CERTI: Evolutions of the ONERA RTI Prototype

Benoît Bréholée,
Pierre Siron
ONERA-CERT
Information Modelling and Processing Department
2, av E. Belin, BP 4025
F-31055 Toulouse Cedex
Benoit.Breholee@cert.fr, Pierre.Siron@cert.fr

Keywords:
HLA, RTI, distributed architecture, security, domains

ABSTRACT: *In this paper, we discuss some works carried out at ONERA (Office National d'Études et de Recherches Aérospatiales) in the distributed simulation field. Particularly, we describe the design and implementation of a RTI called CERTI. These studies started in 1996 and we summarize their present main results: an original architecture of the RTI prototype, security extensions, evolution to the version 1.3 of the HLA specifications and release as free software. Ongoing works concern mainly performance evaluation and prediction, introduction of the domain notion for large-scale simulations and significant applications using this prototype.*

1 Introduction

ONERA is a French governmental laboratory involved in aeronautic and spatial studies and researches. Therefore we are sharing the need, with a large community, for advanced simulation architectures. Examples of simulation studies are the design of new airports [1], the evaluation of new embedded systems, etc.

Consequently we have been very soon interested in the HLA project, a generic distributed architecture, which allows distributed discrete-event simulation studies. This interest is at the origin of the ONERA HLA initiative, that we propose to survey in this paper.

The main objectives of our initiative are the following:

- First of all, to get a better understanding of the HLA architecture from several points of view. For example, what kind of difficulties have to be overcome in designing and implementing the RTI from its specifications. Another important issue is to evaluate the underlying programming methodology of HLA in order to provide relevant support for the potential users inside ONERA.
- Secondly, to initiate new researches, suggested by performance issues and/or new simulation paradigms. We could here apply our skills in distributed systems and in computer security.
- Finally, to provide a HLA architecture with security properties, which is an important issue in both the defense and civil domains, as soon as several companies have to cooperate on the same project.

To achieve these objectives, we have conducted various studies, which we detail in the following chapters:

- Development of a RTI: CERTI. Actually we did not have access to the RTI executable code provided by the DMSO when we started this work.
- Study of the security problem, and implementation and evaluation of security extensions.
- Development of test applications, that are useful for HLA training and for the evaluation of performances of distributed simulations.
- Development of the notion of domains, which is an alternative and a complement to solve both the security problem and the *Data Distribution Management* problem.

These studies have been mainly funded by ONERA; particularly the development of CERTI, that we can consider as our main result. The security studies have

been funded by DGA.[1] ONERA is strongly linked to the French Ministry of Defense and our HLA skills have been largely distributed.

2 CERTI

2.1 Architecture

CERTI is a prototype of RTI developed at CERT.[2] This prototype has some original characteristics, in particular it is built around an architecture of communicating processes. The RTI is a distributed system involving

- a local process (RTIA)
- a global process (RTIG)
- a library (libRTI) linked with each federate

The RTI architecture is depicted by Figure 1.

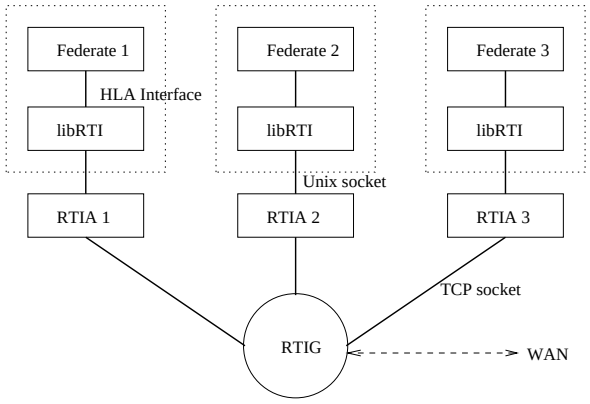

Figure 1: CERTI architecture

2.1.1 The RTIA

Each federate process interacts locally with a RTI Ambassador process (RTIA) through a Unix-domain sockets (this name is unfortunately confusing with the name of a class of the RTI API). The RTIA process exchanges messages over the network with the RTIG process, via TCP (and UDP) sockets, in order to run the various distributed algorithms associated with the RTI services.

The RTIA is always listening to both the federate and the network (the RTIG). It is never blocked because its response to the requests of the federate or to the network does not imply the reception of another message.

A specific role of the RTIA is to satisfy some federate requests immediately, while other requests require to send messages to the RTIG. The RTIA is receiving and sending messages concerning for example *Time Management* or *Declaration Management*, etc.

A message concerning the time management is a NULL message, as explained in [2]. In a first step, we have implemented this old but robust algorithm. With the time stamp included in this kind of message, the RTIA computes the LBTS[3] for the sender federate. This LBTS indicates the lower bound on the time stamp of any subsequent message a particular federate will receive from another federate. The RTIA computes the LBTS of its federate by taking into account the logical time and the lookahead of all the regulating federates in the federation.

A message concerning the object management is, in particular, associated to the implementation of the *Update Attribute Values*, *Reflect Attribute Values* †, *Send Interaction* and *Receive Interaction* † services. The *Reflect Attribute Values* † and *Receive Interaction* † messages are directed to two waiting files, one for the TSO[4] messages and the other one for the RO[5] messages. This is the function of the RTIA to manage the memory allocation, and the fact that RTIA and federate memories are separate is an aspect of the security.

2.1.2 The RTIG

The first main function of the RTIG, or RTI Gateway, is to manage the communications between the RTIAs, and therefore between the federates. In the reliable mode, this is an obliged passing point between two federates, for example $RTIA_i$ and $RTIA_j$. In fact, we do not want to use a direct TCP connection between $RTIA_i$ and $RTIA_j$ but two TCP connections: one from $RTIA_i$ to the RTIG, the other one from the RTIG to $RTIA_j$. Globally we have n TCP connections, if n is

[1] State Organization responsible for Armament Programmes
[2] Research Centre of Toulouse of ONERA
[3] Lower Bound on Time Stamps
[4] Time-Stamped Order
[5] Receive Order

the number of federates.

For distributed simulations with a large number of federates, which are often allocated to different local networks connected by a WAN, our project is to multiply the RTIG and to implement the necessary routing of messages in the logical network of connected RTIGs.

The second main function of the RTIG is to simplify the implementation of some HLA services, because it is a centralization point in the architecture. It manages the creation and the destruction of federation executions. It records the identity of federates willing to publish the attributes of an object class (resp. an interaction class) or to subscribe to the attributes of an object class (resp. an interaction class). The RTIG uses these data to forward messages from a RTIA in a software multicast approach. We have so an emulation of a reliable multicast service.

2.1.3 The libRTI library

This is a little library in which HLA service calls are transformed into messages sent to the RTIA. Messages are built (this includes a type and input parameters) and sent to the associated RTIA. The service execution then waits for a response from the RTIA (usually an acknowledgment) and provides the output parameters.

A *tick* primitive is added, allowing the execution of the RTI initiated services (user code associated to *Time Advance Grant* †, *Reflect Attribute Values* †, etc.) This *tick* function has no parameter, it causes the execution of one callback or it returns immediately if the RTIA cannot deliver anything (in this case, the federate has often to wait).

The allocation of the CPU resource to the federate and the RTIA processes is exclusively managed by the operating system. We make the hypothesis that the user can remain unaware of this problem (well exposed in [3]).

A relatively easy optimization would be to replace the communication by Unix socket between the libRTI and the RTIA by a communication using shared memory. Nevertheless, our performance measurements do not show that this communication is a bottleneck in the architecture. We are much more hesitating to transform our multiprocessing approach into a multithreading approach.

2.1.4 Data-transfer scenario

Figure 2 illustrates the exchange of messages involved in *Update Attribute Values* (this is like a data transfer). The message file is not represented in the right-side RTIA, nor the delivery condition which depends on the time management. UAV is the acronym for *Update Attribute Values*, and RAV refers to *Reflect Attribute Values* †.

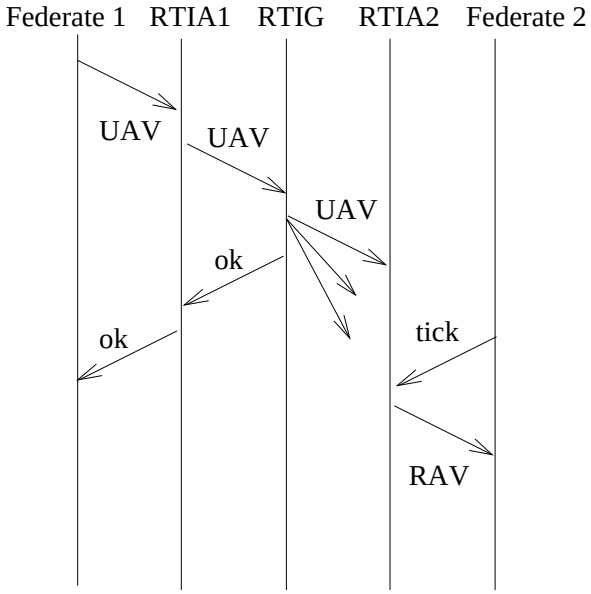

Figure 2: Data transfer scenario

2.2 Schedule

A small team was set up to produce CERTI. The project started by the end of 1996. A first version [4] was available in September 1997. Our objective was a rapid prototyping, therefore we had selected a reduced set of HLA services, which are enough to build significative applications. *Data Distribution Management* and *Ownership Management* were not implemented.

A second version was achieved in May 1998, which included some optimizations. The reliable mode of communication has been privileged, because our main applications are constructive simulations (simulation

for the engineer in coordinated time with conservative option). We have reduced the number and the size of the exchanged messages and we have chosen better options of the TCP protocol.

The third version integrated security mechanisms (see 3). The second and third versions were robust and efficient enough for us to develop a fair number of applications. But these versions were only compliant with the version 1.1 of the HLA specifications.

The fourth version, in 2000, adapts the implemented services of HLA to the version 1.3 of the specifications. This work concerns mainly a new version of the libRTI.

The fifth version (2001) integrates the *Ownership Management*, and we are currently working on the *Data Distribution Management*.

The spirit of this incremental work must be recalled. We are mainly concerned by understanding and following the philosophy of HLA. Our goal is not to achieve a commercial product, nor to immediately satisfy a DMSO HLA verification process. But we can write distributed simulations with CERTI in a well controlled environment, and adaptation effort to use the RTI-NG, for example, is minimal.

2.3 Portability

The C++ language is used for the CERTI code and the development of federates. We are using standard protocols, such as TCP/IP, for the communications between the different components. We are also using standard Unix libraries for the processes management.

With these assumptions, we have very few portability problems, and CERTI is currently running on workstations (Sun Solaris, SGI, HP), PC, and clusters of PC (Linux).

2.4 Release

We are planning to release CERTI as free software. As previously said, our goal is not commercial and the free software development model is usually efficient to improve portability and code quality.

CERTI will be released under a copyleft free software license (GNU GPL for the programs, GNU LGPL for the libraries) that doesn't prevent proprietary federates to link against the libRTI. Details on these licenses can be found on the GNU web site.[6]

The home page is at http://www.cert.fr/CERTI/. The project development web site is currently at http://savannah.nongnu.org/projects/certi/.

3 Security Aspects

3.1 Compatibility with existing infrastructures

Firewalls are often used for low risk environments. They provide controlled and audited services both from inside and outside a private network, by allowing, denying or redirecting the flow of data. Our problem is to run distributed simulations between different laboratories protected by firewalls.

An application level firewall employs proxies to screen traffic. We cannot develop a new proxy, dedicated to HLA and CERTI. That would be difficult to evaluate and to accept by the network and security administrators of the laboratories. But, as we know very well the communications needed by CERTI, we can ask for a simple TCP proxy, that allows the TCP connection between the RTIA process running in a first laboratory, and the RTIG process running in another laboratory at the known TCP port and host.

This TCP proxy exists soon for the majority of commercial firewalls and our request has been accepted by our administrators. Experiments have been made between three laboratories : IRIT, LAAS and ONERA.

3.2 Security extensions

Work on distributed simulation security was conducted for the DGA, in particular in the framework of the SAIDA[7] project in cooperation with DERA (UK). It was aimed at providing an answer to problems of inter-company simulation, for which an unlimited information exchange must take place via the RTI between the federates of two different companies to preserve the confidentiality of their know-how. (Fig. 3)

[6] `http://www.gnu.org/licenses/licenses.html`

[7] Security Assurance In Distribution Applications

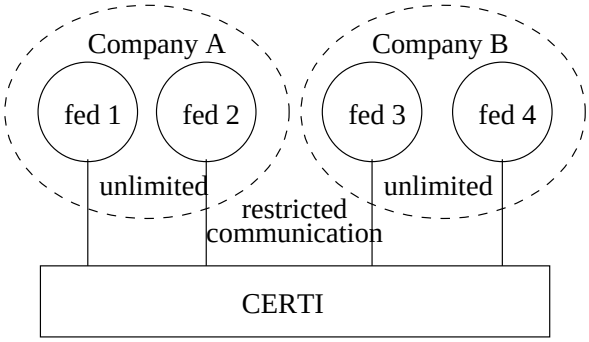

Figure 3: Inter-company simulation

We have conducted a complete security study [5] : threat analysis, definition of security objectives and security functions, implementation. The result is a security architecture based on CERTI and the trusted third party principle. We suppose that the RTIG process is under the control of a third party that is not in competition with the companies owning the federates, for instance a public organization, and we have implemented secure communications and access control mechanisms.

3.2.1 Secure communications

To secure communications between remote federates and the RTIG we use the GSSAPI.[8] This interface hides from its callers the details of the specific underlying security mechanism, leading to better application portability, and moving generally in the direction of a better interworking capability. We have used the implementation of [6].

3.2.2 Access control mechanisms

We propose to extend the description of the federation (the `FED` file) with security attributes. Each class, attribute and federate is associated with a security domain. Then the RTI filters the message according to the security domains of the object and of the federate (this is an application of research on multilevel security.)

This control is performed by the RTIG because, in our architecture, this component is already in charge of recording the publication and subscription. So the RTIG will now check the security labels of the federate and of the class whenever this federate has sent a subscription message for this class. The RTIG will record for each published class a list of authorized subscribers. As the RTIG transmits the *Update Attribute Values* messages only to authorized subscriber RTIAs, a federate from one company will never receive *Reflect Attribute Values* † messages for a private object of another company because its subscription request are blocked by the security label control in the RTIG. In that case, in accordance with HLA rule, any federate are informed of the updates on any object belonging to any class in the FOM.

We have studied the impact of these security mechanisms on the federation real-time behavior [7]. This impact is very weak for the access controls.

4 CERTI Applications

The first test application is a billiard game. There is one main object class, the ball with position attributes. The federation is composed of any number of federates, each federate modelizing and simulating only one ball instance. It publishes and subscribes to the position attributes, so that each computer could graphically represent the current situation. The collisions are simulated by interactions. This is a time-coordinated example, with an initial synchronization point.

With this example, the performances are bounded by the performances of the RTI and the physical architecture. A good criterion is the number of simulation steps per second. Figure 4 indicates performance obtained with CERTI and the RTI provided by the DMSO.

The two curves are similar. We do not explain why CERTI seems faster. This is a rapid comparison and, probably, we have not reached the best results for the RTI-NG on our physical architecture (the configuration file of this RTI has a lot of parameters).

Another application [8] is a fight simulation of aircraft attacking defense units. Patrols of aircraft's are equipped with anti-radar missiles. Air defense units are composed of a command post supervising multifunction radar devices and surface-to-air missile ramps.

[8] Generic Security Services Application Program Interface

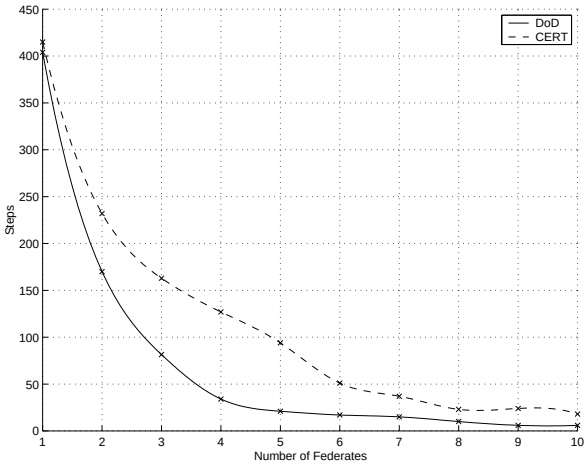

Figure 4: Performances of the billiard federation

We have chosen this problem because it is more significative and a little more complex. The object classes are Aircraft, Radar and Missile. The interactions are Radar Emission, Destruction, etc. The federates are associated with Aircraft's, the Defense Units (which both simulate the Missiles) and a Monitor like in the previous example.

With this application, we have investigated the multiresolution problem. [9] details the multiresolution of entities : patrols and aircraft's, and discusses the chosen mechanisms allowing triggering aggregation from an entity-level representation, and conversely, triggering disaggregation from an aggregate representation. A multiresolution management service has been proposed.

More complex distributed simulations are simulations of airports of the future [1], and simulations of radar detection of missiles and drones designed by the Departments of Physics of ONERA (a paper has to be written).

5 Domains and HLA

5.1 Concept of domain

Several on-going works concern the introduction of the domain concept in CERTI and/or in federation executions. These works are mainly motivated by the scalability and the security problems.

Domains provide a solution to data distribution management with the specification of interest regions, allowing communication optimizations. Data filtering is used with relevance criteria, but other implementations of domains can provide data filtering for security purposes. This general domain concept appears in several projects:

Interfederations They are the connection of several federations. These federations are chosen (or designed) depending on the objectives of the global interfederation (for example security, scalability, or interoperability). The federations are linked to each other with one or more bridge federate (see 5.3).

Distributed RTIG In the current architecture, optimizations can be done in the RTIA, thus preventing useless messages to reach the RTIG. This is for example the case of *Declaration Management* services and the subscription/publication mechanism. But one of our main concerns is that despite optimizations carried out in the previous versions of CERTI, the RTIG is still a significant bottleneck. With network architecture, it is the main obstacle to large-scale simulations. A distributed RTIG would add a domain level between local treatments (in RTIAs) and global mechanisms (involving every RTIG component), and should improve federation executions.

DDM development The *Data Distribution Management* services of the HLA propose the use of domains with the definition of routing spaces. This has to be implemented in CERTI (see 5.2).

Multicast Multicast is used to send messages to multiple recipients: the interest is that the task to route data is handled by the underlying network architecture. Possible applications of multicast concern messages sent from the RTIG to the RTIA, messages between RTIG components in the case of the above distributed RTIG. And of course *Data Distribution Management* implementations can use multicast to group recipients by interest region.

5.2 Data Distribution Management

The purpose of HLA *Data Distribution Management* services is to reduce the amount of irrelevant data

exchanged between federates. Such relevance filtering reduces both traffic over the network and the number of messages federates have to filter. *Data Distribution Management* services allow federates to refine the publication/subscription information they provide to the RTI. Without these services (ie, using only *Declaration Management* publication and subscription), a federate subscribed to some object class attribute receives all information related to this attribute. *Data Distribution Management* introduces relevance of data a the attribute instance level.

The principle is still a publication/subscription mechanism similar to *Declaration Management* services, but parameters help federates to refine the description of the information they are interested in. Such descriptions are possible thanks to the concept of routing spaces. Routing spaces are defined by their dimensions, which correspond to attributes existing in the federation. Federates subscribe to particular regions of routing spaces, thus defining the only ranges of data values they are interested in.

Routing spaces help to reduce communications by preventing lots of irrelevant data to be transfered, but they have to be managed in the RTI which requires significant resources. In large-scale simulations, very numerous regions represent a time cost for the RTIG and mean additionnal message traffic over the network.

Therefore reducing the number of routing regions is often an interesting compromise between *Data Distribution Management* optimizations and run time overhead. In particular, this is necessary in *Data Distribution Management* implementations using multicast groups to send messages to multiple recipients: since the number of multicast groups is limited, it's not possible to associate one group with each region in large-scale federations. Several approaches and alternatives to the traditionnal fixed-grid implementation are proposed in [10].

We do not plan to use multicast groups in our first implementation of *Data Distribution Management* services in CERTI: the objective is to implement the routing regions just as they are defined in the HLA, without simplifying or grouping them. More complex implementations with this kind of management improvement are scheduled for later versions of CERTI.

Actually, we are focusing on taking advantage of the architecture of CERTI. Optimization with *Data Distribution Management* depends heavily on the coordination between the RTI internal strategies and the federation design (particularly the use of regions). With CERTI, region management can be carried out both in RTIG and RTIA.

For example, managing *Data Distribution Management* only in the RTIG helps to reduce irrelevant messages sent to RTIAs. But another approach is to partially manage regions in the RTIA. If the RTIA is aware of all the subscription information, then it can filter update messages. The inconvenience is that RTIAs have to get informed of all subscription region modifications.

5.3 Bridge Federates

A HLA bridge federate is an application that connects two or more federations, and is in charge of representing each federation to the others. There are many variants depending on the objectives, for example a bridge can represent a federation only partially, to create private objects not seen in other federations.

A common design of bridge federates is based on a core component linked with federate components; each of these federates joins one federation. Then the general behavior of the interfederation should be the same as a federation based on the same federates, using only one RTI (there are exceptions, depending on the objectives of the bridge). (Fig. 5)

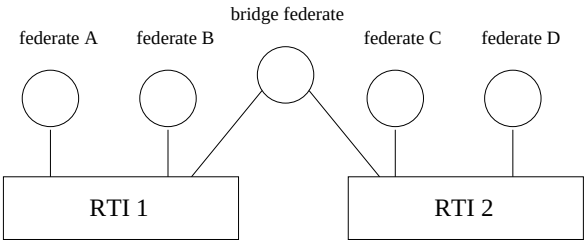

Figure 5: Interfederation architecture

The bridge can be used for several purposes:

Interoperability A bridged federation can use several RTIs. This is useful if some federates are optimized for one particular RTI. It is also a solution to some language binding problems. In the case of

CERTI, only C++ si supported, which prevents federates written in another language to use it. With a bridge federate, it's possible to use another RTI for this kind of federate, and to link them to a CERTI-based execution.

Optimization In some simulations, the execution can be improved by running homogeneous federates in their own federations, and bridging the federations. This is the case when there are very dissimilar federates (considering SOMs or event rates) as explained in [11]. We can also think of the use of fixed grids in *Data Distribution Management*: one problem is that the grid resolution depends on the federate scale and behaviour, which may greatly differ, making it impossible to choose an optimal granularity. Homogenous federates can be grouped. Then the simulation uses several grid resolutions, one for each federation.

Security Another use of the bridge federate is similar to the inter-company simulations problem described in 3.2. It's possible for different parties to participate to a unique simulation, based on several federations, each of which may have private parts: the only need is to share a common FOM, but inside one federation, the FOM can be larger and include attributes that will be filtered out by a bridge federate.

So the general bridge federate architecture consists of a central component and several federates. Another important component is a translator, in charge of associating the identities of every entity existing in more than one federation. For example, an attribute id or a save label, may differ from one federation to another.

[12] describes the way a bridge can translate HLA mechanisms from one federation to another, so that from a federate view, everything seems to be held in the same manner as in a single federation. One must remember that the bridge is not a part of a RTI or any other privileged entity of the simulation: it is an HLA application, and can only use the services available to federates. This is why some direct and unconditional mechanisms such as updating/reflecting attributes are possible through the bridge with the usual services, while some global mechanisms require the use of the MOM.[9]

[9] Management Object Model

For example, the MOM is usually required to handle an interfederation save process: the bridge monitors the MOM to get each federate save response. If the save succeeds in every federate of a federation, then the bridge declares itself as correctly saved to the other federations.

At first, we didn't consider the use of the MOM in a bridge since it is not implemented in CERTI yet. We considered some restrictions on the federations to make the bridge work without the information provided by the MOM. For example, we can use the save mechanism over a bridge if the only federate to use the result of the save process is the one which sent the save request: the bridge just has to declare itself saved in other federations, then the result of the federation save may be wrong for some federates, but is always correct in the federation were the save request was initiated. Other kinds of mechanisms such as publication/subsription can be handled without the MOM. Actually, the resulting optimization may be very poor if, for example, the bridge publishes and subscribes to every object and interaction class. But this doesn't prevent the interfederation to be executed, which can be useful when the purpose is interoperability. The goal of this first implementation is just to allow federates to use different RTIs in an interfederation.

Of course, a significant improvement is the implementation of the MOM in CERTI to handle global mechanisms. We are also particularly interested in another interfederation architecture, using several cooperating bridges, as detailed in [11].

6 Summary

In this paper we described the evolution of CERTI and its associated projects. We recalled the original architecture of this prototype and the project history.

We consider CERTI as our main result, but aside from the development of this RTI towards the HLA specifications, CERTI provided a distributed simulation platform for many works. Particularly, studies concerning security extensions have been carried out. Also, test applications have been developed.

Eventually, several on-going works focus on the im-

provement of the use of domains in federation executions: in particular, the implementation of *Data Distribution Management* services and the development of a HLA bridge federate are considered to improve interoperability, security and scalabillity.

Acknowledgments

We could not complete this paper without naming all the contributors to this project: M. Adelantado, P. Bieber, S. Bonnet, B. Bréholée, P. Desseaux, F. Fayet, A. Harzi, Ph. Hautesserres, S. Lemanceau, P. Nortier, S. Prunet, P. Siron and G. Zanon.

References

[1] M. Adelantado. Experimenting the HLA framework for the ONERA Project Airport of the Future. In *Proceedings of the 1999 Fall Simulation Interoperability Workshop*, 1999.

[2] K. M. Chandy and J. Misra. Distributed Simulations: A Case Study in Design and Verification of Distributed Programs. In *IEEE Transactions on Software Engineering*, September 1979.

[3] R. D. Wuerfel. RTI-NG Process Model. In *Proceedings of the 2001 Spring Simulation Interoperability Workshop*, 2001.

[4] P. Siron. Design and Implementation of a HLA RTI Prototype at ONERA. In *Proceedings of the 1998 Fall Simulation Interoperability Workshop*, 1998.

[5] P. Bieber, J. Cazin, P. Siron, and G. Zanon. Security Extensions to ONERA HLA RTI Prototype. In *Proceedings of the 1998 Fall Simulation Interoperability Workshop*, 1998.

[6] D. P. Barton and L. J. O'Connor. Implementing Generic Security Services in a Distributed Environment. Technical report, CRC for Distributed Technology, April 1995.

[7] P. Bieber and P. Siron. Design and Implementation of a Distributed Interactive Simulation Security Architecture. In *Proceedings of the 3rd IEEE Int. Workshop on Distributed Interactive Simulation and Real-Time Applications*, October 1999.

[8] M. Adelantado and P. Siron. Air-Ground Combat Simulation through the ONERA HLA Run-Time Infrastructure. In *Proceedings of the Advanced Simulation Technologies Conference*, April 2000.

[9] M. Adelantado and P. Siron. Multiresolution Modeling and Simulation of an Air-Ground Combat Application. In *Proceedings of the 2001 Spring Simulation Interoperability Workshop*, 2001.

[10] A. Berrached, M. Beheshti, O. Sirisaengtaksin, and A. deKorvin. Alternative Approaches to Multicast Group Allocation in HLA Data Distribution. In *Proceedings of the 1998 Spring Simulation Interoperability Workshop*, 1998.

[11] T. Lake. Time Management Over Inter-Federation Bridges. In *Proceedings of the 1998 Fall Simulation Interoperability Workshop*, 1998.

[12] W. Braudaway. The High Level Architecture's Bridge Federate. In *Proceedings of the 1997 Fall Simulation Interoperability Workshop*, 1997.

Author Biographies

BENOÎT BRÉHOLÉE was graduate from a french engineer school of aeronautics and computer science in 1999. Then he is a PhD student at ONERA. He is a member of the Design and Validation of Computer Systems research unit.

PIERRE SIRON was graduate from a french engineer school of computer science in 1980, and received his doctorate in 1984. Then he is a Research Engineer at ONERA and he works in parallel and distributed systems and computer security. He is a member of the Design and Validation of Computer Systems research unit.