

IESTA: evaluating air transport systems using a rich, modular, generic, managed platform

Judicaël Bedouet, Thomas Dubot, André Elie, Romain Kervarc

ONERA – DPRS/TCS
BP72 – 29 avenue de la Division Leclerc
92322 Châtillon cedex – France

judicael.bedouet, thomas.dubot, andre.elie, romain.kervarc@onera.fr

Phone: (+33) 1 467 / 34 532, 34 903, 34 950, 34 740

Fax: (+33) 1 467 / 34 149

Abstract

IESTA is a platform designed at the ONERA in order to achieve the simulation of air transport systems. IESTA relies on several technologies, especially the HLA (High-Level Architecture) standard. This paper focuses on the technical aspects of IESTA, and presents, on the one hand, the development approach used in this project in order to ensure a high quality and reliability and, on the other hand, the various components that are necessary for the platform to fulfil its purpose.

Keywords

Distributed simulation, high-level architecture, fast-time simulation, systems of systems, air transport system

Introduction

IESTA (Infrastructure for evaluating air transport systems) is a project aiming at providing the aeronautical community with advances in modelling and simulation support tools through a global evaluation platform.

Current systems created to measure the impact of an aircraft on its environment are integrated software, designed for only one goal. Modular platforms designed to support as many studies as imaginable are rare, and rely heavily on their makers' field of expertise. A general solution to extend the narrow possibilities of such integrated software is developing a fully modular platform, aimed at connecting specialised modules and ensuring the good communication between them. Our platform focuses on modularity, to allow customers to bring their own models to our system. A good solution to connect various modules and ensure effective, traceable communication between them is the HLA norm, which connects systems regardless of the computing platforms. This architecture is further enhanced by Genesis, an ONERA tool.

IESTA does not aim at developing a new Air Traffic Simulator, but at offering a true laboratory providing the users with a complete simulated environment. The IESTA platform user can connect its own modules that may complete or replace existing ONERA modules, to build a new simulation environment.

In this paper, we present the IESTA platform, focusing on its technical aspects. In a first part, we describe the various technologies that are being used in this project: the HLA norm, the various graphical tools necessary for the exploitation of the platform, and the low-level tools ensuring to the project high quality and traceable development. In a second part, we describe the platform itself, its general architecture and the notion of scenario, the underlying database structure, the various models that are to be integrated, and we sketch the possible uses of such a platform. In a third part, we focus on the software upon which the core of the platform relies: data processing software, administration software, and federate software integrated into the simulation.

1. Technical aspects

1.1. HLA/Genesis

In its conception, the IESTA project is planned to make extensive use of distributed simulation. Therefore, one of the main technologies on which it is based is the High Level Architecture (HLA) paradigm. Its purpose is to help formalising the various components of a distributed simulation and their interactions at a high abstraction level, without going into implementation details but focusing on the exchanged data and the coordination between components, in order to allow a huge interoperability and the possibility of capitalising models.

Due to these interoperability and capitalisation possibilities, HLA acquired a particular importance in the industrial context, which led it to become an international IEEE standard [9]. These communications happen among a set of federate modules [10], called a federation, regardless of the type and pieces of software of the computing platform. It consists of:

- an object oriented interface specification, describing how the platform will interact with the communication manager software: the Run-Time Infrastructure (RTI);
- an object model template (OMT) [11], defining how information is shared;
- a set of rules ensuring the compliance with the HLA standard.

The object model template is composed of documents describing the federation at various levels. The Federation Object Model (FOM) describes the interactions and attributes of the whole federation. It is composed of all the Simulation Object Models (SOM), which describe the same features for every single federate. Each SOM is unique and evolves together with one federate of the simulation, while the FOM is common to the whole federation and must be updated every time a new component, requiring new interactions, is added to the platform.

HLA is a very powerful technology but may be rather heavy, especially because of some redundancies in its development process. To simplify this part of the

work, a tool, GENESIS [2], has been developed by the ONERA to simplify most of these redundant parts, being helpful both to the developers and the specifiers. Its main purpose is helping the developer in all automatable phases of the writing of an HLA component, that is the consistency between the object model and the federate software and the handling of existing federates, object models, and federations, and this during the whole development process, from the description until the production of fully functional federates, without additional work.

To do this, the tool uses an ad-hoc language which allows specifying the entire object model, federates and federation and encompasses every HLA concept and some additional concepts ensuring general consistency. It operates as follows:

- based on the SOM of the federates, it builds a description of the simulated system, linking the HLA functionalities to the physical models;
- this description is then used as a base to accomplish various actions, such as a consistency check of the simulation description, documentation generation or automated source code and makefile productions.

GENESIS is useful for capitalizing work, enhancing productivity, guiding software development and simplifying the coding phase. Its use in the IESTA project allows maximising the profit drawn from HLA while minimising the tedious parts of the work and ensuring global consistency.

1.2. Graphic tools

The external libraries, which were not developed inside the ONERA and used to enhance graphically the user software and the management of the platform, are described here. The user has access to several tools to manipulate simulation data in IESTA. In order to improve their overall appearance, a set of external tools have been selected to offer state-of-the-art visual interface, and to accelerate and ease the software development in fields not directly connected to the core goal of IESTA: precise fast-time simulation. The needs to fulfil are described as follow.

First, a widely used, stable, and easy to implement graphical library, compatible with the IESTA main coding language, C++, and working on several exploitation systems, was needed. Qt [17] is a cross-platform application framework, also offering many integrated tools for Graphic User Interface (GUI) development, and benefiting heavily from its large user-base. One of these tools is the Qt Designer, which offers an easily accessible interface to prototype quickly and efficiently most types of interfaces. It was used for the mock-ups of most user-oriented pieces of software. The main library of Qt was then extensively used in the development of the client software, for every aspect of the user interface; while some of the administration software was designed using the Qt Designer tool.

Second, to allow a precise representation of all types of aeronautical data to be used, created and modified, through an intuitive interface, a library specifically dedicated to geospatial data handling and display was needed. LuciadMap [13] is a library working with the Java programming language. It offers a full high-end display system, specialised in the Air Transport Management field, and compatible with a very large panel of aeronautical standards. Commonly used in

the aerospace industrial context, it takes advantage of its long experience in the field of geospatial data representation. Supporting all the common types of projection for its world representation, and allowing easy change between them, it recognises most of the format used by IESTA (Navigraph for the procedure and navigation point data, GRIB for the weather representation ...). It also includes a small aircraft model, which is used to simulate accurately the trajectory from a procedure, in three dimensions. It is used mostly in the scenario elaboration software (where its world representation is synchronised with the editable data to offer a real-time view of the situation) and the ATC viewer (where it allows simulating an ATC station view, with all aircraft tracks and air traffic data, to translate into a realistic display the raw data transiting through the simulation).

Finally, a world-realistic 3D display engine, open-source and easily implementable was needed to ensure a 3D representation of the simulated world during the simulation. The Delta3D simulation engine [7], initially a U.S. Department of Defence tool, now a free open-source project, gives all possibilities of the higher-end 3D engine, with the support of a very strong community.

1.3. Building tools

The importance of the IESTA project made it necessary to adopt rigorous methods to ensure a high quality building process. This section describes the tools used to achieve this, and presents the improvements induced in the developing process.

Due to the number of developers taking part in IESTA, it is important to find an easy way to integrate all the pieces of software, while, although the IESTA platform will run under Linux, some developers are working under other operating systems. This is consistent with the HLA approach, in which each simulation component may be produced independently and then interoperate with others. So, a framework is needed to integrate these different ways of working.

CMake [14], an extensible, open-source tool to compile software independently from operating systems and compilers, was chosen to manage the IESTA build process. It proceeds so: the developers write simple text files, which CMake uses to generate native build project files (UNIX makefiles, IDE workspaces ...). In addition to specifying how files should be compiled, CMake supports platform inspection, easy configuration, user-customization builds and out-of-source builds. It may search for platform properties, programs or libraries that may be required for the software to be compiled. All this information is stored in a cache file and may easily be modified by the means of the CMake GUI.

For IESTA, each project administrator provides CMake files to build the module that he has in charge. All modules are integrated in a main CMake module that allows managing IESTA platform builds. Hence, a new developer may easily compile and deploy some of the IESTA modules by selecting them. Any other task is performed by the CMake tool: checkout precise modules and their dependencies, configure, compile and install them ...

To avoid compilation problems, the code must be regularly built and tested. Hence, IESTA developers are encouraged to develop unitary tests, but considering

the large amount of code, it seems difficult to continually check if tests are passed, and may be difficult to know which code change has introduced an error. Kitware provides a second powerful tool to answer to this problem: CDash [14].

CDash is an open-source, web-based software-testing server that collects the results of test processes submitted by clients. For IESTA, we automate the building process by compiling each night the last versions available from scratch, and the testing process by running each night unitary tests, code coverage and dynamic memory analysis. All the results of these processes are submitted to a CDash server. CDash is at the centre of the quality building process. Each night, a report is submitted to the server, which may be analysed each morning by developers. CDash may even be configured to send an email to developers who checked in the portions of source code that cause building or testing errors. This allows quicker correcting, and testing several platforms in a single night.

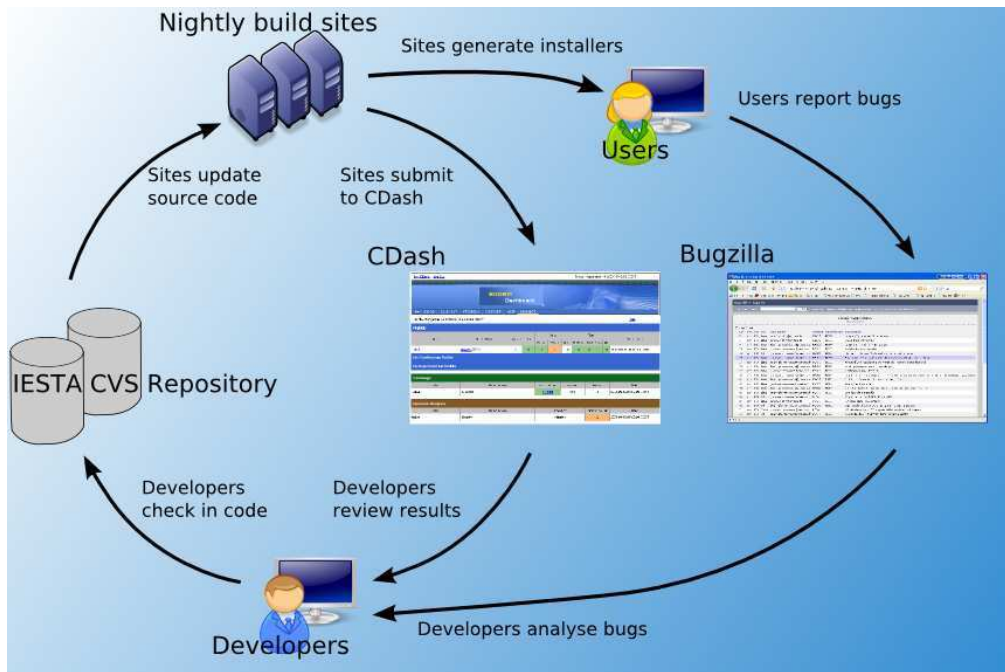


Figure 1: IESTA quality building process

Two other auxiliary tools are extensively used in the IESTA project: CPack [14] and Bugzilla [3]. CPack is used to create platform specific installers (NSIS Windows installer, RPM, Debian package, tarball). Bugzilla is a bug tracking system, and is also used to notify bugs on IESTA modules.

2. IESTA platform infrastructure

2.1. General architecture

The IESTA project aims at providing a complete simulated environment for evaluations of air transport systems. Based on an HLA architecture, the IESTA platform is fully modular and allows users to connect their own federates in order to build a new simulation environment by completing or replacing those of

ONERA. In addition to the HLA-based distributed simulation framework, it includes a set of utilities to prepare, manage, run, monitor and analyse experimentations, called process in the IESTA platform terminology.

IESTA data storage is ensured by four databases:

- BDGC (Configuration and Management DataBase) is a relational database containing the hardware and software description of the platform and all available processes. A process is composed of sequential tasks, each being a set of executables. There are three main tasks: pre-processing, simulation, where HLA federates play the main part, and post-processing.

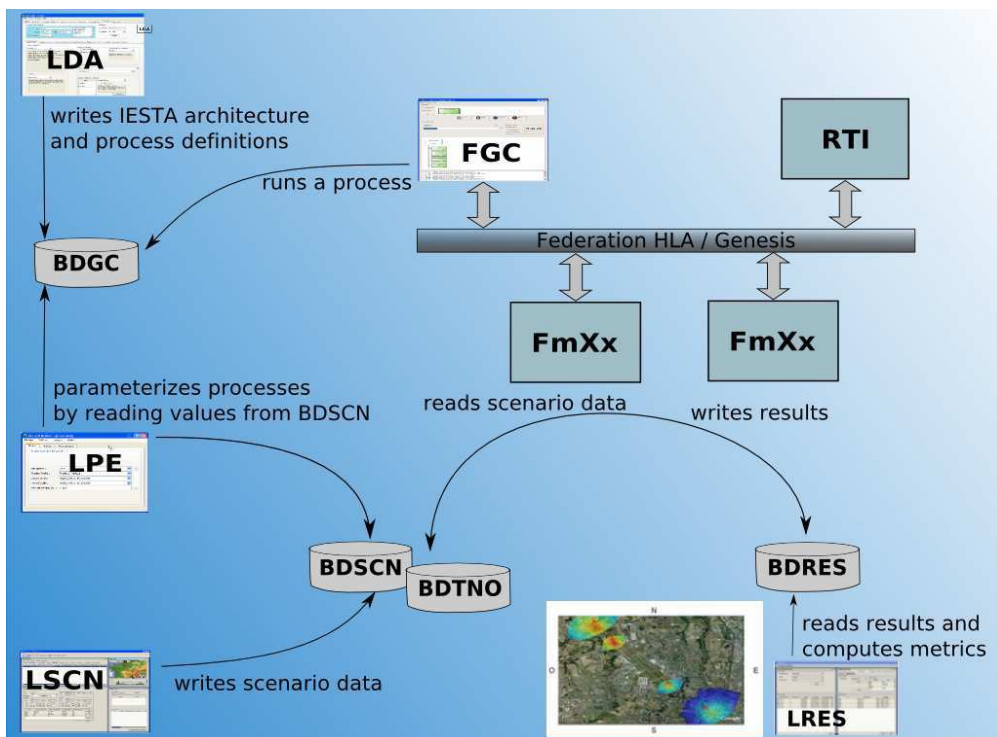


Figure 2: IESTA overview

- BDSCN (Scenario DataBase) is a relational database storing scenarios. A scenario is a set of aeronautical information (aircraft and engine models, waypoints, procedures, flight plans ...).
- BDTNO (Technology DataBase) is a set of data files provided by external organisations like Eurocontrol or generated by pre-processing models, such as the ground traffic planning or acoustic models.
- BDRES (Results DataBase) is a set of result files generated during simulation or post-processing.

Three tools aim at managing and preparing experimentations:

- LSCN (Scenario Definition Software) fills in the BDSCN database.
- LDA (Architecture Definition Software) allows IESTA administrators to describe the IESTA platform and create processes.
- LPE (Process Parameters Definition Software) allows IESTA users to give values to the parameters of the process they want to run.

The experimentation is run and monitored by the FGC (Federate of management and control). The FGC runs processes parameterised by the LPE: it launches pre-processing tasks, then manages the HLA federation, in collaboration with the RTI, and finally launches post-processing tasks. HLA federation is composed of federates called FmXx. FmXx federates encapsulate ONERA or customer models. They read aeronautical data from the BDSCN and BDTNO databases and write results of the experimentations in the BDRES database.

Finally, the LRES (Result Software) tools process results. They can, for instance, generate a map of the traffic noise on the ground or compute concentrations of CO₂ in the airport atmosphere.

2.2. Database

Database development is an important part of the IESTA project. Different tools have been set up to insure easy and reliable access to databases. PostgreSQL [16] has been chosen as the database management system of the IESTA platform. As a free software, PostgreSQL relies on a huge community of developers and is often rewarded as the best database tool. Thus, Postgres was selected to store the massive and confidential data of the IESTA platform. Database schemas have been conceived with DB-MAIN [6]. When developing data-centred applications, it is necessary to write SQL queries in order to insert or get data from database. For simple applications, SQL queries may be mixed to the code of the application, typically in the APIs used by IESTA tools to interact with databases. Unfortunately, SQL errors can only be detected during execution. Instead of losing time recompiling applications, we use stored procedures, which help separating the SQL implementation from the C++ implementation, and, because they are stored in the database, allow quicker validation of the SQL queries and improve performance at execution.

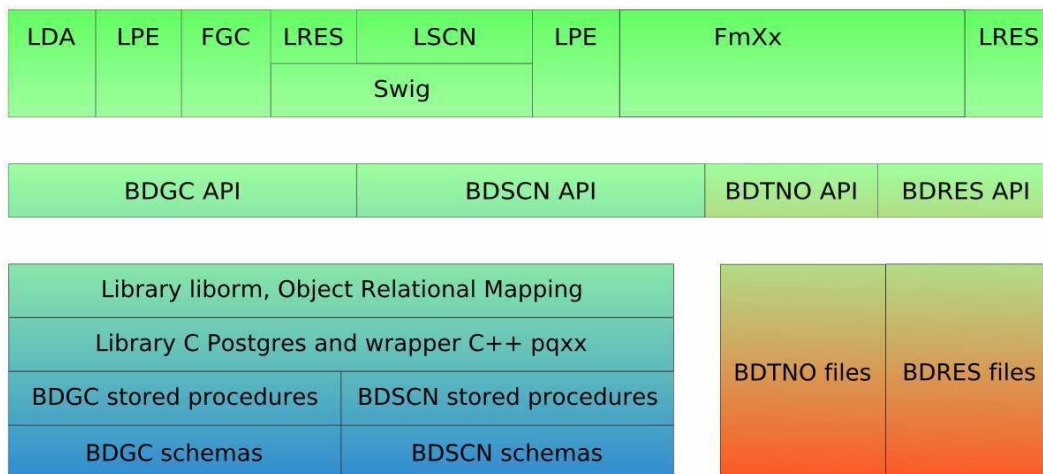


Figure 3: IESTA software infrastructure

IESTA tools and APIs used to access databases are developed in C++ and Java. Unfortunately, types in relational databases and types in object-oriented languages are mostly incompatible. Postgres APIs can only manipulate scalar values such as

integers and strings, or aggregate of scalar values. BDGC and BDSCN APIs developers must then convert objects into groups of simpler values for storage in the database. This programming technique is called Object Relational Mapping.

A free and efficient library proved difficult to find. It was decided to develop a library for Object Relational Mapping, and help APIs developers write as few code as possible by generating as much code as possible. This code generating process is based on a meta-language that defines classes needed by IESTA tools and SQL types needed by stored procedures.

Two tools, LSCN and LRES, are developed in Java. Using Swig, they are able to call C++ methods to access database. Swig is an open-source software development tool that simplifies the task of interfacing different languages to C and C++ programs.

2.3. Federates and encapsulation

This section describes three models that simulate the events of the simulation and the way they are connected to the platform federation. In order to properly simulate realistic aeronautical behaviours, a set of physical models have to be set up to reproduce each agent of the system's behaviour. These models have to simulate the noise generated by the engines of an aircraft and its propagation, the gases emitted and their propagation in the atmosphere, the behaviour of an aircraft while following a defined flight plan (attitude and position, engine power and fuel consumption), the evolution of weather above the airport and the behaviour of aircrafts on the ground. These models may either be designed by the IESTA team of developers, or brought to the platform by a user for specific purposes.

The models used for a first version of the IESTA platform are the followings:

- CEDRE [5], an ONERA Computational Fluid Dynamics numerical code able to simulate turbulent combustion;
- CESAR [4], an ONERA numerical code able to compute the aircraft noise installation effects;
- SIMOUN [15], an ONERA numerical code able to compute the propagation of acoustic rays;
- an aircraft ground traffic model to compute ground trajectories;
- an air traffic model to compute air trajectories, based on Eurocontrol BADA [8];
- a weather forecast model to take weather conditions into account;
- an aircraft engine model to compute engine data.

To successfully plug modules of various origins and design into one single federation, it is necessary to add software capsules between them and the RTI. These capsules are specific to each module and translate the HLA communication protocol into the format expected by the module. They are developed by the ONERA development team for each external module used in the simulation. In order to be as efficient as possible in the integration of foreign models, the encapsulation process must be very generic. This is ensured by the use of a design pattern separating as much as possible the HLA technical part of the code from

the simulation models code. As seen on Figure 4, the simulation code is linked only to the FederateX class, while the HLA code is mainly concentrated in the AbsFederateX class, thus making the connection between the two codes easier.

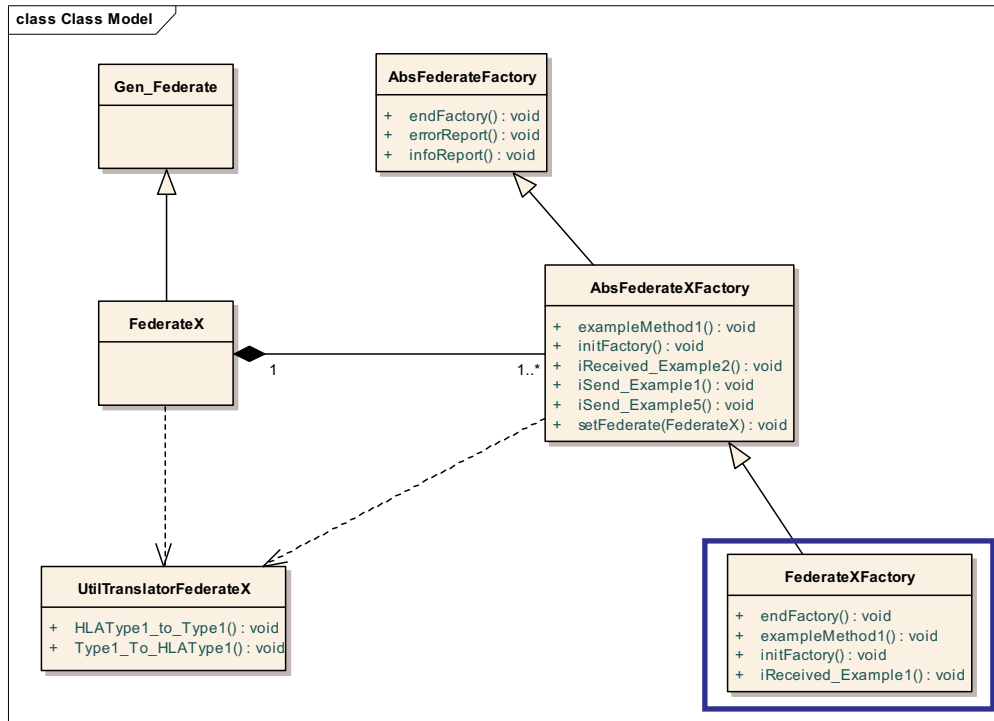


Figure 4: Model capsule design pattern

As some of the existing models are time-consuming at runtime, which is not compatible with the fast-time constraint, they will have to be divided into several parts and compute data which do not depend on the simulation, performed at a pre-processing time, or to compute results depending on a high number of data at a post-processing time. They are not encapsulated to be used in a federation.

2.4. Using the IESTA platform

Based on the HLA architecture, the IESTA project provides a fully modular platform, ideal to test various concepts. For example the fast-time simulation could be used to test the effects of new aeronautical procedures. Thanks to the IESTA result tools (cf. §3.1), the user can compare different procedures together accorded to defined criteria. The platform will thus be applied to test CDA (Continuous Descent Approach) concepts in cooperation with the French National Air Services. The platform could also be used to test new air transport systems such as the UAV (Unmanned Air Vehicles) systems. The platform could also be employed within the framework of the IFATS (Innovative Future Air Transport System) [6] project.

The first version of the IESTA platform will be used within the Clean Airport context. The main purpose is to test new concepts that could reduce acoustical and chemical pollution. The first scenario will thereby consist of the analysis of the

emissions around an airport. The IESTA platform will be composed of ONERA numerical codes such as CEDRE [7] for the chemical emissions, or SIMOUN [8] for the acoustical emissions.

After this first application, a second version of the IESTA platform will be implemented to test new concepts with larger-scale scenarios. The calculated emissions could be for instance “gate to gate” from the take-off to the landing of the aircraft and not only around a specified airport. This implies the integration of a larger traffic, the management of a larger map and new interconnections, including the possible connection with another traffic simulator.

3. IESTA software

3.1. Data managing tools

This section describes the software helping the user to create and organise the data needed for his experiment.

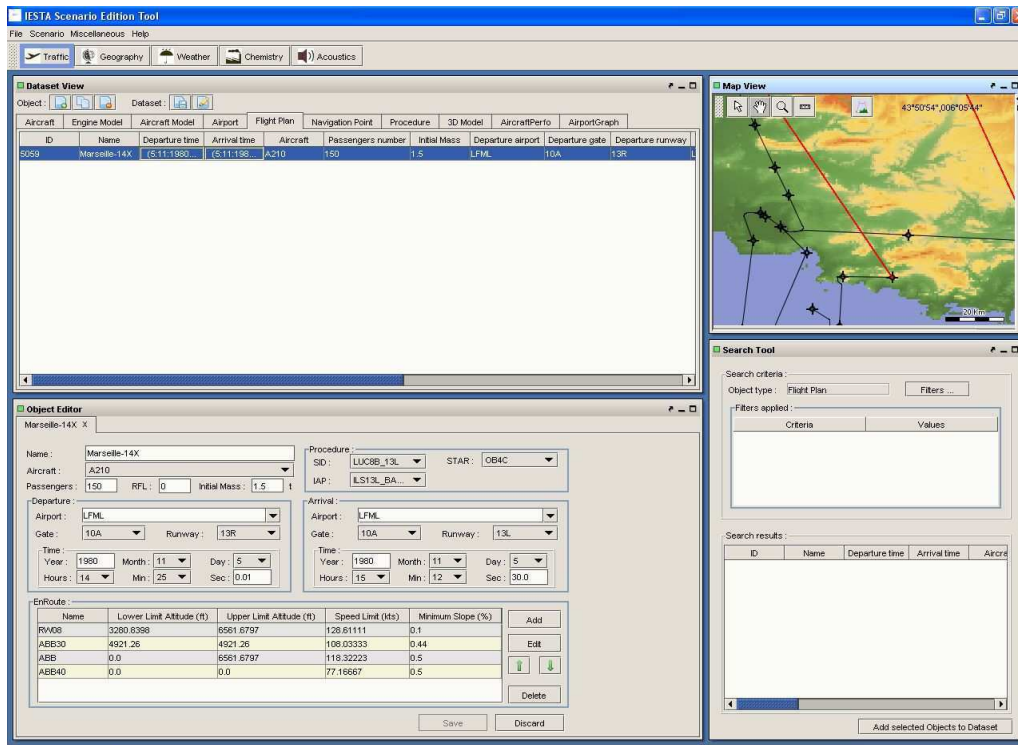


Figure 5: LSCN interface

To allow creating a functional and realistic experiment, the IESTA platform includes a tool, the LSCN, developed specifically to feed the scenario database with relevant and accurate data. The tool is developed in Java, with several external libraries, Luciad among them. It has been designed as generic as possible, making it quick and easy to enrich its supported data types. With the help of a graphic user interface, the user is able to create and link all the elements of an experimentation by domain (air traffic elements, geographic and weather data, chemistry and acoustics standards and sensors), each of which is composed of

several datasets of objects of different types. These objects can depend on each other, meaning that they cannot be fully defined unless objects of another type exist and are linked to them. These dependencies are visible on the world map view offered by the Luciad library. An import function also allows the user to use external data for its simulation. It supports the Navigraph standard for aeronautical data (including navigation points position, airports description, aircraft procedures and flight plans), and XML files which are used to add data from user-made files, mixing several data types.

With this tool, global data (such as worldwide navigation points or airports) can be directly imported in the database, to be later added to any scenario. A powerful research tool is implemented, which, using sets of filters specific to each datatypes and their metadata, gives access to the desired objects. The Luciad library allows calculating the trajectories of the aircraft along procedures. The dynamic update of the two-dimension world view after the data the user is working on allows to fine-tune every aspect of the simulation. With this tool, all the data relevant to the simulated world and actors can be easily and graphically entered by the user. They are then stored in the Scenario database, to be linked by the LPE to an experiment.

IESTA is a complex platform that gives back very rich data on completion of its simulation. One of the problems is to be able to sort these data and to gain access to the ones relevant to the simulation initial scope. In order to do this, a set of tool, the LRES, has been designed. The first tool lets the user select the type of results he needs among all the results stored. Its user interface allows selecting a precise experiment, accessing to all the results it contains and selecting the desired type of data. These data can then be extracted from the Results database. Using the second tool, the user can then load and process the data to convert them into the suitable format. A cartographic tool, such as GoogleMap, with three-dimensional data for the buildings and the landscape of the scenario scene, can then be used to display and animate the results in a three-dimensional photorealistic environment.

3.2. Administrative tools

The LDA tool deals with all the management information of the IESTA platform. It is directly linked to the database BDGC and is used for every management action such as hardware or software updates. The LDA is a data input interface with a large choice of information. Every actor of the platform is described: clients, databases, networks, hardware tools, software, licences, and applicative frameworks. The LDA can be used to manage all these management data, and to store the information in the BDGC database. The LDA is also used to create a new process while entering information such as the process parameters, the tasks and the programs linked to the process and the rights allocated to the different groups of IESTA users. This tool is thereby essential within the platform tools as the first element of the IESTA “assembly line”. It is a windowing WIMP¹-style interface. Its main complexity is the large amount of displayed information. Thus, it is composed of specific tabs for every type of data inputs.

¹ Windows, Icons, Menus and Pointing device

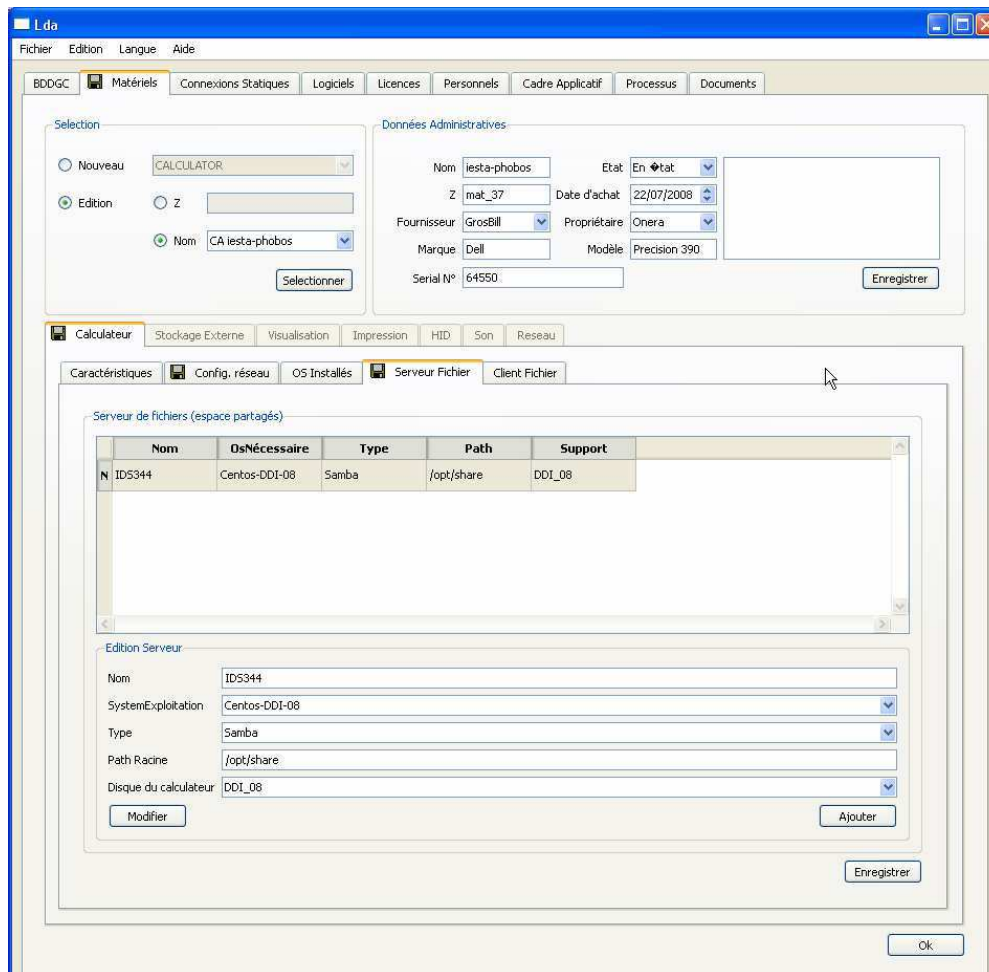


Figure 6 : Input of hardware data in the LDA

The LPE tool aims to manage the different process parameters. These parameters arise from the data in the BDGC and the inputs via the LDA tool. They are then used by the FGC tool in charge of the process run. These data are from different types: string, integer, Boolean, etc. An undefined type “Expression” is also used to allow the different tools (LPE, FGC) to launch a request after its definition and using a dedicated environment to evaluate some parameters. This tool is an interface used by every client of the platform to configure the data of the study he wants to run. As the LDA is an administration tool managed by the platform administrators, the LPE is the main interface between the client and the IESTA chain. Via the LPE the user can select a study in a context (applicative framework, family), display all the parameters of this study, modify them and store this study in the BDGC database. If the process needs to be compared to another process, the LPE displays two sets of parameters:

- the parameters of the process to be filled
- the parameters of the referenced process

The user can then modify the parameters in order to compare the new process with the parameters of a process case or a process already run.

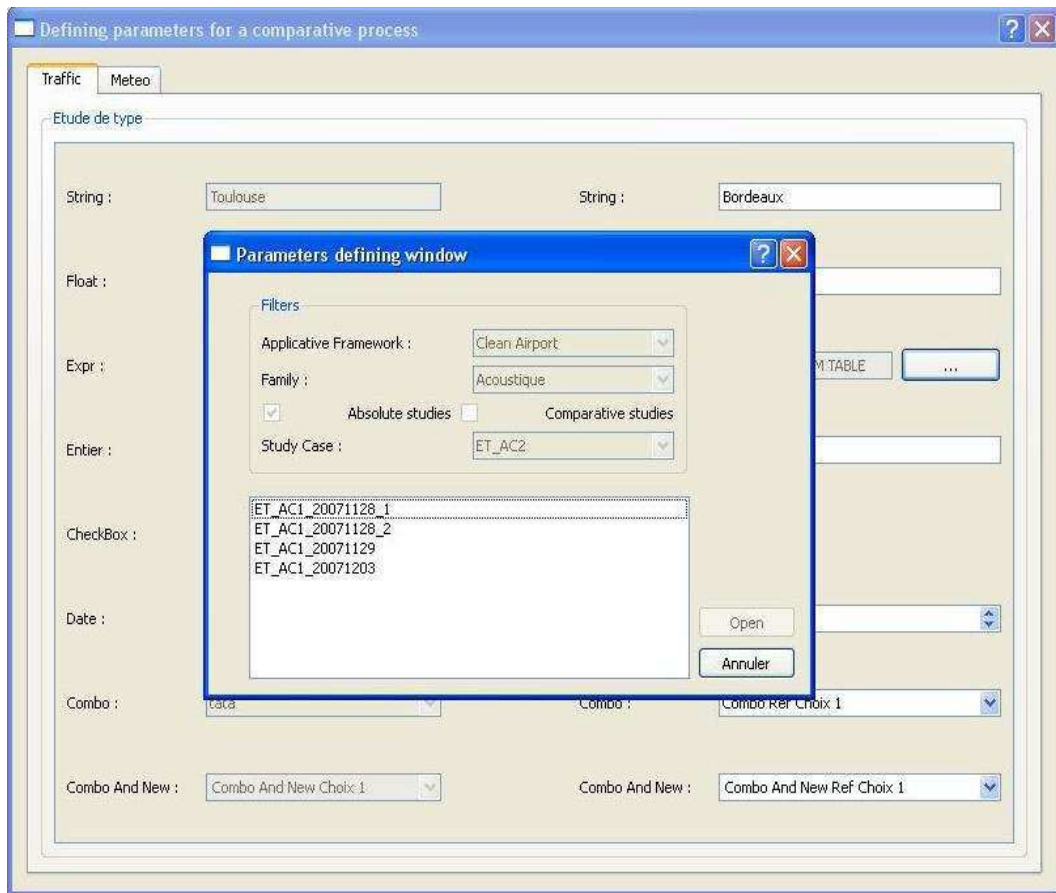


Figure 7 : Selection of a reference process for a comparative process

Finally the LPE can launch the FGC tool with the reference of the process previously saved. The LPE is thus in charge of checking the hardware and the software configuration of the platform for this dedicated process. Moreover, if the process is a comparative process, the LPE checks if a reference process has been selected by the client.

3.3. Federate tools

Federates tools are implemented according to the HLA standard, which allows them to be plugged into the simulation and communicate with the other federates directly.

The FGC is a federate tool dedicated to the management of the IESTA federation. It is directly connected to the BDGC database in order to get the process information, its attributes, and the attributes of the tasks and the programs it is linked to. It uses the process data entered in the LDA tool, and the process parameters modified via the LPE tool. It may be used in two modes: a simple “batch” mode or via a graphical interface. It may be launched either as a standalone or via the LPE tool.

For every process, the user can select an automatic mode or a manual, step-by-step mode. The interface displays the different tasks, highlights the current task with a progression toolbar to visualise the progression of the execution, and allows the user to manage the tasks (deletion, organisation), control the programs and check the hardware configuration of the federates.

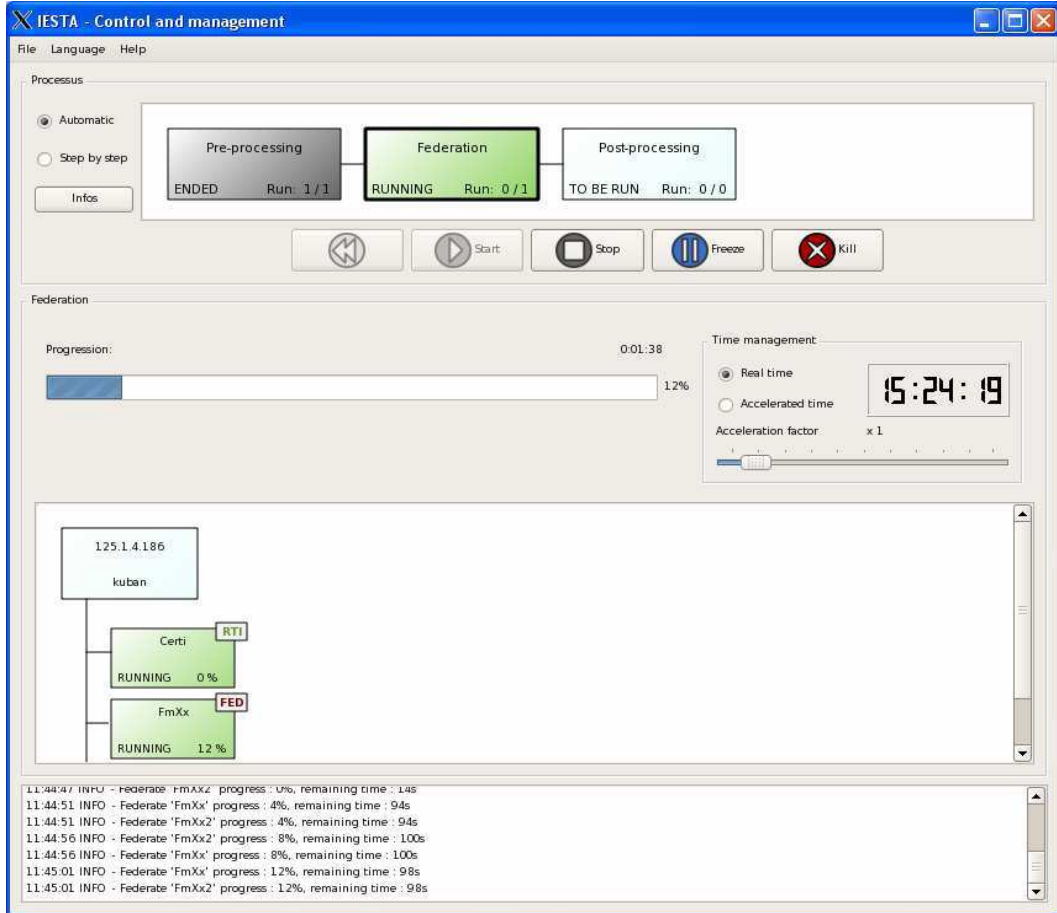


Figure 8: FGC interface

The Stealth Viewer is a federate tool used to visualise aircraft trajectories and the three-dimensional airport “scene”. This interface manages HLA objects of type “Aircraft” and it provides a three-dimensional representation of these aircrafts, with all the available parameters, such as callsign, position, speed, meteorology, etc.

A replay function allows playing a recorded sequence. Two modes are available for the visualization: inside or outside the cockpit. With the outside view, the current aircraft is displayed in three-dimensional and the user can observe its evolution.

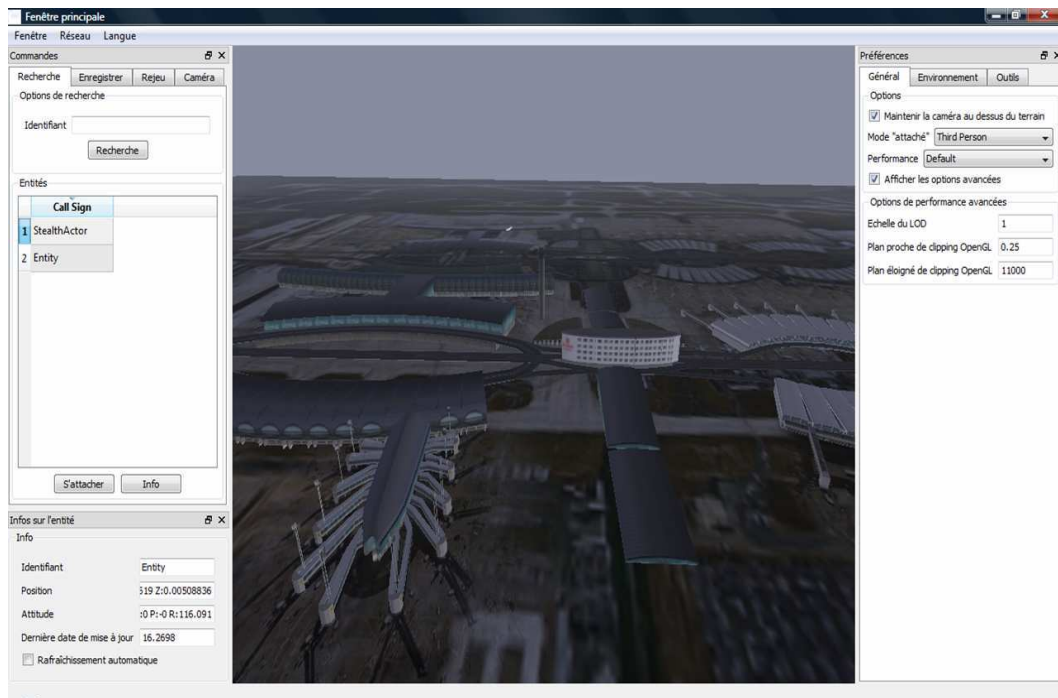


Figure 9: Stealth Viewer interface

Conclusion and perspectives

IESTA simulates in fast-time a full environment on a network of computers, using the HLA and Genesis technologies to ensure a safe and efficient communication between highly different programs, on various operating systems. The simulation is managed in real time using a user-friendly software (FGC). The management of its massive data is done through four databases, each specialised in a type of data, and using optimised APIs compatible with major programming languages. The management of all the aspects of the platform is done through a limited number of graphic interfaces: the LDA manages the software and hardware configuration, the LPE manages the experimentations description process. For the simulation data, the LSCN manages the entry data for the simulated environment. Each element of aircraft traffic on and around the airport is to be simulated through a different dedicated model. Once the setup is done, the modules are plugged into the system. The whole platform can then be controlled as a single system. All communication of every module is recorded, to allow later replays of the scenario. The study results are stored in a database by the LRES deals to be parsed and organised by dedicated tools, and formatted to be easily interpreted.

By ensuring a mean of having completely different pieces of software communicate, through a robust architecture, and with the help of simple user-oriented administrative tools, the IESTA platform stands high on the presumably huge market of generalist, large-scale simulation. Its first use will specifically attend to the problem of aeronautical acoustic and chemical pollution in an airport environment, and should give large quantities of realistically simulated data to the

research in this growing field. On the long term, the platform is to be used for various other means, such as new air traffic management concepts simulation, or military tactical and strategic planning.

Acknowledgements

The authors would like to thank the whole IESTA team (AMO and IMI) for their various contributions and insights on the project, and would like to express their special gratitude to Jean Bourrely and Nicolas Huynh for their useful comments on the first versions of this paper.

References

- [1] D.M. Beazley & P.S. Lomdahl, Lightweight Computational Steering of Very Large Scale Molecular Dynamics Simulations, in *Proc. of Supercomputing'96*.
- [2] J. Bourrely, P. Carle, M. Barat, F. Lévy. GENESIS: an integrated platform for designing and developing HLA applications (2005).
- [3] Bugzilla: <http://www.bugzilla.org/>
- [4] J. Bulté, *Simplified Acoustic Modelling of Aircraft Noise during Take-off and Landing*, ONERA Technical Report RTI2/F00201DPRS, 2003.
- [5] P. Chevalier et al., CEDRE: development and validation of a multiphysic computational software, in *Proc. of the 1st European conf. for aerospace sciences*, 2005.
- [6] DB-MAIN: <http://www.db-main.be/>
- [7] Delta3D: <http://www.delta3d.org/>
- [8] EUROCONTROL, *User Manual for the Base of Aircraft Data (BADA)*, Revision 3.6, 2004.
- [9] IEEE 1516-2000: *Standard for Modelling and Simulation High Level Architecture – Framework and Rules*.
- [10] IEEE 1516.1-2000: *Standard for Modelling and Simulation High Level Architecture – Federate Interface Specification*.
- [11] IEEE 1516.2-2000: *Standard for Modelling and Simulation High Level Architecture – Object Model Template (OMT) Specification*.
- [12] C. Le Tallec & A. Joulia, *IFATS, an Innovative Future Air Transport System Concept*, 2005.
- [13] LUCIAD: <http://www.luciad.com/>
- [14] K. Martin and B. Hoffman, *Mastering CMake: A Cross-Platform Build System*, Kitware Inc., 2003
- [15] G. Ménéxiadis, J. Varnier, Long-range propagation of the sonic boom from Concorde airliner: analyses and simulations. Accepted in May 2008 to be published in *AIAA Journal*.
- [16] Postgres : <http://www.postgresql.org/>
- [17] Qt: <http://trolltech.com/products/qt/>