Network Strategy

# 1.0 Hosting and Connecting

## 1.1 Host

The Host will either create a new campaign (for creating) or load campaign (which will load from the database). From here they can click on the Host button to allow others to connect. The hosting will set the game up using unity’s network solution – it will publish an entry to a list of game servers so they can easily be looked up.

## 1.2 Client

The client will click the join button, displaying the available games. Clicking on a game will select it to be the one to join. Session manager will be handling this for both the client and the host.

# 2.0 Campaign Screen Synchronization

## 2.1 Basic Setup Prior to Synchronization

Both the server and client will have the blank squares as the backdrop for this screen.

## 2.2 Synchronized Data

When rooms are present on the campaign, RPC calls are used to put a new gameobject on top of the existing background square. This will allow all clients to see the new object on top of the same square (if one joins later they will receive the call once they are in and it will be created as well). Only rooms that are marked visible or are connected to visible rooms should be visible to clients, therefore the manual RPC approach is used instead of Network.Instantiate method which would require the object not exist before creating.

No Information on the room is visible to clients, just the position and whether it has been explored.

# 3.0 Room Screen Synchronization

## 2.1 Basic Setup Prior to Synchronization

Both the server and client will have a black and white grid as the backdrop for this screen.

## 2.2 Synchronized Data

When a character or creature is put in the room Network.Instantiate is used to place it. The details of creatures is stored as a child so is not sent over. State of the creature will also be stored in a child, this will be linked via RPC and will be synchronized.