Digital Twin Interoperability in Space Exploration: Integrating Kafka, MySQL, HSML, and EC2 for Real-Time Simulation

NASA Jet Propulsion Laboratory
SIRI Spring 2025 Internship

Sydni Yang (US 345C-Affiliate)
Mentors: Dr. Thomas Lu, Dr. Edward Chow

Abstract

This report outlines my findings and research contributions at NASA's Jet Propulsion Laboratory (JPL) to support research on Digital Twin Interoperability in space exploration. By leveraging tools such as Apache Kafka, MySQL, FastAPI, AWS EC2, and more, the objective was to streamline real-time communication and registration across various environments (Unreal Engine, Omniverse, Unity) using the HSML (Hyperspace Modeling Language) schema. My contributions included setting up AWS EC2 cloud infrastructure, configuring Kafka producer-consumer models, creating a MySQL registry on EC2, and testing the HSML API for user test cases. Findings show the potential of a shared virtual test environment for multi-platform simulation, which greatly contributes to future mission planning and the development of Web 3.0 infrastructure.

Overview

Digital Twins have become more widely used in aerospace simulations and mission planning, as they are virtual representations of real-world systems. My internship project at NASA JPL focused on creating seamless communication between Unity, Unreal Engine, and Omniverse by integrating Kafka servers, such as Zookeeper, an EC2 cloud infrastructure, and MySQL databases into a cohesive integration. The center of this project revolved around the HSML schema (credit: intern Alicia Sanjurjo-Barrio), which standardizes the interaction of spatial entities. Some key areas of contribution include Kafka producer-consumer architectures of real-time data streaming, MySQL database creation + HSML entity registration, HSML API testing and integration, and EC2 setup and management for remote access. This relies on the HSML schema, which uses a JSON-based schema to encode entities and their relationships. I was tasked with integrating Apache Kafka (for message exchange), MySQL (identity storage), and AWS EC2 (hosting services) into a functioning architecture to help manage DT (Digital Twin) communication.

Results

Kafka Integration: Kafka was deployed inside an AWS EC2 instance and configured with dual listeners to support both internal (localhost) and external (public IP) communication. I created multiple Kafka topics—such as unity-hsml-topic, omni-hsml-topic, and unreal-hsml-topic—to handle simulation-specific messaging.

MySQL Setup: I created a database called did_registry inside the EC2 instance to store HSML entities. I manually added six test entities (*Alicia_Sanjurjo, Jared_Carrillo, ViperA_Unity, Cadre_A_Omni, Credential_ViperA_allows_CadreA, Credential_CadreA_allows_ViperA*) and verified correct registration using SQL queries.

HSML API Testing: I cloned the HSML API from GitHub, configured it in a virtual environment, and connected it to the Kafka and MySQL services. I tested registration and message publishing through FastAPI, confirming end-to-end functionality.

Real-Time Communication: Messages sent via Kafka from one simulation environment (e.g., Omniverse) were successfully consumed by another (e.g., Unity), enabling synchronous state updates across platforms.

Conclusions

This research is a stepping stone toward achieving complete interoperability across digital twin environments for aerospace simulations—and potentially for industries such as healthcare, business, and energy. By integrating Kafka, MySQL, and the HSML schema using either AWS EC2 or JPL FUTURAMA Wi-Fi, the team created a scalable system for identity registration and real-time data exchange. As the framework grows, it will support future space mission simulations where reliable communication between rovers and spacecraft is essential.

This project also highlighted challenges such as dynamic IP management in EC2 and Kafka listener configurations, which were affected by security limitations. Future research will involve testing in additional environments and expanding the HSML schema to support more entity types.

Methods

AWS EC2 Setup

1. Launched multiple Ubuntu instances on AWS EC2
2. Configured SSH access and opened ports (22, 3306, 9092, 9094)

3. Installed necessary packages: Docker, Kafka, MySQL, Python environments

Kafka Configuration

1. Deployed Kafka using Docker and configured it with a ./start_kafka.sh script to dynamically insert the EC2's public IP into server.properties
2. Created Kafka topics

MySQL Database

1. Installed and set up MySQL inside the EC2 instance
2. Created the did_registry database and did_keys table with fields for did, public_key, metadata, and more
3. Manually added six entities and verified each using SQL queries

HSML_API Setup

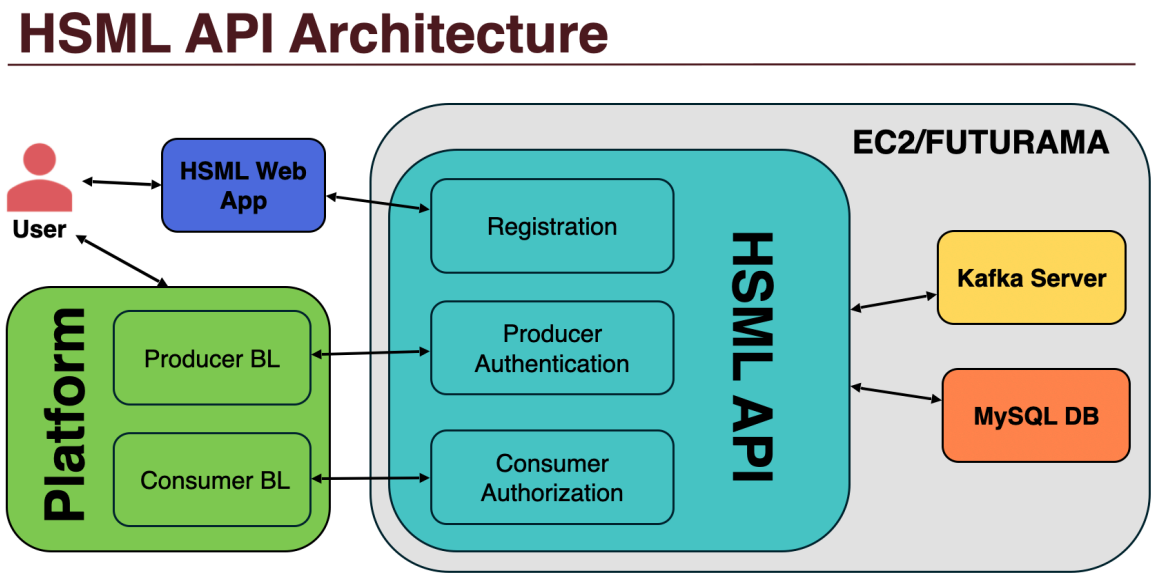
1. Cloned the HSML_API repo from GitHub
2. Activated a Python virtual environment and installed dependencies
3. Tested FastAPI endpoints for registering DIDs and pushing messages to Kafka topics

Schema Validation and Testing

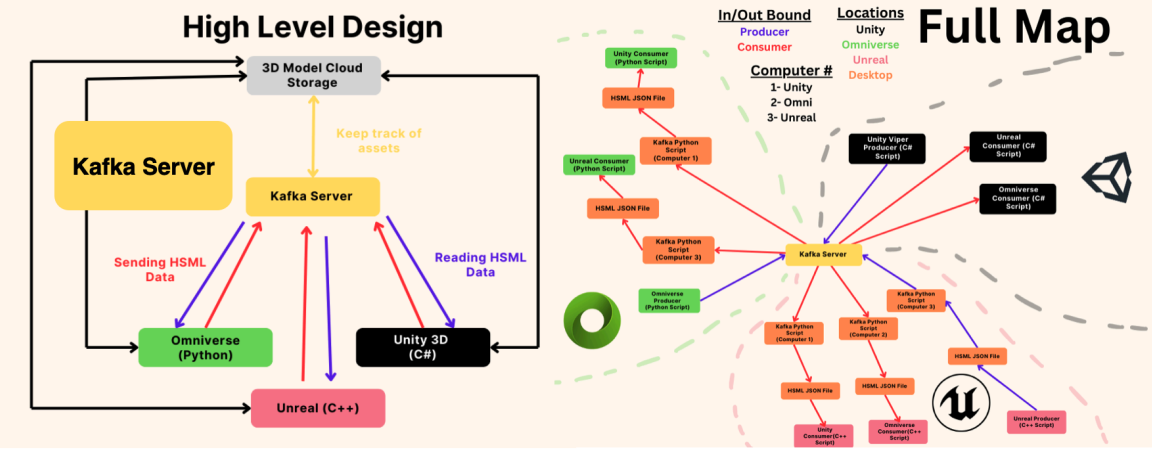
1. Used the HSML schema to validate identity formats and implement tests to ensure correct functionality

Figures

1. HSML API Architecture by Alicia Sanjurjo-Barrio



2. DoD Implementation Design by Jared Carrillo



References

1. AWS Engineering. *Deploying a Django Project on AWS EC2 – A Step-by-Step Guide*. https://medium.com/@awseng12/deploying-a-django-project-on-aws-ec2-a-step-by-step-guide-787f646b5e77
2. Brownhall, I. (2023). *AWS: Hosting a Python API on an EC2*. https://medium.com/@indy.brownhall/aws-hosting-a-python-api-on-an-amazon-ec2-with-a-certified-https-domain-name-c919984dd315
3. Cobb, C. (2021). *Kafka Producer and Consumer*. https://medium.com/@cobch7/kafka-producer-and-consumer-f1f6390994fc
4. Digital Twin Interoperability GitHub. https://github.com/Digital-Twin-Interoperability/hsml_schema
5. Docker Documentation. *Install Docker Engine*. https://docs.docker.com/engine/install/
6. FastAPI Documentation. https://fastapi.tiangolo.com/
7. Flask Framework Documentation. https://flask.palletsprojects.com/en/stable/
8. W3Schools. *MySQL Installation on Windows*. https://www.w3schools.com/mysql/mysql_install_windows.asp

Acknowledgments

Thank you to my mentors, Dr. Thomas Lu and Dr. Edward Chow, for their guidance and expertise. I am also extremely grateful to the intern team—Alicia Sanjurjo Barrio, Jared Carrillo, Gerald Parish, Subhobrata Chakraborty, Gabriel Venezia, Niki Namian, Sohee Kim, Joshua Drye, Aaron Hui, and Diego Cordova—for their collaboration and support. Lastly, I want to thank the SIRI internship program team for giving me this once-in-a-lifetime opportunity to research and learn more about space exploration at NASA JPL.