B. Hérissé, G. Hervieux, K. Dahia, J.-M. Allard, J.-C. Sarrazin
(ONERA)

E-mail: bruno.herisse@onera.fr

DOI: 10.12762/2016.AL12-15

Component-Based Simulation for Real-Time Experiments of Advanced Aerospace Systems

This paper presents recent advances at ONERA in simulation engineering. Simulation is used for many purposes, from performance evaluation of algorithms to realistic emulation of the real world, including real components such as humans. To design general and scalable simulation software, a flexible architecture needs to be designed. A component-based architecture approach that allows multilevel capitalization and high collaborative sharing between systems experts and computing experts has been developed at ONERA. This architecture allow hybrid simulation involving real devices in the simulation loop to increase the Technology Readiness Level of novel algorithms by emulating real embedded environment. Human-In-the-Loop simulation is also an important tool to study the human interactions with a system. Some recent research projects are presented to illustrate our simulation platforms.

Introduction

Simulation is a more and more demanding tool

The main role of computer simulation is to simulate physical or non-physical phenomena that intervene in a system. This is achieved using analytical and numerical models but also black box functions stemming from an unknown subsystem, for example. Using these models, simulation platforms are often dedicated to facilitating design of systems and evaluation of their performance, their safety and their reliability (for controller design, mechanical design, etc.). For technical or cost reasons, using simulation for testing new systems involving complex algorithms, new technological components or testing human behavior in specific contexts can be the best way to achieve the project requirements according to its Technology Readiness Level (TRL). The scientific projects in which ONERA is involved have different TRL. Therefore, simulation platforms have to be sufficiently flexible to meet all research needs.

Simulation has evolved along with computers. From the 80s, object programming languages allowed the ability to work with advanced software components to be exploited. The first simulations were often monolithic, but tended to complex architectures that address systems and system of systems [1] [2] very well. Software design took a new direction, with business issues considered in the architecture design to meet new objectives: the capitalization and the maintainability of code. The component-based approach that is being developed at ONERA and detailed below illustrates this need to produce a reference code useful for business experts as well as for software experts.

With the increasing power of computers, simulation tools can provide results with more and more precision and always faster. However, scientist and engineer demand is also increasing with this respect. Indeed, the improvement of output precision implies more complex models that need more computation power on specific architectures. Therefore, in order to respond efficiently to the requirements, hardware and software architectures for simulations have to be heterogeneous and multiple. Thus, a key challenge for software engineers is to design simulations that manage multithreading, distributed architectures and hardware acceleration (using GPU or FPGA chips for instance). Another challenging problem consists in ensuring repeatability, reliability and portability of the simulation tool.

Simulation interacts with the real world

Increasingly, the use of simulation is extended to interact with the real world, including humans, using physical devices as interfaces (network communication devices, haptic interfaces, flight pilot simulator with emulated cockpit, etc.). Such simulation platforms are made to ensure a complete immersion of the target component being assessed in its environment. For example, the analysis of human behavior (stress, panic, etc.) in a cockpit can be evaluated correctly only if the simulation environment is sufficiently accurate and coherent in terms of motion and vibration especially. Moreover, training and education can use high fidelity simulation for a better experience (3D simulation with haptic devices in medical education for example).

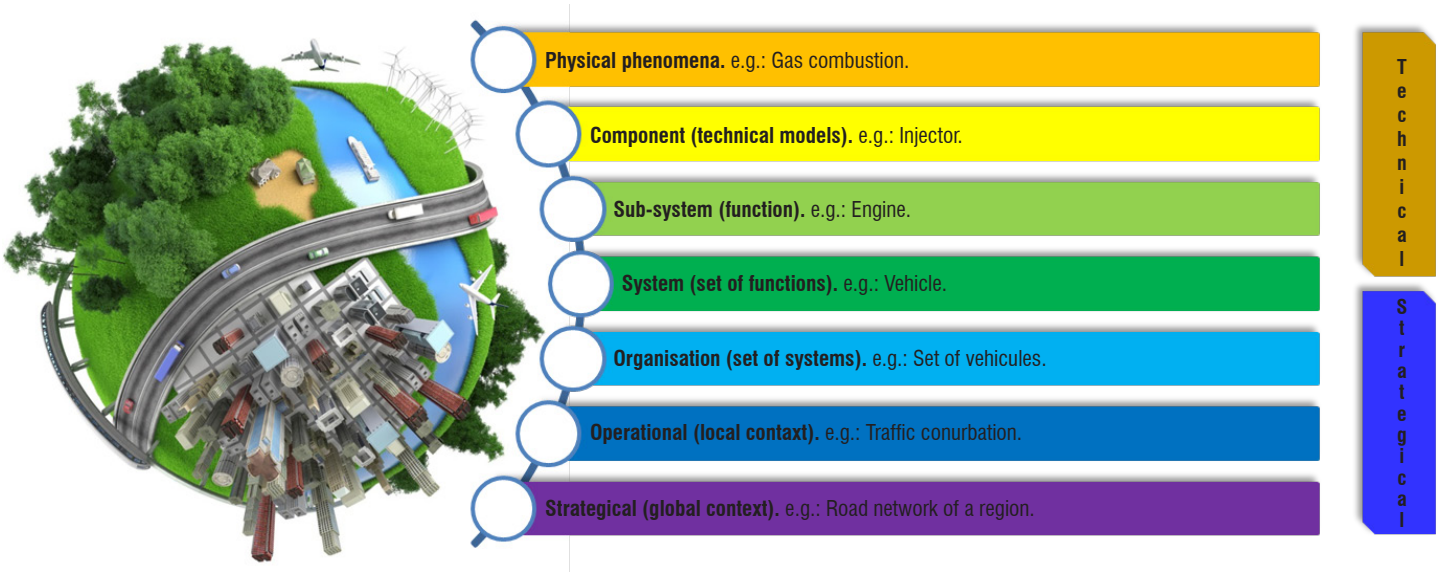


Figure 1 – Simulation levels of Systems

Thus, simulation of complex systems can include several specific simulators for each actor of the considered system, as well as real data stemming from a running system, such as a radar detection system or a decision platform. The simulation can also interact with humans via a flight simulator or a supervision platform. Moreover, hardware devices, such as embedded computers and sensors, can be incorporated into the simulation to assess their performance under realistic conditions before considering their use onboard a real system. In this paper, some of ONERA's simulation means are presented through some past and current research developments that involve real components in the simulation loop.

Designing relevant simulation

For a simulation platform to be fully reliable, the right behavior is fundamental to correctly emulate the real world. Therefore, evaluation tools are necessary. Taking advantage of the fact that all input and output signals are observable in a simulation test, such simulation tools can be designed. For example, in a Simulink simulation [3] or a LabVIEW simulation [4], all wires connecting component blocks can be checked in real-time with available viewers. Most real-time operating systems can be simulated to observe the full system behavior for every time step, using the associated tool such as WindView for VxWorks [5]. Profiling tools are also available for the recent Robotic-Operating-System [6] used for robotic applications. However, work remains to be done to interpret this flow of data in real-time and autonomously, so that supervision of the simulation and reconfigurations are possible. For example, communication based on an Ethernet network must be robust to packet loss using dead reckoning techniques. Such tools make it possible to verify that the simulation provides the right output related to the given input only empirically. To be fully reliable, formal proof should be provided in order to ensure a deterministic behavior [7].

Another important challenge raised by simulation is to make both the input data easy to enter in the simulation and the output result useful for the user. Ideally, the user should not have to understand how the tool is designed. This necessitates suitable autonomous interpreters that bridge the gap between the simulation and the user. Moreover, the simulator should be able to adapt its models to the required output precision by means of automatic model reduction, for example.

Simulation platforms at ONERA

This paper is aimed at presenting some of the simulation platforms available at ONERA through some recent or ongoing projects. It is organized as follows. In the next section, the architecture of the simulation Platform developed by engineers at ONERA is presented. Then, we focus on three topics of interest for our current and future advanced testing developments: interconnection of simulators in a distributed system, Hardware-In-the-Loop simulation and Human-In-the-Loop simulation.

Architecture of the simulation Platform

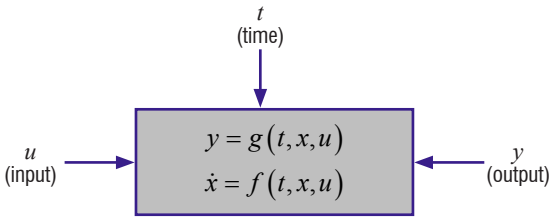
To efficiently design a software architecture understandable by both business experts and software engineers, it is necessary to develop a methodology based on a multilayer approach (Figure 1). Indeed, system studies sometimes require variable granularity models: very coarse to very fine simulation considering complex physical phenomena. The architecture implementation must be robust to this scaling problem. This implies the definition of the adequate scope for each component, as well as for the corresponding interfaces.

In the case of systems and systems of systems, this subdivision can be performed by functions or by physical architectures. Functional separation will be preferred for macroscopic levels and technical cutting up for low levels. To facilitate the work, it is helpful to use a pattern for arranging components in relation to each other. In our case, we used the SCA paradigm (Sensor / Controller / Actuator) well-known in the field of automatic control and especially suitable for addressing engineering systems.

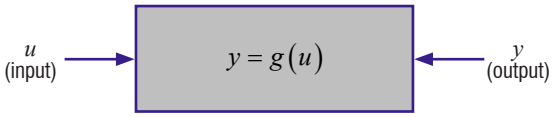
Component-based subdivision of the system

An actor in the simulation, also called agent in some cases, consists of components. A component is a block consisting of a model.

The purpose of this component consists in processing input data (the control u) to provide output data y. For technical components modeling the dynamics of a physical system, a simplified abstract representation is given below:



where g is the output function, x is the state and f is a continuous state function. For functional components, the representation does not involve time or internal state:



Definition of interfaces

For the communication between the components to be possible, it is necessary for their interfaces to be compatible. In our approach, an event mechanism is used. Once a component has finished a job, a new event is produced as a message and broadcasted directly to components that have subscribed to it. Each component is responsible for its subscription to the messages of interest. An initialization step is performed before the simulation is launched, to ensure that all of the communication channels are set up. During the simulation, a callback is run by a manager responsible for calling the broadcasting function when it is necessary.

Multi-level compatibility is ensured by a scale effect on each component. Indeed, it is possible to divide a component A into sub-components {A.1, A.2, ..., A.n}. The interface between a component A and a component B implies the interfacing between the components {A.1, A.2, ..., A.n} and B. This means that the inter-component interfaces should remain the same. However, the intra-component interfaces can be defined regardless of the outside and may be more specific or less generic.

Implementation of the simulated system

The architecture presented in Figure 2 is organized around a master (the *SimulationManager*), it is able to access all of the players (the *Entity*) in the simulation, as well as process them over time. These players consist of components that can be functional if the time is not a necessary input data for updating, or technical if a temporal integration mechanism is needed. Communication between components is ensured by a specific object (the *CommunicationManager*) attached to the master. It is responsible for transmitting messages between players. In order to manage time in the simulation, a specific component (the *Sequencer*) is dedicated to managing the execution sequence.

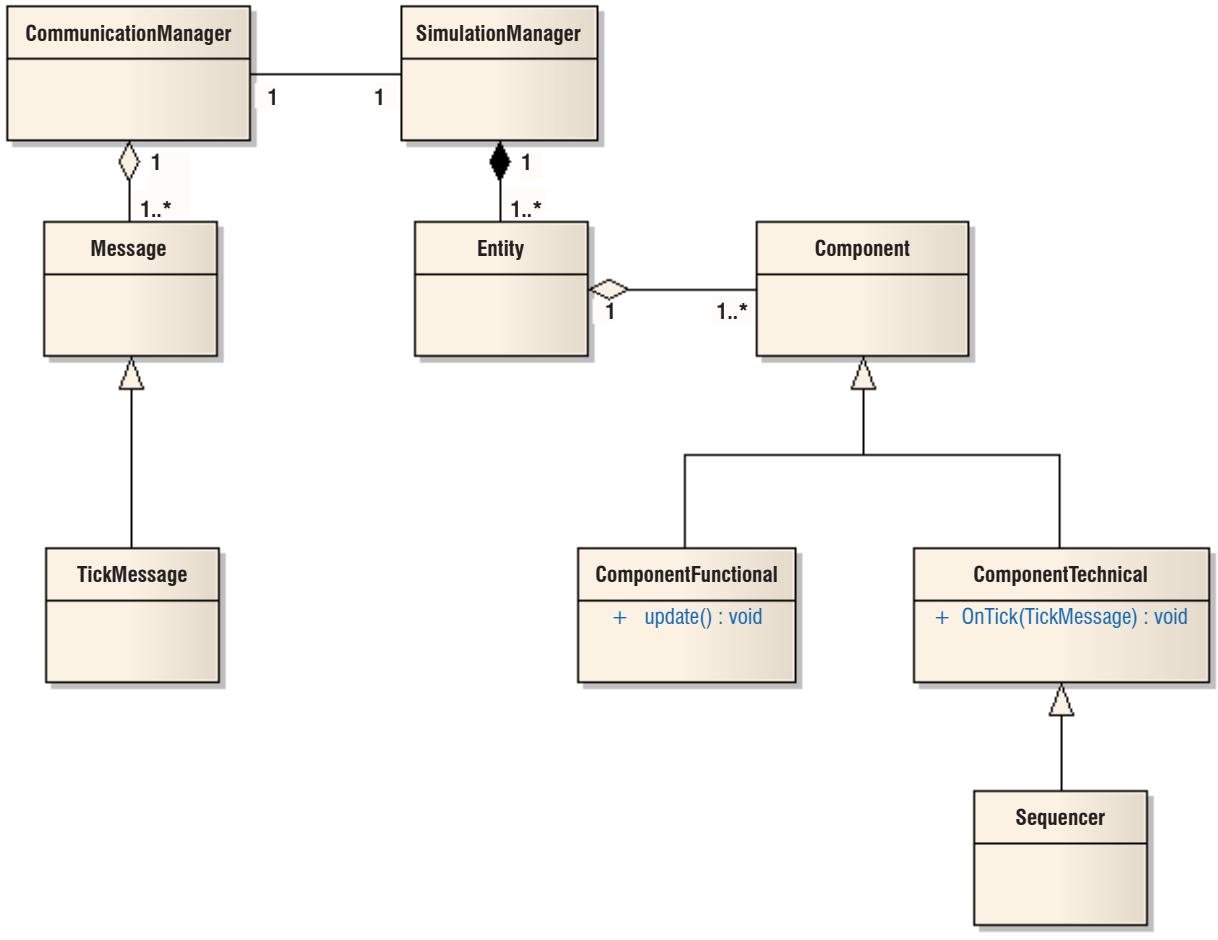


Figure 2 – Architecture of the simulation (class diagram)

Managing discrete and continuous events

The management of discrete and continuous events is done by sending messages. In the discrete case, a message can be any event sent punctually that will be processed upon receipt of the message modifying the internal state of the component. In the continuous case, a time message (*TickMessage* in Figure 2) is sent at each step of the simulation. This time message contains the current simulation time and the time step thereby achieving, for example, an integration of the component state. Messages are communicated via a callback when it is necessary. Event-messages are sent instantly. Time messages are sent at each step (*tick*) of the simulation. For the purposes of specific inter-component communication, messages are sent instantly, thereby making available up-to-date data to meet the needs of each component.

As previously said, the management of simulation is performed by a master managing the events and the progress of each player and component. For specific system needs, a new feature was implemented to allow each component to work at a different time step. Indeed, in most systems, sensors, controllers and actuators do not operate at the same speed. To be more representative of the physical reality, it is necessary to consider this time constraint in order to ensure that the system remains stable. For example, guidance and control modules of an aerospace vehicle do not operate with the same speed. Therefore, sub-tick management has been implemented in the *Sequencer*. Moreover, components need to be activated in a right order so that the loop progresses correctly over time. In our simulator, actuators, then sensors, then controllers are activated successively in the simulation loop.

Interconnected simulation

The interconnection of a simulator with an external tool is quite common. Often, some processing and even some simulators are third party tools that cannot be integrated either with the used technologies or due to the intellectual property. The interconnection is very often based on a standard of communication so that it can be facilitated. The standard must be implemented on both sides and can sometimes be expensive, especially when a specific implementation meets a specific need. Two cases are considered here: a first case addressing interoperability between simulations through the use of standard High-Level-Architecture and a second case dealing with a specific interoperability with an external tool (which is not a simulator) processing data during the simulation.

Distributed simulation

Distributed simulation offers the possibility of playing a simulation on several remote machines. In order to implement this type of simulation, the High-Level-Architecture (HLA) [8] [9] [10] standard has been used at ONERA for twenty years [11]. The previously described simulator provides a HLA gateway to map messages to the HLA standard. This standard describes the objects and shareable messages in a reference RPR-FOM [12]. The master of the simulation is a Run-Time-Infrastructure (RTI) that manages time. Messages are timestamped and a dead-reckoning mechanism allows the Federated actors to have the right information at time t knowing the information at time $(t - dt)$.

This kind of distributed simulation has been implemented many times at ONERA. In particular, tests in collaboration with the DGA (the French department of defense) in the simulation network SimDEx Defense (see Figure 3). In this experiment, some actors (Federated) were played at ONERA and the others at the DGA, thereby demonstrating the feasibility of large-scale simulations involving many actors (industry, state, etc.).

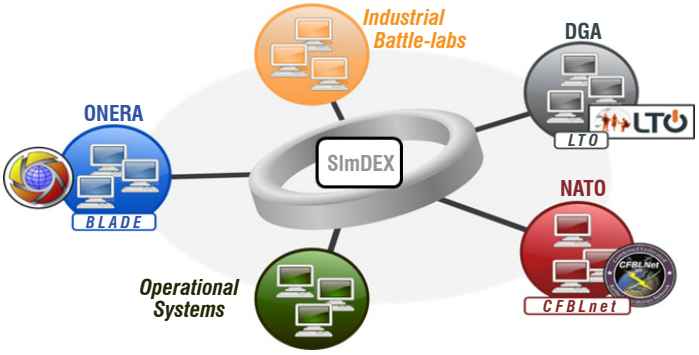


Figure 3 – HLA distributed simulation in the SimDEx network

In this example, interconnection consisted in providing the simulator with an external tool that is responsible for carrying out the fusion of data from the simulation. This tool is also involved in the simulation chain, since it may send requests for additional information via an operator. Therefore, this simulation also deals with the Human-In-the-Loop problem with this contribution of a human during the simulation. From the point of view of computer implementation, a specific gateway has been developed for converting the simulation events into SOAP messages. These XML SOAP messages are specific to our implementation. The data fusion tool only reacts to data from the simulation by processing them according to the current time of receipt. These messages are timestamped and sent by the simulator. A buffer mechanism was provided at the gateway for more flexibility in the transmission of data over the network. A synthetic scheme is proposed below in Figure 4. Note the specified processing loops representing a process. It was necessary to create a thread for receiving the simulation data so that the simulation loop is not blocked. The exchanges are ensured by two SOAP connections: sending simulator data (uplink) is done by the server to the client from the external tool; receiving data from the external tool simulator is done by the client simulator (downlink).

This work has contributed to highlighting the interconnection capacity of the simulator with an external tool. The effort involved is more important when using a standard communication protocol is not possible. Preference is given to the use of gateways that can be capitalized without creating a strong connection with the simulator. Indeed, in our case, the gateway converts the simulation messages to SOAP messages without any useful gains (only a translation is carried out).

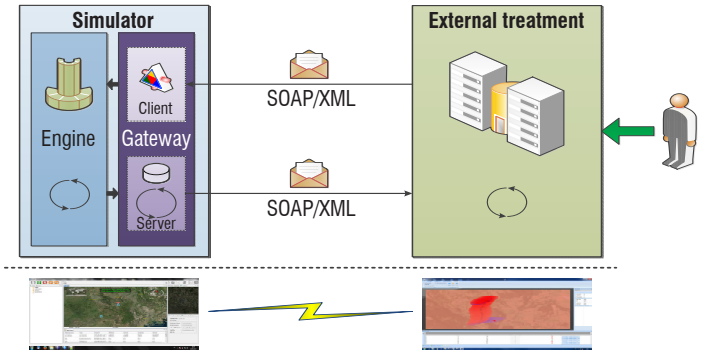


Figure 4 – Example of a specific interconnection

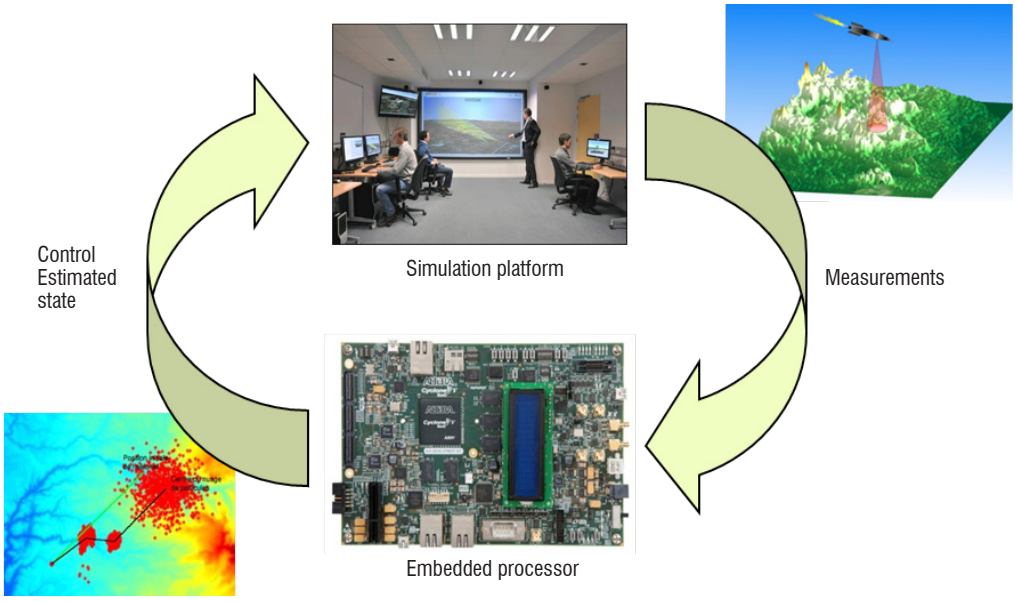


Figure 5 – Presentation of a HIL Simulation

Hardware components in the simulation loop

The purpose of the Hardware-In-the-Loop (HIL) simulation is to evaluate a real sub-component of an embedded system inside a simulation of the other components of the system. This method is widely used for system validation and verification. This is particularly interesting when the system is very complex and includes many hard components (sensors, actuators, controllers, interfaces, etc.). Testing each component with real experiments could be very expensive, unacceptable or even impossible. For example, the control of a vehicle used under critical conditions requires many tests to ensure its robustness to every possible scenario, the HIL simulation then offers a cheap, safe and repeatable method for this purpose.

A HIL simulation is a real-time loop including three main components. The embedded hardware is the component that needs to be evaluated. This often consists of a controller alone. However, real sensors and actuators can also be plugged into this controller. In that last case, another hardware component is necessary to emulate the plant model. For example, if a vehicle motor and its controller are tested in a HIL simulation, the motor load needs to be emulated by another physical component. If a vision-based algorithm is under test including the camera, images of the modeled environment need to be generated [13]. The HIL simulation also includes a real-time computer implementing the plant model. Eventually, an I/O device is used for the communication. Indeed, for a complete evaluation of the embedded component, the I/O interface is needed to communicate with the simulation using the real physical signals that are either feeding an actuator or generated by a sensor. Such a testing approach is often used in the design chain of a system controller. The HIL simulation constitutes the last test before validation with the real physical system after the other validation steps have been performed, that is, Model-In-the-Loop (MIL), Software-In-the-Loop (SIL) and Processor-In-the-Loop (PIL) simulations. Only the HIL and the PIL simulation can be used to verify that the execution time in the embedded processor fits within the required time. For PIL simulation, the interface between the embedded controller and the plant model is ensured by a standard communication such as Ethernet, while for the HIL simulation, the plant model is implemented on a hard real time computer, so that full real time simulation can be performed.

When a full complex system, such as the aerospace systems encountered at ONERA (missile systems for example), cannot be physically built and tested, PIL and HIL simulations offer a way for experimenting new algorithms in real time and under real conditions. Using this approach, the next section presents the experimentation of an advanced navigation and guidance algorithm currently being done in the DCPS department at ONERA.

Experiments of embedded advanced GNC algorithms in a HIL simulation

Experiments on advanced navigation and guidance algorithms are currently conducted in the DCPS department using the presented approach to demonstrate their performance in a realistic complex scenario.

The problem considered consists of both an interceptor missile and a cruise missile. The goal of the interceptor is to intercept the possibly maneuvering target (Figure 6). A ground station involving a radar detects the target and sends a predicted intercept point [14] to the

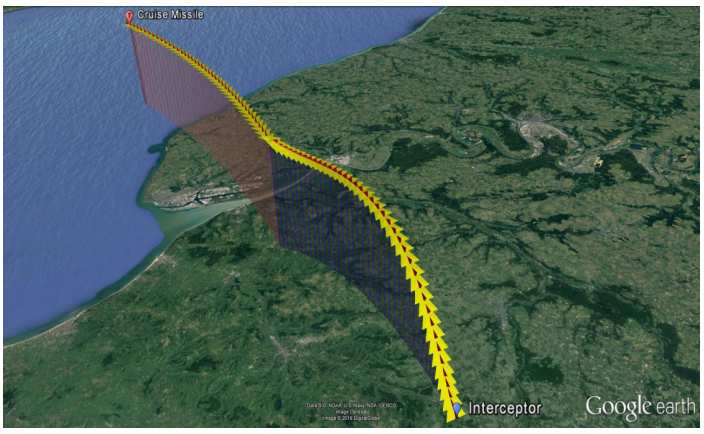
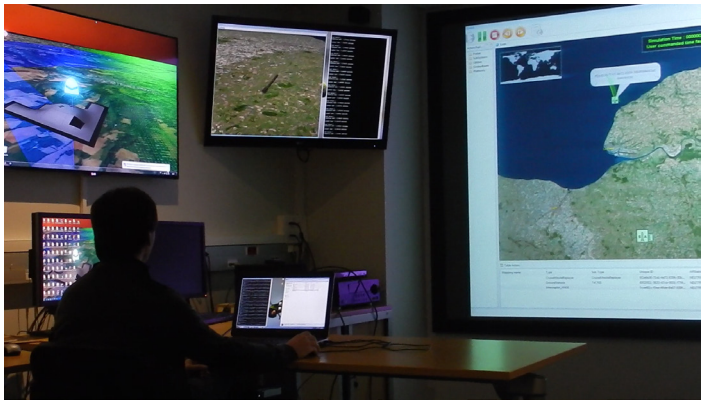


Figure 6 – Presentation of the intercept scenario

interceptor so that it can ensure the rendezvous. For this scenario, a controller is embedded in the interceptor for the midcourse guidance [15] and a second controller is embedded in the cruise missile for navigation [16]. A short video presenting this project is available at the following URL: http://www.aerospacelab-journal.org/sites/aerospacelab.onecert.fr/files/playlists/al12-15-video1.flv



Video 1 – Hardware-In-the-Loop simulation for missile interception

The computing consumption of classical algorithms used in most aerospace systems is low. Indeed, these algorithms often rely on analytical equations that can be computed very fast. Thus, cheap and low-power embedded processors are sufficient to ensure real time performance. Such classical algorithms are not satisfactory for complex missions involving navigation without GPS or optimal guidance, as considered here. Thus, advanced algorithms to address such problems have been developed. For the navigation of the cruise missile, specific particle filters were designed [16] [17] and for midcourse guidance of the interceptor, an indirect shooting method was designed [15]. These two algorithms are computationally demanding, since the first relies on the Monte-Carlo method and the second relies on iterative Newton methods [18]. Good performance is demonstrated in simulation results. However, it remains to be shown that these techniques can be embedded with the same level of performance, that is, real time capabilities need to be verified on an embedded processor.

To this end, a PIL simulation has been carried out: Figure 7 briefly presents the architecture of the simulation. A computer ensures the simulation of the system consisting of three main components: the interceptor component, the cruise missile component and the ground station component. The interceptor includes a subcomponent that ensures the computation of the control and the target component includes a subcomponent for navigation. Two modes are available: a full MIL simulated mode (control and navigation are modeled) and a PIL mode (control and navigation are ensured by embedded processors). For the PIL mode, the subcomponents directly transmit input data to the embedded processor through an Ethernet interface and the output data are acquired using the same interface. The processors that are used here are two Cyclone V SoC Development Kits from Altera that allow computing acceleration using the integrated FPGA [19]. Since the interface between the simulation and the embedded processors does not emulate real signals yet, this simulation is not fully HIL. However, as a first step, the PIL simulation is sufficient to verify that the processors can compute output data within the required time.

The proposed simulation loop offers a tool that allows the team to experiment new algorithms in realistic situations, so that the practical feasibility of our algorithms can be demonstrated. However, critical aerospace applications demand a safe and deterministic behavior. Therefore, to increase the TRL of our novel algorithms, a formal verification of the developed software may be required [20]. For instance, some advanced algorithms developed at ONERA rely on optimization methods, such as the Newton method or the interior point method [21]. A formal analysis of such methods needs to verify that a given precision can be achieved for a given number of iterations. These deterministic properties are necessary to ensure that the implemented algorithms can be qualified and transferred to industry. Future work will concentrate on these important issues.

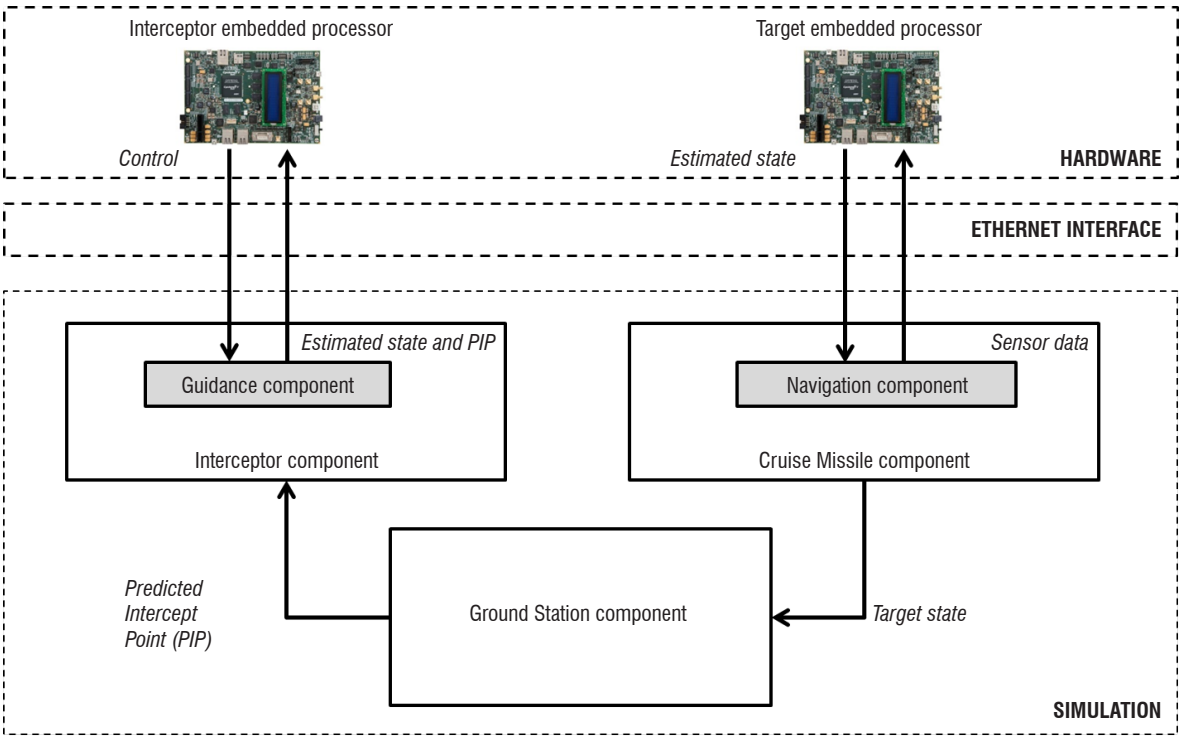


Figure 7 – Architecture of the PIL simulation

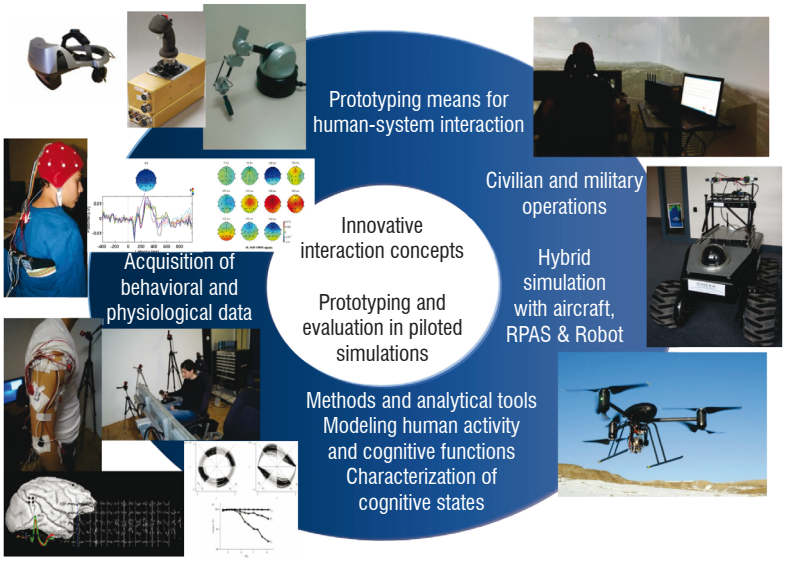


Figure 8 – Some experimental means available at ONERA

Human-in-the-loop simulation

When a Human interacts with the simulation, the simulation is called Human-In-the-Loop. Such simulations can be used for training, for example (flight simulators for pilot training), or to assist in the design of virtual objects by creating a simulated environment. Human-In-the-Loop simulation has been a focus of many research projects [22] [23] [24]. These projects generally consider how automation and humans can work together or in a joint environment. However, testing these algorithms and setting up experiments is often difficult, mainly due to lack of realistic simulations. Quantifying simulation fidelity, using an engineering metrics approach, underpins the confidence in the successful completion of the conception-design-build-test/qualification-production-operation cycle of aircraft, yet has been neglected in the aeronautical world. For fixed wing aircraft, the concept of zero flight time training using flight simulation is accepted and deemed necessary from a safety and cost standpoint. This must become the modus operandi for rotorcraft training. Simulators are commonly used to assess handling qualities and to develop crew-station technologies. Attempts to quantify overall simulation fidelity within the framework of handling quality engineering have been presented in a number of forms in recent years. In [25] [26] [27], an approach based on pilot-aircraft modeling has been developed and the handling quality sensitivity function was introduced as the basis of a quality metric. In [28] and later in [29] the use of the handling quality standard (ADS-33E PRF 22) was proposed, for deriving metrics, the rationale here being that if the simulator is to be used to optimize handling qualities, then what better parameters to judge fidelity than those defining the predicted handling. In [30] and [31], an approach using comparative measures of performance and control activity, correlated with handling quality ratings given for the same tasks when flown in simulation and in flight, was presented. In all of these approaches, the philosophy has been to try to develop a rational and systematic approach to the identification of the differences between simulation and flight, hence directing attention towards areas of deficiency. The partial success of these methods is encouraging, but only serves to highlight the need for fidelity criteria for use in design, development and product qualification. In these areas, flight simulation can be a primary source of data from which knowledge is derived, decisions are made and significant resources are committed; similar arguments can be made for the development of flight training.

A Human-In-the-Loop project at ONERA

An assessment methodology for human-in-the-loop simulation is under development at ONERA, using available experimental platforms (Figure 8) that addresses the Spatial Disorientation (SD) phenomenon, where pilots experience erroneous sensations about their orientation. One situation in which SD can occur is the go-around procedure. In some circumstances, go-around can lead to a somatogravic (i.e., vestibular) illusion (also termed false-climb) prevalent during high accelerations (or decelerations) when a pilot has no clear visual reference [32]. The "illusion" is a strong pitching sensation up (or down) when the body is exposed to high accelerations (or decelerations) [33]. This illusion is due to the limitation of the vestibular system, which detects changes in orientation without differentiating between head tilt backward and forward acceleration.

How to optimally simulate self-motion using motion simulators is still an unsolved problem, despite the fact that self-motion simulation is an essential part of all commercial flight and driving simulators. Flight simulators used for pilot training and also most driving simulators strive to simulate motion trajectories that are considerably larger than the actual range of the physical simulator device. To do this, motion cueing algorithms attempt to mimic the accelerations that act on the body during self-motion. While a larger range of movement allows for more accurate motion cueing, increasing the number of degrees of freedom and enlarging the movement range of the simulator raises the costs of the device considerably, and there are also technical limits to what kind of trajectories can be performed in a simulator due to the limited motion envelope and actuator power. It is therefore important to find techniques to believably simulate large trajectories using smaller movements that are within the limited movement range of the simulator. Presenting believable physical accelerations is an issue for psychophysical experiments that investigate the perception of self-motion in motion simulators.

Conclusion and perspectives

In this paper, the state of the ONERA's simulation developments was presented. A component-based approach has been adopted to enable researchers and engineers to integrate their system developments as easily as possible into a common and stable software. For

researchers, this offers the possibility of increasing the maturity of their innovative solutions using a very flexible simulation tool. Moreover, thanks to the modularity of the simulator, the accomplishment of multidisciplinary projects is facilitated for system engineers. In particular, the simulation platform includes hybrid simulation involving humans and real components. This allows the gap between pure simulation and real experiments to be bridged.

Future work will consist in feeding innovative technologies into the hybrid simulation platforms at Onera, so that complex future systems can be designed and tested efficiently. Thus, the simulation platform will constitute a wonderful tool to demonstrate proof of concepts for future systems and to facilitate transfer of technology. For example, using all Onera technologies integrated into a simulation platform, future complex systems such as Reusable Launch Vehicles could be fully simulated and validated, including aerodynamic phenomena, navigation, guidance and control, specific sensors and actuators, etc.

Moreover, scientifically challenging problems still need to be explored, especially with regard to the management of simulation uncertainties: how can errors and inconsistencies be detected, and how can they be managed autonomously? Furthermore, the question of choosing the right granularity of models with respect to the assessed output remains open: both the real-time property and the output consistency need to be ensured at the same time. From a technical point of view, the simulation engine has to be improved to deal with studies that require very high precisions. The current engine is compliant with technical-operational studies, but for other studies improvements must be made in the numerical analysis domain. The Discrete Event System Specification (DEVS) is a formalism especially designed for the modeling and analysis of discrete event systems, as well as continuous state systems [34]. The idea is to manage the precision required by the experts in a transparent way. The engine will perform a fully integrated control of the time and of the events, in order to achieve the goal defined by the experts at the beginning of the experiment. ■

Acronyms

TRL	(Technology Readiness Level)
HLA	(High Level Architecture)
SD	(Spatial Disorientation)
HIL	(Hardware-In-the-Loop)
PIL	(Processor-In-the-Loop)
SIL	(Software-In-the-Loop)
MIL	(Model-In-the-Loop)
DCPS	(System Design and Performance Evaluation Department)

References

[1] P. CARLE *et al.* - *Simulation of Systems of Systems*. AersopaceLab Journal, 2012.

[2] R. CUISINIER, M. BRUNEL, and S. PRUDHOMME - *Using Open Source to Build Comprehensive Battlespace Simulations*. Proc. of SimTecT, 2010.

[3] https://fr.mathworks.com/products/simulink/.

[4] http://www.ni.com/labview/.

[5] http://www.windriver.com/products/vxworks/.

[6] http://www.ros.org/.

[7] E. CLARKE, D. KROENING, and F. LERDA - *A Tool for Checking ANSI-C Programs*. International Conference on Tools and Algorithms for the Construction and Analysis of Systems, March 2004, 168-176.

[8] J.S. DAHMANN - *The High Level Architecture and Beyond: Technology Challenges*. Proceedings of the Thirteenth Workshop on Parallel and Distributed Simulation, 1999, 64-70.

[9] IEEE 1516-2010. Standard for Modeling and Simulation High Level Architecture - Framework and Rules.

[10] IEEE 1516.1-2010. Standard for Modeling and Simulation High Level Architecture - Federate Interface Specification.

[11] J. BOURRELY, P. CARLE, M. BARAT, and F. LÉVY - *Genesis: an Integrated Platform for Designing and Developing HLA applications.* Proc. of Simulation Interoperability Workshop, 2005.

[12] IEEE 1516.2-2010. Standard for Modeling and Simulation High Level Architecture - Object Model Template (OMT) Specification.

[13] N.R. GANS, W.E. DIXON, R. LIND, and A. KURDILA - *A Hardware in the Loop Simulation Platform for Vision-Based Control of Unmanned Air Vehicles*, Mechatronics. vol. 19, 2009, 1043–1056.

[14] P. PHARPATARA, R. PEPY, B. HÉRISSÉ, and Y. BESTAOUI - *Missile Trajectory Shaping Using Sampling-based Path Planning*. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2013.

[15] R. BONALLI, B. HÉRISSÉ, and E. TRÉLAT - *Analytical Initialization of a Continuation-Based Indirect Method for Optimal Control of Endo-Atmospheric Launch Vehicle Systems*. IFAC World Congress, vol. to appear, 2017.

[16] A. MURANGIRA, C. MUSSO, K. DAHIA, and J.-M. ALLARD - R*obust Regularized Particle Filter for Terrain Navigation*. IEEE International Conference on Information Fusion (FUSION), 2011.

[17] N. MERLINGE, K. DAHIA, and H. PIET-LAHANIER - *A Box Regularized Particle Filter for Terrain Navigation with Highly Non-Linear Measurements*. IFAC Symposium on Automatic Control in Aerospace, 2016.

[18] E. TRÉLAT - *Optimal Control and Applications to Aerospace: Some Results and Challenges*. Journal of Optimization Theory and Applications, vol. 154, 2012, 713–758.

[19] https://www.altera.com/products/boards_and_kits/dev-kits/altera/kit-cyclone-v-soc.html.
[20] V. WIELS *et al.* - *Formal Verification of Critical Aerospace Software.* (4), p-1., AerospaceLab, 2012.
[21] J.F. BONNANS, J.C. GILBERT, C. LEMARÉCHAL, and C.A. SAGASTIZÁBAL - *Numerical Optimization: Theoretical and Practical Aspects.* Springer Science & Business Media, Ed., 2013.
[22] G.D. PADFIELD *et al.* - *Simulation Fidelity of Real-Time Helicopter Simulation Models*. 61st Annual Forum of the American Helicopter Society, 2005.
[23] S.J. HODGE, J.S. FORREST, G.D. PADFIELD, and M.D. WHITE - *Determining Fidelity Standards for Maritime Rotorcraft Simulation.* Maritime Operations of Rotorcraft. London: The Royal Aeronautical Society, 2008.
[24] V. SHIA *et al.* - *Semiautonomous Vehicular Control Using Driver Modeling*. IEEE Transactions on Intelligent Transportation Systems, 2014, 1-14.
[25] R.A. HESS - *Identification of Pilot-Vehicle Dynamics from Simulation and Flight Test*. Advances in Aerospace Systems Dynamics, vol. 31, 1990, 151-175.
[26] R.A. HESS and T. MALSBURY - *A Methodology for the Assessment of Manned Flight Simulator Fidelity*. Journal of Guidance, Control and Dynamics, vol. 14, 1991, 191-197.
[27] R.A. HESS and W. SIWAKOSIT - *Assessment of Flight Simulator Fidelity in Multiaxis Tasks Including Visual Cue Quality*. Journal of Aircraft, vol. 38, no. 4, 2001, 607-614.
[28] G.D. PADFIELD, M.T. CHARLTON, and A.T. MCCALLUM - *The Fidelity of Hi-Fi Lynx on the DERA Advanced Flight Simulator Using ADS-33 Handling Qualities Metrics*. DRA/AS/FDS/TR96103/1, 1996, 1-152.
[29] A.T. MCCALLUM and M.T. CHARLTON - *Structured Approach to Helicopter Simulator Acceptance,The Challenge of Realistic Rotorcraft Simulation*. RAeS conference, 2001.
[30] S.K. ADVANI and C.H. WILKINSON - *Dynamic Interface Modelling and Simulation-a Unique Challenge*. Society Conference on Helicopter Flight Simulation, 2001.
[31] M.F. ROSCOE and J.H. THOMPSON - *JSHIP's Dynamic Interface Modeling and Simulationsystem: a Simulation of the UH-60A Helicopter/LHA shipboard Environment Task*. 59th Annual Forum of the American Helicopter Society, 2003.
[32] T. WILSON - *Aircraft Human Performance & Limitations*. Civil Aviation Safety Authority, 1995.
[33] A.T KERN - *Flight Discipline*. McGraw Hill Professional, 1998.
[34] B.P. ZEIGLER, H. PRAEHOFER, and T.G. KIM - *Theory of Modelling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*. Academic Press, Ed. London, 2000.

AUTHORS

Bruno Hérissé received the Engineering degree and the Master degree in 2007 from the *École Supérieure d'Électricité* (Supélec). He obtained the Ph.D. degree from the University of Nice Sophia Antipolis in 2010. Since 2011, he has been a research engineer at ONERA, Palaiseau, France. His research interests include optimal control and vision-based control of aerial vehicles.

Gyslain Hervieux has been a research engineer in computer science at ONERA since 2006. He received his Engineering degree from "*Ecole Nationale Supérieure d'Arts et Métiers*" (ENSAM 2002) and his Advanced Master in Simulation and Virtual Reality from "Institut Image of Chalon sur Saône" (2006). His research interest is focused on computer simulation and on computer graphics.

Karim Dahia is a senior navigation engineer with the ONERA French Aerospace Lab in Palaiseau, France. His Ph.D. from the *Université Joseph Fourier* – Grenoble (France) focused on the application of particle filtering to aircraft motion estimation. Dr. Dahia's research interests include robust and optimal navigation as well as filtering for aerospace systems.

Jean-Michel Allard received his Engineering degree from the "*Ecole Nationale Supérieure d'Arts et Métiers*" (ENSAM 1996) and the "*Ecole Supérieure des Techniques Aérospatiales*" (ESTA 1997). He has been working at ONERA since 2001 as a research engineer for the System Design and Performance Evaluation Department (DCPS). His research interest is focused on inertial measurements hybridization and simulation for navigation of aircraft. He is carrying out expertises to the benefit of military programs managed by the Direction Générale de l'Armement (DGA) of the Ministry of Defence.

Jean-Christophe Sarrazin is currently Senior Research Scientist at ONERA. He received his PhD in 2003 in Health and Life Sciences from the University of the Mediterranean (Marseille, France). After two postdocs in computational neuroscience (postdoc INRIA, Nancy, France, one year) and cognitive neuroscience (postdoc Marie Curie, Université Libre de Bruxelles, two years) respectively, he obtained a permanent position at ONERA. In the Information Processing and System Branch, in which he coordinates the Human System Integration Team, he is a neuroergonomist and works on the scientific and technical development of human system integration studies. He works on the identification and the modelling of the computational principles of motor control and its modulation by high level cognitive functions. As he considers that a key to the development of HMI technologies lies in the integration of the neurosciences by industrials, he oeuvres at ONERA for the acquisition of neuroscientific knowledge and methods of investigation, specifying simulation scenarios, and the use of this newly found knowledge in improving design methods of new concepts of interaction.