Design Proposal Outline for Digital Twins Project
By: Ian Fleming and Lucien Lee

Goals V1:

We will be developing a Digital Twin that will tell the user different readings from the place that they are located. As an example of this, if we are on a random planet then our twin will be able to model the various oxygen levels, the humidity, the temperature, etc. of the outside world. To visualize this we will be constructing a spaceship that the user will reside in that will receive the data from the sensors and send the data to the ship. This way the user inside the ship will be able to see if the world they are in is safe, or if it is not and can choose to go outside into the world they have landed in or they can choose to stay inside of their ship and continue to monitor the incoming data.

To summarize -
- The visualization will get data on the world such as oxygen levels, humidity, temperature, etc. This will be both random and manual.
- The visualization will also show data about the ship, such as fuel level and power consumption.
- This data will be simulated in Python and sent to the visualization via MQTT publisher (the visualization will subscribe).
- The user can see the data, and choose to exit the ship or stay in the ship as they see fit.

Objectives V1:

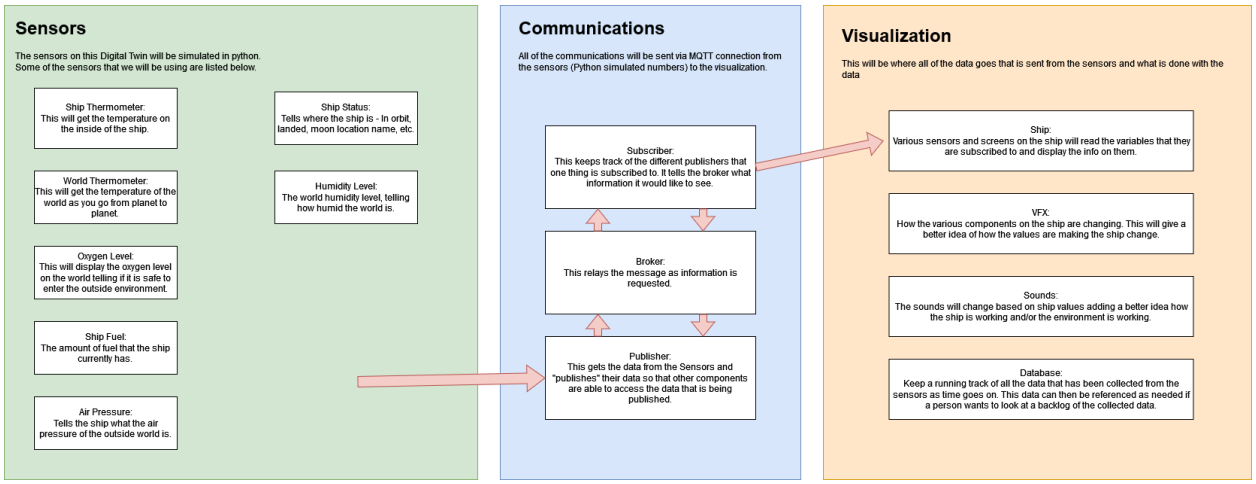
Our objectives will be the following -
- We will develop a virtual ship to show the world and the ship data. This will update as it happens in "real time." All of this info will be simulated with a Python script.
- We will develop python scripts to generate data to send to the digital twin and have the displays in the twin update as they update from the script.
- Linking the script to the ship using MQTT for subscriber to publisher communication.
- Making interactable with the in game player for a higher level of immersion.
- Have animations and interactable objects in the virtual world to read the data in a better way.

Architecture V1:

We will be using Unreal Engine in order to develop the virtual simulation and have it interface with the publish/subscribe system of MQTT. We will have separate topics that will all be sent to the same broker bus. This will work well because of the high output capabilities of the MQTT service. This will function better then serial because it doesn't have to be serial, and comes synchronized out of the box. We will then be developing some python scripts that will allow us to generate the simulated numbers before sending them to our Digital Twin that is implemented in Unreal Unreal Engine.

Design V1:

Our design will consist of the above mentioned tasks that will allow us to complete our Digital Twin to the best of our capabilities. A diagram of the flow of the design will be included below -



Goals V2:

—

No changes were made to the goals of this project.

Objectives V2:

—

No changes were made to the goals of this project.

Architecture V2:

—

No changes were made to the architecture of this project.

Design V2:

—

No Changes were made to the architecture of this project.

Data Exchange:

—

Below is a list of the data that we believe will need to be exchanged between the Digital Twin and the Real Device. These data points are as follows:

Name:	Type:	Description:	Implementation Priority:	Real Device Component:	Likely Digital Twin Visualization Actor:
Hologram emitter	Structure	This will display the ship and the components on it in real time as they update.	High	Essentially a database for ship component information	Hologram ship actor color or look will change as it updates.
Communication Antenna	Boolean (True or False)	The communication antenna will tell if we are connected to MQTT telling us if we are online or offline	High	This will connect to the MQTT publisher. If we can't connect to this then nothing will connect.	Antenna located visually on the ship hologram.
Computer System	Structure	This component will a computer terminal that will allow the user to see all the data being sent via MQTT	High	This will visually display all of the different components that are being received	A computer terminal actor will be constructed to connect all of the different elements to the visual aspect of the twin.
Thruster 1	Float (Between 0 and 100%)	This will output the level of burn (burn rate) will measure how fast the ship is moving.	Medium	There will be thrusters on the ship hologram telling the user information at the ship holoram	These will be primarily visualized on the back of the ship and on the ship hologram.
Thruster 2	Float (Between 0 and 100%)	This will output the level of burn (burn rate) will measure how fast the ship is moving.	Medium	There will be thrusters on the ship hologram telling the user information at the ship holoram	These will be primarily visualized on the back of the ship and on the ship hologram.
Engine	Boolean (Running or not running)	This will tell stats about the ship and how it is running. Will	Low	The ship hologram will show the engine status	This will be visualized on the ship hologram and the color will change as the engine

		interface with the ship thrusters and the ship fuel level		and how the engine is functioning with colors to differentiate the status of the engine.	status changes.
O2 Sensor	Float (Percentage between 0 and 100%)	This will get the O2 levels in the world and interface with the ship to tell the user the current O2 levels of the ship and the world.	High	The O2 sensor will get oxygen levels about the world and be sent over MQTT	This will be visualized on the computer terminals, and the warning lights. If the oxygen gets too low then the warning lights will start going off in the world.
Humidity Sensor	Float (0 and 100% and will be using a hydrometer)	This will get the humidity levels in the world and interface with the ship to tell the user the current humidity levels of the world.	Medium	The humidity sensor will get the world humidity levels and tell the ship what they are via MQTT	This will be visualized with fog emitters that will make the inside of the ship foggy when there is a higher humidity level in the world.
Ship Thermometer	Float (Between 0 and 200 Degrees) Will have fahrenheit and Celcius	This will get the current temperature of the ship and tell the user the temperature.	Low	The temperature will be sent to the ship via MQTT	There will be a thermometer in the ship that will visually tell the user how warm the ship currently is, and the temperature will change based on the part of the ship that the user is currently in.
World Thermometer	Float (Between 0 and 200 Degrees) Will have fahrenheit and Celcius	This will get the current temperature of the world and tell the user the temperature	Low	The temperature will be sent to the ship terminal via MQTT.	This value will be visualized on the ship terminal so that the user can see the data about the world temperature visually.

Ship Fuel Level	Float (Between 0 and 100%)	This will be the current fuel level of the ship, and the main hologram will display this level on several different components	Low	This will be sent to the ship via MQTT.	This value will be linked to several different components that have to rely on fuel to function. With this implementation the fuel will be shown in several different visual ship components.
World Air Pressure	Float (Millibars, not really a cap, but likely between 0 and 3000 millibars)	This component will change the air pressure in the world to impact various components on the player.	Low	This will be a generated variable from MQTT.	This value will alter the gravity and the jump height og the player, so their movement will be changed based on the current air pressure of their surroundings.
Ship Air Pressure	Float (Millibars, not really a cap, but likely between 0 and 3000 millibars)	This component will change the air pressure in the ship to impact various components on the player.	Low	This will be a generated variable from MQTT.	This value will alter the gravity and the jump height og the player, so their movement will be changed based on the current air pressure of their surroundings.
Lights	Boolean (On or Off)	This component will either turn the lights in the ship on or off	Low	This will be a randomly generated variable in MQTT	As this value changes based on other aspects of the ship, the lights will have the ability to turn on and off based on the other current variables that the ship is receiving.

Design V3:

We will be implementing an Arrow Database in order to load the data that we are generating for our digital twin within our MQTT script. When we are doing this, we are able to get the data from MQTT, and store it so that in the event that there are other ships in the simulation, there will be a much more efficient way to get the data and send it to other people. This would work in the following way:

- The simulation generates the data
- MQTT sends the data
- Two scripts will be created:
    - One will send the data directly to Unreal Engine 5
    - The other will send the data to the Arrow Database
- The storage manager script will take the data and send it directly into UE5
- The query manager script will send queries from Unreal to the database in which it will send the data back to Unreal with a successful query

This will allow us to add timestamps to the data when we store it, and we can provide graphs in our simulation to show the historical data of the specified query and display that data in Unreal. This also allows us to extrapolate the data to make it not only easier to use, but also it allows us to easily expand the simulation side. We are able to create a larger distributed system to show the analytics to multiple systems, since we are able to populate and process the data in a python script instead of Unreal Engine directly.

Using the above functionality, we will attempt to create 2D graphs in order to better visually show the data that the simulation is generating. A couple examples of what the graphs could show are:

- Past temperature graphs
- Ship power consumption over time
- The signal strength of the antenna over time
- Engine temperature over time
- Fuel used during the lifetime of the ship

The above examples are a couple of different ideas for what we will be able to show with the historical data that we will be receiving through the Arrow Database.