CERTI, an Open Source RTI, why and how

Eric Noulard
Jean-Yves Rousselot
ONERA/DTIM/SER
Centre de Toulouse
2 avenue Edouard Belin
31055 Toulouse Cedex
France
rousselot@cert.fr, eric.noulard@onera.fr

Pierre Siron
Université de Toulouse, ISAE
10 avenue E. Belin
31055 Toulouse Cedex
France
pierre.siron@isae.fr

Keywords:
Open Source, RTI, Distributed Simulation, Collaborative development.

ABSTRACT: *CERTI is an HLA RTI developed since 1996 by ONERA, the French Aerospace Lab. The initial purpose of CERTI was to develop a home made RTI in order to: learn HLA usage and HLA RTI internals (e.g. time management), 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...). CERTI became open source in 2002: https://savannah.nongnu.org/projects/certi. Since then, Open Source CERTI project has had variable activity periods, mostly driven by research project needs and funds. CERTI development has started again since the end of 2006, with an increased interest from the open source user community. After a brief status survey of CERTI, this presentation will focus on the Open Source objectives of CERTI and explain why this is not a product but a project driven OSS initiative, pushed by a Public establishment like ONERA. We will further explain how open sourceness CERTI stimulates its development and the community itself and why every stakeholder benefits from this.*

1. Introduction

There is a certain amount of non-commercial HLA RTIs out there [1], but many of them seem discontinued or do not have native C++ support. Is this due to the fact that commercial ones are mature, which lowers the need for non-commercial ones which were mostly developed for research purposes? Is it hard to maintain a working non-commercial RTI? ONERA, the French Aerospace Lab manages the development of one of those non-commercial RTIs, the CERTI, which is an Open Source RTI available on various platforms. We would like to share here our past and historical knowledge, as well as ideas for the future of non-commercial RTIs.

After a historical survey of CERTI and its current status, we will explain why CERTI is an Open Source project and how it all works.

2. CERTI Project History and Status

This first chapter gives a general idea of the CERTI initiative. It includes the presentation of different projects of distributed simulation at ONERA. Many results have already been published but we will emphasize, for each project, how the availability of CERTI has been useful and necessary for us as well as the connection with various research domains.
Another important aspect is that past results should not be forgotten as they can still be valid regarding the different requirements that could have distributed simulations.

2.1 CERTI project history

CERTI project started back in 1996 when ONERA wanted to continue research work on distributed systems [2]. We wanted to study distributed simulation itself as the primary research objective. We wanted to learn, study and experiment with HLA. So we decided to build our own RTI.

Let's not forget that at the beginning of this project the DMSO RTI was not available and that during the project the development and distribution of the DMSO RTI NG were stopped. For the moment, the status of CERTI is stable, thanks mainly to the support of ONERA.

CERTI is recognizable through its original architecture of communicating processes. The RTI is a distributed system involving two processes, a local one (RTIA) and a global one (RTIG), as well as a library (libRTI) linked with each federate. The RTI architecture is depicted in Figure 1.

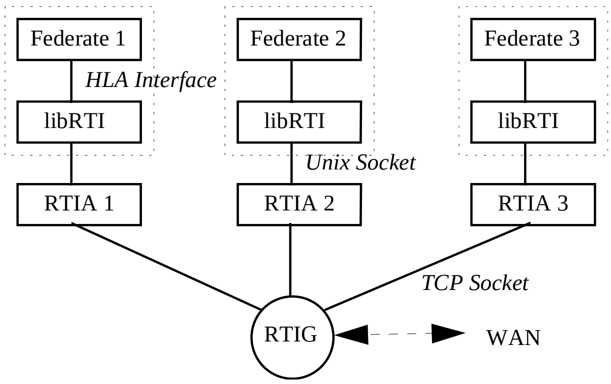

Figure 1: CERTI architecture

Each federate process interacts locally with an RTI Ambassador process (RTIA) through a Unix-domain socket. This point evolved when we ported CERTI to Windows systems and on multiprocessor architectures. The RTIA processes exchange messages over the network, in particular with the RTIG process, via TCP (and UDP) sockets, in order to run the various distributed algorithms associated with the RTI services.

A specific role of the RTIA is to immediately satisfy some federate requests, while other requests require network message sending or receiving. The RTIA manages memory allocation for the message FIFOs and always listens to both the federate and the network (the RTIG). It is never blocked because the required computation time is reduced. It also plays a great role in the implementation of the tick function.

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.

Based on this architecture, we started with the implementation of the federation management, object management and time management services. We had a concrete application of theory of distributed algorithmic and distributed simulation (from the Chandy, Misra and Bryant algorithm to the Fujimoto zero lookahead algorithm). We really hope that we have brought a little contribution to these research domains. Obviously we did not have the opportunity to apply new reliable multicast protocols developed by the networking community.

The first prototype was a success for us. Despite the distribution of commercial products, we have pursued its development and extension with respect to the HLA standard. CERTI guarantees a forward compatibility: a federate developed and tested with CERTI will be compatible with a certified RTI even if all the services are not yet implemented (some HLA services were not necessary for our usage). CERTI has been extensively used in the projects summarized in the following paragraphs.

2.2 Security of distributed simulation

An expected use of HLA/RTI [1] is to allow simulations developed by various companies to interoperate. However, some firms are reluctant to join an HLA federation because they fear that some confidential data could leak to their competitors. Hence, there is a need for an HLA/RTI that guarantees secure interoperation of simulations belonging to various mutually suspicious organizations.

Although we have carried out a complete security analysis (threat analysis, definition of security objectives and security functions, etc.), we will only mention here the implemented security architecture.

We have implemented TTP (Trusted Third Party) security architecture. At the core is a local area network operated by the trusted third party which includes a machine that contains the RTIG process. A company may trust the federate processes it has written to behave correctly with respect to security concerns. It might also trust some components of the RTI and in particular the RTIG. But a company would certainly not trust federate components developed by other companies.

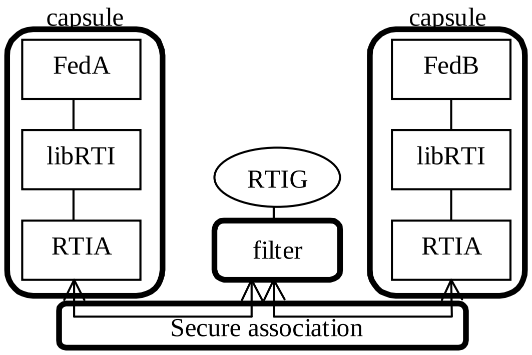

Figure 2: Security architecture

There is no communication between company machines except those mediated and authorized by the RTIG. The description of the federation (FOM) must be completed to include security domains. Security domains we have considered include PrivateX where X is the name of a company and Public. Security domains may be organized hierarchically.

The extension to RTI services is very limited. We propose to add security domain filters for the publication and subscription services (messages PublishObjectClass, PublishInteractionClass, SubscribeObjectClass and SubscribeInteractionClass). These messages are erased whenever the security domain of the requesting federate does not dominate and is not equal to the security domain of the requested class. As the RTIG transmits UpdateAttributeValue messages only to authorized subscriber RTIAs, a federate from one company will never receive ReflectAttributeValue messages for a private object of another company.

To secure communication between remote federates and the RTIG we use the Generic Security Services Application Program Interface (GSS-API) [3]. 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.

Mastering the code and the architecture of an RTI was essential:

- To make communications secure or to go through existing security mechanisms (firewalls for example).
- To add some access control mechanisms.
- To do some code analysis (and to avoid Trojan horses).

The link with researches in the security domain was obvious (security of distributed systems, code analysis and multi-level security policies).

2.3 Multi-resolution

Conventional simulations represent entities at just one level of resolution. Multi-resolution representation of entities consists in maintaining multiple and concurrent representations of entities. We tackled the problem of how HLA services may allow multi-resolution modeling and simulation to be achieved. Our goal was not to provide a general framework as a basis for designing concurrent simulations of entities at different levels of resolution. We focused on experience feedback that we obtained by migrating a single-level resolution HLA federation to a multi-level resolution federation. The selected application is an Air-Ground Combat simulation involving aggregated patrols of aircraft engaged against a surface to air defense system.

Aggregate and disaggregate levels are respectively shown in Figure 3 and Figure 4. At the aggregate level, a patrol of aircraft has to attack a set of ground radars. On entering the engagement area, the patrol automatically disaggregates into its individual entities. The engagement is then managed according to the rules described for the single resolution application.

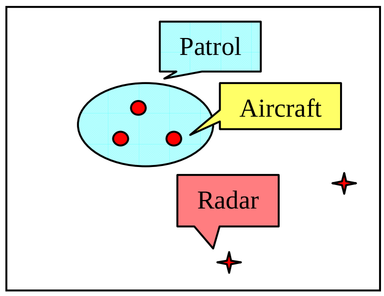

Figure 3: Aggregate level

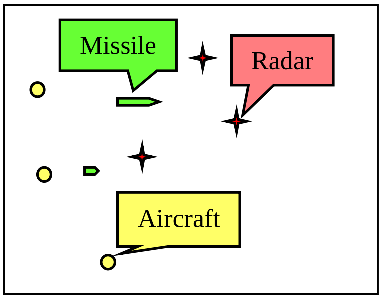

Figure 4: Disaggregate level

We have tested a centralized approach (a single federate handles the patrol and its aircraft) and a fully distributed approach (one federate per entity). The first results showed that a fully distributed approach facilitates the migration from a single-level resolution application to a multi-level resolution one, in that the underlying models of the components can be directly re-used.

Although HLA interactions are useful and sufficient to implement the communications between federates that are dedicated to the multi-resolution management, they are too low-level oriented. Therefore we are investigating HLA-based higher level services, encapsulating both aggregation/disaggregation interactions and transfer of control from one level of resolution to another.

With CERTI we have added these services to our library but we have implemented them by using the standard HLA services.

This was an interesting subject of distributed algorithm, specifically the way the aggregation was produced (a rendez-vous problem).

2.4 High-performance simulation

While HLA was initially designed to support fully distributed simulation applications, it provides a promising framework for composing not necessarily distributed simulations, from existing reusable components.

Simulation composability allows users to construct federations from a set of communicating components according to the needs of the decision makers. Composability provides a means to build integrated simulation platforms with increased coverage of decision support. In such simulation applications, distribution becomes a means to achieve high-performance computing, while remaining a constraint since existing components are reused.

To face the aforementioned performance and availability requirements, ONERA has designed the HP-CERTI package [4], an optimized version of CERTI, including two main development issues. The first one, named SHM-CERTI, deals with a shared memory communication scheme between RTIG and RTIAs, in order to achieve high-performance simulation of federations running on the same shared memory execution platform. The objective of the second issue is to increase both availability and performance of composable simulations running on high-performance cluster platforms. In the SHM-CERTI architecture, TCP socket based communications between the RTIG and RTIAs are replaced by shared memory segments.

We have checked the performance optimization gained with classical benchmarks and especially with a pilot application of distributed cooperative simulation under HLA. This last application, named SICODIS [5], has involved various departments from ONERA: Automatics, Computer Science and Physics. The subject was the study of a new passive radar concept and its evaluation for the recognition of different objects in different scenarios. HLA has been a good tool, making communication and work easier between departments and research groups with very different skills. Moreover, the developed federation can be considered a tool with satisfying performances.

The availability of CERTI has made possible its porting and optimization on various multiprocessor architectures and the use of the latest research results in the parallelism domain.

2.5 Hard real-time simulation

ONERA and CNES (the French space agency) had common projects of new spatial systems, in particular projects of in-formation flight of satellites. To simulate these distributed systems, we could do distributed simulations. This would allow us to re-use existing simulators; however, we have new requirements for HLA: hard real-time requirements.

In these new federations, each federate is time-stepped driven. The constraints are to respect the deadlines of each step and to synchronize the different steps of the different federates. The time step of the more complex federate is 5 ms. Satellites obviously have to communicate to maintain their relative position or for their payload. So there will be HLA data communication between federates that are to be achieved every step.

CERTI and Linux operating system have been chosen to perform many experiments, to define a simulation architecture and have some guidelines for this new type (at least for us) of HLA simulation.

New real-time mechanisms and practices were necessary [6] to meet real-time requirements:

- For the operating system, we are setting a real-time scheduling, locking memory pages, using real-time timers in a new preemptible kernel.
- For CERTI, we have changed the implementation of the tick function.
- For the federates, we have found that the use of the time management services is more efficient for these real time simulations (the use of the real-time term in HLA should change…).

The different interactions between the application level, the RTI level and the operating system level are very difficult to understand but are essential to a successful study. The availability of CERTI implementation was a key point to understand global scheduling and to adapt one function that impacts this scheduling.

We are still working on this subject, in order to extend the use of HLA (and CERTI) to the study of new embedded systems, in parallel with researches in the real time domain.

2.6 CERTI current status

CERTI is an Open Source project whose forge is hosted at Savannah [7] and project home is at ONERA [8]. CERTI software includes an HLA 1.3 RTI which is compatible with the latest RTIG-NG release with some missing services (MOM and part of Notification Service). CERTI currently runs on several platforms including various flavors of Linux, Microsoft Windows, Solaris, FreeBSD and IRIX. The CERTI RTI aims at HLA Evolved support [9] including SISO DLC compliance [10] (or EDLC when available). The level of completeness will depend on projects demands and contributions.

As briefly explained later, CERTI is not a ***product*** to be compared with commercial RTI; it is an ***Open Source Project*** structured around a user community which thus provides other software components or companions projects and other kinds of services.

3. Open Source CERTI Project

CERTI is not only some freely available software. It is an ***Open Source project*** hosted on the Savannah [7] free software forge. We will explain hereafter why CERTI is an Open Source project and how it works for users and contributors.

3.1 Home-made, then Open Source CERTI - Why?

As shown in CERTI history, §2.1, it was both necessary and natural for ONERA to have its own RTI implementation in order to be able to pursue its research activities in the distributed simulation domain. Note that having source code control capability over the RTI is a requirement that has been shared by others in the past [11] [12]. The reasons range from being able to add slight modifications in order to support a new platform up to have full understanding of the RTI internals in order to tune/optimize the federation execution for performance [6].

We can imagine other future research needs like studying the possibility of an HLA RTI as embedded middleware, experimenting with different implementations of HLA Evolved SURR [13] in order to totally constrain the bandwidth, adding more fine-grain QoS delivery than Reliable or BestEffort... For all those test cases, we need to have our own RTI implementation. Moreover, many such projects usually do not need a fully HLA-compliant RTI but a tunable one which includes all services mandatory for the particular usage.

The next question is: *Why should we be interested in going Open Source?*

Back in 2002, when CERTI was open sourced (from contribution of Benoit Bréholée [14]) the immediate and foreseen reasons were:

1. It was easier to exchange CERTI between different project stakeholders,
2. CERTI could be freely used for teaching purposes in universities,
3. Students and PhDs could easily contribute using collaborative tools and open licenses.

Recently, two ONERA projects brought new development activities into CERTI. The first one was the joint CNES/ONERA formation flying satellite program [5], that needs an RTI with real-time capabilities, and more recently the IESTA project [15], an infrastructure capable of evaluating new air transport system concepts. For the latter, ONERA wanted to ease the use of the underlying HLA RTI for all stakeholders; it therefore suggested its Open Source RTI as a possible solution which does not prevent the usage of other RTIs.

With these needs in mind, we decided to re-stimulate the CERTI Open Source community through the open source Savannah forge [7]. After a little less than two years of standard project activities, we have seen noticeable open source contributions coming from outside of ONERA project CERTI users:

- A small and self-explaining HLA Tutorial application,
- Parts of an embryo of an HLA Test Suite,
- A Matlab/HLA13 binding [16],
- A Fortran90/HLA13 binding [16],
- A FlightGear/HLA plugin [17],
- An IEEE-1516/HLA Evolved compliant C++ encoding library [18],
- A Python/HLA binding [19],
- Many patches for fixing bugs in CERTI or in the new companion software components.

All those contributions could certainly not have been possible if CERTI was not an open source project. Moreover there would not have been any contribution if there had not been any publicly interested people. For example, the FlightGear/HLA plugin would not have been developed if CERTI was not open source and/or freely available for the FlighGear potential users which are used to open source software. The Matlab or Fortran90 bindings are part of a PhD work [16] for which free and open availability of software is a great advantage.
As a preliminary conclusion, we can say that there is a potentially growing open source community for Open Source RTI, which confirms the opinion that seems to have been shared at the Open Source Session organized during the 2008 Fall SIW [20].

Therefore, our revised and updated reasons for supporting the Open Source CERTI project are:

1. Having an RTI for which we can make fast modification or add-on for specific project needs: real-time simulation, embedded middleware …
2. Federating an international user community which contributes to the enhancement and maintenance of the open source software component,
3. Having freely usable HLA tools for teaching,
4. Having some piece of software usable for pursuing research in the area of distributed and/or high-performance simulation.

As an illustration, this open source user (and contributor) testimonial summarizes the advantage of an Open Source CERTI:

I'm using CERTI because I need a free HLA RTI with C++ API that could be used by individuals/organizations that cannot afford to purchase a commercial HLA RTI. [...]
I need C++ API because most of the simulation software I'm using is in C/C++. I like CERTI because:

- *it's free (see above)*
- *it's open, so we can fix it quickly if necessary*
- *the license allows inclusion of CERTI in a proprietary software*
- *it has satisfying quality*
- *it's still evolving*
- *it works both under Linux and Windows*
- *it has no Java inside, so it doesn't have poor performance, complex installation and startup*
- *it has a friendly and supportive mailing list ;-)*

That said, one may easily understand that CERTI is not just "another RTI product", CERTI is not a product. Thus, CERTI is *not* competing with others: commercial MAK [21], Pitch [22] or even other Open Source RTI like Portico [23], we are complementary. CERTI is an Open Source Project managed by a government entity with a living community of users and contributors. Now let us explain how the CERTI Open Source project works.

3.2 Open Source CERTI - How?

An open source project is structured around its community of users and contributors/developers. Unlike "usual standard" software, users may participate in the evolution of the software, through their contribution. This is somehow a new way to use software which is already widespread including in firms [24]. A striking difference between closed-source and open source software is that a user may be a contributor. Each contribution may enrich the globally shared open source components and promote reuse as shown in Figure 5.

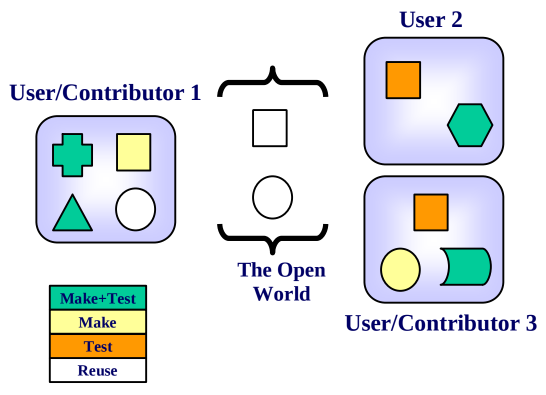

Figure 5: Open Source user/contributor

This does not necessarily prevent the different users from having private, non open source in their project. The mixing of open and closed-source parts in a project using CERTI is possible because CERTI Open Source License is LGPL [25] for libraries and GPL for applications. Contributing to the open source software is not mandatory; some users may be "standard users", as shown in Figure 5.

The essence of an Open Source project is the open collaboration between all stakeholders. This collaboration is usually supported by collaborative tools. We will describe hereafter the CERTI Project stakeholders and collaborative tools. The description is not that specific to CERTI and certainly similar to other Open Source projects.

CERTI Project stakeholders: like any other open source project, CERTI has different kinds of actors. Each actor is a person who may (or may not) represent his/her company.

- Project administrators: people who have the right to perform administrative actions (add a member, remove a member, lower or raise privilege for a member regarding the usage of the different collaborative tools, moderate messages on project mailing lists …). CERTI project currently has 2 administrators representing one institution (ONERA). A project administrator usually defines the project roadmap and ensures the consistency of the project when merging contributions.
- Project developers: people who have [autonomous] write access to the source code of one or several software components in the project. They may add/remove/modify software. They integrate external contributions, they fix bugs, carry out the release, etc… Note that a "developer" may be someone who only takes care of documentation; he may not be a computer scientist even if most of them are. A developer reports bugs. A developer will voluntarily answer questions raised on the mailing list, etc… There is at least one developer responsible for the development of each software component in the project.
- Project contributors: people who use the software components and sometimes provide bug fixes and/or new features such as a patch (a piece of source code), documentation, translation, new companion software modules… The contribution may be merged (or not) by a project developer. The decision to include or reject the contribution is discussed with potentially all interested project stakeholders using collaborative tools (mailing list, trackers); the developers plus the administrator make the final decision.
- Project users: people who use any software component found in the project. Users do ask questions on the mailing lists, they are invited to directly report bugs using project trackers. They are invited to contribute; they may become developers if they apply for it and have recognized knowledge within the project.

There are no deeply hierarchical responsibilities in a relatively small project like CERTI, besides the roles described previously. The community is open and friendly. It is important to notice that an Open Source project is essentially a set of people. Some of these people work on the project for a company and some of them not but merely contribute as a hobby. That's another key difference between a commercial product and open source software: a user may not ask for a bug to be fixed or a new feature to be added within a firm deadline unless he hires someone to do it. Some companies offer commercial support for open source software too, but this is not currently the case for CERTI. There exist several business models around open source [26]. CERTI evolution is currently only driven by the needs of its users and primarily by the research needs of ONERA. However, ONERA welcomes contributions which go beyond its own needs.

CERTI Collaborative tools: almost all open source software projects use a set of collaborative tools in order to support a worldwide team. Most of these tools are web-enabled and grouped in a project portal called a "forge". The CERTI project forge is hosted by Savannah forge [7] as shown in Figure 6.

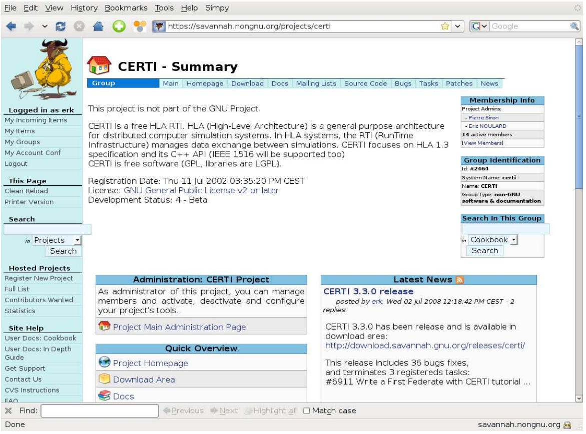

Figure 6: CERTI project at Savannah

We describe below the collaborative tools offered by Savannah forge and used by the CERTI project. Note that the use of those tools is open as much as it can be. When an access right is enforced, it is either because we want to avoid SPAM and/or because we have to ensure project consistency. There are 4 levels of access to the tools:

1. Anonymous: anyone on the internet,
2. Registered Savannah user: anyone who has a valid Savannah account (which is free and may be obtained through a simple request),
3. Developer: a registered Savannah user who has been added to the CERTI project by a CERTI administrator,
4. Administrator.

The CERTI administrator may set the required level of privilege for using any tool. In the following description of the collaborative tools, we will recall the required level of privilege for each one.

The collaborative tools used by CERTI are:

- Download Area: this is where the CERTI software may be downloaded. The access level is Anonymous; it is therefore freely accessible by anyone at http://download.savannah.gnu.org/releases/certi/. The primary distribution format is a source archive to be compiled by the user. CERTI currently compiles easily on several combinations of Operating Systems (Windows, Linux, Solaris, FreeBSD …) and compilers (gcc, Visual Studio, Sun Studio, MinGW …). If a user is not used to software compilation, he may consult the online documentation on this subject http://www.nongnu.org/certi/certi_doc/index.html or just kindly ask for help on the mailing list.
- Mailing List: this is the primary communication means for the CERTI community. The list archive is freely accessible to anyone. However, one must subscribe to the list, http://lists.nongnu.org/mailman/listinfo/certi-devel, in order to be able to send a message. This is a necessary measure in order to avoid SPAM. The CERTI community is currently small so there is no separate user/developer mailing list.
- Bug Tracker: this is where bugs are reported and handled, https://savannah.nongnu.org/bugs/?group=certi. Bug report is open to Anonymous users but it is better to be a Registered Savannah one in order to get automatic follow-up. Every CERTI user is invited to file bug reports, since this is the primary means to improve CERTI. As soon as the bug is assigned to a developer, the reporter will get e-mail follow-up concerning his bug report. The bug tracker is the place to look at when you face trouble using CERTI; searching the bug tracker may lead you to an already fixed issue (possibly in a forthcoming release).
- Patch Tracker: this is where contributors should drop their contribution, https://savannah.nongnu.org/patch/?group=certi. A patch is a file containing the source difference between the current CERTI version and the contribution. It will be reviewed by CERTI developers and eventually merged in the next CERTI release.
- Task Tracker: this is where planned CERTI evolutions are listed, https://savannah.nongnu.org/task/?group=certi. You have to be a CERTI developer to create a task.
- SCM: Source Control Management, this is the versioned repository of CERTI software component, CERTI is using CVS (Concurrent Versioning System). Anonymous users have read-only access to the repository whereas CERTI developers do have read/write access to the source repository. Even if CERTI is an Open Source software it has a structured and controlled source revision policy. Using the SCM we may rebuild any version of CERTI since day 1 (unless the compiler you use at that time is no longer available).

Collaborative tools are essential to the CERTI community where we have already people from France, Germany, Italy, United-States, etc. Most of them have never met but are working together using the collaborative tools; we hope to be able to organize CERTI users' meetings some day. Contribution may be as small as a single bug report up to the whole implementation of a missing feature.

Potential CERTI users must be prepared to use at least the bug tracker and the mailing list and they should quickly be pleased with these tools.

When using open source software of this kind you just have to remember that this is not a commercial product. You cannot file a bug report and expect the community to do it as fast as you want. With open source software there are two solutions to this problem:

1. Hire someone to fix the bug, which would be simple because the person you hire would have the full source at hand,

2. Fix it yourself: you have the source too.

You should now have a fairly good idea of how to use and contribute to CERTI; for any remaining question, you should find an answer on the CERTI Mailing List: http://lists.nongnu.org/mailman/listinfo/certi-devel.

3.3 CERTI software components

As already introduced, the CERTI Open Source project is now composed not only of the CERTI RTI itself but of several other software components that may be useful to potential HLA users. You will find hereafter the list of current CERTI project software components with a small description.

HLA Tutorial application: this is a small HLA federate which implements a controller/process application. The application is simple, with 2 federates; it uses basic HLA 1.3 services and is self-explanatory when running. This is an HLA sample application which is designed to be self-contained from source. It works with CERTI but should work with other RTIs too.

HLA Tests Suite: this is a suite of standalone federate applications whose purpose is to test a particular RTI feature (parsing FED file, create/join/resign/destroy federation execution, synchronization point…). Ideally, the suite should cover all HLA [Evolved] features and work either as a validating suite and/or benchmarking suite. Each application is standalone and has been designed to be easily run as a batch job. The output of each "test" may be easily parsed for success or failure. CERTI is using DTest [27] in order to automate the launch of the whole suite; the result of the tests may be automatically sent to a CTest dashboard [28]. The whole Tests Suite is used as a regression test suite for CERTI RTI. The current suite is far from complete and any contribution in the suite is welcome.

Matlab or F90/HLA13 bindings: The objectives of these bindings are the following [16]:

- provide engineers with HLA access within their usual working environment,
- provide a way to easily extend existing code to HLA federates,
- increase acceptance of HLA in the engineering domain.

The current version of the MatlabHLA-Toolbox [29] fully supports federation management, declaration management, object management and time management of the HLA 1.3 standard. A Matlab federate invokes RTI services by calling the appropriate MatlabHLA-Toolbox m-function. The m-function directly calls a C++-wrapper function. All necessary type conversions are then done. Following this, the actual RTI library function is called. All RTI services immediately return after execution. RTI initiated calls are handled first on the wrapper layer. LibRTI calls the implemented federate services. There, a type conversion from C resp. C++ types into Matlab types takes place. Finally, the appropriate m-file service is called through invoking the MEX function mexCallMatlab(). The execution then returns to libRTI.

Simulation model design and execution in the engineering and scientific domain are often characterized by the use of Scientific and technical Computation Environments (SCE) like Matlab, a famous commercial SCE. Other free SCEs already exist, like Octave or Scilab. These systems increasingly replace traditional Fortran coding. But existing Fortran programs are used daily throughout the scientific and engineering community. Especially for the supercomputing community, Today Fortran is still the primary programming language. Due to the increasing importance of HLA, there is a need to provide engineers with a native HLA interface in their "daily working environments". This approach promises to minimise the effort of applying HLA in the engineering community. The F90HLA library closes this gap for Fortran [30].

A FlightGear/HLA plugin: The Virtual Air project [17] intends to provide a standardized (HLA 1.3 based) framework for distributed air traffic simulation.
The project currently includes a FlightGear HLA plug-in which is an HLA 1.3 interface for the FlightGear flight simulator.

XPlane/HLA plugins: there are 2 plugins. The first one makes it possible to get information from XPlane to an HLA Federation and the second one makes it possible for an HLA Federate to send commands to XPlane.

An IEEE-1516/HLA Evolved compliant C++ encoding library: The library [18] implements efficient access functions that provide direct access to IEEE 1516.2 compliant data buffers. The data are manipulated "in situ", no temporary variables are created. The extensive use of template metaprogramming allows many operations to be pre-calculated during compile-time. The library has similar features as those described in [31]; however, the two have been independently developed and the CERTI component is freely available since it is an open source component.

A Python/HLA binding: The PyHLA module [19] provides Python language bindings for the Modeling & Simulation High Level Architecture (M&S HLA). The PyHLA module aims to enable rapid development of HLA federates, i.e. to simplify the activity 4.3 of FEDEP [IEEE 1516.3]. Integrating HLA into the Python language may reduce the development and maintenance effort (compared to C/C++). PyHLA provides:

- Python language HLA API, that is compliant with the HLA 1.3 standard (implemented as a Python wrapper for the C++ HLA API),
- Pack/unpack methods providing IEEE 1516.2 encoding,
- HLAuse python function that is able to directly import OMT DIF data types (the XML format described in IEEE 1516.2)

The PyHLA module can be built on a variety of platform/compiler combinations, including Windows, Linux and Sun Solaris. The module relies on the Classic Python interpreter (version 2.4 or higher) and requires an HLA 1.3 compliant RTI with C++ DLC API.

As a side note, we would like to remark that CERTI users need to open the HLA interface to "non-native" HLA standard bindings like Java and C++. We did call those Matlab, F90, Python "bindings" but they do not currently intend to add another officially normalized binding to HLA standard, by the way, those "bindings" do not cover the whole HLA standard. Those user bindings aim at bringing HLA usage to a wider audience.

4. Conclusion

We have described the history and the current state of the CERTI Open Source project, its initial motivation and evolution. CERTI was initially built from the internal ONERA research project needs [2] [14]; it will keep its roots in these fundamental needs but with a firm will to federating a worldwide user community interested in distributed simulation.

The open source CERTI project has two main goals. The first is to spread HLA usage and knowledge for research purposes inside and outside ONERA. This already was the case in the past [32] and this is still the case nowadays [16] [33]. The second goal CERTI is to federate a wide international open source community around distributed simulation. We thus invite universities and laboratories to use CERTI and maybe contribute or help other CERTI users with their CERTI software component. Contributions, new CERTI usage, or collaboration with other open source projects are welcome.

5. References

[1] Wikipedia: "Run-Time Infrastructure" http://en.wikipedia.org/wiki/Run-Time_Infrastructure(Simulation), January 2009.

[2] Pierre Siron: "Design and Implementation of a HLA RTI Prototype at ONERA", 1998 Fall Simulation Interoperability Workshop, 98F-SIW-036, ftp://ftp.cert.fr/pub/siron/98f-siw-036.ps.

[3] Pierre Bieber, Jacques Cazin, Pierre Siron, Guy Zanon: "Security Extensions to ONERA HLA RTI Prototype", 1998 Fall Simulation Interoperability 98F-SIW-086, ftp://ftp.cert.fr/pub/siron/98f-siw-086.ps.

[4] Martin Adelantado, Jean-Loup Bussenot, Jean-Yves Rousselot, Pierre Siron, Marc Betoule: "HP-CERTI: Towards a high Performance, high Availability Open Source RTI for Composable Simulations", 2004 Fall Simulation Interoperability Workshop, 04F-SIW-014, ftp://ftp.cert.fr/pub/siron/04F-SIW-014.pdf.

[5] Pierre Siron: A pilot application of distributed cooperative simulation under HLA (SICODIS PRF), Scientific and Technical Activities, ONERA, 2005.

[6] Bruno d'Ausbourg, Pierre Siron, Eric Noulard: "Running Real Time Distributed Simulations under Linux and CERTI", 2008 Euro Simulation Interoperability Workshop Proceedings, 08E-SIW-061, ftp://ftp.cert.fr/pub/siron/08E-SIW-061.pdf.

[7] CERTI Open source project at Savannah: https://savannah.nongnu.org/projects/certi, January 2009.

[8] CERTI Home at ONERA: http://www.cert.fr/CERTI, January 2009.

[9] Björn Möller, Katherine L Morse, Mike Lightner, Reed Little, Robert Lutz: "HLA Evolved – A Summary of Major Technical Improvements", Proceedings of 2008 Spring Simulation Interoperability Workshop, 08F-SIW-064, September 2008, http://www.pitch.se/images//08f-siw-064.pdf.

[10] Dynamic Link Compatible HLA API Standard for the HLA Interface Specification (IEEE 1516.1 Version), SISO-STD-004.1-2004, 20 April 2004.

[11] Pamella Knight et al. : "Evaluation of Run Time Infrastructure (RTI) Implementations", Huntsville Simulation Conference 2002, http://www.scs.org/confernc/hsc/hsc02/hsc/papers/hsc017.pdf.

[12] Len Granowetter, Douglas D. Wood: "Opening the Black Box: The Effect of RTI Implementation

Details on the Success of an HLA Federation", MÄK Technologies whitepaper, http://www.mak.com/pdfs/wp_blackbox.pdf.
[13] Björn Möller, Mikael Karlsson: "Developing Well-Balanced Federations Using the HLA Evolved Smart Update Rate Reduction", Proceedings of 2005 Fall Simulation Interoperability Workshop, 05F-SIW-087, http://www.pitch.se/images//05f-siw-087.pdf.
[14] Benoit Bréholée, Pierre Siron: CERTI, Evolutions of the ONERA RTI Prototype, 02F-SIW-018
[15] IESTA Project: An infrastructure capable of evaluating new air transport system concepts, http://www.onera.fr/iesta/index.php.
[16] Christian Stenzel, S. Pawletta: "CERTI - Bindings to Matlab and Fortran", Magdeburger HLA-Forum, 27 february 2008.", http://www.mb.hs-wismar.de/~stenzel/talks/hla_forum10.pdf.
[17] Petr Gotthard, "Virtual Air: a Multiplayer framework for flight simulators", http://sourceforge.net/projects/virtualair/ see also http://wiki.flightgear.org/index.php/Virtual_Air.
[18] IEEE-1516.2 compliant encoding task on CERTI Project: http://savannah.nongnu.org/patch/?6534 .
[19] Petr Gotthard: "Using M&S HLA in Python", http://www.nongnu.org/certi/PyHLA/index.html.
[20] Informal private communication with Tim Pokorny about 2008 Fall SIW Open Source Session.
[21] MAK RTI: http://www.mak.com/products/rti.php.
[22] Pitch RTI: http://www.pitch.se/products/pitch-prti/pitch-prti-overview.html.
[23] Portico Project: http://www.porticoproject.org.
[24] Gartner Group: "Using Open-Source Software Increases, Companies Must Adopt and Enforce an OSS Policy", November 17, 2008 http://www.gartner.com/it/page.jsp?id=801412.
[25] LGPL: GNU Lesser General Public License, http://www.gnu.org/copyleft/lesser.html.
[26] APRIL: "The Economic Models of Free Software", white paper, http://www.april.org/files/economic-models_en.pdf, December 2007.
[27] Lionel Duroyon: "What is DTest ?", internship report, Onera DTIM, 11 april 2008, http://download.savannah.nongnu.org/releases/tsp/dtest/what_is_dtest.pdf.
[28] CERTI CTest dashboard: http://www.cdash.org/CDashPublic/index.php?project=CERTI%20HLA%20TestsSuite.
[29] MatlabHLA-Toolbox: http://www.mb.hs-wismar.de/~stenzel/software/MatlabHLA.html.
[30] F90HLA: http://www.mb.hs-wismar.de/~stenzel/software/F90HLA.html.
[31] Jay Graham: "Creating an HLA 1516 Data Encoding Library using C++ Template Metaprogramming Techniques", 2007 Spring Simulation Interoperability Workshop Proceedings, 07S-SIW-035.
[32] Valery Raulet, Vincent Rodin, Alexis Nédélec: Using HLA to provide a Collaborative Multi Agent Virtual Prototyping platform, 2003 Spring SIW, 03S-SIW-51.
[33] Hassen Jawhar Hadj-Amor: Contribution au prototypage virtuel de systèmes mécatroniques basé sur une architecture distribuée HLA. Expérimentation sous les environnements OpenModelica-OpenMASK, PhD Thesis, 2008.

Author Biographies

ERIC NOULARD graduated from a French High School 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 & Telecom domain for BT C&SI, mostly building high-performance tests & validation systems, he joined ONERA research center in Toulouse as a Research Scientist. He now works on distributed and real-time systems and is actively involved in the development of the CERTI and TSP Open Source projects.

JEAN-YVES ROUSSELOT is a long-standing ONERA engineer, so long that no one dares ask when he joined ONERA. He has black feet but a clear mind and a stormy voice. We are pretty sure that he will have peacefully retired when you read this. The co-authors would like to thank him warmly for giving his last working years to CERTI.

PIERRE SIRON graduated from a French High School for Engineers in Computer Science (ENSEEIHT) in 1980 and received his doctorate in 1984. He is currently a Research Engineer at ONERA and works on parallel and distributed systems. He is the leader of the CERTI Project. He is also a Professor at the University of Toulouse, ISAE, and the head of the computer science program of the SUPAERO training (French High School for Engineers in Aeronautics & Space Sciences).