stuff” through a more standard interface. |

|  |
| --- |
| How have you identified the different components of your design? |
| I identified each component using story mapping and user journeys. Each component was then broken down into what each action or task that it needed to perform. |
| How are you going to describe your design? |
| The bulk of my design will be carried out using MASCOT2 ACP diagrams, Entity-Relationship-Diagrams and Finite-State-Machines. |

|  |
| --- |
| What issues arose during the design process? |
| The biggest issue that arose during the design process was the best way to represent the program’s function. How threads/tasks interact, behave and, are launched; by what, when and by whom.  Designing interactions between systems I have no prior experience with is another major hurdle. And abstracting those concepts out into more generic overviews has proven to be relatively challenging.  Further, making sure the designs are accurate, robust and useful is a major concern as it’s perfectly possible that I reacha stage where there’s a major error in my design and I’ll have to backtrack to a previous stage to correct it. |

|  |
| --- |
| What test data did you specify? |
| The main test data I specified was either control values used in unit testing or via testcases in XML format.  The testcases will test various aspects of the interpreter and make sure that the data is correctly saved and passed onto the data generator.  To test the data generator, I’ll be writing protocol plugins that can be used by the core to read the data gained from the testcases. An SNTP and a simple “Hello World” protocol plugins, as a proof of concept, to prove that it sends the correct/receives/handles data correctly and to demonstrate how the data from testcases might be used and interpreted.  For the sake of interest, panache and more tangible testing/demonstration I will also be writing plugins to interact with 1) an embedded system (likely to simply blink a light, but mainyl to show it is being controlled) 2) a terminal (to simply print things that have been defined within a test case) |

Implementation

|  |
| --- |
| How will you implement your project and why? |
| My project will be implemented using a cmake build system, compiled on GCC for linux, written in C++17. Using C++ gives me more control over memory and thus it can be fine-tuned more to my liking without having to worry about garbage collection as I would in a managed language. However, there are new facilities in C++11/14/17 that allow a great deal of memory management to be handled in code, taking some of the burden off me.  I’m using GCC as it is a proven compiler that has implemented a great deal of the new features available in the later C++ standards (compared to MSVC, which isn’t even available on Linux). It is also a more familiar compiler that has more support generally.  Linux is my target system as the program is aimed at large scale testing and development where the servers are likely to be running linux in some form, and so, it makes more sense to make sure everything is targeted there. |

|  |
| --- |
| How will you record how you implemented it? |
| My record of implementation will be done via short notes that state:   * What I was doing. * What went wrong? * What can I do to fix it? * What is the best course of action to take? * Did it work?   My record of implementation will take the form future me looking back and evaluating each step as it was taken within the library. Making note of any initial concerns, issues that arose, considered evenues for how I overcame them, whether the fix was successful. |

|  |
| --- |
| What issues arose during implementation? |
| Moving from raw pointers to special pointers, factories,  logger and testing,  setting up gtest/gmock,  setting up cmake,  project structure,  Generic thread pool Parsing complications How to model chaos |

|  |
| --- |
| How will you demonstrate that it works? |
| To demonstrate that it works I can demo it against a locally running server. Both a “Hello World” server and an SNTP server. I will also be running other demos for interacting with an embedded system and a terminal. Just to show a greater range of interaction and possible use cases (and really the full range of ability that the software has).  These tests will cover the test case analyser, protocol plugin system, target plugin system, threading, speed, chaos (if it gets implemented), the API (If a web page is ever written for it, though the CLI should cover many of the web areas), the logging facilities. Which is more or less everything a user will every really interact with in the program.  As far as the “cloud” aspects of the system go they may be demonstrated using virtual machines or with a serpate instance running on a different remote. |

|  |
| --- |
| How will you show what has been achieved? |
| I will show what has been achieved by proving that the software performs tasks as related to my requirements specification, acceptance tests and passing unit tests.  As soon as an initial product that can do some basic functionality is up and running I will begin a system of versioning based on “Phases”. Phase 0 will be the product as it stands within that moment. To reach Phase 1 I must have implemented the highest priority requirements. Phase 2 can only be reached once all the next highest priority ffeatures have been implemented. And so on and so forth.  This way I can easily track what still needs to be done as soon as a simple product is up and running. This means I can implement some nice-to-have features that will help with testing and interaction. While they aren’t *vital* they will be useful. |

|  |
| --- |
| Has anything not been finished? |
|  |

**Testing (part of Implementation)**

|  |
| --- |
| How will you test your code? |
| My code will be tested using GoogleTest/GoogleMock for the unit testing and module testing. These systems provide a way to organise unit tests and provide features such as Test Fixtures, Mocking Frameworks, easy to parse output, and a great deal of macros that can perform a large variety of checking and roles.  For full system tests I’ll be writing my own bash/ruby/python testing framework that I can use to automatically test user journeys and other various functions within the program.  Due to their nature, many of the threaded parts of the code are untestable in a consistent and guaranteed manner. As such I will be writing separate executables that I can use to manually verify that the threaded components are working, where I am able. |

|  |
| --- |
| How do your tests prove that your code works correctly? |
| My unit tests and module tests prove that it works at a very basic level. They totally prove that each method can perform its desired task and behaves exactly how it should. This is achieved by passing control data in and verifying any outputs or checking for exceptions. However, interaction between certain units could prove to be problematic, especially when threads are involved (and generally in large projects).  The scripted testing framework will work by using the API and performing tasks that a normal user might do, and trying to find issues with how the system is run and any mistakes that a user will make because they’re retarded and a user. |

|  |
| --- |
| How will you document your test results? |
| Test results will be documented in a table. Each test will have an ID and that ID will relate to either a unit test or area of the BASH-Framework.  The test ID will be labelled simply as <TESTID>. For each “version” tested, I will be listing the results for the test. |

**Evaluation**

|  |
| --- |
| Do your findings support or refute your claim? |
|  |

|  |
| --- |
| Have you met your original project goals? |
|  |

|  |
| --- |
| What limitations does your product have? |
|  |

|  |
| --- |
| What thoughts do you have now about your product? |
|  |

**Conclusion**

|  |
| --- |
| How could you improve your product? |
|  |

|  |
| --- |
| How could you improve the development process? |
|  |

|  |
| --- |
| What would you do differently? |
|  |