P. Carle, R. Kervarc, R. Cuisinier, N. Huynh, J. Bedouët, T. Rivière, É. Noulard
(Onera)

E-mail: patrice.carle@onera.fr

Simulation of Systems of Systems

Complex systems have become a particular area of interest in the past years and the study of such systems, seen as a whole, has yielded many and varied approaches. One of the difficulties frequently encountered is that such systems display emergence, i.e. their global effect or behavior is greater than the sum of the behaviors of their agents and depends strongly on the interactions between these agents. Simulation is a very interesting approach for this study, since it allows focusing on the dynamic study of the system, thus pointing out these behaviors and interactions. Moreover, it is particularly interesting for aerospace systems, which are nowadays clearly studied as complex systems, since these systems are generally poor candidates for real experimentation, due to many diverse reasons (e.g. cost for space systems, criticality in terms of human life for air transport systems, etc.). Hence, they must be studied quite thoroughly before their full design or realization is possible and simulation plays an important part in this. This paper presents methods that can be used to study complex systems by simulation, as well as some techniques that take into account their specific nature, in particular distributed simulation, a paradigm quite intensively used at Onera. It also deals with the difficulties that may be encountered in setting such large simulations up and also addresses the problem of data in such a large simulation, from the handling to the exploitation.

Introduction

"System" is a very generic term, which is used in all fields of science and may describe practically anything studied: roughly speaking, a system is a collection of elements in interaction, this collection being considered as an object. Systems may be very complicated, but, regardless of that, an orthogonal notion of *complex system* has emerged over the past years. While there is no generally accepted definition of what a complex system is, there are some essential properties that a system needs in order to be considered complex, upon which most authors agree. Mostly, the three main required properties are the following:

- A complex system consists of a significant number of interacting entities (often called *agents*), which can each be considered independently (and may potentially have diverging purposes).
- A complex system exhibits *emergence*, i.e. global behaviors that are greater than the sum of the behaviors of each agent, or at least that are difficult to predict from the knowledge of the individual behaviors of the agents.
- A complex system is not centralized: the emergent behaviors do not result directly from the coordination of a central controller (which does not necessarily exclude the presence of central supervising systems).

Such systems may be found in many various branches of science: biology (e.g. eusocial insect colonies), medicine (e.g. epidemic propagation), engineering (e.g. system design in various fields such as aerospace), chemistry (e.g. molecular self-assembly), sociology (e.g. human networks), computer science (e.g. grid computation), economy (e.g. stock market), etc. Since they encompass very different aspects, they may be studied by various different approaches, which can be roughly divided into two families: static approaches, focusing rather on the properties of the system and dynamic approaches, focusing rather on its behavior.

These dynamic approaches are especially interesting because of the very nature of complex systems, in which the interactions between components make new behaviors emerge. In this regard, simulation is an interesting tool, since it allows data reflecting emerging behaviors to be produced. This data may be used for various purposes, from learning and prediction to the analysis by various methods of the produced data for assessment purposes.

Simulation is an interesting axis for the study of complex systems [7]. It allows both the stochastic and temporal aspects of complex systems to be addressed by developing simulation models, i.e. algorithms or computation codes representing the behavior of a part of the system in response to a defined scenario and putting them together.

Various types of simulation exist and may be used to put a system "in situation" and assess its behavior. However, this simulation approach still sets issues. First, simulations generate large amounts of data that need be exploited, which can be done by various means. Second, since complex systems result from the interactions of various agents, it would seem suitable for the models developed for the simulation of an agent to be able to be reused: this capitalization concept is quite important when studying families of systems (e.g. defense systems). So is the linked question of simulation interoperability: a complex system simulation may be built using the knowledge of various partners and interoperability is the key for such cooperative constructions. Hence, the architecture of the simulation is a very important point.

Simulation may be monolithic or *distributed*. Monolithic simulation, where a single model is developed for the entire system, is easily set up and allows studies on the parameters of the system, or easy running of Monte-Carlo computations over a family of scenarios with slight variations. Distributed simulation is a quite different paradigm, where each individual component of a system is simulated on its own, interactions between these components being modeled as message exchanges between the individual simulators, these exchanges being standardized, or at least complying with a set of common rules. This type of simulation is also sometimes called a functional simulation, because it focuses on the functions[1] of the system. An advantage of distributed simulation is that it allows heterogeneous systems to be taken into account naturally, and facilitates the capitalization of models and the parallel work within a multi-disciplinary team. Moreover, simulation may be set up with the assistance of various techniques, such as the use of a simulation framework or a higher-level description language related to code generation. In addition, various possible points of view may be considered for the use of simulations, depending on the type of study considered: constructive simulation (purely numerical), hybrid simulation (involving hardware in the loop), interactive simulation (with human agents in the loop, mostly for human factor studies or for training simulators).

This paper presents some ideas on how simulation may be used for the study of complex systems. The first section details the various benefits of this approach. In a second section, various tools supporting this simulation approach are presented. In a third section, the issue of simulation exploitation is dealt with. Finally, a fourth section gives some hints regarding possible applications.

Benefits of simulation for the study of complex systems

Interaction and dynamic aspects of a complex system

As has already been pointed out, interaction and dynamic aspects are paramount in the study of complex systems, because of behaviors emerging from these interactions. Hence, dynamic approaches, such as temporal simulation, allow these to be evidenced.

In the aerospace industry, this approach by simulation is particularly needed, one of the main reasons for this being that the considered systems (air transport system, space missions, etc.) generally do not allow real experimentation before a very advanced stage of development. Indeed, this is a consequence of their being, most often, either very critical or tremendously expensive, when not both. Functional simulation is particularly suited in this case: the agents of the system are identified, each of them is separately simulated with the desired (or possible) degree of granularity and the simulation then focuses on the interactions between those agents.

Capitalization and knowledge protection

Even though capitalization aspects appeared very early in the distributed simulation paradigm (concept of "reuse of federates" - federates being a common name for the components of a distributed simulation, which is often considered as a "federation of simulators"), this idea of an "available model catalogue" is in fact just part of the knowledge and know-how that is capitalized upon in using distributed simulation techniques. Indeed, the interaction between models developed by different multidisciplinary teams and their participation in a large scale simulation require an efficient collaboration between the teams designing the various capitalized models.

However, there are in general two possible difficulties in the process leading to capitalization and model reuse, which may be either technical or related to knowledge protection.

Technical problems, typically linked to the fact that models developed independently must be put together, may be solved by offering assistance for model integration, either by using simplifying tools (SIMCORE, Genesis, GAMME - cf. the following sections) or by establishing a close cooperation between integration engineers and modeling engineers (IESTA[2] - cf. the following sections).

Knowledge protection issues can be dealt with, since distributed simulation standards (e.g. HLA, an IEEE standard that will be introduced in a later section) already offer a first level of knowledge protection, since capitalization is done for interoperable "black boxes" and the model remains totally under the responsibility of its initial designer, even though the "black box" must be provided with a clear definition of its possible use, since the link must be established between the design and the use of the models.

This granted, feedback from various projects such as IESTA shows that, as long as providing a model does not mean losing control over its development, capitalization offers a very interesting framework for the participating teams, since it offers an interesting situation of validation and verification of their model, which would not easily be found otherwise.

Therefore, distributed simulation is a good paradigm, even offering a way to concurrent societies to carry out common simulation, by only sharing the necessary interface, protection being ensured by the HLA mechanisms. Moreover, the increasing complexity of systems of systems makes simulation more and more useful and distributed simulation seems unavoidable to ensure a more operational cooperation between teams, e.g. for defense.

Multidisciplinary and model-driven approach

Modeling and Simulation (M&S) technology [26] [27] is an innovative approach for the design and conception of complex systems, which

[1] in the sense of functional analysis

[2] http://www.onera.fr/iesta-en/index.php

appeared in 2000. This framework is aimed at verifying and validating any significant project in simulation as much as possible, before any significant realization (either in terms of cost or criticality). In this context, many simulations are carried out using distributed simulation as a support to construct a multidisciplinary complex system, where the consistency between disciplines is ensured by the integration of the various models into a temporal simulation.

A major interest of the distributed paradigm in this case is that each specialist may have at hand the data needed to describe the evolution of their model and that, since the system is closed, its evolution at time $t-1$ has been taken into account in the acquired data. This approach being valid for all disciplines involved, it is referred to as cosimulation, as described by I. Nakhimovski [24].

Onera has been using these M&S techniques to assess and compare aircraft approach procedures in the first application setting of the IESTA project. This setting was meant to assess the ecological impact of such procedures in terms of noise and chemical pollution around an airport. The multidisciplinary approach involved is illustrated on figure 1.

Important feedback in this regard has been provided by the validation of the IESTA acoustic chain, with measurements from the Aviator campaign. Indeed, the results obtained by the simulation of the conditions of the campaign proved the representativeness of the acoustic model to be good. This validation also needed the other models involved in the simulation, which permitted the flight to be replayed, simulating the reactors.

Simulation frameworks for complex systems

From the problematic stated above, a crucial point emerges for the simulation of systems of systems: time management. In a simulation of a system of systems, it is indeed necessary to maintain (at the system level) a global state divided into subsystems in interactions, all of this depending on time with possibly very different paces. Therefore, it is necessary to have a lean time management, allowing each component to respect its own rhythm, while at the same time keeping a global state as close as possible to the real state.

Distributed simulation does not only bring the benefits of its capacity to make models from various fields of physics (flight mechanics, engine, acoustics, chemical dispersion, etc...- cf. figure 1) interoperable. It also allows the time management issue to be addressed: HLA offers a lean time management relying on various mechanisms allowing a harmonious coexistence of various paces: each system publishes information with a validity period, i.e. a certain time span within which they will not be altered. This way, the system does not need to be studied at the smallest pace of all its components, which would be too demanding.

Moreover, another advantage of distributed simulation is its adaptability with respect to models: it allows these to be integrated by encapsulation, which means that the simulation is fitted to the models, rather than the models fitted to the simulation. This kind of simulation, dealing with models close to those developed by researchers, allows a much easier validation, even though the integration cost is of course higher, since ad hoc encapsulation techniques must be applied for each of the models.

Unfortunately, this approach by distributed simulation is not well disseminated. The main reasons for this are probably the complexity of the standard, which is not easily set up, the important development efforts, and the strong link to support means (COTS) that is required for the exploitation of the simulation [25].

Onera has been investing in the HLA distributed simulation standard for many years and today has a broad range of proficiency and tools for setting up such simulations easily (cf. the following sections).

Simulation exploitation

Considering the very large scale of industrial simulations (which in an aerospace context typically may include a whole day of traffic around an airport, the study of a satellite constellation over its lifetime of 15-20 years, etc.), the amount of data produced by any of these is tremendous and may not be treated by a human operator; hence the need for assistance analysis tools. This amount of data is actually

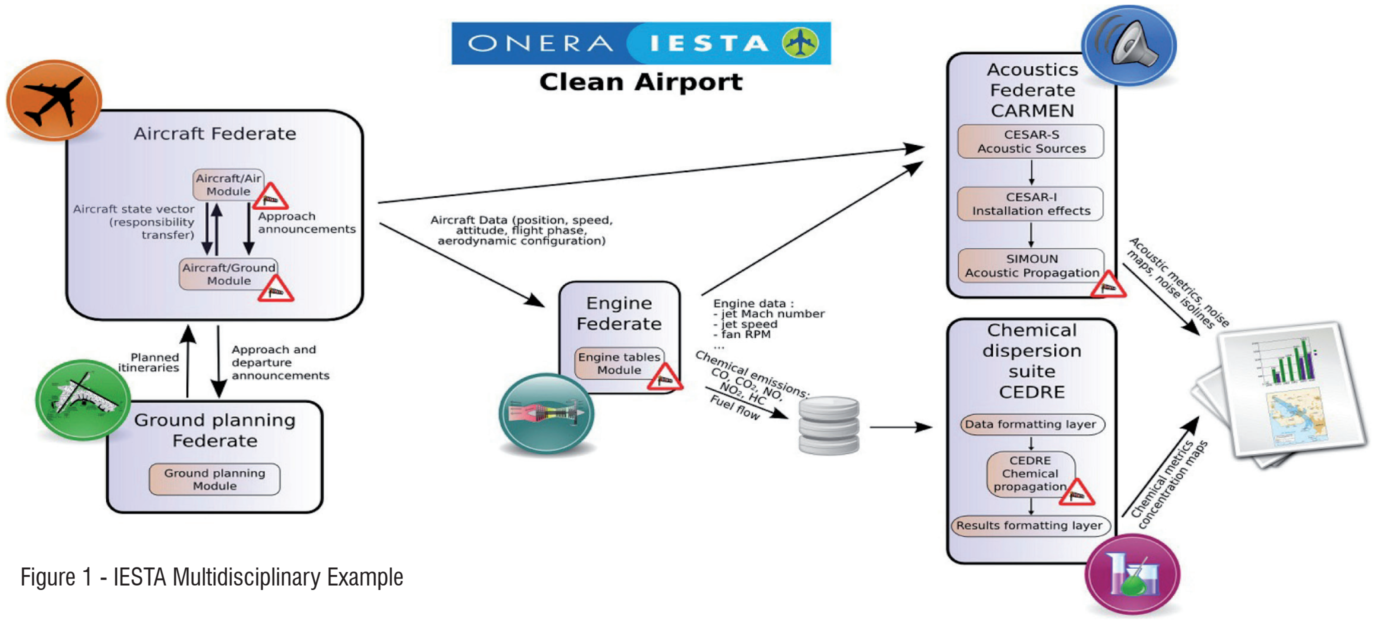
ONERA IESTA
Clean Airport
Aircraft Federate
Aircraft/Air Module
Aircraft state vector (responsibility transfer)
Approach announcements
Aircraft/Ground Module
Planned itineraries
Approach and departure announcements
Ground planning Federate
Ground planning Module
Aircraft Data (position, speed, attitude, flight phase, aerodynamic configuration)
Engine Federate
Engine tables Module
Engine data :
- jet Mach number
- jet speed
- fan RPM
...
Chemical emissions: CO, CO2, NO, NO2, HC
Fuel flow
Acoustics Federate CARMEN
CESAR-S Acoustic Sources
CESAR-I Installation effects
SIMOUN Acoustic Propagation
Acoustic metrics, noise maps, noise isolines
Chemical dispersion suite CEDRE
Data formatting layer
CEDRE Chemical propagation
Results formatting layer
Chemical metrics concentration maps

Figure 1 - IESTA Multidisciplinary Example

sometimes so huge that it may not even be saved for future treatment and must be analyzed online, which is impossible without some automated assistance.

This section presents one example of techniques that may be used for this purpose: chronicles, a formalism focusing on the description and recognition of behaviors. Chronicles rely upon the idea that the events generated by the simulation contain enough information on the behaviors occurring within the simulation to be directly exploited by identifying behavior signatures within their time-ordered succession. These signatures are expressed by logical expressions called chronicles.

Chronicles

For this purpose, a temporal approach is naturally particularly suited and temporal logic is a very powerful tool for the analysis of the behavior of large systems of systems. It may be used in order to make macroscopic assessments on the global behavior of a system, as in [6] [21] [23], but this last approach is a global one, not particularly suited for distributed simulation.

A very interesting approach focuses on following a simulation for the instantaneous, certain detection of behaviors throughout time, rather than on the global assessment of the likeliness of a behavior, providing an expression power and the possibility of detecting behaviors through the observation of their characteristic traces. This technique is not only useful for the exploitation of simulation, but also potentially for their development (cf. e.g. [10])The characteristic traces are expressed through correlations of interactions (called events) in the system and these correlations are represented using a formalism called chronicles for their fine detection. This formalism was introduced by Dousson et al [14] and developed by Carle and Ornato [8] and Dousson et al [13] [15] [16].

In this approach, events cannot occur simultaneously and durations are not associated with events, thus considered as instantaneous, but time may indeed be taken into account using special events corresponding to clock ticks. A chronicle describes relationships between the events of a sequence ordered with respect to time. The goal is to identify all instances of the chronicle schema within an observed event flow (where events are ordered).

Chronicle identification is achieved through the matching between events of the flow and events in the chronicle description, while flow events that do not contribute to the chronicle recognition are simply ignored. In addition, it may be of interest to save the piece of information stating which events in the flow contributed to the chronicle recognition, because it may help to find the causes of the observed events. A first modeling of chronicle recognition and its application to the analysis of distributed simulation was proposed in [1] [2] [3] [4] and a reworked version of the previous work in [9] [22].

Simulation Tools

The previous section gave some flavor of the kind of techniques that can be used for distributed simulation; here, some actual techniques will be presented, focusing around the High Level Architecture (HLA), which has been the NATO recommended standard for distributed simulation since as early as 1998[3] . HLA has been used also in other non-military contexts, for instance, to design civilian air traffic simulations. This section first describes the HLA standard. Then, it presents three Onera tools that were developed to assist distributed simulation designers and make the development of such simulations go more easily and smoothly: the first one, CERTI, is an HLA implementation; the second one Genesis, relies on a description language allowing to automate and make consistent as much as possible from the development process of a simulation; the third one, SimCore, consists in a framework providing a generic model integration process. Finally, another approach for the development of distributed simulation is presented, the originality of which is that it is data-centered and may be used to generate models complying with various standards or tools, including HLA.

HLA

A very popular standard for distributed simulation is the IEEE High Level Architecture [17] (abbreviated HLA). Its principles were originally developed by the United States department of defense [12], but quickly spread to industry and finally led to an IEEE standard, in which the simulation consists in various components - called federates - which are linked in a federation, in which communications and interactions take place [18], regardless of the characteristics of the computing platform.

A High Level Architecture model consists of:

- an object oriented interface specification, describing how the platform will interact with the communication manager soft ware: the Run-Time Infrastructure;
- an object model template (OMT) [19], defining how information is shared between the various agents composing the federation;
- a set of rules ensuring compliance with the HLA standard.

Box 1 presents an example of a run-time infrastructure: Onera's CERTI.

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

[3] "The High Level Architecture (HLA) is the preferred Simulation Interoperability Standard recognized by NATO as early as 1998", quoted from NATO Modelling and Simulation Group (NMSG): http://www.rto.nato.int/panel.asp?panel=5

Box 1 - CERTI

CERTI is an HLA RTI developed since 1996 by Onera, the French Aerospace Lab.

The initial purpose of CERTI was to develop a homemade RTI in order to: learn HLA usage and HLA RTI internals (e.g. time management) and have total control over source code in order to use this particular RTI with specific modifications in several research projects (security mechanism, multi-resolution, high performance distributed simulation, etc.).

CERTI became open source[5] in 2002. Since then, the Open Source CERTI project has been mostly driven by research project needs and funds. CERTI is drawing increasing interest from the HLA

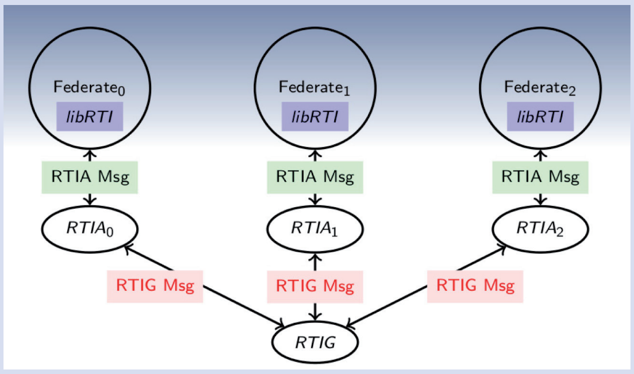

Figure B1-01 - CERTI messaging architecture

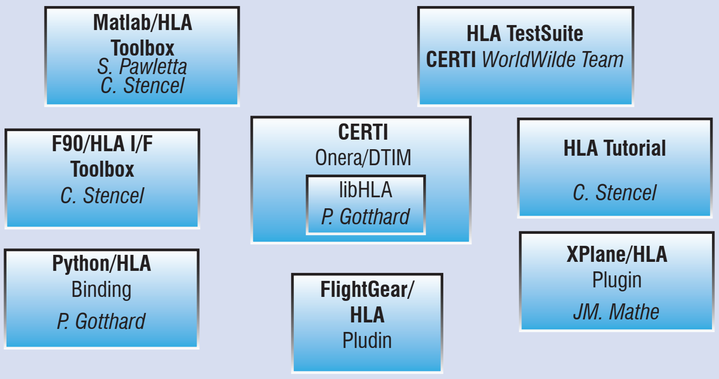

Figure B1-02 - Components of the CERTI

CERTI has a classical communicating process architecture depicted in figure B1-01, making it very portable on various operating systems. CERTI currently works on various Unix platforms, including Linux and various Windows OS flavors. CERTI is natively written in C++ but offers binding in Java and Python.

The CERTI messaging infrastructure makes it generic, a message-oriented middleware. Every HLA service is implemented using a predetermined set of message exchanges between Federates, its RTIA, and RTIG.
The RTI Gateway (RTIG) is a centralization point in the architecture. Its function has been to simplify the implementation of some services. It manages the creation and destruction of federation executions and the publication/subscription of data. It plays a key role in message broadcasting, which has been implemented by an emulated multicast approach. When a message is received from a given RTIA, the RTIG delivers it to the interested RTIAs, avoiding a true broadcasting [see: CERTI messaging architecture]

A specific role of the RTIA is to immediately fulfill some federate requests, while other requests require messages with the RTIG. The RTIA manages memory allocation for the message FIFOs and always listens to both the federate and the RTIG. It is never blocked, because the required computation time is reduced. It also plays a great role in the implementation of the Time Management HLA Service.

HLA is a DoD defined simulation standard, which is now the IEEE-1516 standard. It is publish/subscribe oriented middleware that could be compared with OMG DDS. However, HLA has a unique feature, which is Time Management service. The Time Management service makes it possible for each simulation stakeholder (called a Federate in HLA wording) to ensure that all of the data or events that it receives or sends are causally ordered. CERTI makes no exception to this and implements the HLA time management service using the well-known Chandy-Misra-Bryant NULL message conservative algorithm. CERTI even has a unique feature, which is a modification of this protocol to greatly enhance the protocol performance in some event-driven situation: this is the NULL message PRIME conservative algorithm.

More than a simple HLA compliant RTI implementation, CERTI is an ever growing Open Source community that contributes original software within or around CERTI as depicted in figure B1-02. CERTI is currently evolving towards the HLA 1516 standard [17].

[5] components of the CERTI

Genesis

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 [5], has been developed by Onera to simplify most of these redundant parts and is helpful both to the developers and the specifiers. Genesis consists in a language, in which a high-level description of an HLA simulation component or a whole HLA federated simulation may be specified, and an engine, processing such specifications in order to generate the source code for the components, along with the documentation required by the standard and the configuration files, ensuring that they will be able to interact properly with each other in the federation. Figure 2 displays the principles of Genesis.

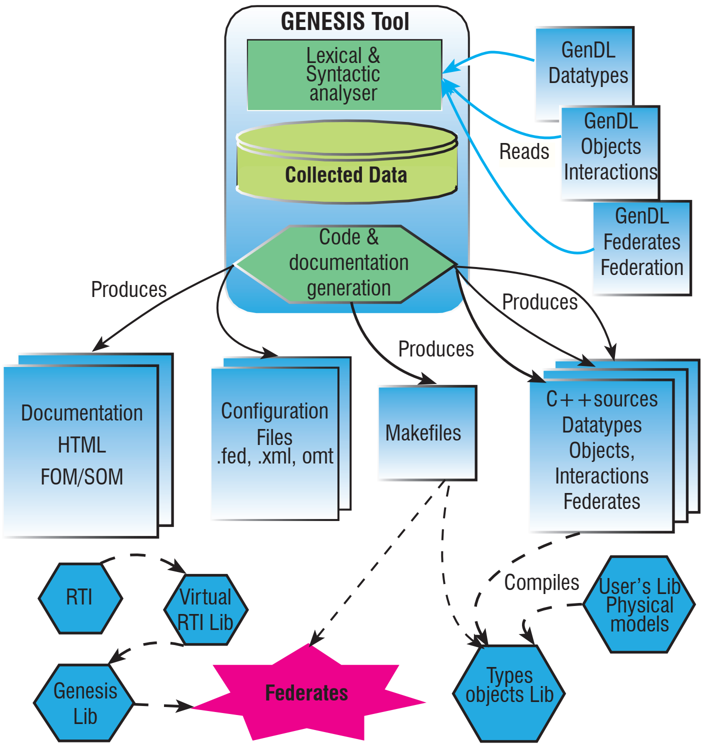
GENESIS Tool
Lexical & Syntactic analyser
Collected Data
Code & documentation generation
GenDL Datatypes
GenDL Objects Interactions
GenDL Federates Federation
Reads
Produces
Produces
Produces
Documentation HTML FOM/SOM
Configuration Files .fed, .xml, omt
Makefiles
C++sources Datatypes Objects, Interactions Federates
RTI
Virtual RTI Lib
Genesis Lib
Federates
Compiles
User's Lib Physical models
Types objects Lib

Figure 2 - Principles of Genesis

Its main purpose is helping the developer in all automatable phases of the writing of an HLA component, i.e. 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. Besides, Genesis is also an engineering tool, allowing the generation of project-related files, in order to facilitate the development process and offering developers a support and dedicated tools from the design and specification process to the testing process.

Simulation Framework Approach

The need for a framework

Since system of systems simulations involve more complex operational scenarios, simulation engineers face very challenging requirements, so as not to lag behind.

To cope with these increasing expectations, a global approach needs to be undertaken to find the most generic solution. A "framework" provides an extensible and flexible structure, to ease both the development and the exploitation of new applications. It covers a large spectrum of system of systems applications; it also supports engineers across the entire simulation process: developing new simulations, preparing complex scenarios on a map, executing and controlling their simulation, using 3D visualization and processing the results.

The framework composition and services

A framework is made of a set of coherent components, which are listed hereunder:

- a simulation engine

The framework relies on a simulation engine that serves as a core for model integration and tool development. Its applications may concern both civilian and military domains. Within this engine, our simulation entities (aircrafts, ground vehicles, etc.) are considered as actors. Actors exhibit attributes, which have a type, a value and systematic getter/setter functions. They communicate within a simulation through messages. A manager directs the actors and is responsible for the entire simulation process (including time management and scenario loading). A set of common core services is also available for models: it concerns mathematic functions, input/output handling and geographic services (numeric terrain model, atmosphere and ephemeris model, etc.).

- a library of capitalized models and actors

The engine comes with a standard library of capitalized actors and models. For defense applications, this library (provided by MEFISTO) involves models for sensors, weapons, platform mobility, artificial intelligence, tactical data links, battle damage and decision processes. All platform actors publish kinematic attributes in a standard way, know how to take damage from different kinds of weapons, understand damage states, can perform dead reckoning for both local and remote vehicles, and have the infra-red and radar cross-section attributes useful for sensor models

- a set of exploitation tools

Scenario preparation: in this phase, the different actors of the simulation (platforms, sensors, weapons) are defined, together with their technological parameters, their force affiliation and their initial behavior. The GIS capabilities of the tool help us to precisely locate the actors in the world. At the end, the scenario is saved in XML format. The scenario editor relies on the generic concepts of the underlying engine; nevertheless it is possible to customize it for a particular application, thanks to a plug-in architecture.

Simulation execution and control: the execution mode depends on the user context. It is both agile and scalable. It is agile because we can choose between three modes: a batch mode with no GUI (for development and automatic tests), an execution mode from a stand-alone tool called Scenario Manager (for fast simulations and visual checking) and a full deployment mode on a HLA federation (for demonstration purposes, performance optimization and interoperability). It is scalable because, in HLA mode, each federate can be parameterized with a list of actors to physically simulate. The simulation can also be visualized in three dimensions thanks to a Stealth Viewer.

Result processing: The framework provides result processing and analysis tools: chart visualization of simulation results, easy trajectory replay from Scenario Manager, and federation execution replay from the Stealth Viewer.

The framework status and perspectives

The current version of the framework was developed by an in-house research project called AMAO [11]. The purpose of this project was to study the role and the concepts of Unmanned Aerial Vehicles (UAVs) in future offensive missions. The resulting experimentations can use both constructive and virtual simulations to assess various issues, such as the role of the different actors in the decision making process, the overall mission timing and coordination process, the efficiency of subsystems (sensors, weapons) and algorithms (image processing and data fusion) and the robustness of the proposed system concepts.

The current MEFISTO research project is aimed at extending the framework with new models to perform new applications. Since the framework turned out to be an agile and generic backbone, we would like to keep on capitalizing models for new scenarios, filling the needs of future Onera projects.

GAMME & SIMSKY

When designing distributed simulation systems, data management is a particularly crucial issue. Data management is about storage, access, distribution and representation of entities that are shared by different simulators. The approach presented here puts data management at the core of the simulation development process.

We distinguish between two kinds of data: persistent and non-persistent (transient). We could also say static and dynamic data. Persistent data is used to initialize the simulation. It may be stored in a relational database or some files, like XML files. Access to it can be obtained by querying the database or reading the files. Transient data is about dynamic data evolving during the simulation. For example, an airplane will be initialized with static data (number of passengers, initial mass, etc.) and its dynamic behavior will be described by a state vector (position, speed, current mass, etc.). Dynamic data is not stored, but distributed between the different components of a simulation, e.g. an aircraft simulator will publish state vectors, whereas an ATC radar simulator will subscribe to them to display radar tracks.

All of this data may have different representations, especially in heterogeneous simulations where simulators are developed in different languages. An entity may be represented by some tables in a relational database, classes in object-oriented programming languages, some textual or binary file formats, etc. Consistency must be ensured between these different representations, but maintaining them may become very repetitive, painful and error-prone.

This is why Onera has developed GAMME, a tool aimed at deriving many different representations starting from a single "unified" data model. It consists of a conceptual data editor and transformation mechanisms. GAMME relies on the Model-Driven Engineering (MDE) methodology. The users define domain-specific data models, that is, abstract representations of the knowledge shared by the different components of a simulation, and apply transformations on them. Transformations result in pieces of code, database schemes, XML schemas, etc., which are then used by simulator developers to access or distribute data. Thus, developers can concentrate on their domain-specific code (physics or mathematics) and not on IT-related code, for instance, focus on how an aircraft flies and not on how aircraft data is stored or shared with other simulators. This approach is meant to offer a common design, independent from the target representations, and to increase productivity by reusing transformations and generating huge and errorless code.

The first application of GAMME to distributed simulation is the SimSky project.

SimSky is the second version of the IESTA platform. It is aimed at assessing innovating concepts in the ATM system of systems: for example, Onera is currently involved in projects studying fully automated air transport systems, the impact of introducing drones in general aviation airspace, a concept of personal air transport, etc. Such studies are based on refining concepts increasingly through validation steps which rely on iterative simulation.

In this perspective, Onera needs a tool for rapid and iterative prototyping of ATM concepts. Many distributed simulation platforms already exist, however they do not fulfill all SimSky requirements, for instance because of lack of flexibility, complex programming interfaces, proprietary code, limited number of programming languages, etc. Conversely, SimSky aims at proposing a number of facilities:

- a domain-specific data modeling thanks to a user-friendly GUI, promoting iterative prototyping of concepts,
- the rapid interconnection of multi-disciplinary models (e.g. flight mechanics, conflict detection and resolution, strategic planning, etc.) through distributed simulation and simple APIs,
- an open infrastructure, available on several operating systems and programming languages dedicated to various needs and programmer skills: scientific computation (like Matlab/Fortran), algorithmic (CAML), performance (C/C++), GUI (Java), rapid development (Python), etc.,
- tools for the supervision and monitoring of simulations, like air traffic visualization,
- a set of default simulation models (air traffic simulation, weather model) allowing algorithms to be quickly tested in an ATM simulated environment,
- fast-time and real-time / manned and unmanned simulation capabilities.

All of the IT-related code is automatically generated with GAMME: database access, file management, data distribution over the network through middleware, such as the rapid prototyping-oriented Ivy[4], or the reliable and interoperable HLA.

We now present some actual and representative applications of distributed simulation in which Onera is involved: the first is set up in civilian air traffic context, whereas the second deals with a framework for the testing of defense concepts.

Applications

ASTRAL & 4DCoGC

One of the first applications using SimSky will be ASTRAL. This research project is aimed at deeply studying the concepts introduced in the European Project 4DCoGC.

[4] http://www.tls.cena.fr/products/ivy/

These projects study a revolutionary post-SESAR concept for a future ATS (Air Transportation System) by adding as much onboard autonomy to the aircraft as necessary to fulfill the overall requirements of improved efficiency and safety of air transportation.

The system of systems studied by both ASTRAL and 4DCo-GC is composed of Airline Operation Centers, a centralized ground-based ATSM (Air Transport System Manager), aircraft, data-link communication systems, etc. The ATSM and aircraft are themselves systems, some parts of which can be individually simulated (for example, FMS, auto-throttle & auto-pilot, local aircraft re-planning system). Such a system is illustrated in figure 3.

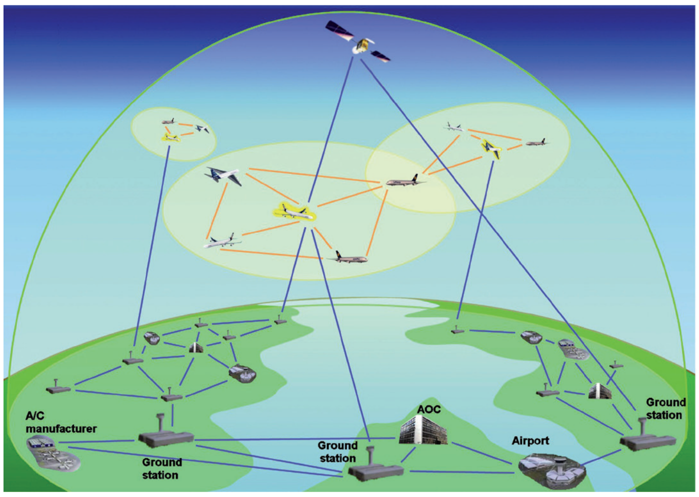

Figure 3 - An ATM system of systems

The central idea of "4D-contract" is that each aircraft must be at the right place at the right time and implies that the aircraft is fully responsible for following the 4D-contract that has been assigned to it by the ATSM. If the aircraft finds that it cannot comply with the contract (due to unpredictable weather conditions or unexpected events, such as a runway closure), then it re-negotiates a new contract with the system. Margins around the expected positions are represented by "freedom bubbles", where the aircraft can move without having to re-negotiate and "security bubbles" to ensure aircraft separation.

The main objectives of ASTRAL are to define and implement models and algorithms for 4D planning and 4D-contracts guidance and monitoring, put the concept under stress via iterative simulations, then present results about the concept's robustness towards the input parameters of planning algorithms (take-off weight, weather, etc.) and finally draw lessons from this to make recommendations on freedom and security bubbles.

MEFISTO

The MEFISTO research project objective is to provide a battlelab for the French Aerospace Lab. This battlelab relies on a simulation framework filled with a library of capitalized Onera models. It focuses heavily on interoperability, and will be connected to the French Army laboratory for technical and operational simulation (DGA LTO) through a dedicated collaboration network (EXAC and the ITCS gateway). Thus, it will authorize collaborative simulations and experimentations with DGA and the other connected battlelabs (e.g. of major industry groups like EADS, Thales, MBDA, etc.).

The main achievements of MEFISTO are:

- the development of two applications :
  - PANTHERE dealing with the study of missile penetration in air defense assets,
  - SITAC dealing with tactical situation awareness on the battle field;
- an interoperability experimentation with the DGA: an independent unmanned air vehicle strike mission will be simulated, part of the models being carried out by the DGA LTO in Arcueil and the other part by the Onera in Palaiseau.

Conclusion

Simulation is a very important approach for the study of complex systems. Especially in the case of aerospace systems, which cannot be directly tested because of their cost or criticality, simulation is a perennial, indispensable activity for designing and testing new concepts.
As this paper shows, simulation of complex systems goes far beyond the simple virtual restitution of a complex system. It takes into account multidisciplinary aspects and also secondary purposes such as capitalization, knowledge protection, easy prototyping, and mutualization, which do not only allow simulations to be set up, but also go beyond the study of the systems to focus on their interactions and allow the consequences of the use of system concepts to be tested.
Onera has a broad panel of tools for the simulation of complex system and the exploitation of the data produced by such simulations, thus positioning itself for the study of future techniques or system designs. These simulation techniques are a new and innovative means to design, test and even validate complex system concepts. It also offers access to a larger audience to the possible consequences of such new concepts, since they allow the system to be viewed in its environment and also provide a way to show these results to the research teams involved in the development of a simulation.
As for exploitation techniques, which are paramount in the simulation approach, they constitute a very active field of study, as has being outlined in this paper. Some interesting perspectives to this extent can be seen in the use of description languages (e.g. the NAF — NATO Architecture Framework) in order to specify the complex system in such a way that the difficult points to be particularly considered in the simulation will be outlined ■

Acknowledgements

The authors would like to thank several people who were a great help on the various topics covered by this paper, especially by providing useful insights on various tools or applications: A. Elie for the SimSky project, A. Joulia for the 4DCoGC project, P. Siron and M. Adelantado for the CERTI. The authors would also like to express special gratitude to Dr. H. Piet-Lahanier for her careful reviewing and comments on this paper.

References

[1] O. BERTRAND, P. CARLE, C. CHOPPY - *Chronicle modelling using automata and coloured.* Petri nets, Proc. of the 18th International Workshop on Principles of Diagnosis, 2007
[2] O. BERTRAND, P. CARLE, C. CHOPPY - *Vers une exploitation des simulations distribuées par les chroniques.* Proc. of the 8ième Rencontres nationales des Jeunes Chercheurs en Intelligence Artificielle, 2007
[3] O. BERTRAND, P. CARLE, C. CHOPPY - *Towards a Coloured*. Petri nets semantics of a chronicle language for distributed simulation processing. Proc. of the CHINA (Concurrency metHods: Issues aNd Applications) workshop, 2008
[4]O. BERTRAND, P. CARLE, C. CHOPPY - *Coloured Petri Nets for Chronicle Recognition*. Proc. of the 14th Int. Conf. on Reliable Software Technologies, 2009
[5] J. BOURRELY, P. CARLE, M. BARAT, & F. LÉVY - *Genesis: an Integrated Platform for Designing and Developing HLA applications*. Proc. of Simulation Interoperability Workshop, 2005
[6] J. BOURRELY, R. KERVARC, C. QUILLIEN - *Performance Evaluation for Complex System.* F. Pistella, R. M. Spitaleri (eds.), Proceedings of the 7th IMACS/ISGG Meeting on Applied Scientific Computing and Tools (MASCOT07), IMACS Series in Computational and Applied Mathematics, 2007
[7] P. CANTOT AND D. LUZEAUX - *Simulation et modélisation des systèmes de systèmes*. Hermès, 2009
[8] P. CARLE, P. BENHAMOU, F.-X. DOLBEAU, M. ORNATO, La reconnaissance d'intentions comme dynamique des organisations, in: Proc. of the 6iemes Journées Francophones pour l'Intelligence Artificielle Distribuée et les Systèmes Multi-Agents, 1998
[9] P. CARLE, C. CHOPPY, R. KERVARC - *Behaviour Recognition Using Chronicles.* Z. Duang, L. Ong (eds.), Proceedings of the 5th IEEE International Conference on Theoretical Aspects of Software Engineering, p. 100 - 107, IEEE Computer Society Press, 2011
[10] P. CARLE, C. CHOPPY, R. KERVARC - *Detecting Behaviours Within HLA Distributed Simulations with Added Analysis Components*. Proceedings of IEEE Aerospace Conference, 2012
[11] R. CUISINIER, M. BRUNEL, S. PRUDHOMME - *Using Open Source to Build Comprehensive Battlespace Simulations*. Proc. of SimTecT, 2010
[12] J. S. DAHMANN - *The High Level Architecture and Beyond: Technology Challenges*. Proceedings of the Thirteenth Workshop on Parallel and Distributed Simulation, p. 64–70, 1999
[13] C. DOUSSON - *Extending and Unifying Chronicle Representation with Event Counters*. Proc. of the 15th European Conference on Artificial Intelligence, 2002
[14] C. DOUSSON, P. GABORIT, M. GHALLAB - *Situation Recognition: Representation and Algorithms.* Proc. of the International Joint Conference on Artificial Intelligence, 1993
[15] C. DOUSSON, P. LE MAIGAT - *Improvement of Chronicle-Based Monitoring Using Temporal Focalisation and Hierarchisation.* Proc. of the International Workshop on Principles of Diagnosis, 2006
[16] C. DOUSSON, P. LE MAIGAT - *Improvement of Chronicle-Based Monitoring Using Temporal Focalisation and Hierarchisation.* Proc. of the International Joint Conference on Artificial Intelligence, 2007
[17] IEEE 1516-2010, *Standard for Modeling and Simulation High Level Architecture* - framework and rules
[18] IEEE 1516.1-2010, *Standard for Modeling and Simulation High Level Architecture* - federate interface specification
[19] IEEE 1516.2-2010, *Standard for Modeling and Simulation High Level Architecture* - object model template (OMT) specification
[20] M. JAMSHIDI (editor - Systems of Systems Engineering: Principles and Applications (chapter 5), CRC Press, 2009
[21] R. KERVARC, K. BOURRELY, C. QUILLIEN - *A generic Logical-Temporal Performance Analysis Method for Complex Systems*. Mathematics and Computers in simulation 81: 717–730, 2010
[22]R. KERVARC, C. CHOPPY, P. CARLE - *Chronicles: a Temporal Logic Framework for the Study of Large Simulations*. R. M. Spitaleri, A. Plaza, F. Pistella (eds.), Proceedings of the 10th IMACS/ISGG Meeting on Applied Scientific Computing and Tools (MASCOT10), IMACS Series in Computational and Applied Mathematics, 2010
[23] R. KERVARC, C. LOUYOT, S. MERIT, S. BERTRAND, J. BOURRELY - *Performance Evaluation Based on Temporal Logic*. F. Pistella, R. M. Spitaleri (eds.), Proceedings of the 9th IMACS/ISGG Meeting on Applied Scientific Computing and Tools (MASCOT09), IMACS Series in Computational and Applied Mathematics, 2009
[24] I. NAKHIMOVSKI - *Contributions to the Modeling and Simulation of Mechanical Systems with Detailed Contact Analyses.* Ph.D. thesis, Department of Computer and Information Science, Linköpings University, 2006
[25] M. S. SHEPHARD, M. W. BEALL, R. M. O'BARA, B. E. WEBSTER - *Toward Simulation-Based Design*. Finite Elements in Analysis and Design 40(12): 1575–1598, 2004
[26] R. SINHA, V. LIANG, C.J.J. PAREDIS, P.K. KHOSLA - *Modeling and Simulation Method for Design of Engineering Systems*. Journal of Computing and Information Science in Engineering 1(1): 84- 91,2001
[27] H. ZHANG, H. WANG, D. CHEN, G. ZACHAREWICZ - *A Model-Driven Approach to Multidisciplinary Collaborative Simulation for Virtual Product Development.* Advanced Engineering Informatics 24: 167–179, 2010

Acronyms

4DCoGC (4-Dimensional Contract Guidance and Control of the aircraft)
AMAO (Autonomie des Missions Aéroportées Offensives - Onera federative project on the autonomy of offensive air missions)
ASTRAL (Automatisation des Systèmes de TRansport Aérien à Long terme - long-term air transport system automation)
ATC (Air Traffic Control)
ATM (Air Traffic Management)
ATS (Air Transport System)
ATSM (Air Transport System Manager)
CERTI (Onera's RTI)
DGA (Délégation Générale pour l'Armement - Division of the French Defense Ministry)
EXAC (EXpérimentation pour l'Acquision de Capacité - DGA secure network for experiments)

FOM (Federation Object Model)
GAMME (Génération Automatique pour la Métamodélisation Métier Enrichie - Onera project on metamodelling and data-centered techniques for code generation)
GENESIS (Onera's assistance tool for the generation of HLA-compliant simulations)
GIS (Geographic Information System)
GIS (Graphical User Interface)
HLA (High-Level Architecture)
IEEE (Institute of Electrical and Electronics Engineers)
IESTA (Infrastructure d'Évaluation des Systèmes de Transport Aérien - Onera infrastructure for the evaluation of air transport systems)
ITCS (Infrastructure Technique Commune dédiée à la Simulation pour l'acquisition - Technical M&S infrastructure for supporting SBA process)
LTO (Laboratoire Technico-Opérationnel - technical and operational laboratory)
M&S (Modeling & Simulation)
MEFISTO (Moyens d'Études Fédérés et Interopérables pour la Simulation Technico-Opérationnelle - Onera federative project for the study of technical and operational simulation)
NATO (North Atlantic Treaty Alliance)
OMT (Object Model Template)
RTI (Run-Time Infrastructure)
RTIA (Run-Time Infrastructure Ambassador)
RTIG (Run-Time Infrastructure Gateway)
SBA (Simulation-Based Acquisition)
SESAR (Single European Sky ATM Research)
SimCore (Simulation Core)
SimSky (Onera's fast-prototyping simulation architecture)
SOM (Simulator Object Model)
XML (EXtended Markup Language)

AUTHORS

Patrice Carle graduated from the Université Paris VI, where he obtained a Ph.D. in mathematics and artificial intelligence in 1992. Since then, he has been a full-time researcher at Onera, first at the Department for Modeling and Information Processing, then at the Department for System Design and Performance Evaluation. He is currently head of the TCS ("system design and simulation techniques") research unit. His research interests focus on distributed simulation, domain specific languages and code generation.

Romain Kervarc graduated from the École Normale Supérieure de Lyon and obtained a Ph.D. in formal logic in 2007. Since then, he has been a full-time researcher at Onera, where he is also a member of the scientific council of the branch for Information Processing and Systems. His research interests include formal methods, logic, modeling and evaluation of complex systems. He is also a member of the Supervisory Board of the European Union Initial Training Network "From Mathematical Logic to Applications".

Raphael Cuisinier graduated as an engineer from the École Polytechnique in 1999, after which he obtained his Engineering Diploma from Sup'Aéro in 2011. After five years at the DGA (Directorate for Armament of the French Ministry of Defense), he joined Onera in 2006, where he is currently head of the S2IM ("simulation, infrastructure, model integration") research unit and responsible for Onera project MEFISTO, aimed at developing an Onera Battlelab.

Nicolas Huynh graduated from the French national college of aeronautics and aerospace (Sup'Aéro) in 1989, and holds two MSc in aerospace engineering and computer science. He worked for fourteen years at the French Air Navigation Service Provider on many Air Traffic Management (ATM) simulation projects and, since 2007, has been a research engineer at Onera, where he has been in charge of simulation infrastructure and integration activities in several projects and now leads project SimSky, aimed at rapid prototyping of ATM concepts.

Judicaël Bedouët is an engineer from the École Nationale d'Aviation Civile and holds a M.Sc. in security of software jointly from Université de Toulouse II/III, Institut National Polytechnique de Toulouse and Institut Supérieur de l'Aéronautique et de l'Espace. He entered Onera in 2008 as a research engineer at DCPS. His research interests include distributed simulation applied to air transport systems, data-driven approaches for simulation, automated generation of code and software quality.

Thomas Rivière is an Onera research scientist with ten years of research experience in computer science applied to air traffic management problem solving. He is currently focusing on the environmental evaluation of air traffic. He is also an expert in research activities for air traffic management and computer science for the European Commission Community Research and Development Information Service.

Éric Noulard graduated from a French Higher Education Institution for Engineers in Computer Science (ENSEEIHT) in 1995 and received his PhD in computer science from Versailles University in 2000. After 7 years working in the Aerospace and Telecom domain for BT C&SI, mostly building high performance tests and validation systems, he joined the Onera research center in Toulouse as a research scientist. He works on distributed and/or embedded real-time systems and is actively involved in the development of the CERTI Open Source project.