CONCEPTUAL ARCHITECTURE

FLIGHTGEAR

GROUP 3 – PROJECT FLIGHT FORGE

VIDEO URL: https://www.youtube.com/watch?v=IorS11IWSLQ

OUR TEAM

- Team website: https://18ds25.github.io
- **Leader:** Alan Teng (Derivation process, Evolution)
- **Presenters:** Divaydeep Singh (Architectural overview, subcomponents 5.1, 5.2, 5.3)
  Lucas Coster (Introduction, Data Flow)
- Members: Zhongqi Xu (Lessons learned, conclusion, Divisions of responsibility)
  Antonio Sousa-Dias (Use cases/Sequence Diagrams)
  Donovan Bisson (Subcomponents 5.4, 5.5, Concurrency)

INTRODUCTION

- What is FlightGear?
- Derivation process.

WHAT IS FLIGHTGEAR?

- Free and Open-Source Flight simulator.
- Initial Release in 1996.
- Includes real-time weather updates.
- Multiple aircrafts and environments.
- Used in aerospace research and industry.

DERIVATION PROCESS

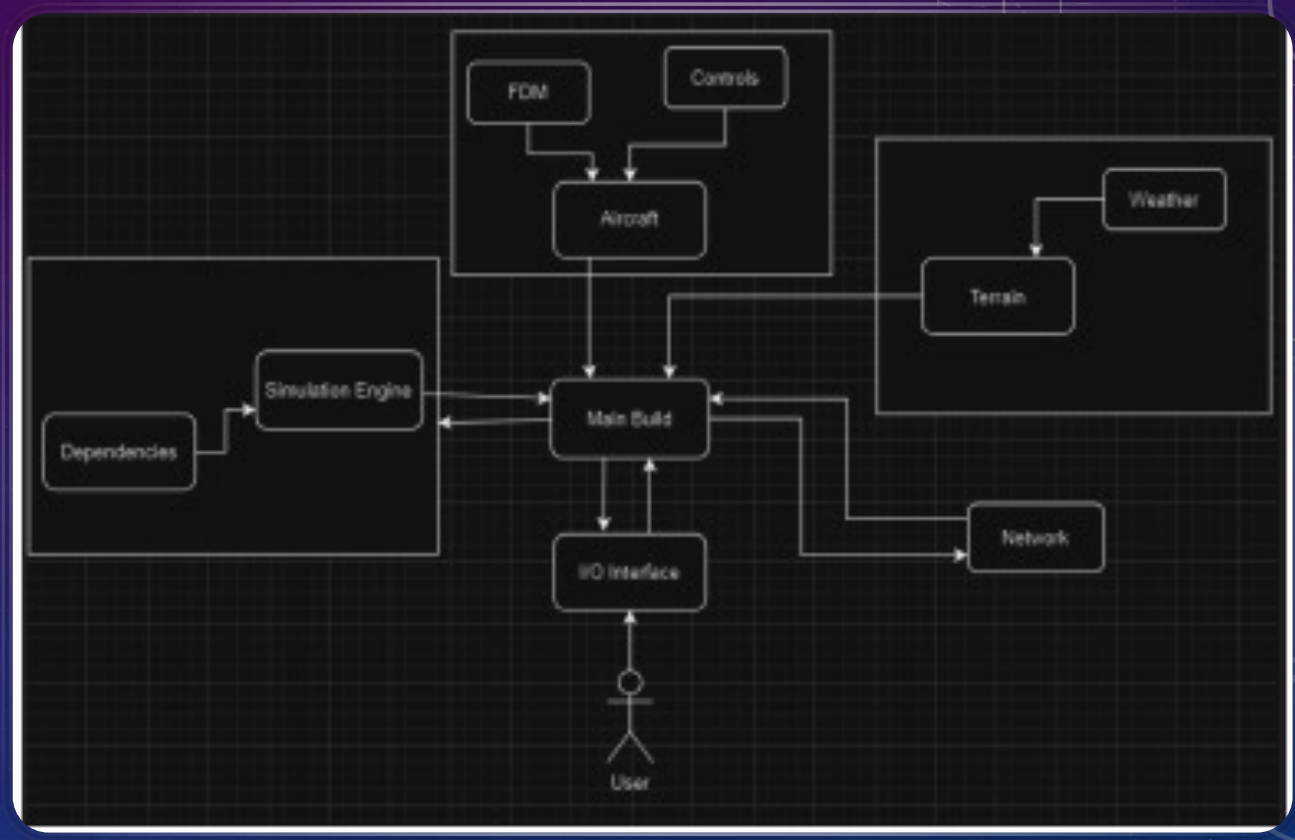
Books

Scholarly Journals

Code

ARCHITECTURE OVERVIEW

- FlightGear is highly modular.
- Data is routed through various specialized submodules.
- Each module processes its own data.
- Overall architecture is believed to utilize a combination of various styles.
- Object-oriented approach for various simulation aspects.

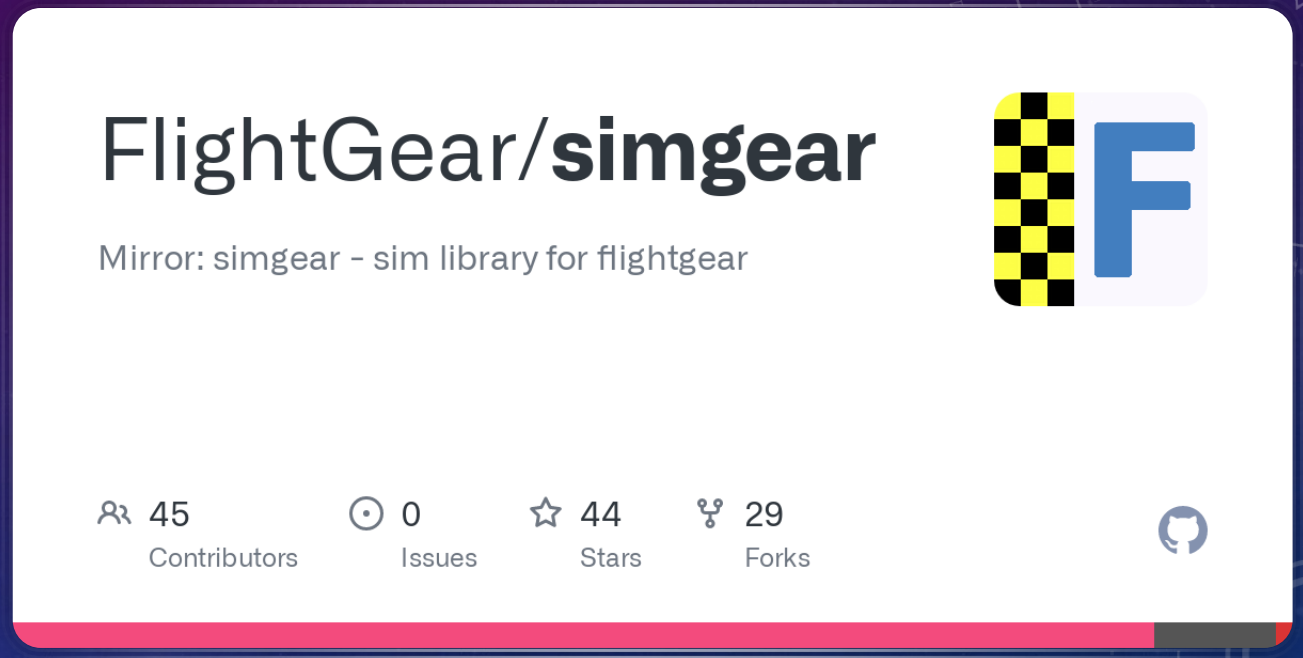


MAIN BUILD

- The simulation host.
- Coordinates data processing between various submodules.
- Runs essential background processes such as Terrasync.
- Accepts user inputs and displays simulation results.
- Outputs system conditions and failure scenarios.

SIMULATION ENGINE

- FlightGear is dependent on SimGear.
- Main engine responsible for generating visuals.
- Converts data from main build into relevant inputs.



TERRAIN

- Responsible for files that generate terrain features.
- Dynamically fetches terrain information from HTTP server.
- Uses TerraSync through main build for terrain generation.
- Requires user’s virtual location as input.
- Fetches files containing weather, terrain, and airports.

AIRCRAFT

- The virtual 'hardware' the user pilots
  1. APPEARANCE
     - How the aircraft LOOKS
     - Aircraft model, VFX, SFX
     - Cockpit USER INTERFACE (altitude, fuel, engine status...)
  2. FLIGHT DYNAMICS MODEL (FDM)
     - How the aircraft FLIES
     - Interaction with ENVIRONMENT (thrust, lift, drag, collision...)
     - Interaction with USER (control mapping, input changes to simulation)

NETWORK

- How FlightGear instances communicate with each other
- Allows for LAN/online MULTIPLAYER
  - Formation flying, air traffic control simulations with other users
  - Google Maps integration (See where other users are flying!)
- Allows for MULTI-SCREEN simulations
  - Synchronizing multiple instances of FlightGear
  - Multiple perspectives, same simulation

DATA FLOW

- Main build initializes all necessary data.
- Aircraft module calculates aircraft performance using FDM.
- Terrain engine retrieves and builds environment.
- Simulation engine processes data to generate visuals.
- Resulting simulation is displayed to user through main build.

EVOLUTION

- Early Years
- Middle Life
- Recent Years

EVOLUTION

- First released in 1996 by David Murr.
- Offered Rudimentary simulation in early stages.
- Experienced significant growth since 2001.
- Version 0.9.10 introduced enhanced flight dynamics, and broad platform support.
- Ongoing development focuses on refining realism, VR support, etc.

CONCURRENCY

- How FlightGear currently works?

- High Level Architecture?

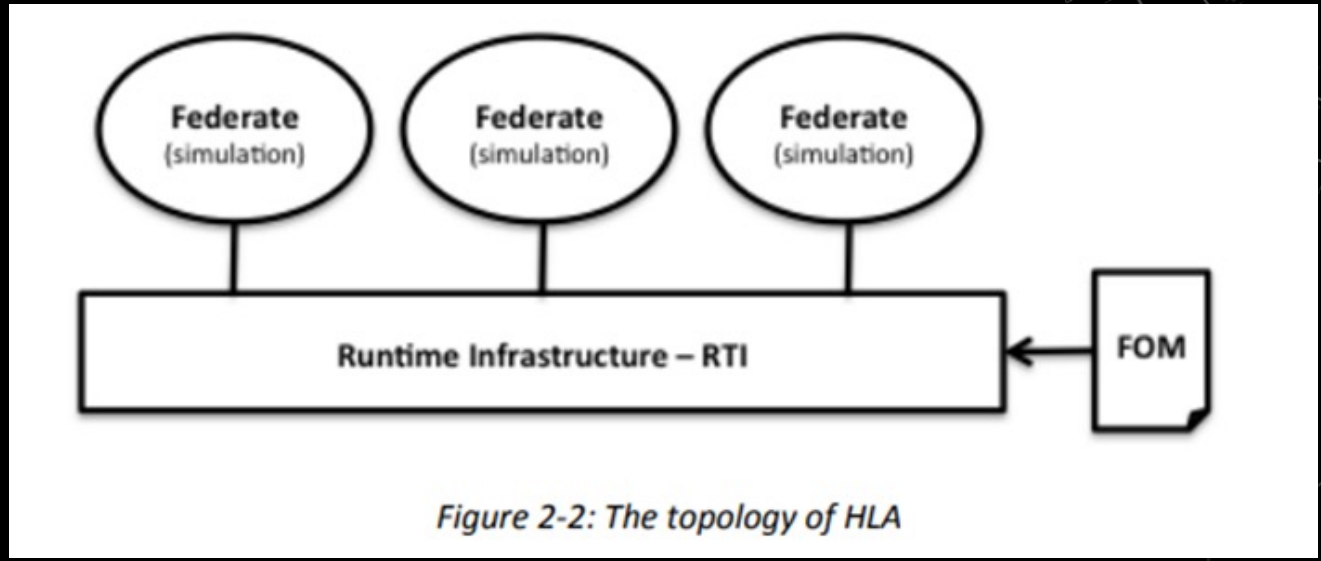


Figure 2-2: The topology of HLA

CONCURRENCY

- Initial belief was single-core performance would improve over time.
- FlightGear runs with repeating main loop.
- Performance of single component can impact entire system.
- Ongoing discussions about supporting multithreading.
- Still a work in progress.

HIGH-LEVEL ARCHITECTURE

- Identified by the community to best meet FlightGear needs.
- Each federate is connected to an RTI.
- Provides information, synchronization, and coordination.
- Each independent federate can simulate aircraft, record data etc.
- FlightGear is implemented in C/C++ which have poor multithreading support.

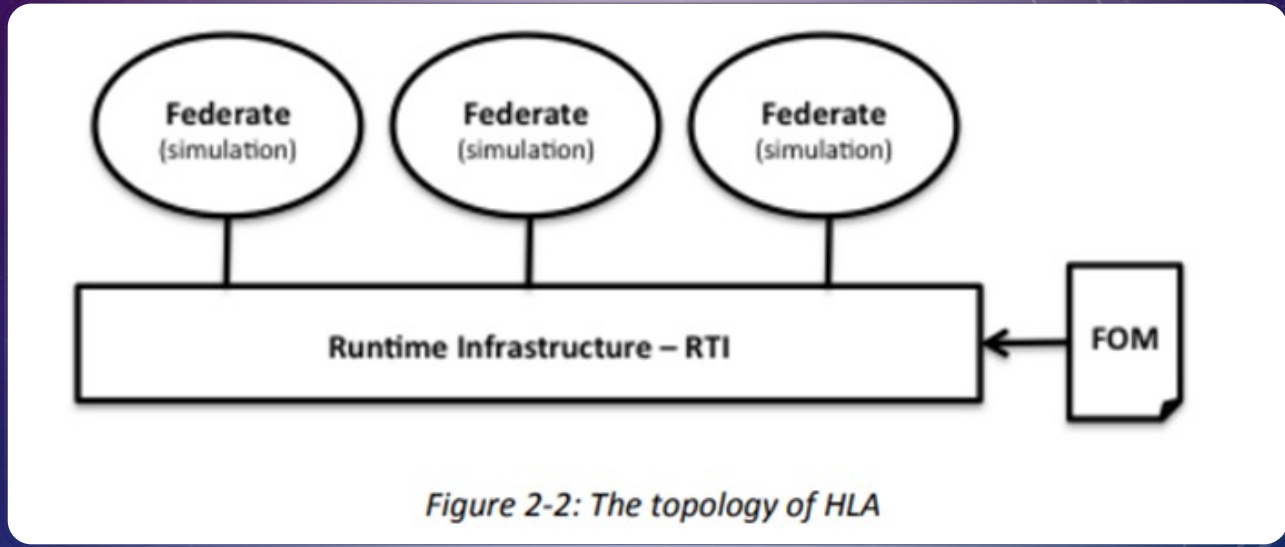


Figure 2-2: The topology of HLA

DIVISIONS OF RESPONSIBILITY

- OPEN SOURCE!
  - FlightGear engine free to download/modify via GitHub repository
  - Any contributor can propose changes/additions to any part of the project
    - Core functionality regulated by veteran community members, for code integrity
  - Scenery toolkit available for creating custom terrain
  - Aircraft are modular – Use the template structure to create custom Aircraft modules!
- AIRCRAFT/SCENERY DATABASES
  - Project supports official Scenery and Aircraft databases, and third-party database integration
    - Official databases are minimally regulated
  - Contributors can submit custom Scenery and Aircraft for public use

LESSONS LEARNED.

- FlightGear has been in continuous development for 30 years.
- Employs a modular architecture style.
- Specialized submodules route data to achieve functionality.

CONCLUSION

- Please reach out to team members for any questions.