JOURNAL OF AEROSPACE COMPUTING, INFORMATION, AND COMMUNICATION
Vol. 1, January 2004

Distributed Simulation for Acquisition and Analysis of Future Airport Operational Concepts

Martin Adelantado,[*] Jérôme Latour,[†] and Alexandre Vincent[‡]
ONERA-CT/DPRS/SAE, 31055 Toulouse Cedex, France

and
Pascale Bonnet[§]
ONERA-CSP/DCSD/PSEV, F-13661, Salon de Provence, France

Air transport systems undergo continuous evolution in order to adapt to future performance, quality, and safety requirements. This implies the operational use of new techniques in navigation, communications, surveillance, regulations, and computer-based assistance/automation systems. In the context of the future air traffic management (ATM) systems, airports will play a critical role because they have to face serious environmental (noise impact) and operational restrictions, which reduce their potential capacity. To meet the projected airport demand and improve efficiency of ground operations while maintaining safety, several new operational concepts are under investigation, including new regulations or operational concepts, and advanced equipment or computer-based assistance systems. The French Aeronautics and Space Research Center (ONERA) is participating in this effort through the design of a modeling and simulation infrastructure for acquisition and analysis of airport operational concepts and equipment. The originality of this work is to provide an open and scalable simulation support aimed at easily plugging in additional components in order to analyze and compare the efficiency of several airport configurations.

I. Introduction

Airports are increasingly complex organizations that can be divided into several stakeholder parties. On the airside part, there are the airport operators, being the airport authority, the air traffic service provider, slot coordinators, and airlines. Next, there are the regulatory authorities and people living in the neighborhood of the airport. Depending on the individual point of view of these stakeholders, everyone is able to identify different priority focus issues, leading to a different set of requirements for problem identification and solving. Several well-known issues such as delay and punctuality, throughput capacity, timetable robustness, safety, security, noise abatement, passenger's quality of service, or third party risk, are important to all stakeholders, however addressed with a different priority. To improve the efficiency of the airport system, while maintaining safety, several initiatives are under investigation.[1,14,16] Among these initiatives, Advance Surface Movement Guidance and Control Systems (A-SMGCS) is one of the most promising concept to meet the growing demand in airport capacity. The International Civil Aviation Organization (ICAO) defines A-SMGCS as a concept "expecting to provide adequate capacity and safety in relation to the specific weather conditions, traffic density and aerodrome layout by use of modern technologies and a high level of integration between various functionality." Collaborative decision making

Presented as Paper 2003-5373 at the AIAA Modeling and Simulation Conference and Exhibit, Austin, Texas, USA, 11-14 August 2003; received 20 June 2003; accepted for publication 28 October 2003. Copyright © 2004 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved. Copies of this paper may be made for personal or internal use, on condition that the copier pay the $10.00 per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923; include the code 1542-9423/04 $10.00 in correspondence with the CCC.
[*]Research Engineer, French Aeronautics and Space Research Center, Long Term Design and Systems Integration, 2, Avenue Edouard Belin; Martin.Adelantado@onera.fr.
[†] PhD Student.
[‡] Student.
[§]Research Engineer, French Aeronautics and Space Research Center, Systems Control and Flight Dynamics, Ecole de l'Air, Base Aérienne 701; Pascale.Bonnet@onera.fr.

(CDM) provides a complementary promising concept, based on a framework of information exchange and data sharing between all ATM actors.

The evaluation of costs and benefits of future operational concepts such as A-SMGCS or CDM, generates, among other things, the need to create analysis and acquisition techniques that can help to validate functionality and advantages of new systems and technologies, and quantify the performance of such concepts to support decision making on investment, as well as provide stakeholders and designers with tools that are more flexible, and fully adaptable to any foreseen system.

Modeling and fast-time simulation (FTS) are nowadays very commonly used in several industrial fields such as energy, factories, economics, or transportation systems.[9,13] This is a consequence of the critical need to know in an early stage of development, the answers to "What if?" questions. In the airport management domain, some available standard tools include SIMMOD (Simulation Model) from the FAA, TAAM (Total Airspace and Airport Modeler) from Preston Group, TARMAC (Taxi and Ramp Management and Control) from DLR, or high fidelity air traffic flow management (ATFM) simulators. Nevertheless, those tools are in most cases, monolithic and "black boxes" packages that cannot be easily extended. In addition, many fragmented and isolated simulation tools are available that only address part of the airport planning, environmental, operations, and development problem.[12] For instance Integrated Noise Model (INM) from the FAA addresses noise impact of landing/take-off operations. Nevertheless, an airport is a very complex system including both airside and landside parties and components. Therefore, following a global approach in modeling and simulation of such organizations is difficult to successfully achieve. Thus, a bottom up approach aiming at integrating existing or enhanced simulation tools seems more cost effective and technically achievable.

The French Aeronautics and Space Research Center (ONERA) is carrying out a federative research and development project to propose solutions which enhance traffic management of future airports while maintaining safety. One goal of this initiative is to provide an open and flexible modeling and simulation infrastructure to evaluate future airport operational concepts (equipment and tools) to assist controllers and pilots. To achieve this goal, the high level architecture (HLA) has been selected as a support for distributed simulation.

The paper addresses the following issues. The next section provides motivations for distributed simulation and gives technical background about HLA. Next, the paper briefly addresses the architecture of both the modeling and simulation infrastructure and the reusable HLA federation we propose. More precisely, we focus on recent models that have been introduced in the reusable HLA federation, a first prototype of which has been described in Ref. 5. These new models and HLA federates provide an operational communication level between the simulation and both pilots and controllers allowing then a more realistic modeling and simulation of communications between these airport actors. Those improvements have been introduced to meet the requirements of acquisition and analysis of A-SMGCS simulation. We then focus on methodological approaches to design more complex airport systems by plugging additional elementary HLA simulators to the kernel reusable federation.

II. Motivations for Distributed Simulation and HLA

A detailed description of the Department of Defense's (DoD) HLA is beyond the scope of this paper. Additional information may be obtained directly from the HLA Web site or from the Mc Leod Institute of Simulation Sciences Web site.

In a few words, HLA[7,10] is a general purpose software architecture for distributed simulations, designed to support a wide range of simulation approaches and applications. The expected benefits of HLA are to facilitate interoperability, portability, and simulations reuse, thus reducing modeling and simulation costs. HLA can support virtual, constructive, and live simulations from the training, engineering, and analysis applications domains.

HLA components include rules, Object Model Template[11] (OMT), and the specifications of an application programming interface (API).[8] An implementation of the programming interface specification is offered by the run-time infrastructure (RTI), a software that provides common services to a set of elementary simulators (federates) participating in a named federation (Fig. 1).

The interface specification partitions the exchanges that take place between federates and the federation into six management areas (federation, declaration, object, time, ownership, and data distribution management). At run-time, the RTI ensures communications between the participating federates through a LAN or WAN network. At any time, a given federate may join or resign a federation execution according to the simulation requirements. Recently, HLA has also been approved as an open standard for distributed simulation through the IEEE (IEEE Standard 1516).

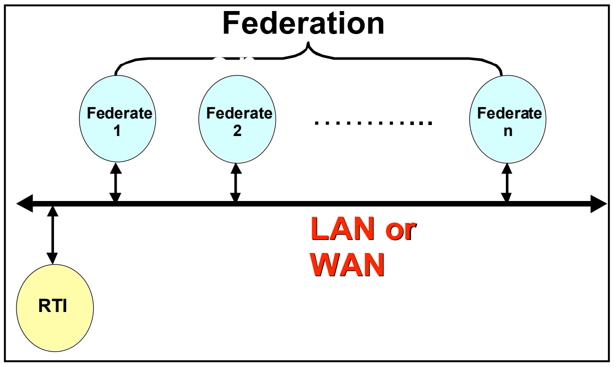


Fig. 1 Components of an HLA federation.

The RTI is the key point in constructing HLA federations. Until September 2002, the Defense Modeling and Simulation Office (DMSO) provided development and support of the Next Generation RTI (RTI-NG), freely available for download. Unfortunately, this facility is not available anymore and HLA has thus begun its transition to a solely commercially developed and sustained product. Nevertheless, alternate solutions exist. For example, ONERA is designing an open source RTI, called CERTI that has been freely available since September 2002 from the Web site.

Motivations for distributed simulation through HLA are the following.

First of all, HLA offers a means to increase the complexity of the modeled global system by introducing additional components that can communicate through the RTI, achieving flexibility and scalability of the simulated system. Therefore, it is not necessary, in a preliminary step of design, to incorporate all the components of the simulated system. This issue is of primary importance, because airports are very large, dynamic, and complex systems with many interacting traffic modes and numerous services that have to share the same resources from the land side (terminal service, luggage service) to the air side (apron, taxiways, runways, airspace).

Secondly, HLA really offers a means to face well-known site-proprietary constraints, and to easily reuse existing models and simulations to design simulations of increasing complexity and coverage.

During the specification phase, we selected HLA specially because it provides a support to easily plug in additional components and not for its capacity to design distributed simulations. Our approach consists then in providing a reusable airport simulation kernel built around a set of distributed elementary simulations inter-operating through HLA. This reusable HLA simulation (federation) is part of a modeling and simulation infrastructure allowing the user to easily define a simulation scenario for data acquisition during simulation, and to provide offline analysis of the simulation results.

III. The Modeling and Simulation Platform

The architecture of the modeling and simulation platform is depicted in Fig. 2. The platform consists of three main components.

Human-machine interface. A human-machine interface (HMI) allowing the user to build up a particular airport scenario in an easy and user-friendly way. This HMI offers a set of facilities for integrating and managing the federation execution, a traffic-sample generator, and software functions using a mouse for creating airport layout.[2] During the simulation, the end-user also has the ability to modify in real-time the airport scenario to increase the realism of the simulation study. Available modifications include delaying or canceling a given flight, or changing the landing/take-off operations. The HMI plays a role of input/output component for the simulation component consisting of the airport federation and the RTI. Input data include both the initial airport scenario and the unexpected events that occur during the simulation exercise. Output functions include post-processing analysis of acquisition results and display facilities.

Computational component. The computational component is the airport HLA federation consisting of the reusable federation and a collection of additional federates. Typically, federates participating in the federation simulate models of operational concepts (equipment or procedures) selected for the acquisition process.

RTI. The RTI implementing the HLA specification interface.

The modeling and simulation (M&S) infrastructure is currently available on two platforms: a cluster of Intel Pentium PCs, connected through both a fast Ethernet and a Myrinet networks, under a Linux Red Hat operating system; a network of SunBlade workstations connected through an Ethernet network, under a SunOS 5.8 operating system. Both distributions use the RTI-NG V6 freely available from the DMSO.

The reusable federation is the first originality of our contribution in that HLA only supports reusability for federates. This concept has a significant impact on the OMT and especially on the Federation Object Model (FOM),[11] but this issue is beyond the scope of the paper. The reusable federation is built around five federates, as depicted in Fig. 3.

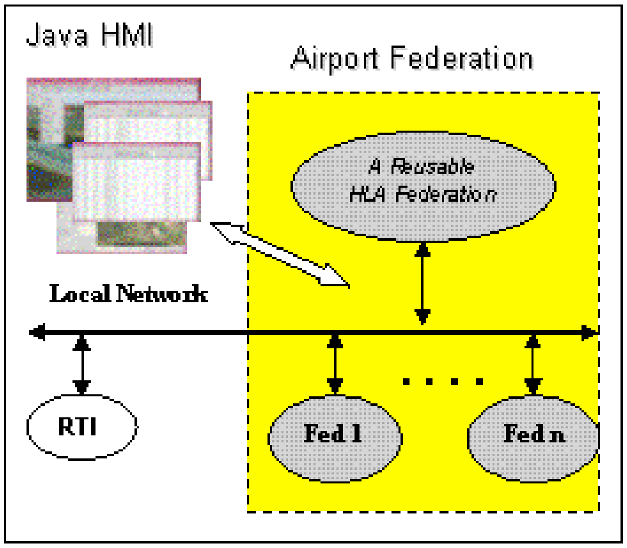


Fig. 2 The modeling and simulation platform.

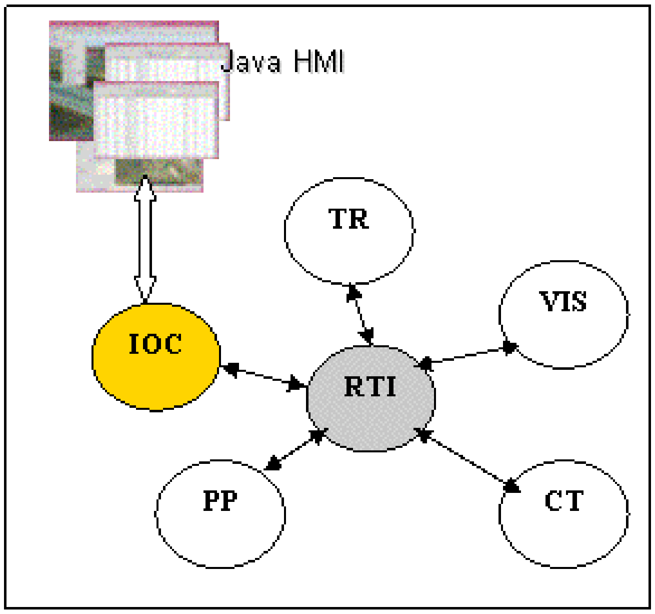


Fig. 3 Architecture of the reusable federation.

The TR federate (traffic federate) moves the aircraft on the airport layout according to the departure or arrival times defined by the initial scenario, the unexpected events (flight cancellation or delay), and operational instructions coming from either the CT federate or the PP federate. The airport layout is modeled through a conventional network of nodes and bi-directional links.[3] Some specific nodes correspond to operational locations such as parking, cross-runway points, or cross-taxiway points. The simulation engine moves aircraft according to the shortest path from a starting node to a destination node of the airport network.

The VIS federate (visualization federate) provides a real-time display of the simulation behavior, showing the aircraft moving on the airport layout. Under Solaris operating system, a two-dimensional display is performed using the C++ MOTIF library.

The CT federate (control tower federate) is intended to support online interaction of the simulation with a controller. The federate, written in Java language instead of C++, displays an operator console built around a set of windows dedicated to both reception of several kinds of operational messages and emission of operator actions, such as instructions to a given flight, or clearance acknowledgments.

The PP federate (pseudo pilot federate), written in Java language, provides a set of windows, buttons, and text fields allowing the user to model the operational communications with both the CT federate and the TR federate. Communications include requesting clearances from the control tower, performing the acknowledged clearances, or executing a given instruction for a given flight. The pseudo pilot is intended to control all the simulated flights.

Finally, the IOC federate (input/output component federate) consists of the interface between the airport federation and the HMI component. It has the responsibility of managing all the files, including a description of the expected scenario, data representation such as airport layout, and simulation outputs. Thus, as soon as the federation has been created and all federates have joined the federation execution, the IOC federate publishes the simulation scenario read from a set of files generated by the HMI. Each federate subscribes to the scenario elements it needs. Alternatively, the IOC federate collects during the simulation, all the results provided by the federation, storing them into a set of files for further offline analysis. Finally, the IOC federate publishes the unexpected events of the scenario introduced by the user through the HMI (flight cancellation or delay for example). In the same way, those unexpected events are then sent by the RTI to the subscribing federates. The HMI module and the IOC federate are thus closely related even if the HMI does not belong to the federation. This architecture has been specially designed in accordance with the HLA "spirit," in that all data shared by the reusable federation and more generally by any airport federation are supported through the RTI. In such a way, fully distributed simulations would be supported by the platform, the single constraint being that both the Java HMI and the IOC federate have to share the same execution platform.

IV. Managing an Acquisition Session

A. Managers and Software Tools

An acquisition exercise involves several people (pilot, controller, and simulation managers) and software tools as illustrated in Fig. 4.

The first simulation manager prepares the use case of the exercise including selection and modeling of a given airport layout and expected timetables. This phase consists in preparing the exercise scenario taken from an actual use case. During the acquisition exercise, this simulation manager has the responsibility of introducing an unexpected operational event according to the use case. For example, he can cancel or delay a given flight at a given time according to the actual scenario. From an operational point of view, this simulation manager performs all the instructions coming from operational actors outside the airport system. For example, slots for departing flights, are provided in Europe, by the Central Flow Management Unit (CFMU) of Eurocontrol. Changes to the landing and take-off direction are performed according to the local weather forecast center. Finally, flight plans and timetables are provided by airline representatives. These tasks are performed through a set of software tools provided by the HMI. Finally, the first simulation manager defines the allocation of federates to the workstations and starts the simulation exercise using the main menu of the HMI.

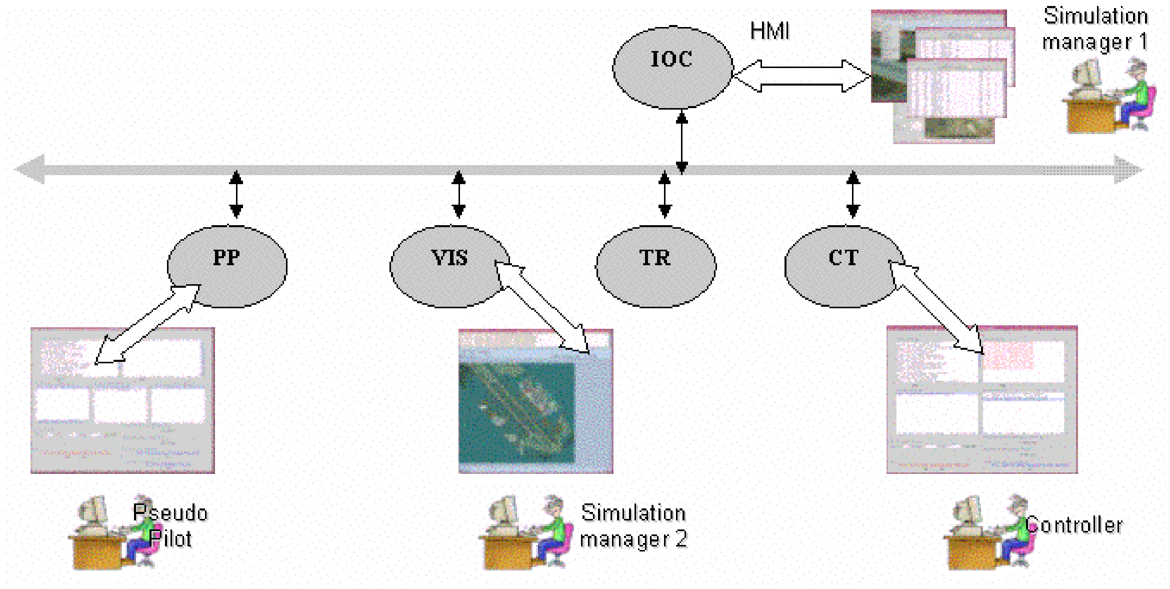
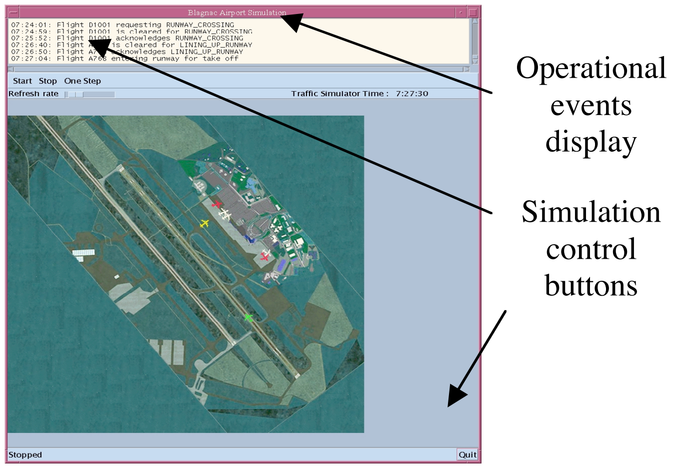


Fig. 4 Organization of an acquisition session.


Blagnac Airport Simulation
Start Stop One Step
Refresh rate
Stopped
Quit
Operational events display
Simulation control buttons

```
07:24:01: Flight D1001 requesting RUNWAY_CROSSING
07:24:59: Flight D1001 is cleared for RUNWAY_CROSSING
07:25:52: Flight D1001 acknowledges RUNWAY_CROSSING
07:26:40: Flight A768 is cleared for LINING_UP_RUNWAY
07:26:50: Flight A768 acknowledges LINING_UP_RUNWAY
07:27:04: Flight A768 entering runway for take off
```

Traffic Simulator Time : 7:27:30

Fig. 5 Simulation display.

The second simulation manager does not represent an operational actor. He controls the simulation, starting it as soon as each federate has joined the federation, and suspending it when necessary. These control operations are performed through a set of buttons provided by the VIS federate, as shown by the screenshot in Fig. 5 (Toulouse Blagnac airport). During an acquisition exercise, the simulation display offers the operational actors (controller and pilot), the actual situation at any time, through both a synthetic environment and a window reporting all the operational events.

The pseudo pilot controls all the simulated aircraft through a set of radio buttons and text editors, displayed by the PP federate. A screenshot of the pseudo pilot console is given in Fig. 6. Window 1 is devoted to operational events broadcast by all the federates participating in the reusable federation. Windows 2 and 3 display both requested and acknowledged clearances. Windows 4 and 5 are dedicated to airport simulation with additional equipment. Finally, a set of radio buttons allows the pseudo pilot to select a clearance management mode, from a fully automatic scheme to a fully manual one. This issue will be addressed later in this paper.

The controller has the responsibility of acknowledging requested clearances and controls ground traffic according to the actual situation and congestion rate. Several facilities are provided by the CT federate, through a set of windows and radio buttons displayed by a Java console (Fig. 7).

Window 1 is similar to the corresponding window of the pseudo pilot console, whereas window 2 is dedicated to clearance management. Windows 3 and 4 are also dedicated to airport simulation including additional components. Radio buttons and fields of text in the bottom part of the console allow the controller to select a management mode for both clearances and instructions, as well as to order instructions to the pseudo pilot.

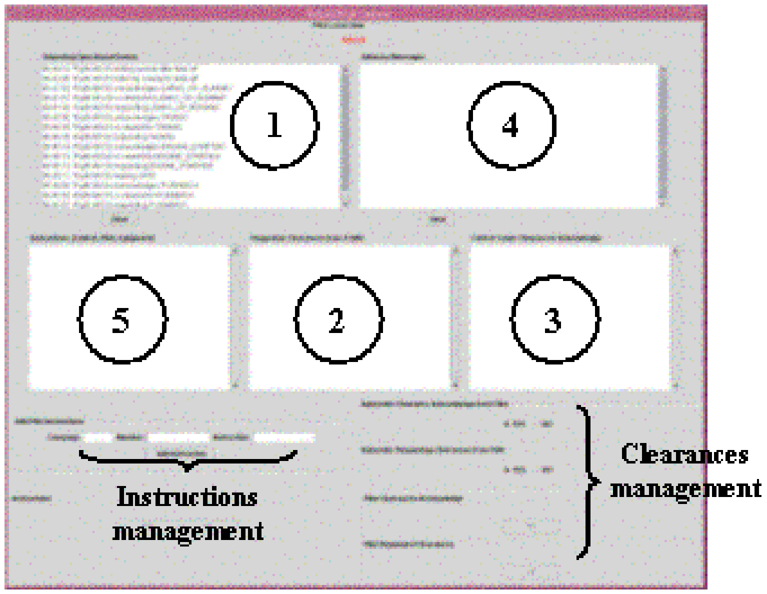


Fig. 6 Pseudo pilot console.

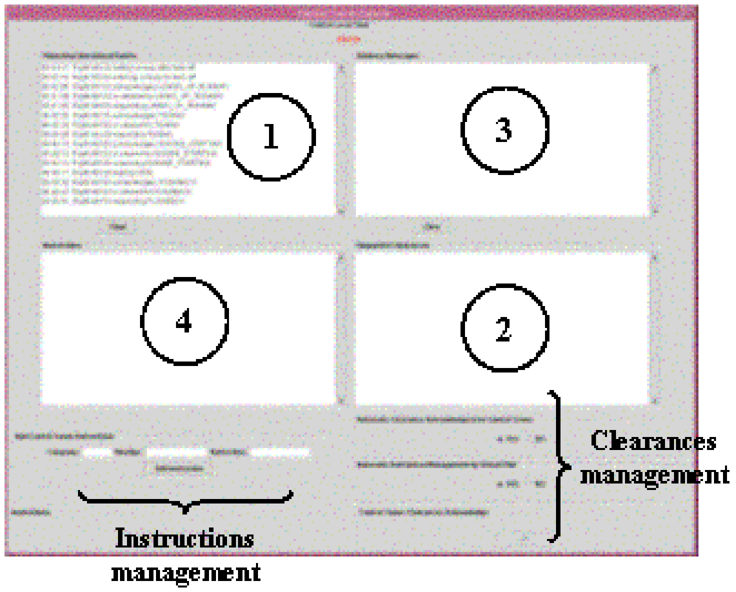


Fig. 7 Controller console.

B. Acquisition Process

In this subsection, we propose to illustrate the facilities offered by software tools through an example, showing how each actor can interact with the simulation execution according to the acquisition requirements. The first step consists in preparing an initial simulation scenario through the HMI, including expected flight timetables, airport selection, landing and take-off direction, simulation time range, etc. This preparation phase needs the involvement of all contributors and managers. Assume that the expected scenario involves only three flights, flight A6103 (expected take-off time, 06:40:00 a.m.), flight A763 (expected landing time, 06:50:00 a.m.), and flight A3147 (expected take-off time, 07:25:00 a.m.). Flights A763 and A3147 are supposed to be related to the same aircraft.

The acquisition process starts then by running the RTI and the federation execution. A preliminary step consists in reading the scenario files (IOC federate) and publishing them to the participating federates. The scenario includes the airport jpeg image that is sent by the IOC federate and received by the VIS federate. Once each federate has received the scenario components it is interested in, all interaction consoles are displayed. The simulation manager #2 may then definitively start the acquisition process through the simulation control buttons provided by the VIS federate. By default, the management modes of both clearances and instructions are set to "fully automatic."

The analysis process is mainly based on post-processing an acquisition file storing all operational events occurring during the acquisition session. Each operational event consists of a time-stamped message describing the corresponding event associated with the time of the physical system. For example, according to the simplified scenario proposed previously, the operational events associated with the departing flight A6103, will be the following:

06:40:01: Flight A6103 requesting PUSHBACK
06:40:02: Flight A6103 is cleared for PUSHBACK
06:40:03: Flight A6103 acknowledges PUSHBACK
06:40:11: Flight A6103 leaving GATE
06:40:12: Flight A6103 requesting ENGINE_STARTING
06:40:13: Flight A6103 is cleared for ENGINE_STARTING
06:40:14: Flight A6103 acknowledges ENGINE_STARTING
06:40:28: Flight A6103 requesting TAXIING
06:40:29: Flight A6103 is cleared for TAXIING
06:40:30: Flight A6103 acknowledges TAXIING
06:41:58: Flight A6103 requesting LINING_UP_RUNWAY
06:41:59: Flight A6103 is cleared for LINING_UP_RUNWAY
06:42:00: Flight A6103 acknowledges LINING_UP_RUNWAY
06:42:16: Flight A6103 entering runway for take off
06:43:21: Flight A6103 exiting runway after take off

Accordingly, the operational events corresponding to the arriving flight A763, are the following:

06:48:31: Flight A763 landing
06:48:39: Flight A763 touching runway after landing
06:50:06: Flight A763 exiting runway after landing
06:50:20: Flight A763 requesting RUNWAY_CROSSING
06:50:21: Flight A763 is cleared for RUNWAY_CROSSING
06:50:22: Flight A763 acknowledges RUNWAY_CROSSING
06:52:26: Flight A763 stopping at GATE

In fully automatic mode, clearances are sent by the TR federate and acknowledged by the controller, one second later. However, both controller and pseudo pilot have the possibility to select a fully or intermediate manual mode using the radio buttons of the corresponding consoles. When a fully manual mode is selected, the information flow followed by a clearance management is depicted by Fig. 8.

Each operational event is broadcast by the corresponding federate to the federation. The IOC federate subscribes to the "operational event" HLA object class, and collects all time-stamped messages for post-processing.

Instructions management follows a quite similar information flow, represented in Fig. 9 (manual management mode). Therefore, in fully automatic mode, when either the pseudo pilot or the controller selects an instruction to a given flight, this instruction is immediately performed by the TR federate. Instructions include stopping or starting a given flight to avoid traffic congestion or conflict. When a fully manual mode is selected, an instruction issued by the controller has to be acknowledged by the pseudo pilot before execution by the TR federate.

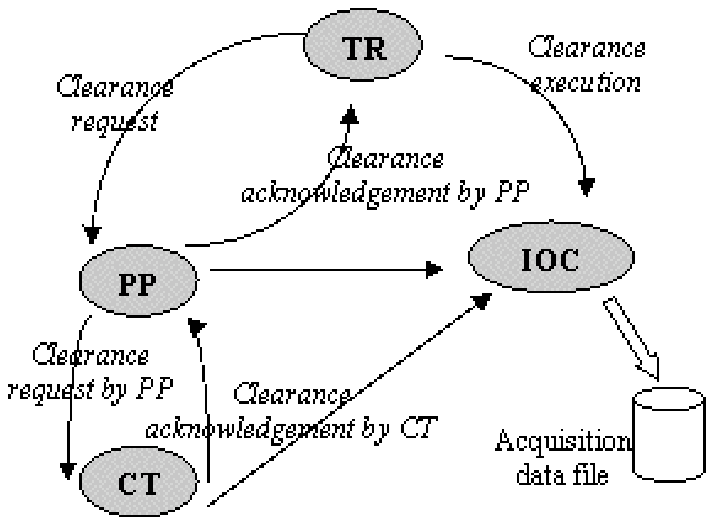


Fig. 8 Information flow for clearances management.

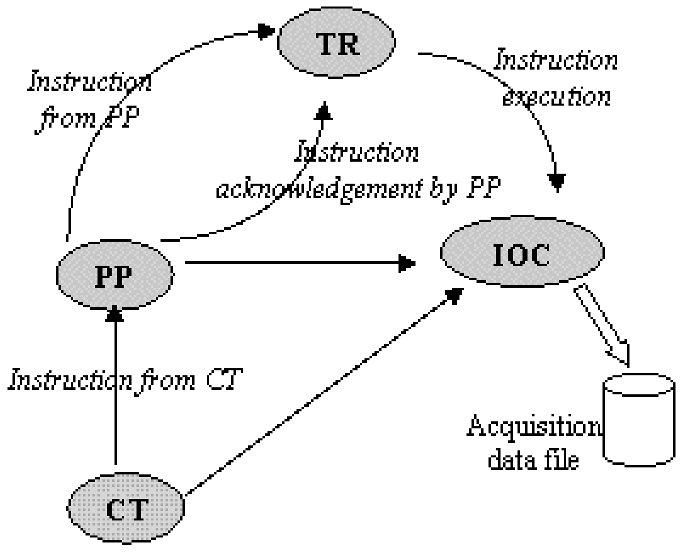


Fig. 9 Information flow for instructions management.

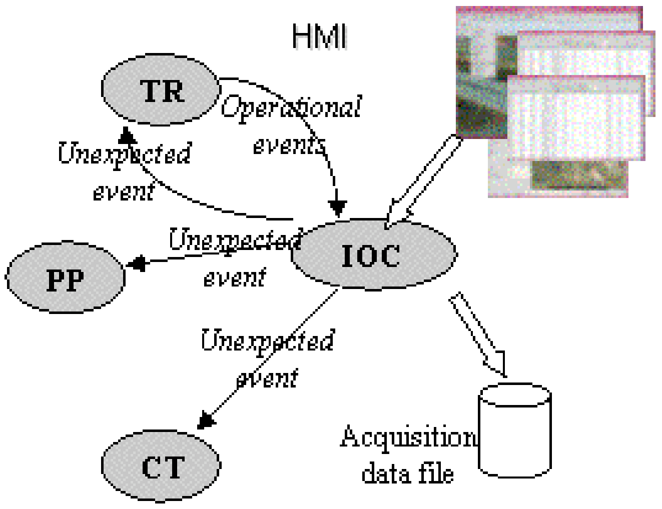


Fig. 10 Information flow for unexpected events.

Available management modes of both clearances and instructions, allow the user to simulate several kinds of operational incidents occurring during the communication protocol between the control tower and pilots. For example, the pseudo pilot has the possibility of crossing a runway without authorization from the control tower. In a similar way, an instruction provided by the control tower should be ignored by the pseudo pilot.

An acquisition session consists in simulating the expected scenario according to the management modes described previously. Nevertheless, in the actual world, several unexpected events may unfortunately occur. The HMI of the modeling and simulation platform is also dedicated to introducing such events. For example, flight A763 (expected landing time, 06:50:00 a.m.) should be delayed at 8:00:00 a.m. due to air traffic congestion. We assume that such delay is known at 06:40:00 a.m. In this case, upon arrival of flight A763, the following flight A3147 will be automatically delayed at 08:31:00 a.m. to allow ground operations such as refueling. The corresponding operational events will then be the following:

```
06:40:16: Flight A763 arriving at 06:50:00 has been delayed at 08:00:00
07:58:31: Flight A763 landing
07:58:39: Flight A763 touching runway after landing
08:00:06: Flight A763 exiting runway after landing
08:00:20: Flight A763 requesting RUNWAY_CROSSING
08:00:21: Flight A763 is cleared for RUNWAY_CROSSING
08:00:22: Flight A763 acknowledges RUNWAY_CROSSING
08:02:26: Flight A763 stopping at GATE
08:02:26: Flight A3147 departing at 07:25:00 will be delayed at 08:31:00 due to late arrival
```

The information flow associated with unexpected events management is depicted in Fig. 10.

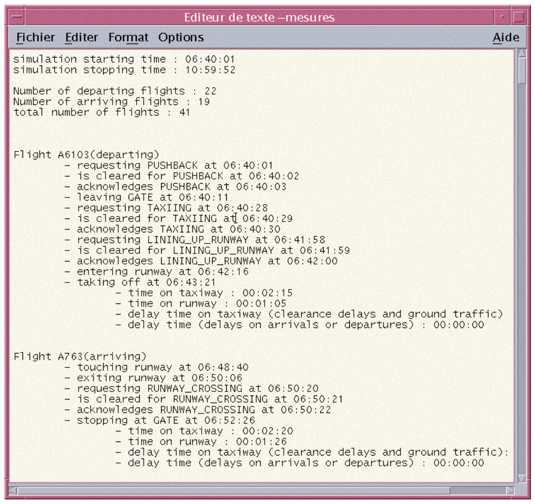
V. Analysis Issues and Software Facilities

The IOC federate has the responsibility of collecting several pieces of information and collecting them into a set of files: an acquisition data file storing all time-stamped operational events occurring during the acquisition process, and a replay file storing at any time, the locations of all aircraft moving on the airport layout. Those locations are sent by the VIS federate and collected by the IOC federate. Two kinds of software facilities are then available for analysis of the acquisition results collected during the simulation, by the IOC federate.

A. Capacity Measurements

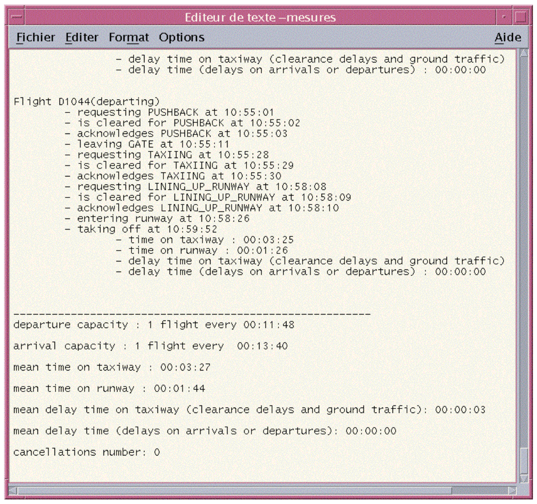
First of all, a post-processing software analyzes the acquisition data file and calculates some metrics related to each flight and the global airport system.

Flight metrics (Fig. 11a) include a description of all operational events related to the flight (requested and acknowledged clearances, instructions, operational events), as well as the time on taxiway and runway, and delay times (clearances delays and delays on arrival or departure).



```
simulation starting time : 06:40:01
simulation stopping time : 10:59:52

Number of departing flights : 22
Number of arriving flights : 19
total number of flights : 41



Flight A6103(departing)
        - requesting PUSHBACK at 06:40:01
        - is cleared for PUSHBACK at 06:40:02
        - acknowledges PUSHBACK at 06:40:03
        - leaving GATE at 06:40:11
        - requesting TAXIING at 06:40:28
        - is cleared for TAXIING at 06:40:29
        - acknowledges TAXIING at 06:40:30
        - requesting LINING_UP_RUNWAY at 06:41:58
        - is cleared for LINING_UP_RUNWAY at 06:41:59
        - acknowledges LINING_UP_RUNWAY at 06:42:00
        - entering runway at 06:42:16
        - taking off at 06:43:21
                - time on taxiway : 00:02:15
                - time on runway : 00:01:05
                - delay time on taxiway (clearance delays and ground traffic)
                - delay time (delays on arrivals or departures) : 00:00:00


Flight A763(arriving)
        - touching runway at 06:48:40
        - exiting runway at 06:50:06
        - requesting RUNWAY_CROSSING at 06:50:20
        - is cleared for RUNWAY_CROSSING at 06:50:21
        - acknowledges RUNWAY_CROSSING at 06:50:22
        - stopping at GATE at 06:52:26
                - time on taxiway : 00:02:20
                - time on runway : 00:01:26
                - delay time on taxiway (clearance delays and ground traffic):
                - delay time (delays on arrivals or departures) : 00:00:00
```

a)



```
                - delay time on taxiway (clearance delays and ground traffic)
                - delay time (delays on arrivals or departures) : 00:00:00


Flight D1044(departing)
        - requesting PUSHBACK at 10:55:01
        - is cleared for PUSHBACK at 10:55:02
        - acknowledges PUSHBACK at 10:55:03
        - leaving GATE at 10:55:11
        - requesting TAXIING at 10:55:28
        - is cleared for TAXIING at 10:55:29
        - acknowledges TAXIING at 10:55:30
        - requesting LINING_UP_RUNWAY at 10:58:08
        - is cleared for LINING_UP_RUNWAY at 10:58:09
        - acknowledges LINING_UP_RUNWAY at 10:58:10
        - entering runway at 10:58:26
        - taking off at 10:59:52
                - time on taxiway : 00:03:25
                - time on runway : 00:01:26
                - delay time on taxiway (clearance delays and ground traffic)
                - delay time (delays on arrivals or departures) : 00:00:00



-------------------------------------------------------
departure capacity : 1 flight every 00:11:48

arrival capacity : 1 flight every  00:13:40

mean time on taxiway : 00:03:27

mean time on runway : 00:01:44

mean delay time on taxiway (clearance delays and ground traffic): 00:00:03

mean delay time (delays on arrivals or departures): 00:00:00

cancellations number: 0
```

b)

Fig. 11 Metrics: a) Flight, b) Global system.

Global metrics (Fig. 11b), provide efficiency measures through a set of metrics, including departure and arrival capacity, mean time on taxiway and runway, mean delay time on taxiway due to clearance delays and ground traffic, and mean delay time on arrival or departures due to air traffic control (slots, cancellations, …).

B. Three-Dimensional Visualization and Replay

An interesting facility is offered by a software package intended to replay the simulation through a three-dimensional visualization of ground movements, using a synthetic environment. The architecture of this software package is shown in Fig. 12.

The first software module trajectory module (TRA) generates a set of *n* trajectory files from the simulation replay file. Each trajectory file contains time-stamped records storing the locations of each flight at any time. Run time replay involves two software modules, a monitor, and a GPF module, including both a viewer and an HMI written in Qt, a C++ toolkit for multiplatform GUI and applications development. All three-dimensional objects are built around several hundred polygons. During the simulation replay, the user has the possibility to select a given movement of the scene observer through the HMI.

The monitor allows the user to select a given three-dimensional database of the airport layout, the type of aircraft, and a set of trajectories to be replayed. The module prepares the simulation replay by loading the selected trajectory files, manages the evolution of the aircraft according to time advancement, and then transmits them to the GPF module.

Finally, the GPF module displays the aircraft on the three-dimensional airport data file. It uses the mouse to move the position of the observer within the synthetic environment. The GPF module uses a C++ API built around the performer library under the Linux Red Hat operating system.

Communications between the monitor module and the viewer are based on shared memory.

Figures 13a and 13b provide two screen captures of the three-dimensional simulation replay using a three-dimensional database of the Toulouse Blagnac airport. In Fig. 13a, the observer has been fixed just in front of an aircraft taking off, while an arriving flight is moving on the taxiway to the parking area. The second figure provides a global view of the terminal area, showing a departing flight moving on the taxiway.

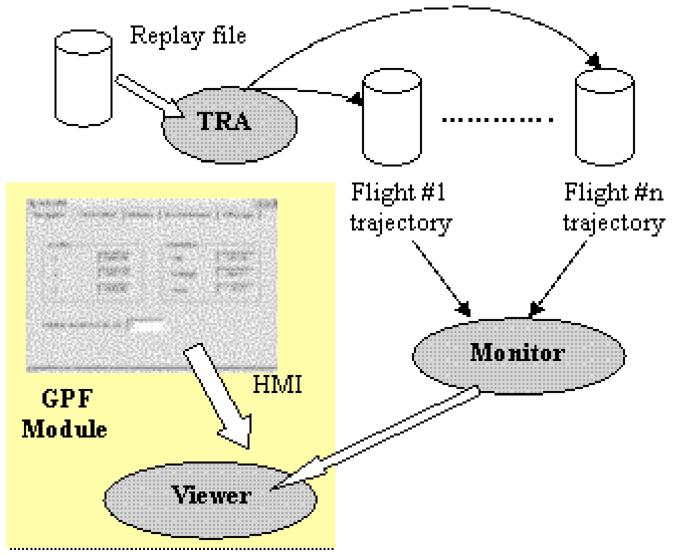


Fig. 12 Three-dimensional replay software architecture.

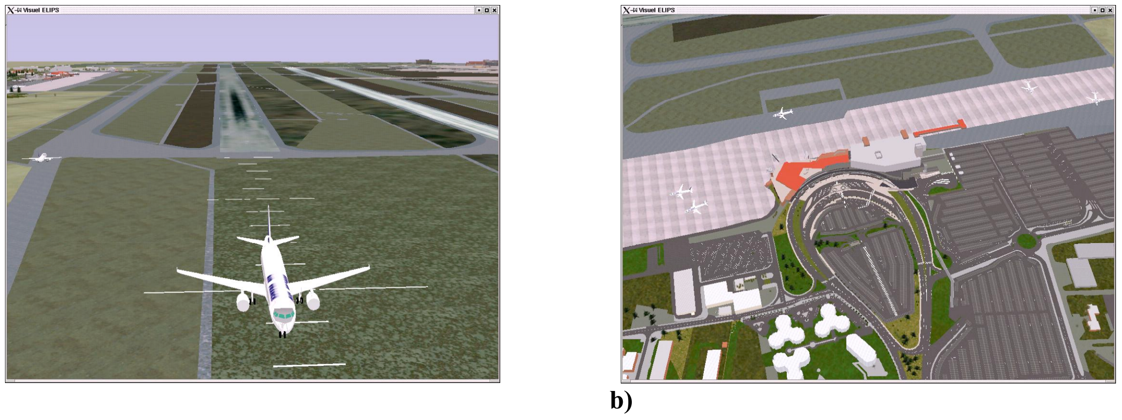


Fig. 13 Three-dimensional replay screen captures.

VI. Plugging Additional Components

As explained before, the modeling and simulation infrastructure has been developed around a reusable HLA federation, aimed at easily integrating additional federates for evaluating benefits of decision-making equipment or new procedures. Integration of new federates participating in a given airport federation needs the knowledge of the HLA FOM[11] associated with the reusable federation. In few words, a conventional FOM describes all class attributes that are shared by the federation. To reuse the federation, we have introduced within the FOM, some class attributes that are not shared by the federation. Consequently, those attributes may be either published by a given federate, or subscribed to by another federate. A detailed description of the FOM is beyond the scope of this paper.

To illustrate how an additional federate can be plugged, let us take an example. Assume that a model of a passive conflict detection equipment is available. The federation FOM includes a subscribed HLA object class, called AdvisoryClass. This class, described by a single attribute (a string), is subscribed by both the CT federate, and the PP federate. Obviously, locations of simulated aircraft are published periodically by the Traffic federate. The additional federate then has to subscribe to the aircraft locations, and to publish the AdvisoryClass

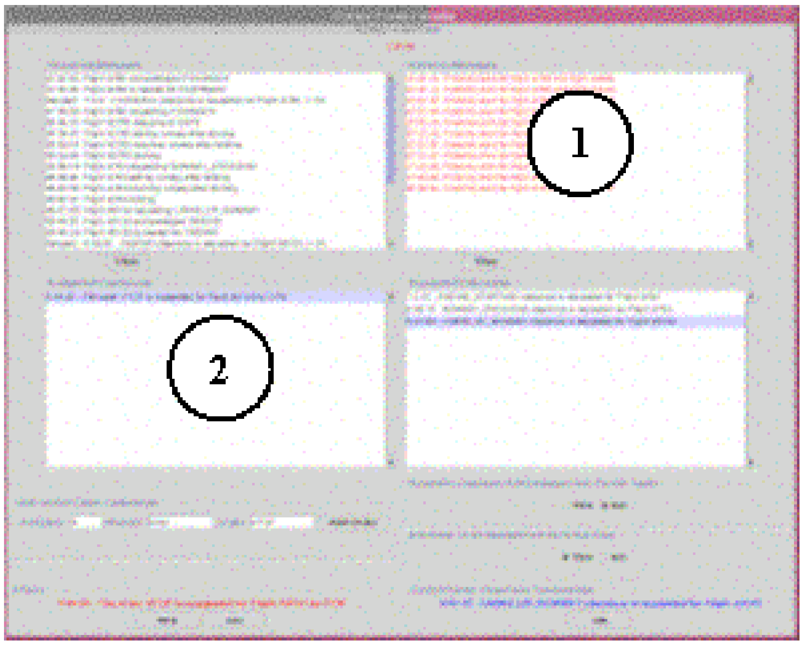


Fig. 14 The CT interaction console.

class. When the single attribute of an instance of this class, is updated by the federate, both the CT and PP federates display the string on a dedicated window of the corresponding interaction console.

As an illustration, Fig. 14 shows the CT interaction console. Window 1 is dedicated to advisory messages published by federates updating the attribute of the AdvisoryClass class. Those messages are similarly displayed by the PP federate. In such a way, advisory messages or conflict alerts may be broadcast to both pilot and controller, according to the conflict detection model supported by the additional federate.

Increasingly levels of assistance or automation can also be modeled and simulated through an HLA object class, called InstructionsClass. Two subclasses inherit of the InstructionsClass class: SuggestedInstructionsClass and OrderedInstructionsClass.

The first one, subscribed to by both the CT and PP federates, is dedicated to instructions published by federates simulating conflicts identification and solving. Those messages are displayed in Window 2 of the CT interaction console. Any instruction suggested by the federate, to solve the identified conflict, has to be validated by either the pilot or the controller, before to be executed by the Traffic federate.

On the contrary, when the attributes of an OrderedInstructionsClass instance, are updated by an additional federate, the corresponding instructions are immediately performed by the Traffic federate, without acknowledgment from neither the pilot nor the controller. Such instructions have thus to be considered as orders coming from fully automatic conflict resolution models.

VII. Concluding Remarks and Further Developments

The aeronautical world expects air traffic to further grow significantly over the next decade, leading to more congestion and delays. Airports could be then the primary source of congestion delay in 2015. Improvement in both ground operations and decision-making equipment is required to support a more sophisticated air traffic management. Accurately simulating the ground segment is thus crucial, because with simulation, it is feasible to test an infinite number of situations where operational conditions, traffic, and influent system parameters are varying in search for optimized performance and capacity. As heavy simulations can nowadays be achieved on a single personal computer, this simulation-based strategy becomes affordable, especially when the design has been modular enough to limit to a few parts, the adaptations required by every further application. The main focus of the work described in this paper has been to achieve these requirements.

Our main concern was to address interoperability and reuse at the functional level of the target transportation application. Preliminary results are encouraging and show that the kernel federation can support additional federates without extensive effort. Nevertheless, HLA is an abstract framework promoting reusability and interoperability of simulation components. Reusability means that simulation models can be reused in different applications. Interoperability means that several components can be combined on distributed heterogeneous computing platforms without recoding. While reusability is achieved through the OMT, achieving full interoperability needs an implementation level point of view, given by the HLA RTI. Preliminary lessons learned show that scalability and interoperability are achieved by both the HLA services and the run-time infrastructure.[4,6] Nevertheless, experience feedback suggests that reusability is much more difficult to achieve. The main reason is that we cannot ensure that a given federate participating in a federation F1 will also be appropriate in participating in a federation F2. In many situations, reusing the federate will need recoding some parts of the federate, including the interface shared with the federation.

Current technical effort focuses on these critical issues. More precisely, we are developing an abstract higher level architecture hiding the low-level services of HLA, and thus facilitating, the design and/or reuse of federates, as well as HLA federations management.[15]

Acknowledgments

This work is sponsored by the French Aeronautics and Space Research Center (ONERA). The authors would like to thank all contributors to this effort, especially all students that were in charge of the design of the simulation platform tools. Our industry partner OKTAL-SE (Synthetic Environment), owner of the three-dimensional Toulouse Blagnac Airport database, is also acknowledged. Finally, we acknowledge David Reitz from the Toulouse Blagnac control tower for his active contribution.

References

[1]Gotteland, J.-B., Durand, N., Alliot, J-N., and Page, E., "Aircraft Ground Traffic Optimization," *4th FAA/Eurocontrol R&D Conference*, Dec. 2001.

[2]Adelantado, M., Guez, M., and Lesot, B., "A Scalable Modeling and Simulation Infrastructure for Fast-Time Airport Simulation," AIAA Paper 2001-4132, Aug. 2001.

[3]Adelantado, M., "Airport Simulation Using the High Level Architecture," *2001 European Simulation Interoperability Workshop*, 25-27 June 2001.

[4]Adelantado, M., and Brunet-Crespin, M., "Sensor Modeling for Airport Simulation through the High Level Architecture," *IASTED International Symposium on Modeling and Simulation*, 16-18 May 2001.

[5]Adelantado, M., Damioli, C., and Dedieu, A., "A Reusable HLA Federation for Simulation of Airport Traffic Assistance/Automation Systems," *5th IEEE Conference on Intelligent Transportation Systems*, 3-6 Sept. 2002.

[6]Adelantado, M., and L'Hernault, R., "Gate-to-Gate Simulation for Validating Collaborative Decision Making Applications," *Advanced Simulation Technologies Conference, Business and Industry Symposium*, 14-18 Apr. 2002.

[7]Kuhl, F., Weatherly, R., and Dahman, J., *Creating Computer Simulation Systems: An Introduction to the High Level Architecture*, Prentice Hall, 1999.

[8]*High Level Architecture RTI-NG Programmer's Guide*, Department of Defense, Dec. 1999.

[9]Offerman, H., "Simulation to Support the Airport Stakeholder Decision-Making Process," *Air and Space Europe*, Vol. 3, 2001.

[10]Fujimoto, R. M., and Weatherly, R. M., "Time Management in the DoD High Level Architecture", Workshop on Parallel and Distributed Simulation, May 1996

[11]U.S. Department of Defense, "High Level Architecture Object Model Template Specification," Feb. 1998.

[12]Odoni, A. et al., "Existing and Required Modeling Capabilities for Evaluating ATM Systems and Concepts," Final Rept. Modeling Research under NASA/AATT, 1997.

[13]Schulze, Th., Strassburger, S., and Klein, U., "Migration of HLA into Civil Domains: Solutions and Prototypes for Transportation Applications," *Simulation*, Vol. 73, No. 5, Special Issue on the High Level Architecture for Simulations, Nov. 1999, pp. 296-303.

[14]Proceedings of the 3rd ATM R&D Symposium on "External Technologies in Support of Air Transport Efficiency", Madrid (Spain) 17-19 June 2002

[15]Cazard, L., and Adelantado, M., "HLA Federates Design and Federations Management: Towards a Higher Level Object-Oriented Architecture Hiding the HLA Services," *Spring Simulation Interoperability Workshop*, 10-15 Mar. 2002.

[16]*Proceedings of the 4th FAA/Eurocontrol R&D Conference*, Dec. 2001.