Running Large-Scale and Hybrid Real-Time Aircraft Simulations in an HLA Framework

Jean-Baptiste Chaudron(✉), Aleksandar Joksimović, Pierre Siron, Rob Vingerhoeds, and Xavier Carbonneau

ISAE-SUPAERO, Université de Toulouse, Toulouse, France
{jean-baptiste.chaudron,aleksandar.joksimovic,pierre.siron, rob.vingerhoeds,xavier.carbonneau}@isae-supaero.fr

Abstract. Aircraft systems and associated technologies have been constantly evolving over the years with an increasing number of components and actors interacting together in a global chain. Given this evergrowing complexity, the use of simulation in the design and validation phases of emerging systems is essential and allows to significantly accelerate some prototyping phases, especially in cross-domain concepts. This article describes analysis and experiments done at ISAE-SUPAERO for running large scale and hybrid real-time simulations for aircraft using the HLA standard (High Level Architecture).

Keywords: Aircraft simulation · Distributed systems · Co-simulation · HLA

1 Introduction

The complexity of aircraft systems in their environment makes it crucial to analyse and understand their characteristics and behavior early in the design phases. Modeling and simulation (M&S) play an important role in aircraft systems development, where simulation models are used for prototyping and evaluating systems in order to identify potential problems or even misconceptions and allow for decisions at all development stages. Aircraft sub-systems simulation targets cross-domain, multi-scale, multi-physics and multi-fidelity models. The high increase of computer processing power enables many possibilities such as high fidelity simulations. Moreover, emerging computer network technologies, especially the rise of Ethernet technology from low [21] to high speeds [14], have led to distributed computing, where different computers are interacting over a communication network to achieve a common goal, thus providing more power and efficiency to solve very complex problems. However, such configurations come with a cost and designers of distributed applications have had to face several problems such as handling the heterogeneity of the hardware and software distributed components. The efforts to tackle these interoperability issues have led to the development of multiple standards such as Common Object Request

Broker Architecture (CORBA) [26], Data Distribution Service (DDS) [26], Functional Mock-up Interface (FMI) [22] or the non-standardised approach proposed in Robotic Operating System (ROS) [10,11,27]. Such middleware standards provide the following advantages:

– *Interoperability*: defining a common and shared framework for the connected systems to interact together and exchange data,
– *Modularity*: easing the operation of adding or replacing entities, and
– *Distribution*: allowing the processing entities to be located on different computers on a communication network.

For simulation purposes, the specialised High-Level Architecture (HLA) standard [31–33] has been developed to provide a shared and documented framework for interoperability and reusability of heterogeneous distributed simulation. Although interoperability and reusability remain essential, both high performance and availability have also to be considered to fulfill the requirements of the simulation. For many years now, we have been using our implementation of the HLA standard (called CERTI) to develop and maintain hybrid real-time aircraft simulations and allow testing of real hardware components within the simulation loop. This article presents our framework using the HLA standard for aircraft simulations, its architecture and particularities, and an open-source implementation. An aero-propulsive aircraft/engine test scenario for distributed simulation from this open source implementation is presented as illustration. The remainder of this paper is structured as follows:

– Section 2 presents an overview of our HLA simulation framework.
– Section 3 continues with an outline of the application architecture and its particularities.
– Section 4 describes the engine models for the case-study.
– Section 5 discusses the experiments and the obtained results.
– Finally, the paper is wrapped up with some conclusions and perspectives for future work.

2 An HLA Based Framework

2.1 CERTI

In distributed systems, "middleware" is the term used to describe a software layer implementing the services and framework (defined in a standard) to connect the distributed entities. For HLA, the middleware is referred to as the Run-Time Infrastructure (RTI) and implements the HLA Interface Specification part of the standard [31]. Thus, it provides a set of software services to all the connected HLA-compliant applications called "HLA federates". As depicted in Fig. 1, the RTI can be seen as the intermediary layer allowing the global execution of the distributed simulation and hiding the complexity of the underlying distributed system. In reality, each computers embed a local component handling the services implementation.

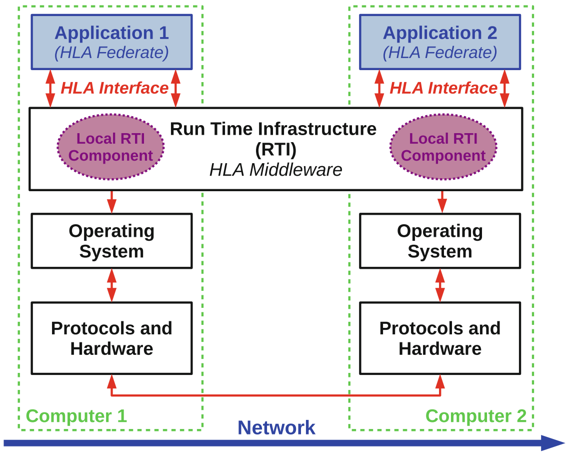

Fig. 1. HLA/RTI as middleware for distributed simulations.

For years, ISAE-SUPAERO has been developing and maintaining an RTI called CERTI. This project started in 1996 [30] and is available as open-source software [25]. CERTI is recognizable through its original architecture of communicating processes implemented in `C++`. Each HLA-compliant simulation (called a "federate") is connected to a local RTI Ambassador (RTIA) process and linked to a library (libRTI) providing the HLA interface services. A global shared process called "RTI Gateway" (RTIG) acts as a software-based router and connects all available RTIAs. Each federate process interacts locally with its RTIA via a Unix-domain socket and the RTIA processes exchange messages over the network, in particular with the RTIG process via Transmission Control Protocol (TCP) sockets. This architecture is depicted in Fig. 2.

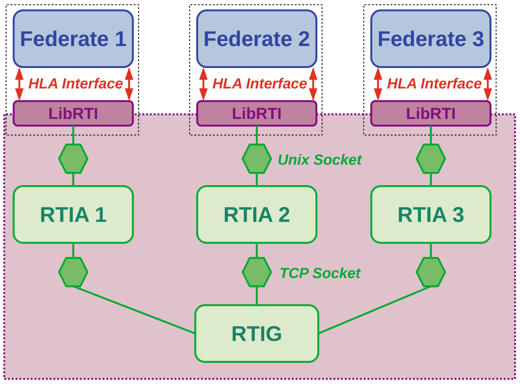

Fig. 2. CERTI architecture.

2.2 Federates and Models

When building a simulation, the studied system and its environment can be considered as a set of components with various interactions between them. From CERTI point of view, we must distinguish the model for each simulation component and the federate process (a `C++` code) that integrates and implements one or more models (see Fig. 3). One way to design and to build a distributed simulation (i.e. a HLA federation) is to associate a federate only one component model. A federate may also encompass several models for the sake of simplicity of the overall architecture or to enable a performance optimization of the global simulation [12].

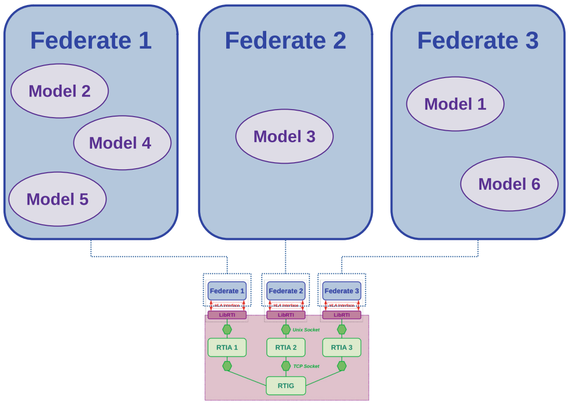

Fig. 3. Illustration of three federates with different models.

The interoperability between the individual models within one federate belongs to the federate designer. However, it is necessary to describe the HLA interfaces of the federate i.e. the data updated by the federate model(s) to the RTI and the data consumed by the federate model(s) from the RTI (data provided by other models within remote federates). The HLA standard requires this description in terms of object classes with their attributes plus some additional information such as the data representation for *technical* interoperability or the unit and referential for *semantic* interoperability.

2.3 Time Management Services

Time management services provided by the HLA middleware form one of the main benefits of this simulation standard and make this standard special compared to others [15]. The time management services enable the access to a consistent global logical time for all the distributed entities involved in the simulation. A time-stamp is assigned to each message and the RTI must ensure that all the messages are delivered with respect to time-stamp order. Therefore no message is allowed to be delivered to a federate in its past (from the logicial time scale point of view). This allows to have a deterministic and reproducible simulation which respects all the computing dependencies expressed with a logical time ordering. Different algorithms can be used to implement time management services and provide this consistent logical baseline. The most used algorithm is the so-called Null Message Algorithm (NMA) from Chandy and Misra [4], implemented in CERTI. The approach is based on a contract for each federate called "lookahead". Each federate undertakes not to send simulation messages with a logical timestamp less than its local time plus its lookahead. This algorithm is suitable for real-time and we have proposed some formal analysis [5] as well a new optimized version, called Null Message Prime (NMP) [9]. In addition, we have shown that these algorithms can be particularly useful when solving distributed differential equations such as in a complex aircraft distributed simulation [8].

3 SMARTIES Project

3.1 Background

In the context of the Research Platform for Embedded Systems Engineering Project (PRISE), we developed a first version of an aircraft flight simulation [7,16], called SDSE (French acronym for Distributed Simulation of Embedded Systems). However, although relevant and functional, this application had two disadvantages:

1. The aerodynamic model of the aircraft (i.e. the definition of its aerodynamic coefficients) as well as the propulsive system model were basic and their realisation could not be validated against real-life data nor any publication.
2. The part concerning the autopilot and flight control systems were integrated in a HLA simulator, but not integrated on a real dedicated on-board system (i.e. not a simulated system). Therefore, this HLA simulation was not a real Hardware-In-the-Loop (HIL) simulation.

Based on these findings, a new project, called Simulation Modules for Aircraft Real-TIme Embedded Systems (SMARTIES), was initiated. In order, to address the first limitation of SDSE, two well-documented flight dynamics models with available open source references have been implemented:

- A high-fidelity model for a Boeing 747-100 as described in National Aeronautics and Space Administration (NASA) references [17,18].
- A high-fidelity model for a F16 as described in Air Force Institute of Technology and NASA references [20,24].

The information provided in these references is very detailed and makes it possible to faithfully reproduce behaviour of these two aircraft and thus allow for realistic simulation on the platform.

The second limitation was addressed by the design and implementation of a full avionics bay connected to the HLA simulator entities and which are running autopilots, fly-by-wire and flight control systems on dedicated on-board computers including the fault tolerance redundancy handling [6].

3.2 Architecture Overview

The SMARTIES architecture, inherited from SDSE, is composed of 10 HLA federates, each representing a specific part or system of the aircraft and its environment (see Fig. 4).

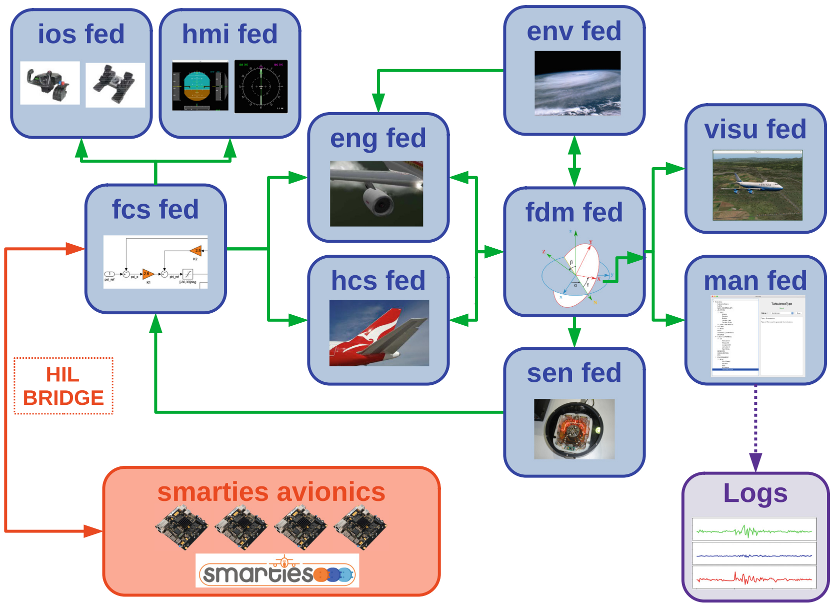

Fig. 4. SMARTIES architecture illustration.

- The Inputs/Outputs federate (***ios fed***) acquires the pilot's inputs in the cockpit from the multiple interfaces such as the side sticks, throttle, etc. The elevator, aileron, and rudder axes can then be commanded through this interface with a processing via the flight control laws implemented in the fcs and avionic systems.
- The Human-Machine Interfaces federate (***hmi fed***) is used to display essential flight information in the cockpit interfaces such as Primary Flight Display (PFD), Navigation Display (ND) or Electronic Centralized Aircraft Monitor (ECAM).
- The Flight Control System federate (***fcs fed***) is in charge of the aircraft control and implements two types of functions:
  1. the autopilot functions such as speed and altitude hold control systems;
  2. the low level flight control functions (i.e. control and stability augmentation systems).

The implemented control laws are a combination of the control laws documented in the reference documents as well as improvements realised in-house. These flight control laws can run either on a simulation entity (within a federate) or on the specialized SMARTIES avionics bay [6].

- The Engine federate (***eng fed***) simulates either 4 high-bypass turbofan engines for the B747 aicraft or 1 low-bypass turbofan engine for the F16 aircraft. The engine characteristics change with the atmospheric conditions as well as the aircraft flight Mach number. This federate receives the required throttle commands from the flight control system federate. A new model for this federate is presented in Sect. 4.
- The Hydraulic Control Surfaces federate (***hcs fed***) implements all the hydraulic actuators behaviors whose deflections change the aerodynamic forces and influence the aircraft motion. The different control surfaces are described in previously mentioned references for each aircraft and modeled either by first or second-order filters with position and rate saturation to enforce realism.
- The Flight Dynamics Model federate (***fdm fed***) represents the core of the simulation and computes the equations of motion. The aircraft trajectory and altitude will evolve with respect the action of aerodynamic, gravity and propulsion forces. The aerodynamic coefficients are implemented in the form of interpolation tables with multiple entries as described in the references.
- The Environment federate (***env fed***) reproduces the weather and climate conditions. It models the US Standard Atmosphere 1976 [23] and depending on the altitude it calculates the corresponding atmospheric variables such as temperature, pressure or air density. It also integrates different types of wind models, such as wind shears and gusts, as well as turbulence models like the ones from Dryden [13,35] and Von Karman [34].
- The Sensors federate (***sen fed***) simulates the different aircraft sensors such as Inertial Measurements Units (IMU) or Global Positioning System (GPS). Each sensor having its own dynamics, we used first or second-order low-pass Butterworth filters [3] augmented with delay, bias, drift, and noise phenomena in order to augment realism.
- The Visualization federate (***visu fed***) is in charge of the 3D virtual environment display and The Manager federate (***man fed***) enables configuration features for the simulation entities along with logs.

3.3 Open-Source Version

We decided to release part of our work for the F16 aircraft simulation as open-source software. In order to ease the usage and deployment of such application for an end-user, the native SMARTIES architecture was simplified. As described in Sect. 2.2, some models can be aggregated within one federate as long as the local scheduling of the entities is guaranteed. Figure 5 illustrates the architecture of our open-source version: 5 models (eng, hcs, fdm, env and sen) are combined within one federate (also named ***fdm fed***). Also, the ios and hmi models are combined into a cockpit federate (i.e. ***ckp fed***). The complex avionics bay composed of several redundant units and corresponding network entities has been replaced to allow users to run the fcs code on an embedded device (having an ethernet interface) of their choice. In the remainder of this paper, the open-source application is considered (the code is available at https://github.com/ISAE-PRISE/smarties_f16).

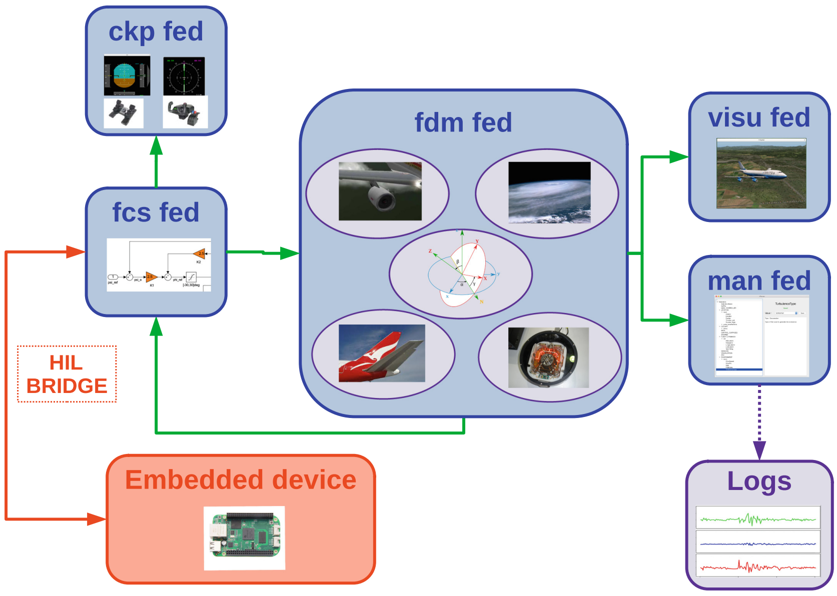

Fig. 5. Open-source SMARTIES architecture illustration.

4 Towards a New F16 Engine Model

4.1 Design Tool

A dedicated software called PROOSIS [2] was employed to create new models for the F16 engine. PROOSIS is an object-oriented software, primarily dedicated to gas turbine engine cycle modelling. It enables the user to both size a new engine architecture and/or to explore engine system design space by evaluating the performance of a pre-sized model through its three-layer structure:

1. Schematic: component-level modelling and whole system architecture assembly,
2. Partition: boundary value definition and initialisation of non-linear terms of the equation system that describes the constructed architecture, and

3. Experiment: engine cycle design and/or off-design simulations in steady state, transient, optimisation scenarios, etc.

Using this tool, the engine cycles were sized for a single operating point corresponding to a typical cruise condition of the F16 aircraft. The boundary conditions chosen to control the model are two engine fuel flows (in core and the afterburner), directly linked to the throttle command of the coupled flight simulator.

4.2 New Engine Models

A new F16 engine schematic (illustrated in Fig. 6) representing an afterburning two-spool low bypass-ratio F110 turbofan architecture was assembled based on the information provided in [1,19].

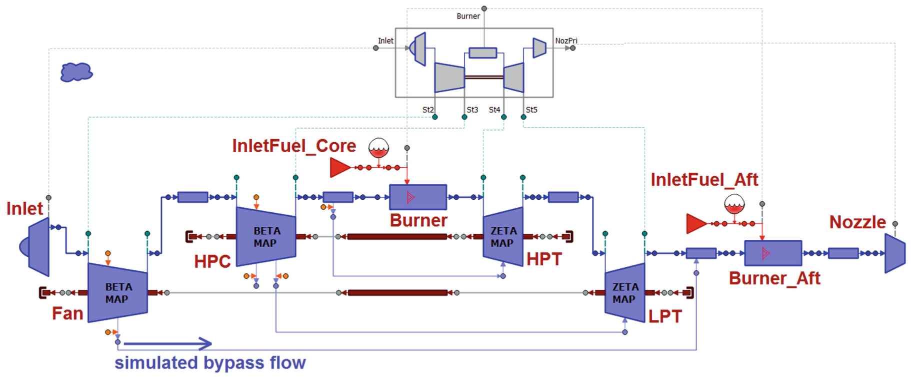

Fig. 6. F110-GE-129 engine architecture (PROOSIS schematic).

The fan was implemented using a compressor component, with the bypass mass flow modelled as bleed at the compressor inlet, re-injected into the main flow right after the last turbine stage. This was done in order to avoid certain inherent difficulties arising from the default library fan model being adapted to single stage components used in civil applications, as opposed to the compact three-stage fan employed in the current engine. The high pressure turbine cooling was modelled as a simple offtake from the high pressure duct re-injected into the high-pressure turbine, since no information concerning this feature was found in the literature. The post-combustion was modelled with an additional burner component situated between the low pressure turbine and the exhaust nozzle. Since no relevant data was found in the public domain, the exhaust nozzle component does not include variable cross-section feature typical for fighter aeroplane engines. As this drawback may only influence the value of the calculated performance, and has no further influence on the framework development at hand, the engine is assumed to have a fixed nozzle, which will be corrected in the future developments if additional data is acquired. Robustness of the resulting cycle model was verified with parametric steady state simulations for the following operating envelope: altitude (from 2000 m to 4500 m), Mach number (from 0.3 to 0.9) and net thrust (from 65000 N to 104750 N).

4.3 HLA/PROOSIS Coupling

There are two possibilities to link the PROOSIS tool with a HLA environment. PROOSIS (based on `C++`) allows the access (via an API) to certain features, internal variables and functions. Therefore, it is possible to implement an internal HLA federate such as depicted in Fig. 7.

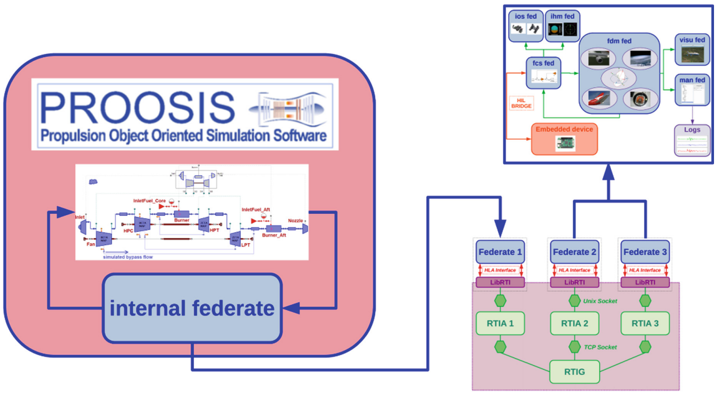

Fig. 7. Illustration of a PROOSIS internal federate.

However, this solution is difficult to handle and requires a lot of knowledge on the internal behavior of the tool. In order to implement a consistent global HLA simulation, it is important to understand the parameters impacting the frequency behavior of models components, their interactions and identify correlations between these models and implemented solvers within PROOSIS. This work is ongoing and not suitable for the open-source version. We therefore generated appropriate look-up interpolation tables (see Fig. 8) that have been integrated into the federate code to provide the necessary engine performance parameters to the simulator.

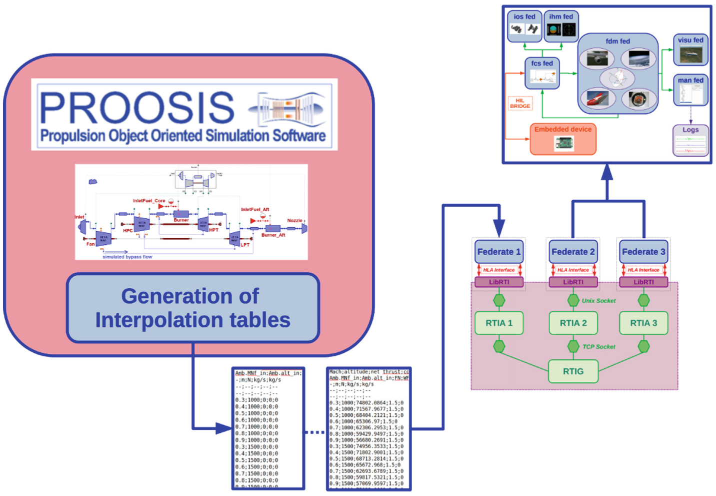

Fig. 8. Illustration of PROOSIS engine performance table generation.

5 Experiments and Results

5.1 Testing Environment

Our experiment takes place at the ISAE-SUPAERO facilities as illustrated in Fig. 9.

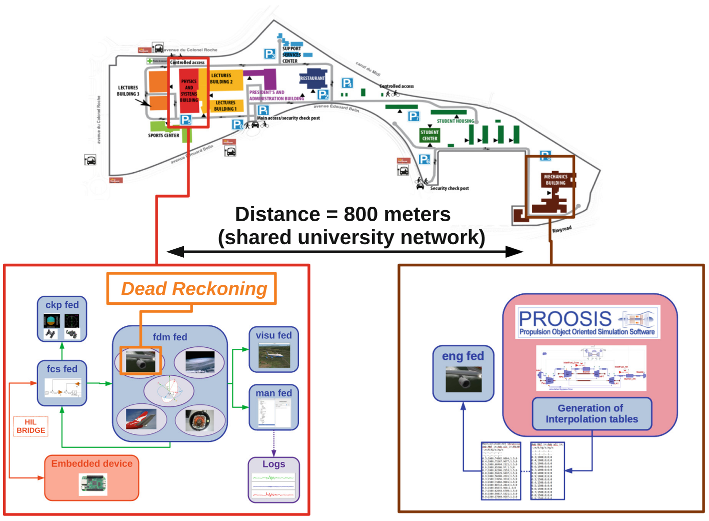
Distance = 800 meters
(shared university network)
Dead Reckoning
ckp fed
fcs fed
fdm fed
visu fed
man fed
HIL BRIDGE
Embedded device
Logs
eng fed
PROOSIS
Propulsion Object Oriented Simulation Software
Generation of Interpolation tables
PHYSICS AND SYSTEMS BUILDING
MECHANICS BUILDING

Fig. 9. Testing environment illustration between DISC and DAEP.

On one side of the campus, the core of the open-source SMARTIES application is running in the Department of Complex Systems Engineering (DISC) building. These entities are sharing a dedicated local Ethernet network. On the other side of the campus at a distance of 800m, the new engine model is running in the Aerodynamics, Energetic and Propulsion Department (DAEP) building on a dedicated computer. The connection between the new engine model and the rest of the simulation is done via the university network, whose bandwidth is shared between all the university students and researchers and can be limited for the simulation data exchanges (i.e. performance issues). This limited performance has to be handled properly to ensure the real-time behavior of the simulation using dead reckoning algorithms [28] combined with HLA time-management services [15] (this is described in Sect. 5.3 and Sect. 5.4). For this experiment, the distribution of the simulation entities is necessary because it enables future possibilities of HIL simulations with real engines running at DAEP facilities and aircraft dynamics models running at DISC facilities.

5.2 Engine Models Comparison

Figure 10 shows the resulting curves comparing the thrust value produced by the new model versus the original values from the NASA reference documents.

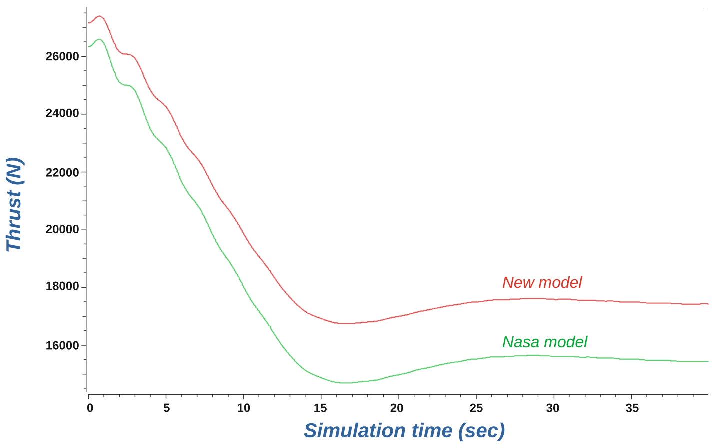

Fig. 10. Thrust comparison between original NASA model and the PROOSIS model.

As it can be observed in the presented data, a discrepancy (thrust overestimation by the PROOSIS model) stabilised at around 7% with respect to the NASA reference is produced for the simulated operating condition. The observed difference is not significant given that the maximum operating thrust (order of 75 kN for cruise without afterburning) is significantly higher than the one currently simulated (order of 20 kN), so the validity of the current model is not compromised. In addition, the NASA model used as a baseline is not a full engine cycle model in itself, but a look-up table providing engine thrust as a function of flight conditions and throttle setting. As such, we have no way of knowing the underlying assumptions concerning either the variable nozzle or any other engine component. It is nevertheless of interest to pinpoint the following drawbacks of the current PROOSIS model, which will inform future improvements:

1. The current engine thermodynamic cycle was sized only for one operating point at 75 kN thrust and in steady state, based on scarce data assembled from several different references. With sufficient reference data, the inherent limitations of single-point cycle sizing method can be overcome in future developments by employing the so-called *multi-point* cycle sizing process [29]. This more complex method is available in PROOSIS and can ensure a broader operating envelope for the sized engine.
2. Variable nozzle cross-section (typical feature to regulate military jet engine performance over its operating envelope) was not taken into account in the developed model.

5.3 Real-Time Performance Results

During the execution, each federate computation and communication is scheduled by time management principles (see Sect. 2.3). A suitable deployment of these techniques ensures a consistent behaviour on the simulation logical time reference. The progress in the logical time is then correlated to an hardware clock to ensure the respect of real-time constraints for federate executions.

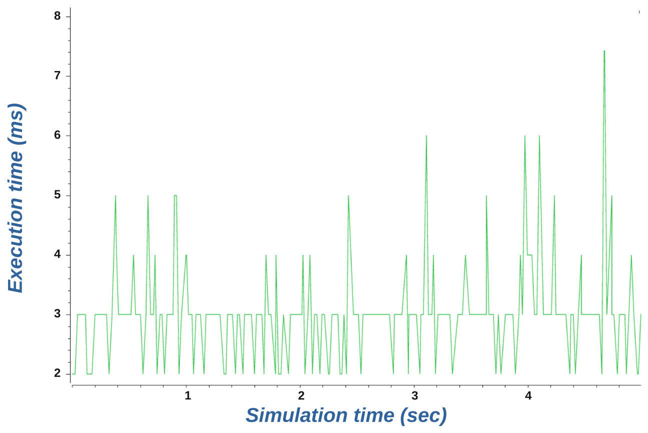

Fig. 11. Fdm federate step times with 10 Hz remote engine federate.

In a nominal mode, the frequency of the simulation 100 Hz which means that a federate must receive, process and send data every 10 ms (deadline). Due to our large scale distributed testing environment, we had to analyse the real-time behavior of the simulation according to the remote engine federate. Our experiments have shown that this federate has to run with a frequency 10 Hz to ensure proper real-time execution. Higher frequencies are causing the 100 Hz federates to wait for the remote federate and violating their 10 ms deadlines. Figure 11 shows the execution steps measurements from the fdm federate with the remote engine federate running 10 Hz; it can be seen that the real-time behavior is correct.

5.4 Dead-Reckoning Performance Results

The real-time performance of the entire distributed simulation is guaranteed using 10 Hz remote engine federate. However, the fdm federate still need relevant thrust inputs 100 Hz. To address this, dead reckoning algorithms [28] have been implemented into the fdm federate in combination with its HLA time management handling. A shadow engine model is integrated into the federate and uses dead reckoning to predict and mimic 100 Hz behavior based on the incoming data received from the remote federate 10 Hz. Two types of dead reckoning techniques have been used: one is only considering the rate of change of the remote value (noted DR1) and one considering the acceleration of change as well (noted DR2). The formulas 1 and 2 are given below with P the value (here the predicted or received thrust value), V is the rate of change of this received value and A its acceleration of change. These predications compared to the remote model are displayed in Fig. 12.

$$P_{DR1} = P_{10Hz} + V_{10Hz} * \Delta t \tag{1}$$

$$P_{DR2} = P_{10Hz} + V_{10Hz} * \Delta t + \frac{1}{2} * (A_{10Hz} * \Delta t)^2 \tag{2}$$

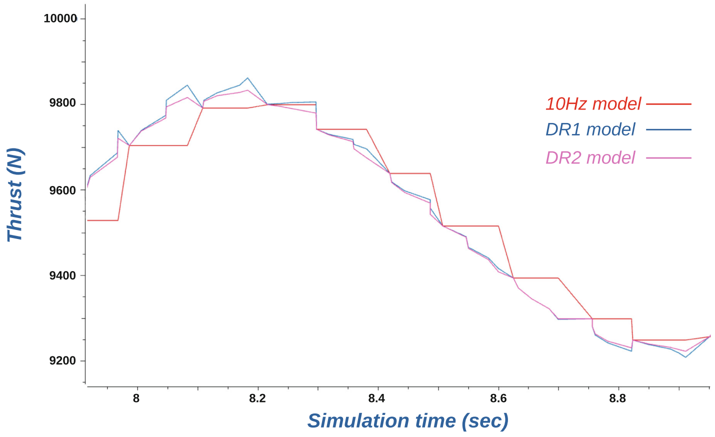

Fig. 12. Thrust comparison 10 Hz model and dead reckoning estimations.

6 Conclusion

This article focuses on the use of an HLA framework to develop, run and extend large-scale, hybrid and real-time simulations for aircraft systems. The approach building on HLA technology was presented, along with a dedicated implementation of a high-accuracy aircraft model from the public domain. The existing engine model was extended using a new dedicated design tool and the new model has been validated with previous model from the literature. A distributed large scale run-time execution experiment was then performed over a university network. Proper handling of the simulation could be confirmed, some enhancements to cover timing issues have been put in place. We have released an open-source package and therefore, our application with its implementation details are accessible, can be used, reproduced and extended. For future work, an extension of the engine model validation is foreseen, along with addressing the mentioned open points, and the overall large-scale simulation scenario will be further enhanced so to allow larger scale simulations to take place. In particular, we are targeting to pursue our effort towards new HIL simulations which will integrate real engines into the simulation loop. We are also working on the integration of multiple aircraft in the simulation loop using different existing simulators such as X-Plane (see https://github.com/ISAE-PRISE/xplane_hla).

Acknowledgements. This work has been realized in the context of the PRISE project (Research Platform for Embedded Systems Engineering) which tends to provide a platform for study, evaluation and validation of new embedded system concepts and architectures to our students and researchers. In addition, this project has inherited from the research work carried out in the scope of the AEGIS research chair, established between SAFRAN Group and ISAE-SUPAERO in 2016 with aim to promote and help develop multi-disciplinary preliminary design and optimisation methods applied to highly integrated propulsive systems.

References

1. Development of the F110-GE-100 Engine, Turbo Expo: Power for Land, Sea, and Air, vol. Volume 2: Aircraft Engine; Marine; Microturbines and Small Turbomachinery (1984). https://doi.org/10.1115/84-GT-132
2. Alexiou, A.: Introduction to gas turbine modelling with PROOSIS, 4th edn. Technical report Empresarios Agrupados Internacional (2020)
3. Butterworth, T.: On the theory of filter amplifiers. Exp. Wirel. Eng. J. **7**, 536–541 (1930)
4. Chandy, K., Misra, J.: Distributed simulation: a case study in design and verification of distributed programs. IEEE Trans. Softw. Eng. **5**(5), 440–452 (1979). https://doi.org/10.1109/TSE.1979.230182
5. Chaudron, J.B., Noulard, E., Siron, P.: Design and model-checking techniques applied to real-time RTI time management. In: Spring Simulation Multiconference-SpringSim 2011. Boston, United-States (2011)
6. Chaudron, J.B., Saussié, D.: Towards the design of a distributed aircraft flight control system connected to simulation components. In: 12e Conférence Internationale

de Modélisation. Optimisation et Simulation (MOSIM 2018), pp. 1–8. Toulouse, France (2018)

7. Chaudron, J.B., Saussié, D., Siron, P., Adelantado, M.: Real-time distributed simulations in an HLA framework: application to aircraft simulation. SIMULATION **90**(6), 627–643 (2014). https://doi.org/10.1177/0037549714531054
8. Chaudron, J.B., Saussié, D., Siron, P., Adelantado, M.: How to solve ODEs in real-time HLA distributed simulation. In: Simulation Innovation Workshop (SIW), pp. 1–12. Orlando, US (2016)
9. Chaudron, J.B., Siron, P., Adelantado, M.: Analysis and optimization of time-management services in CERTI 4.0. In: 2018 Fall Simulation Innovation Workshop (SIW), pp. 1–10. Orlando, US (2018)
10. Cousins, S.: Exponential growth of ROS. Robot. Autom. Mag. IEEE **18**, 19–20 (2011). https://doi.org/10.1109/MRA.2010.940147
11. Cousins, S., Gerkey, B., Conley, K., Garage, W.: Sharing software with ROS. IEEE Robot. Autom. Mag. **17**(2), 12–14 (2010). https://doi.org/10.1109/MRA.2010.936956
12. Deschamps, H.: Scheduling of a cyber-physical system simulation. Ph.D. thesis, ISAE-SUPAERO, Université de Toulouse, France (2019)
13. Dryden, H.L., Abbott, I.H.: The design of low-turbulence wind tunnels. Technical report Report 940, National Advisory Committee For Aeronautics, Washington, D.C., USA (1949)
14. D'Ambrosia, J., Law, D., Nowell, M.: 40 gigabit ethernet and 100 gigabit ethernet technology overview (2010)
15. Fujimoto, R.M.: Time management in the high level architecture. Simulation **71**(6), 388–400 (1998)
16. Gervais, C., Chaudron, J.B., Siron, P., Leconte, R., Saussié, D.: Real-time distributed aircraft simulation through HLA. In: 16th IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications DS-RT, pp. 251–254. Dublin, Ireland (2012). https://doi.org/10.1109/DS-RT.2012.45
17. Hanke, C., Nordwall, D.R.: The simulation of a jumbo jet transport aircraft - volume i: mathematical model. Technical report, Prepared by Boeing Company Wishita Division, Kansas for. National Aeronautics and Space Administration, Ames Research Center, Moffet Flield California (1970)
18. Hanke, C., Nordwall, D.R.: The simulation of a jumbo jet transport aircraft - volume ii: modeling data. Technical report, Prepared by Boeing Company Wishita Division, Kansas for. National Aeronautics and Space Administration, Ames Research Center, Moffet Flield California (1970)
19. Holzman, J.K., Webb, L.D., Burcham, F.W.: Flight and static exhaust flow properties of an F110-GE-129 engine in an F-16XL airplane during acoustic tests (1996)
20. Marchand, M.A.: Pitch rate flight control for the F-16 aicraft to improve air-to-air combat. Technical report AFIT/GGC/EE/77-7, Air Force Institute of Technology (1977)
21. Metcalfe, R.M., Boggs, D.R.: Ethernet: distributed packet switching for local computer networks. Commun. ACM **19**, 395–404 (1976)
22. Modelica association project FMI: functional mock-up interface for model exchange and co-simulation. MODELISAR consortium (2020)
23. National oceanic and atmospheric administration, national aeronautics and space administration, US air force: U.S. standard atmosphere (1976). https://doi.org/10.1016/0032-0633(92)90203-Z

24. Nguyen, L.T., Ogburn, M.E., Gilbert, W.P., Kibler, K., Brown, P.W., Deal, P.: Simulator study of stall/post-stall characteristics of a fighter airplane with relaxed longitudinal static stability. Technical report 1538, NASA (1979)
25. Noulard, E., Rousselot, J.Y., Siron, P.: CERTI, an open source RTI, why and how. In: Spring Simulation Interoperability Workshop, pp. 1–11. San Diego, United-States (2009)
26. Object management group: minimum CORBA specification. OMG available specification, version 1.0, formal/02-08-01 (2002)
27. Quigley, M., et al.: ROS: an open-source robot operating system. In: ICRA Workshop on Open Source Software (2009)
28. Ryan, P.: Performance of dead reckoning algorithms across technology eras (2018)
29. Schutte, J.S.: Simultaneous multi-design point approach to gas turbine on-design cycle analysis for aircraft engines. Ph.D. thesis, Georgia Institute of Technology (2009)
30. Siron, P.: Design and implementation of a HLA RTI prototype at ONERA. In: Fall Simulation Interoperability Workshop (1998)
31. The Institute of Electrical and Electronics Engineers (IEEE) Computer Society: IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Federate Interface Specification. Simulation Interoperability Standards Committee (2010)
32. The Institute of Electrical and Electronics Engineers (IEEE) Computer Society: IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Framework and Rules. Simulation Interoperability Standards Committee (2010)
33. The Institute of Electrical and Electronics Engineers (IEEE) Computer Society: IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Object Model Template (OMT) Specification. Simulation Interoperability Standards Committee (2010)
34. Von Kármán, T.: Progress in the statistical theory of turbulence. Natl. Acad. Sci. J. **34**, 530–539 (1948)
35. Yeager, J.C.: Implementation and testing of turbulence models for the F18-HARV simulation. Technical report NASA/CR-1998-206937, NASA Langley Research Center, Hampton, Virginia, USA (1998)