# SENG2040 – Assignment 1

This assignment should be done in pairs. If you wish to work alone, you must have a valid reason and get approval.

## Critical Note

This is a technically challenging assignment. Do not leave this assignment to the last minute, as you may have unexpected difficulties. No extensions will be given.

## Submission Guidelines

Please read and understand the submission guidelines below. Failure to correctly submit any required component will result in 20% penalty per missing or incorrect component.

### Paper Submission

There is no paper submission required for this assignment.

Do not submit this assignment specification as a part of your document.

### Electronic Submission

Exactly three items will be submitted electronically. Only one of your team needs to make a submission.

* Upload your complete report as a SINGLE document in PDF format.
* Upload a single zip file of your CLEANED source code.
* Upload a git bundle of your project, by running (in your project directory)
  + git bundle <<your\_names>>.bundle --all

All files should be named with your user IDs (example jsmith1234\_bjones5678.pdf). Complete file naming and upload guidelines are provided near the end of this document.

# Your Task

An excellent discussion and tutorial on the difference between TCP and UDP can be found at <https://gafferongames.com/tags/networking/>, where the author goes through an in-depth discussion of enhancing UDP for reliable communications. (Particularly <https://gafferongames.com/post/udp_vs_tcp/> and <https://gafferongames.com/post/reliable_ordered_messages/>)

Read the series on game networking, which culminates in an implementation of reliable UDP and downloadable sample code that I have made available on eConestoga (again, note this is NOT my code – yes I know it is NOT to SET standard). You may be tempted to refactor this code – do not do it. You may document the provided code, if you wish.

You will used this code as a starting point as shown in your first lab and will now add functionality to it.

Please note – you MUST use the provided code – you may not use your own code instead. If you do not use the provided reliability code, you will receive a zero. If you remove functionality present in the existing code (this includes diagnostic messages), you will have a hard cap of 55%.

Using the example source code (provided on eConestoga) develop an application that allows you to reliably send a file of the user’s choosing via UDP. Make sure both binary (images, for example) and ASCII (text, for example) files may be transmitted. You must test transmission with both text and image files.

## VPN

In order to be able to test your application across the Internet, you and your partner will have to use ZeroTier VPN that we set up in our first class. Instructions are available on eConestoga.

Please note that you will need to ensure that your application listens on and connects to the appropriate IP addresses.

## Testing the VPN

To ensure that UDP communications is working as expected, we tested the ability to connect in our Week 1 task. Please see the Week 1 task document for details.

## Requirements

* Your code must be developed in C or C++ on Windows.
* You will have to develop a brief protocol that transmits necessary file information and the contents of the file itself.
* You MUST implement a method of determining that the WHOLE file was correctly transferred (you may choose your method – CRC / LRC / MD5 etc.) You may use publicly available code (be sure to credit the origin of the code if you do so).
* You may choose a GUI or command line interface.
* No hard-coded parameters (i.e. IP address, port). You must include reasonable defaults.
* You MUST devise and demonstrate a test that shows that your whole-file error detection works. This test must be integrated into your software (i.e. you may not manually interrupt communications)
* Your code should accurately and automatically calculate the transmission time. Please ensure that you can measure fractional seconds accurately. Based on the transmission time and the file size, calculate and display transfer speed in megabits per second when the transfer is completed. Hand timing is not acceptable.
* Your tests (and demonstration) must take place between physically separate machines (i.e. not between instances of the application running on a single machine).
* You must find files of different sizes and types for testing purposes.

## Warning

You must be able to transfer both ASCII and binary files. Because of this, you must not use string functions for any of your data manipulation. Please note, changing a file’s extension does not make it a binary file.

## Testing

Note that it is not enough to barely meet the proposed requirements and test only the happy path. It is expected that you test your solution for normal functionality as well as error cases. As an example, consider file sizes – it is expected that you test with small, medium, and large files of both ASCII and binary types.

## Evaluation

You will be graded on the following basis:

* Report Completeness and Accuracy
* Demonstration
* Requirements Fully and Properly Implemented
* Usability
* Testing / Bug Free Operation
* Meaningful git commit history (ensure all of YOUR source files are included)

A letter grade will be assigned for each of these dimensions. An overall letter grade will be assigned accordingly.

The following is a guideline for the letter grade evaluation that will be used.

|  |  |  |
| --- | --- | --- |
| Letter Grade | Achievement | Description |
| A+ | Exceptional | Thorough demonstration of concepts and/or techniques and exceptional skill or great originality in the use of those concepts/techniques in satisfying requirements. |
| A | Excellent | Thorough demonstration of concepts and/or techniques together with a high degree of skill and/or some elements of originality in satisfying requirements. |
| B+ | Very Good | Thorough demonstration of concepts and/or techniques together with a fairly high degree of skill in the use of those concepts/techniques in satisfying requirements. |
| B | Good | Good level of demonstration of concepts and/or techniques together with considerable skill in using them to satisfy requirements. |
| C+ | Competent | Acceptable demonstration of concepts and/or techniques together with considerable skill in using them to satisfy requirements. |
| C | Adequate | Acceptable demonstration of concepts and/or techniques together with some skill in using them to satisfy requirements. |
| D | Barely Passing | Minimum demonstration of concepts and/or techniques needed to satisfy requirements. |
| F | Unacceptable |  |

# Guidelines

Please consider the following guidelines for your assignment.

## Document

### Completeness

Document includes all required content with no missing or inadequately short sections and adheres to the format specified.

## Demonstration

### Readiness

Barring documented extenuating circumstances (failure to compile is not one of these), you are ready to demonstrate during the expected lab period.

You clearly state non-functional elements of your software in advance.

You must demonstrate the version of the code you have electronically submitted. Note that failure to demonstrate will result in a score of zero for all demonstration-based criteria (i.e. bug free operation, etc.)

## Technical

### Results and UI Quality

Your results are complete and well stated. Anomalies or unexpected results have been identified. Your results can be duplicated in your demonstration.

It is easy to run your application as desired. It is not necessary to recompile to change common settings.

### Bug Free Operation

Your application runs as expected. You have properly tested edge cases and variations.

### SET Coding Standards

All source code completed by you includes proper commenting and adheres to general SET coding standards. Any code from other sources is clearly identified.

# Report Sections

Your report should adhere to the following format. Please do not include this text or the header above.

# SENG2040 Reliable UDP

<< Name1 >> <<ID 1>>

<< Name2>> <<ID 2>>

Date: << date>>

## Whole File Error Detection

<<Description of whole file error detection implemented>>

<<Reference for source of whole file error detection being used>>

## Whole File Error Test Method

<<Describe how your whole file error detection test works>>

## File Transfer Protocol Details

<<Protocol packet contents (please use a chart)>>

<<Written description of how your protocol works>>

## Transfer Speed Calculation and Results

<<Describe how you calculate transfer speed>>

<<Run tests to fill table below. Select a range of file sizes, ensure that you have tests that take times ranging from a few seconds up to approximately 60 seconds or more.>>

|  |  |  |  |
| --- | --- | --- | --- |
| Filename | Size | Transfer Time | Transfer Speed (Mbit/s) |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Work Estimate and Breakdown

For each team member, provide an estimate of the hours worked and tasks completed in the table format shown below.

Table 3 - Team Member Work Breakdown

|  |  |  |
| --- | --- | --- |
| Team Member | Hours | Task |
|  |  |  |
|  |  |  |