# Minor Skilled Analysis

Pieter Schreurs 318775

# Index

Contents

[Minor Skilled Analysis 1](#_Toc62220563)

[Index 2](#_Toc62220564)

[Introduction 3](#_Toc62220565)

[Start Minor 4](#_Toc62220566)

[Expectations 4](#_Toc62220567)

[Learning Goals 4](#_Toc62220568)

[Setting up the project 4](#_Toc62220569)

[Phase 1 - Analysis 5](#_Toc62220570)

[Getting started 5](#_Toc62220571)

[Planning 5](#_Toc62220572)

[Progress 5](#_Toc62220573)

# Introduction

The minor skilled is about developing yourself to be a more knowledgeable about a certain topic.  
I took this time to increase my knowledge of multiplayer games and how they work, and eventually making a multiplayer game myself.

The reason why I chose this is because we’ve had a networking course for my study, and it appealed to me. The big thing you could’ve done with is, is to create a game in the Project that came after that course. Unfortunately, we as a group didn’t chose this approach. We made a single player game.

My biggest reason that I would have liked to make a multiplayer game is because that brings new challenges I’d like to have figured out. Plus, most of the games, practically all AAA, are played in multiplayer. It would not be a big overstatement to say that if you want to develop at a successful game company you would require the tools to make a multiplayer.

In this report I will describe my journey, my choices and following results which I had in my Minor.

# Start Minor

## Expectations

My expectations for this project were to make a functional multiplayer game.  
I chose to make a RTS game first of all because I really like that sort of games and played it a lot in my youth.  
My second motivation is because it looks like a big enough project that it can be a real challenge for me. I never made such a big project on my own and I’m curious whether or not I can make it work.  
It will require some good planning and thinking before programming which in my opinion we haven’t really had in most project groups. I’m looking forward to seeing what comes in my path during development.

## Learning Goals

I set up the following learning goals for myself:

Learning how to make a correctly functioning multiplayer game.

This because the multiplayer part is the functional part of the thing I’m making. Actually being to showcase it is quite important.

Learning how to organize a bigger well coded game on my own.

Having an organized base is important to a project, if you were to make some changes or re-use some code you can easily identify what parts you need or edit. Plus being able to document your code is a valuable skill that’s often overlooked.

## Setting up the project

I made my project in Visual Studio and Unity. Using GitHub for version control in combination with SourceTree. I have 3 directories for my code. One is the Client which is made for Unity. Another one is a C# Console application which I use as a server, and the last one is a Shared repository which is a code library which is used in both directories to set up communication protocols between the projects.

# Phase 1 - Analysis

## Getting started

The first few weeks I spent most time analyzing on what to use in my game.  
Is the goal to make a RTS game or is the goal to learn how to make a multiplayer system.  
I carefully thought out the learning goals of my Minor and came to the conclusion that I want to make a system I can use in other games as well. If I ever wanted to make a project a multiplayer game.

I didn’t want to be dependent on a built-in Unity helper or plugin such as Photon.  
I wanted to create a skill for myself so that I could use it multiple ways instead of just focusing for a game in the Unity engine.

In order to achieve this, I made a planning for myself.

## Planning

My planning for the first phase is quite simple:

* Decide on what network protocol I want to use.
* Look up some projects on GitHub/YouTube tutorials on how to make a multiplayer game.
* Flesh out my game idea more, think of all the requirements of a RTS.
* Make a class diagram in which is visible how data will be handled.
* Re-visit the Networking lessons we’ve had to freshen up my knowledge.

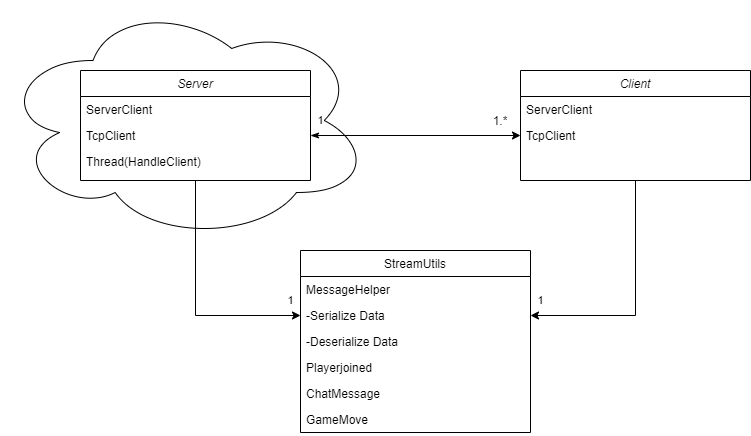
## Progress

The first subject in my planning was deciding on a network protocol.  
I chose to go for TCP, this because TCP is more reliable and makes sure to get the data over in a correct way.  
The game is not meant to be a massive player game in where it doesn’t matter whether or not data is correctly displayed. With designing it in mind I have a max of 4 players per game.

The second subject was looking up tutorials on GitHub/YouTube was somewhat fruitful. However most of the projects I noticed were using Unet/Photon instead of a custom networking code. They however still had some information on how the project should be structured and how the client should handle things.

I went through the networking courses again, re-did the lab assignments and the labs. This gave me enough insight on how the standard TCPclient can write and read data across the network.

Using this method I came up with a small visualization of how I thought the networking part is supposed to be handled, visible in this picture below.



There would be one server running which could receive multiple clients.  
Both the server and the client would have access to the StreamUtils class. This is a class which could “translate” the packets which are sent over the network. For example, a client would send a chat message, the StreamUtils class would translate that chatmessage to bytes which could then be read by the server and distributed to the rest of the clients.

## Obstacles

I didn’t have a lot of obstacles during this phase, mostly because it was all research based and repeating of already laid out assignments.

It was mostly thinking out ideas of how it should work instead of testing the practical side of it.

## Feedback

The big feedback I received was that my initial idea to make a RTS game is somewhat too ambitious.  
I made a big document on researching what components should be part of a RTS, did some market research on RTS. I should put less emphasis on the making of the game, but more of the multiplayer side of things. Take my learning goals and mold them into the game form you can use to support your learning goals instead of making a RTS.

I came up with a new idea:   
Make a lobby system in which people can join and play different games with each other.   
The emphasis is in the multiplayer system, not the creation of the game.

# Phase 2 – Design

## Planning

The planning for this phase concluded of a few things:

* Set up User stories for the game I was going to make.
* Make a flowchart to show how data is handled.
* Make a prototype in which I showcase the way network protocols are handled.

## Progress

Using the feedback I received from the first phase, I came up with a small prototype server in which multiple people could connect and send messages.

I had to write this prototype quickly so that I quickly could find out what’s possible in terms of network code so that I could make my game.

The server is just a C# console application which runs and continuously listens for incoming connecting clients and puts them in a separate thread once connected.

Each message the client can send via the StreamUtil class.  
The way this StreamUtil class works is the following:

Each “command” be it a request or sending of a message is a separate class.  
Each class can be serialized or deserialized in which it details how to convert it to something which can be sent over the TCPnetwork, or be read when it’s incoming from the TCPnetwork.   
These classes all have their own variables and an identifier.

Each command is built up as the code shown on the next page.

You basically have a messagehelper which is called whenever a new message is received, it will read the first int so it can identify what kind of message it received, be it a Playerjoined, chatmessage or gamemove etc. Once it’s been identified what kind of message it is, it is then further deserialized. Since it can be parsed as a class instead of a row of bytes.

This method was enough to make a prototype working of a lobby system in which people can send chat messages to eachother.



## Obstacles

The big obstacle came once I completed my main goal of having a lobby system. The objects I could send to over the network were quite basic. For example the list of players I would need to send an array/list of objects, with the current implementation I couldn’t do this. There is a work around to use wildcard characters but it wouldn’t work if a user would use such a wildcard for his username for example.

I was stuck too long on how to make this work before I asked someone more experienced for some advice. I spent a long time on debugging, unfortunately debugging network code is a hard and tedious process. I should’ve asked earlier for help.

## Feedback

After the prototype was stuck on a point I couldn’t make it work. I had a talk with Hans Wichman. I knew that he was capable of helping me out, because he also gave the course for networking.

He said he also encountered the problem previously as he was also making a multiplayer system himself.  
He then referred me to a new and updated assignment from the next Networking course.

He understood what I went for in my initial idea’s and prototype, but said he had some ideas on how to fix the issues I had. They were implemented in a course assignment he gave me.

I will explain in the next phase what I did with that project which would eventually serve as the base as my final product.

# Phase 3 – Production

## Planning

## Progress

## Obstacles

## Feedback

# Phase 4 – Production

## Planning

## Progress

## Obstacles

## Feedback

# The End product

# Self Reflection

## Improvements

## Future Perspectives

# 