**Client Meeting Notes Summary – 05 Mar 20**

**General Notes:**

LVC – live virtual & constructive; a simulation term (live = in real life, virtual = simulation on a computer, constructive = adding non-real entities to an ongoing virtual simulation that has real actors in to too).

* Main use of LVC Integration:
  + E.g. P3D Flight Models used in VBS simulator
  + Link a virtual environment (that isn't perfect) to link a specialist system

DIS (Distributed Interactive Simulation) – standard networking protocol for exchanging information among simulation applications  
HLA (High Level Architecture) – another simulations standard (US Military Standard), not utilised for this project

Open Source Libraries:

* OpenDis – Java, C#, C++
* KDIS

Pattern of Life – important term. Need to simulate both a terrain and entities to simulate life (focussing on entities ONLY in this initial project).

Unity – used to inject pattern of life into simulation (e.g. create a “crowd”, “flock” etc.)

* Geometric objects moving around
* Injected into VBS
* Match traffic into real entities

Simulators – both VBS and Titan 3D Virtual simulators - just a visualisation platform

**Topics:**

Entities

* Unity creates “flock” of entities to inject into middleware -> interpreted/mapped to objects in connected simulator(s)
* Focus on simple animation and entity content (as they can be mapped to various entities in simulator)
* Each entity should have state information/attributes including but not limited to position, velocity (speed with orientation/direction), latitude, longitude and altitude.

Algorithms for Pattern of Life:

* Boids: 3 simple rules to simulate flock behaviour
* Other options include…
  + Look at simulating human behaviour e.g. conversations\

**Other:**

Tools & Concepts

* ‘Data reckoning’ to avoid having to send a large number of packets or not sending enough packets and causing entities to behave unrealistically (e.g. jumping between positions rather than smooth animation or sudden changes in direction).
  + Involves sending each entity’s position and velocity (via packets to the middleware) initially and ONLY sending updates if velocity changes (direction or speed)
  + Can include a threshold (e.g. if change is above ‘x’ send an update).
  + Note: VBS interface should know how many packets are being sent to it
* Right Hand Coordination System (a standard)
  + Focuses on entity state PDUs and parameters
* DIS Protocol
  + Have to configure both the IP and Port ourselves
  + Suggested tool – Wireshark to see the packets being sent from Unity to middleware
    - Ensures we can monitor the amount of traffic in network

Resources

* Open Source Implementation of DIS – KDIS, Portico, OpenDIS
* DIS Spec - https://standards.ieee.org/standard/1278\_1-2012.html
* Entity State PDU Summary - http://faculty.nps.edu/brutzman/vrtp/mil/navy/nps/disEnumerations/JdbeHtmlFiles/pdu/29.htm
* Crowd Behaviour Algorithms: Boids (Bird behavior)

**Outcomes:**

Goal: to create “pattern of life” behaviour that is injected into one or more connected, distinct simulators via middleware.

Tasks:

* Work in Unity to create pattern of life behaviour
* Integrate with middleware utilising DIS libraries
* Display changes in real time in VBS

Show results on at least 2 distinct connected VBS instances.

Bigger picture:

* Connecting 2 or more simulators up together can create an even better (i.e. higher fidelity) simulation
* Want to display how one can use a high fidelity crowd behaviour model to populate simulators.

Key Guidelines:

* Expected meeting once a week – confirmed for every Thursday at 1230 at client location.
* Email preferred method of communication.
* Establish deliverables ourselves, with something presentable every week – for example:
  + Unity Scenario established
    - which algorithms we pick to generate behaviours - finalise behaviour model
  + Two separate VBS instances connected via DIS

Future Opportunities:

* Possibility of being involved with the writing of a paper to be included in upcoming simulations conference