# Unity Project Design Document

## Introduction

This is the design document for comp3000 that will show me think up, plan and develop a project for a dissertation. The project must be for an identified computing problem, with examination of previous approaches used for the problem, with the result being an approved product which is appropriate for the solution.

My project will be a piece of software developed for the company Mind feast for their game called Co-Operation. The piece of software will help with the creation and visualisation of levels for the game, allowing for easier declaration of object definitions and faster iterations of level design to better create fun levels in a shorter time frame.

## Background

Co-Operation is relatively new game in closed alpha stage but with an extensive modding API meaning the community around the game can create and share levels and mods for others to play. One of the most tedious bits of creating a mod for the game is when it comes to level design and creation. There are techniques to help for example creating mock-ups on a drawing application; but when the initial design is implemented and the next iterative version is done, finding and replacing the correct tiles and creating the new object definitions can be difficult when not being able to visually see how it will look.

## Objectives

The objective of this dissertation is to make a level editor that can be used by the community of Mind Feast to create mods and levels for their game Co-Operation. It will be designed to aide in the production of mods and levels and be catered towards people who have little to no understanding of modding in general. This tool will also be for those who do have understanding but don’t want to manually manipulate a text file with no visual way of knowing what the level will look like. This product should hopefully open the creativity of modding to people who normally wouldn’t attempt the creation of one.

## Deliverables

In this project I want manipulation of 3d objects in a scene that can be translated to a set of object definitions and YAML layout. This will also be able to be imported so the user can also edit already created levels.

Object manipulation

* Scene objects
  + File format GLB (used in Co-Operation)
  + Create objects for the scene
    - Objects will be able to be moved before and after being placed
  + Update objects that are currently on the scene
    - Change offset within placed grid square
    - Change orientation of object
    - Change scale of object
    - Change tags and scripts
      * Add and remove tags
      * Add and remove scripts
  + Read objects attributes
    - A UI on the screen to show and allow for modification of object attributes
  + Delete objects from the scene

Importing and exporting of level data

* Exporting of files
  + Each object on the map will be translated to its YAML definition
    - Possible number reduction of definitions if approximation is implemented option is implemented
  + Map grid positioning will be encoded into base 36 (a-z, 0-9) this will allow for considerably big maps. A higher base of encoding won’t be needed due to the fact that if the map is too big, objects become too small
* Importing of files
  + Level files can be imported to be edited further
    - All object definitions will be scanned and placed once the map and tile definitions start to get read
    - Currently only levels created with the editor will be allowed to be imported, but importing of level made without the editor should be possible

## Method of approach

Technology stack and tools

* C#
  + I will be using the programming language C# for this project as the game engine I’m using is unity that primarily utilizes C# for its scripting.
* Unity
  + Unity will be used as I have used this game engine in the past and am most comfortable using it.
  + With the extensive community behind unity as well there will be more support are community made libraries that I will be using in this project.
* Unity library for GLB object importing
  + I will be using a library that has been written for unity to allow for the importing of GLB objects. This will allow for the use of the objects that are also used in Co-Operation
    - <https://github.com/KhronosGroup/UnityGLTF>
  + When exporting and import YAML I will use a YAML parsing library written for C# to make the process smoother as manual text manipulation needs thorough error handling and to make a more stable application an external library will facilitate this.
    - <https://github.com/aaubry/YamlDotNet>
* Co-Operation game
  + The current copy of the game I possess will be used for validation of the YAML output, this will really test if everything that is being exported is correct and useable by the game it is built for.

Architectural design

* The architectural design will be based around a microservice concept using object orientation. Each object will be responsible for handling different aspects of the program, which will be straight forward to implement due to unity splitting up each script, giving each one its own start and update method.
  + Basic overview of script to appear
    - Camera Movement
      * Panning with middle mouse button click
      * Zooming in and out with middle mouse button rolling
    - Object movement and deletion
      * Mouse will be translated to world space and object will follow the mouse
      * When left mouse button is clicked object will be places
      * If left mouse button is clicked while over an object, the object will the picked up instead to be placed again
        + right mouse click might need to be used to select the object and holding right click to move the object.
        + while selected with right mouse click attributes will be able to be modified
    - Object attribute manipulation
      * Objects will be able to be modified while being selected. The attributes will have a on screen panel/window that will show which values can be changed and to what
    - Object attributes
      * This script gets attached to objects to hold values that unity can’t keep track of by default
        + Position, scale, and rotation will be track using the object attributes in the scene
        + Lua script and tag list will be tracked in the attribute file to be added to and removed from

## Legal, social, ethical

## End project report

## Project reflection

## Conclusion

## Reference list