University of North Texas
Department of Electrical Engineering

In Collaboration with
Hermes Autonomous Mobility Systems

Hermes Architecture

Authors
Dr. Kamesh Namuduri, University of North Texas
Dr. Ravi Subramaniyan, Hermes Autonomous Mobility Systems
and
Nelson Ruiz, University of North Texas

Table of Contents

1 Introduction

The primary focus of this document is to let our third parties know what Hermes can do, how and who is Hermes communicating with, and the overall status of Hermes. If any of our third party wishes to understand Hermes' current situation, they may do so by reading this document. This will eliminate confusion, and give our reader a clear understanding of Hermes current status in regards to utility, functionality, and communication. If there is any confusion or missing information within this document then please contact Nelson Ruiz at NelsonRuiz@my.unt.edu to resolve any questions in your queries.

2 REST ARCHITECTURE

Hermes is designed as a REST API Architecture. The client and server are independent of each other: a change in code from the client side will not affect the server and a change in code from the server side will not affect the client. The transmission or reception of data must be in JSON with a header type of "application/json". Every new message that contains data will not affect or delete previous data, each data transmission/reception is stored in our database. A more detailed description of Hermes' database is provided below.

2.1 HTTP Requests

The request method that Hermes uses to send or receive payloads from other clients is HTTP requests. There are three types of HTTP requests that Hermes accepts:

1. POST - Creates a new table
2. PUT - Update an existing table
3. GET - retrieve a table

2.2 Data Type

Hermes accepts and stores JSON data. JSON data types include:

1. String
2. Number
3. Boolean
4. Null
5. Object
6. Array

Figure 1 shows an example of a standard JSON format that Hermes may accept as a payload. The left side of the colon is called a key and it is used to store a value which is at the right side of the colon. For this example, speed stores a value of 200.1, time_measured stores value and format, and lastly telemetry stores time_measured and velocity. In this case, telemetry, time_measured, and velocity are considered objects, since they store keys with values; all other keys are considered simple data types such as numbers and strings.


```
    "telemetry": {
            "time_measured": {
                "value": "2023-07-14T18:23:50.52Z",
                "format": "RFC3339"
},
            "velocity": {
                "speed": 200.1,
                "units_speed": "MetersPerSecond",
                "track": 120.5,
                "speed_type": "GROUND"
}
}
```


Figure 1: JSON Format

2.3 Tokens

Hermes requires a token[1] to three endpoints:

1. /psu_client/v1/operational_intent_references/<string:entityID>
2. /psu_client/v1/operational_intent_references/<string:entityID>/<string:ovn>
3. /psu/v1/telemetry

To be able to access these three endpoints, Hermes token or AAMTEX token is required. The Hermes token can be retrieved by using the POST method towards our endpoint: "/token". A client ID and a secret ID are needed toward this endpoint; otherwise, Hermes will deny the request. If an AAMTEX token is used, Hermes can accept AAMTEX token as well and the client will be given access to our endpoints. To use the token, it needs to be placed inside the headers as a value with a key named "Authorization".

Hermes interacts with other servers that require their token. The tokens that Hermes requests and uses to communicate to other endpoints are from AAMTEX, Avianco, and OAuth2.0. A table is provided below that explains what type of token is used and their utility.

[1]Token - data that allows a client to access a server's endpoints for a given amount of time. Usually, a token expires in 24 hours.

Token Table	
Token	Utility
AAMTEX	Forwarding Telemetry to AAMTEX DSS Subscription Allow clients to interact with Hermes' token required endpoints
Avianco	Forwarding flight command UAV to Avianco Forwarding OI to Avianco Forwarding Telemetry to Avianco Renew Avianco Token
OAuth2.0	Ping FRAIHMWORK
Hermes	Allow clients to interact with Hermes' token required endpoints

3 Communication

3.1 Operational Intent

A client can send an operational Intent (OI) to two of our endpoints. Hermes does not activate OI, instead our server checks for bugs in the payload; if the payload is in good condition, Hermes forwards it to Avianco and stores the payload to our database. Avianco validates OIs and returns an OVN to Hermes' endpoint: /psu/v1/operational_intents. This OVN can be used to change the state of an OI. The different states are listed below, they are followed in sequence.

1. Draft
2. Accept
3. Activate
4. End

The two endpoints used to forward and store the OI are listed below:

- /psu_client/v1/operational_intent_references/<string:entityID>
- /psu_client/v1/operational_intent_references/<string:entityID>/<string:ovn>

3.2 DSS

Hermes is currently subscribed to DSS, where the subscription region encapsulates Discovery Park from the University of North Texas. This subscription allows Hermes to be notified of any OI that is in the subscription region. The OI is then stored in our database. An important distinction, Hermes does not receive the OI from DSS, which is owned by AAMTEX, but gets the OI from Avianco. Where Avianco sends a POST to our endpoint: /psu/v1/operational_intents. Quick recap, Hermes sends a subscription to AAMTEX (DSS), which notifies Avianco, and then Avainco notifies Hermes if there is an OI in the subscribed region. This OI is then stored and can be used for Telemetry.

3.3 FRAIHMWORK

FRAIHMWORK is operated by ResilienX, it is a system that reports a component's health through a graphical user intergace (GUI). As of now, Hermes is sending data to FRAIHMWORK on the health conditions of Hermes. An OAuth2.0 token is used to make this process feasible.

3.4 Telemetry

Telemetry data includes live flight positional data, with parameters like altitude, speed, latitude, and longitude included. Air Commons sends Telemetry data to Hermes with their OI that they activated through Avianco, this payload is then forwarded to AAMTEX, Lone Star, Avianco, and ResilienX.

3.5 Constraints

Weather constraints refer to any weather advisories along the specified flight path during pre-flight or in-flight. CASA/Delmonte Systems sends the weather constraints to a PSU in response to operational intent and telemetry data, and the PSU forwards it to the Hermes web server, which then passes the constraints, if there are any, back to the operator.

Figure 2 shows the inter-communication between Hermes and its third parties.[3]

[3]NOTE: There are a few mistakes in figure 2. The OI and Constraints come from Avianco, not from DSS. For UMEX to communicate with Hermes, they need a token from Hermes or AAMTEX, not just from Hermes. This will be fixed.

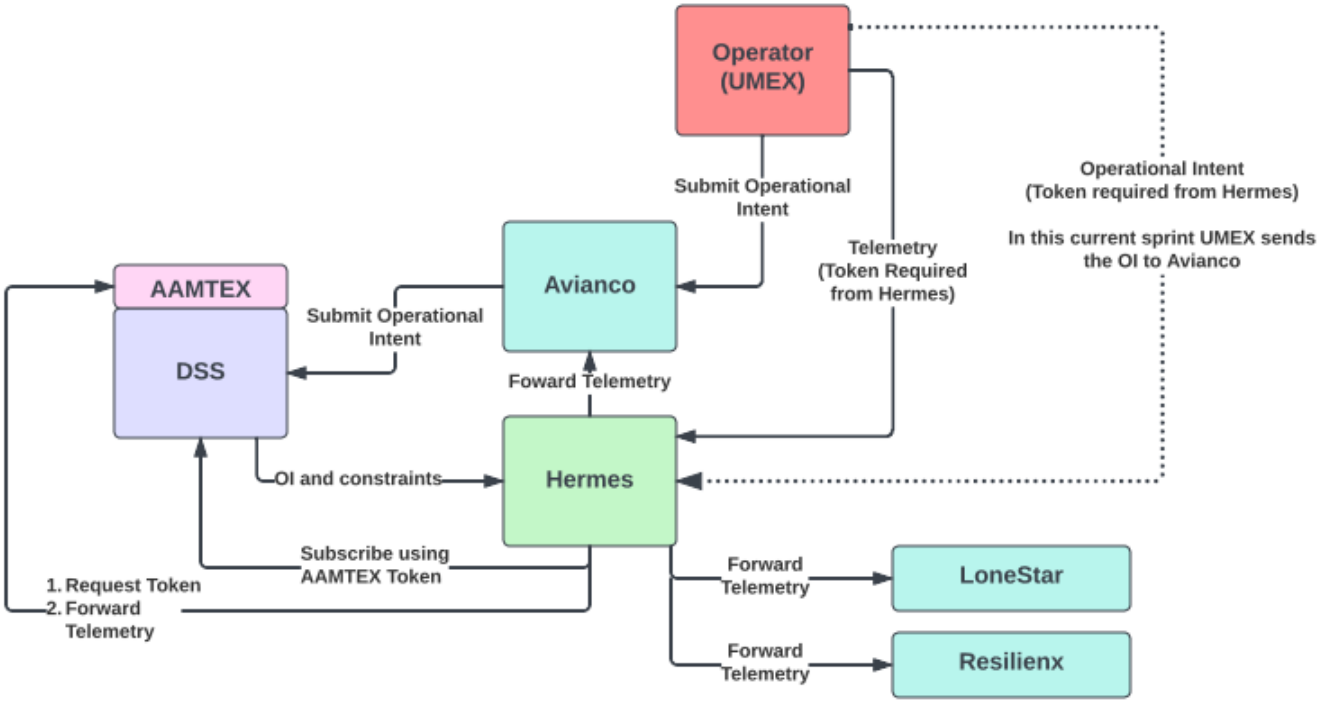

Figure 2: Hermes Interactions

4 Database

Hermes database supports 12 tables[2], where each table is associated with one or more endpoints. Below is a list of tables and their respective endpoints with their HTTP method.

Database Table	
CommandUAV	POST /psu_client/v1/flight/commandUAV
ConstraintDetails	POST /psu/v1/constraints
ConstraintReference	POST /psu/v1/constraints GET /psu/v1/constraints/<string:entityID>
ConstraintSubscriptions	POST /psu/v1/constraints
OperationalIntentReference	PUT /psu_client/v1/operational_intent_references/<string:entityID> PUT /psu_client/v1/operational_intent_references/<string:entityID>/<string:ovn> POST /psu/v1/operational_intents
OperationalIntentSubscription	POST /psu/v1/operational_intents
PsuClientOperationalIntentReference	PUT /psu_client/v1/operational_intent_references/<string:entityID> PUT /psu_client/v1/operational_intent_references/<string:entityID>/<string:ovn>
PsuClientTrajectory	PUT /psu_client/v1/operational_intent_references/<string:entityID> PUT /psu_client/v1/operational_intent_references/<string:entityID>/<string:ovn>
RequestLog	GET /psu/v1/logs
Subscriptions	POST /psu/v1/constraints
Telemetry	POST /psu/v1/telemetry
Trajectory	POST /psu/v1/operational_intents

[2]Table - a subsection of a database where columns represent the variable type with their specified value (similar to JSON) and rows represent the quantity of data.