Hybrid Simulation of Distributed Large-Scale Critical Infrastructures

Massimo Ficco
Department of Industrial and Information Engineering
Second University of Naples
Via Roma 29, Aversa (CE), Italy
massimo.ficco@unina2.it

Giovanni Avolio, Luigi Battaglia and Vittorio Manetti
SESM S.C.A.R.L
Via Circumvallazione Esterna di Napoli
I-80014, Giugliano (NA), Italy
{gavolio, lbattaglia, vmanetti}@sesm.it

***Abstract*—The distributed and complexity nature of modern critical infrastructures that have to provide integrated services through the interoperability of heterogeneous subsystems, even spread among different countries, require new methodologies and tools to dominate overall systems complexity. In particular, in order to get knowledge about their real behavior and define dependability improvement actions, such complex and distributed systems should be reproduced and simulated locally. On the other hand, the extraordinary large number of their components cause a large-scale of the resulting model, limiting its resolution by current simulators. This paper presents a framework to implement hybrid simulation of distributed large-scale critical infrastructures, such as Air Traffic Control (ATC) and Vessel Traffic System (VTS). High Level Architecture (HLA) has been introduced into the engine simulations platform as its design and development foundation, whereas cloud-based virtualization techniques have been exploited in order to reproduce the overall distributed system on a local adaptive testbed. The use of such a framework can result in a considerable reduction of costs in all the system life phases, as well as an increased system dependability level.**

***Index Terms*—Hybrid simulation; critical infrastructures; network emulation; HLA; cloud computing.**

I. Introduction

Critical infrastructures represents the pivotal assets and resources upon which the current society greatly relies to support welfare, economy and quality of life. Nowadays, the trend is to restructure these infrastructures by applying a System of Systems (SoS) concept, where the sparse islands are progressively interconnected by means of proper middleware solution through wide-area networks. The huge complexity of such systems makes more complicated for designers and developers the task of facing integration and configuration issues of both pre-existing and under development systems. Indeed, integration among components may introduce unexpected system behaviors on dependability and performance that usually manifest during systems installation and execution time. Additionally, as they cannot be detected earlier, they require on-site maintenance operations resulting in increased maintenance costs and overspending in terms of personnel resources. Such systems open new scenarios in which the companies have to deal with both third party (COTS, Commercial Off-The Shelf) and in-house (named target hereafter) developed components [1].

For this reason, in order to dominate systems complexity and prevent exponential increase of costs, industries need novel solutions to face the migration from centralized and monolithic systems to open and distributed interconnected SoS. Academic and industrial research are investing significant effort in this direction, especially in the field of mission and safety critical systems which are required to exhibit predictable dependability behaviors.

A promising way to cope with these new systems, and to lower maintenance costs, is to reproduce such complex and distributed systems locally, and let them run prior to the actual execution on-site in order to get knowledge about their real behavior and define mitigation means and improvement actions. Hybrid and distributed simulation strategies, supported by novel technologies for resources virtualization and working environment reproduction, represent the most promising way to define the needed strategies to actually support such paradigm shift.

Therefore, this paper aims to propose an open-source framework to implement local testbed, representative of large-scale critical systems through the integrated use of distributed and hybrid simulation techniques. The framework is able to support the setup of simulation platforms, compliant with the reference standards, which act as a glue layer among simulated components and targets. The proposed solution intends to provide effective support to the daily industry work in all the phases of systems life cycle. The framework will allow reproducing on local testbeds, by combining simulation and emulation techniques, hardware and software working environments in which real systems are intended to run. On the one hand, this will enable systems designers to evaluate potential actions to undertake both for refining existing systems or designing new ones, considering system requirements evolution and operational environment changes. On the other hand, it will allow trying out maintenance operations needed to mitigate occurred system failures and/or to prevent estimated potential faults and security threats.

The framework provides configuration and emulation facilities, as well as virtualization capabilities to be exploited in order to reproduce the overall distributed system, on a local testbed. Interoperation between different simulated and emulated subsystem is achieved by using the High Level

Architecture (HLA) paradigm [2], which is a general purpose architecture for distributed computer simulation systems. The implementation is based on the open source poRTIco RTI tool, which is adopted as interoperation framework for the simulation of hybrid systems [3]. However, in order to overcome HLA shortcomings due to lack the capabilities of management and scheduling of simulation resources, especially during periods of peak load, as well as to emulate the network infrastructure, a cloud-based virtualization solution has been exploited. Finally, the air-to-ground warfare simulation system is taken as an example. The proposed solution has been designed in order to support testing and maintenance phases of the complex and critical infrastructure, such as Air Traffic Control (ATC) and Vessel Traffic System (VTS).

The rest of this paper is organized as follows. Background and related work are presented in Sec. II. Sec. III shows the functional view of the proposed framework. Sec. IV presents the simulation platform. The network emulation engine and the virtual infrastructure are presented in Sec. V and Sec. VI. Conclusions and future work are presented in Sec. VII.

II. Background and Related Work

A. VTS Simulation Scenario

VTS objective is to support safety and efficiency of navigation, as well as protection of the marine environment, adjacent shore areas, work sites and offshore installations from possible adverse effects of maritime traffic. A Commercial VTS system is designed to provide vessel traffic management and informative services to the maritime community. VTS supports the maritime community needs to monitor, control, interact and identify all the ships in ports and in the proximity areas, providing all the necessary informative contents, and showing an interactive 'traffic picture', updated on a real-time basis. Normally, a VTS system is organized into hierarchical levels, although it is not necessary for its proper operation, but rather to streamline its management. For our analysis, we refer to a VTS organized in three hierarchical levels, including:

- *VTS Local Control Centers* (VTSL), they are responsible for:
  - local traffic image compilation;
  - interaction with traffic;
  - port management;
  - local actions planning;
- *VTS Area Centers* (VTSA), they are responsible for:
  - fusion of VTSL data;
  - contiguous VTSL hand-over;
  - data link with patrol units;
  - search and rescue (SAR) planning;
- *Head Quarter* (HQ), they are responsible for:
  - distribution of Long Range Identification and Tracking (LRIT) [5], Vessel Monitoring System (VMS) [6] and legacy ship reporting systems;
  - nation and world wide traffic image compilation;
  - central archive;
  - overall SAR coordination.

In order to provide the expected traffic management services, the VTS architecture has to include at least two kind of sites: remote Sensor Sites and VTSLs. The former includes all the available sensors to track ship position and environment evolution. The latter is the first level of control, where operators can interact with the system. VTSs can integrate a wide variety of sensors (e.g., radars, Electro Optical Sensor (EOS), Automatic Identification System (AIS) [7], direction finders, etc.) and of external systems (e.g., LRIT, weather), allowing for data importing/exporting in many formats (e.g., ITU1371 supported both in version 1 and 3, NMEA). Starting from the tracks recognized from the Sensor Sites, VTSLs are responsible to process and store the system tracks. The VTS system is a database-centric system, therefore all the system elements (servers and operators) are able to establish remote connections with database.

In a distributed architecture, in which are involved more than one VTSL, each VTSL shares the same traffic global picture, and display it in a real-time application at the operator position. Based on this model, the VTS system can manage situations of network degradation or network breakdown among VTSLs in the distributed architecture, by means of 'disaster recovery' techniques.

B. High Level Architecture

High Level Architecture (HLA) was originally proposed by the US Department of Defense in 1995 as a general architecture for connecting several computer-based simulation systems, which is evolved to HLA 1516 in 2000 [2], [4].

However, although HLA was firstly developed for military applications, it has been widely used in non-military industries due to its many advantages. In particular, according to the HLA simulation paradigm, a $federation$ is a distributed simulation system for a particular purpose, which consists of a number of interactive federation members. All application programs that participate in simulation can be called *federate*. The interface specification of HLA describes how to communicate within the federation through the implementation of HLA specification: the Run Time Infrastructure (RTI). Federates interact using services implemented by the RTI. They can take on the role of *Publisher* to inform about an intention to send information to the federation, and *Subscriber* to receive some information created and updated by other federates. The information exchanged in HLA is represented in the form of classical object oriented paradigm. The two kinds of objects exchanged in HLA are *Object Class* and *Interaction Class*. Object class contains object-oriented data shared in the federation that persist during the run time; Interaction class data are just sent and received information between federates. These objects are implemented within XML format. In the HLA framework, the logical structure of a typical federal simulation is shown in Fig. 1.

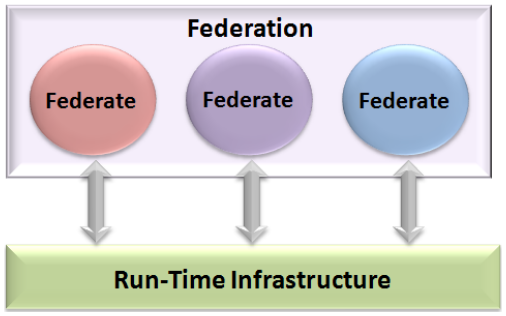

Fig. 1. Logic structure of HLA silmulation.

Although, HLA itself is not able to achieve full interoperation, but it defines the system structure and mechanisms, which means by running the RTI (Run-Time Infrastructure) to achieve interoperability. RTI implements all services of HLA interface specification, and provides a range of service functions which support interoperability of the federal members. It provides support service program and separates implementation of simulation functions, as well as the simulation running management and the underlying communication transport.

C. Related Work

In the past years, some effort has been made to simulate hybrid systems. Most of the research has concentrated on specifying a unified methodology for hybrid stand-alone simulation. The standalone simulation of hybrid systems means that the systems are modeled and simulated by only one modeling tool or methodology. For example, the SimEvents toolbox supports MATLAB/Simulink to perform discrete event simulation, and the Event Translation block that enables communication between SimEvents and other continuous operational blocks [8]. Ptolemy II is a computational framework for embedded systems that focuses on concurrent systems, and HyVisual is a hybrid system modeling tool that provides a visual syntax [9]. AnyLogic is a tool for modeling and simulating hybrid systems and a way of HLA support integration in the tool simulation engine [10]. PowerDEVS proposes discrete event methods for the numerical integration of ordinary differential equations (ODEs) [11]. These approaches commonly propose the use of a unique language for the specifications of the overall system, which consists of different model types. Therefore, existing models may not be reusable.

Some research has simulated models separately. Examples of integrated simulation methodology are expressed by MATLAB/Simulink interfaces, such as CODIS [12] and BCVTB [13], and is applied to most of the hybrid control systems. CODIS is a framework for simulating continuous and discrete systems, and it has a co-simulation bus to integrate a SystemC model and a MATLAB/Simulink model. BCVTB is based on the Ptolemy II software environment, and it integrates with a MATLAB/Simulink model. These methodologies use a specific interface between two models, such as Inter-Process Communication (IPC) and function calls. The two models are executed using such an interface in one process and tightly coupled. Thus, certain limitations have existed on the simulation of models developed in different modeling environments.

Agent-based simulations have recently become a popular way to model and study complex multi-actor systems, complementing the long-established system dynamics and discrete-event simulation approaches [14], [22]. Agogino [15] uses agent-based simulation for incremental enhancements of the air-traffic management (ATM). Krozel [16] uses the Future ATM Concepts Evaluation simulation Tool (FACET) to model air-traffic in inclement weather. Gorodetsky [17] uses an agent-based simulation for ATM within the air-space of large airports. Hwang [18] uses simulation for verification of collision avoidance algorithms. Finally, the AgentFly [19] system built using the AglobeX Simulation platform aims to provide a model of the entire air traffic.

Another simulation method consists of the interoperation between discrete event system models and continuous system models using HLA/RTI. The models are spatially separated explicitly. In a few studies, the HDEVSimHLA [20] framework provides an HLA/RTI interface for interoperating the DEVS models and the MATLAB/Simulink models. The framework uses the time management service of HLA/RTI for time synchronization between heterogeneous models, and it uses the analog-event and event-analog functions for data exchange. In addition, AnyLogic [21] and MATLAB provide HLA/RTI interfaces as a package of the application. Developers have to use the HLA support modules and implement new interface codes in each model. This interoperation methodology is most efficient method for reusing existing models and modeling systems in different environments.

III. The Framework Functionalities

Figure 2 shows a functional view of the proposed framework. It provides basic functionalities at system-level, and services at user-level accessed by special API and GUI for configuring and managing the entire simulation process.

At system-level the involved logical components are:

- *Simulation Manager* - It is enabled to configure, manage and monitor the process of hybrid simulation. It allows to define the system behavior in terms of simulated interactions between the involved subsystems, that best identify the simulated application scenario. It interfaces with the Configuration Manager to offer its capabilities at user-level in order to operate on the simulation process at real-time.
- *Time/Event Manager* - In order to simulate distributed environments, it is necessary that the individual involved components perceive the progress of time in a uniform manner, regardless of their world of origin (real, emulated, simulated). Time/Event Manager is responsible for the advancement of time that should be transferred to the real and emulated components downstream of an alignment to the simulation time which, therefore, constitutes the reference variable. The progress of time between simulated and emulated components is based on

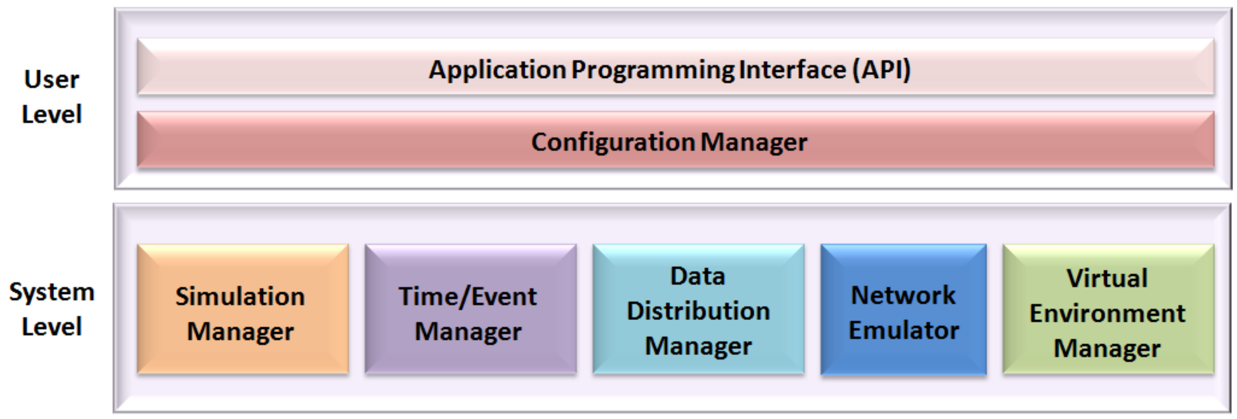

Fig. 2. Functional view of the proposed framework.

events. To be successful in sharing among the various components are implemented wrappers for the real systems, which will be waiting for specific events received by the involved components, and interact with them for the time progress. To enable full synchronization of all components, Time/Event Manager is responsible for receiving all the time events by simulators and select the smallest non-negative value, which is sent to all components and wrappers as the actual time to run.

- *Data Distribution Manager* - It manages the interaction between the parties in the context of simulation process, which requires the constant exchange of information between the components involved in the simulation, for the distribution data and synchronization information.
- *Network Emulator* - Given the nature of network-centric systems, it is adopted an emulated network infrastructure in order to reproduce reality with a high degree of verisimilitude. The service is implemented by emulation techniques based on virtualization technologies.
- *Virtual Environment Manager* - This component is enabled to dynamic deployment, management and monitoring of virtual resources necessary for implementation and orchestration of the local testbed. The virtual environment manager exposes its functionality to the Configuration Manager, through a dedicated interface for configuring virtual infrastructure resources by the end user.

At user-level specific interfaces are provided to configure the simulation process for both its static behavior (i.e., at the architecture level) and dynamic (i.e., at the level of functional process). Specifically, the user through descriptors can describe the structure of exchanged data and the interconnections of the simulated components. This descriptors are provided by the Configuration Manager to the Simulation Manager, at the system-level, in order to realize the connection between the components according to user requirements. Similar operations can be performed to describe exchanged data, interactions and synchronization among the subsystems. Specific interfaces are provided to configure the virtual infrastructure.

IV. Application of HLA in the Simulation Platform Engine

The proposed simulation platform aims to provide an expandable hybrid simulation environment, which allows to integrate and make interoperable different models of simulation.

As a framework for advanced distributed interactive simulation, HLA meets the needs of the proposed simulation platform engine, such as software re-usability, interoperability and extensibility of large-scale systems. HLA provide standardized guidance for development of simulation engine for large-scale critical systems.

Therefore, according to the architecture (HLA) framework, the Simulation Manager provides mechanisms for specifying the exchange of data and coordination among members of the federation. It uses a common, standardized formalisms for describing the capabilities of potential federation members. In particular, the subsystem (federate) to be simulated are modeled according to the HLA object model. Each federation member is represented by an HLA simulation object model (SOM), whereas the iterations among the federates is described by the federation object model (FOM). SOM is a specification of the types of information that an individual federate can provide to HLA federations and the information that an individual federate can receive from other federates. FOM is a file that contains a description of the data exchange in the federation, for example the objects and interactions that will be exchanged. This can be seen as the language of the federation. During a federation execution, all exchange of FOM data among federates shall occur via the RTI. Moreover, RTI specifies a group of the interface services to support members of the federate in accordance with the provisions of the FOM. Federates shall use these standard interfaces to interact with the RTI.

In order to be able to correctly exchange simulation data among federates that evolve according to a different temporal model (real time, emulated, and simulated), the Time/Event Manager exploits the RTI to coordinate how fast the simulators advance logical or scenario time in which the correct delivery of data is based on time stamped. Moreover, it allows to define synchronization points that enables the members of the federation to coordinate when they have reached a certain state, for example, when they are ready to start the simulation of the next phase of a scenario. Finally, in the federation, the model of data exchange is implemented by the Data Distribution Manager by exploiting the publish/subscribe paradigm offered by the RTI.

The Real-Time Infrastructure is based on PoRTIco, which is an open-source cross-platform HLA RTI implementation. PoRTIco is a modularity and flexibility platform, which provides a production grade RTI environment that can support continued research and development.

A. Scalable Large-Scale Simulated Architecture

According to the HLA standard, the RTI system uses a centralized manager to provide the RTI services, such as joining of federates, data exchange, synchronization of federates, publishing/subscription management, etc. On the other hand, in a complex large-scale simulation, with a large amount of federates like that presented in Sec. II-A, a centralized Simulation Manager would tend to be overloaded because of busy processing the requests of the federates, degrading the simulation performance.

For this reason, we design a multi-layer architecture. Specifically, federates are partitioned into several groups. Each group is managed by a local Simulation Manager. All Simulation Managers cooperatively manage the execution of the whole federation. In particular, in order to coordinate the federation, each Simulation Manager needs to synchronize information with each other when states update. For example, when a federate publishes an object class, the corresponding local Simulation Manager will synchronize this information with other Simulation Managers. We assume that, each Simulation Manager can communicate with each other.

When a federate joins to a local Simulation Manager, the RTI system must assign to the federate a unique ID in the simulation environment. Therefore, in order to get a globally unique ID in such distributed architecture, we use a two layer ID assignment scheme [23]. In particular, the Simulation Manager assigns to the requester federate, a pair of numbers of the form [*Manager_ID*, *Federate_ID*], where the Manager_ID is the ID of the local Simulation Manager, whereas the Federate_ID is a unique ID in the group of the local Simulation Manager. After the assignment process, the local Simulation Manager broadcasts the ID to other Simulation Managers, thus each Simulation Manager can know that a new federate has joined the federation.

The first Simulation Manager created in the federation becomes the master, and it will be assigned to ID 0. The other Simulation Managers created after master, in the same federation, will become slave with ID starting from 1. To generate Federate_ID for each federate, every Simulation Manager maintains a local federate ID counter starting from 1.

Moreover, when a federate requests the RTI services, the message has to be multicast to federates which are interested in this event. The multicast operation consists of the following steps. When the source federate sends the message to a federate of the same group, the local Simulation Manager becomes the source, which determines the federates that will receive the message according to the publish/subscribe relationship. If the message is directed to a federate of another group, the local Simulation Manager analyzes the Manager_IDs of the destination federates and finds the destination Simulation Managers that manages the destination federates. The local Simulation Manager sends the message to each destination Simulation Manager. Every destination Simulation Manager determines the local federates which will receive the message according to the publish/subscribe relationship. The multicast from the destination Simulation Manager to the destination federates in different groups can be done in parallel.

Finally, in order to coordinate the execution of the simulation processes with large amount of federates, we use the following synchronization schema:

- *Step 1*: In each group, the federates start the operation of synchronization. After the federates in a group finish the synchronization, the federate with the smallest Federate_ID will notify the Simulation Manager that the local synchronization is done.
- *Step 2*: The Simulation Managers start the operation of synchronization after they receive the notification from their local federates. When the Simulation Managers finish the synchronization, all federates are synchronized.
- *Step 3*: Finally, each Simulation Manager triggers the callback to the federates in its own group.

V. Network Emulation Engine

According to the commission of the European Communities, both ATM and VTS systems can be considered Critical Infrastructures, with particular reference to the transportation category [24], [25]. Such systems rely on a distributed organization, and consist in interconnected heterogeneous infrastructures usually spread over national areas. Both kind of systems are bound by strong international safety constraints to ensure 24-hours/365-days business-continuity.

Recent studies concerning the ATM field, highlighted a remarkable increase of air traffic in Europe, confirming in such a way the need to adopt new technologies and approaches for the design and implementation of innovative ATM systems. The Single European Sky ATM Research (SESAR) [26] stands for the most effective response of the ATM community to the aforementioned needs/challenges: within the European research program, new ATM operational concepts have been introduced to allow the management of the estimated ATM throughput. Concerning in particular the management of information, those concepts claim to implement a strong cooperation among the different ATM actors to allow sharing of relevant procedures, information, and infrastructures, through SWIM, the System Wide Information Management unit designed under the SESAR Research Program. In order to support SWIM and other new operational concepts, an heterogeneous IPv4/6 network backbone namely the Pan European Network Services (PEN), has been built up to cover the whole European area.

The network emulation concept plays a key role in the framework we propose. Realizing complex Wide Area Network (WAN) scenarios in line with the real ones, requires the adoption of a Network Emulation module. We investigated and evaluated a number of open source tools and projects in order to select the more suitable for our framework [27]–[30]. After an accurate comparison we selected the Common Open Research Emulator, namely CORE [31]. In addition to a number of common capabilities for a network emulation module, CORE is able to run: (i) over a guest Operating System in a Virtual Machine, allowing in such a way the adoption in Cloud Computing platform implementing the IaaS approach; (ii) in conjunction with the ns-3 Network Simulator. On the other hand, aiming at increasing flexibility on the management of the Data Link level in our IaaS platform, we introduced the OpenvSwitch kernel module (OVS) [33], that stands for one of the major solutions for the Local Area Network (LAN) emulation. We chose OVS since it supports the Open Flow protocol [34], an implementation of the Software Defined Networking (SDN) paradigm, which can be considered as a new way of thinking about the network. The framework we propose relies on the aforementioned tools. The adoption of CORE and OVS allow us to integrate real VTS and ATM subsystems into an emulated network environment, and to reproduce a relevant number of real operational scenarios.

VI. The Cloud Infrastructure

The proposed framework allows the deployment of simulation and emulation instances in a virtual cloud environment. In particular, the simulation cloud infrastructure is composed of a number of virtual nodes. Each federate is assigned to one or more cloud nodes on the implementation. Therefore, the Virtual Environment Manager is enabled to monitor real-time allocation of resources (by Nagios monitor services [32]), and to optimize dynamically the allocation of resources to deal with unexpected overload of the simulation process.

To this aim, a private cloud IaaS has been realized by means of the OpenNebula [35] open-source technology. Born as a research project in 2005, with the main purpose of designing and implementing an efficient and scalable management platform for virtual machines (VMs) on large-scale distributed infrastructures, the OpenNebula (ONE) package is now a fully open-source software distributed under the the Apache License Version 2.0. The ONE platforms assume that the underlying physical infrastructure is built up in line with the classical cluster-like architecture, where the hypervisor-enabled nodes are managed by a so-called front-end node, which is responsible for the centralized management and monitoring of all ONE services.

The front-end node manages the whole life-cycle of clusters, hosts, VMs, storage areas and other virtualized resources. The aforementioned IaaS platform has been realized on the top of a cluster consisting of 24 Dell PowerEdge M610 Blade servers, each of which equipped with two Quad-core Intel Xeon E5420 2.50GHz processors, 16GB of RAM memory, and four Gigabit Ethernet adapters. Two L3 Switching modules namely Dell M6220 provide network connectivity to the nodes.

Two L3 Switching modules, namely Dell M6220, provide the nodes, on which the Linux CentOS 6.4 is adopted, with high-speed Local Area Network (LAN) interconnections.

VII. Conclusions

In this parer, we presented an open-source framework to implement local testbeds, used to simulate large-scale critical systems. In particular, the proposed framework allows to perform (i) early analysis of design and development decisions on real targets, (ii) to change and adapt taken decisions according to system workload changes, and (iii) to plan and verify the effectiveness of maintenance operation for foreseen or already known faults, both spontaneous and malicious, thus optimizing maintenance time and costs.

The presented solution will result in two main benefits (i) a significant reduction of costs in all the system life phases, and (ii) an increased system dependability and security level to improve the software product quality. It can bring a deep innovation into industrial design and development processes, especially with respect to very large and critical systems.

In the future work, the framework will be validated on real world industrial case studies in the field of VTS, as well as training systems for several civil applications.

Acknowledgment

This work has been partially supported by the Italian Ministry for Education, University, and Research (MIUR) under Project PON02_00485_3487784 "DISPLAY" of the public-private laboratory "COSMIC" (PON02_00669).

References

[1] A. Venticinque, N. Mazzocca, S. Venticinque, and M. Ficco. Semantic support for log analysis of safety-critical embedded systems. in Proc. of the *10th European Dependable Computing Conference (EDCC'14)*, Newcastle, UK, May 13-16, 2014.

[2] IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA), Federate Interface Specification, Std 1516.2-2000. New York: Institute of Electrical and Electronics Engineers, Inc.

[3] poRTIco RTI tool, available at http://www.porticoproject.org/index.php?title=Main_Page. Last update April 2013.

[4] IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA), Federation Development and Execution Process (FEDEP), Std 1516.3-2003. New York: Institute of Electrical and Electronics Engineers, Inc.

[5] LRIT - Long Range Identification and Tracking, available at: http://www.imo.org/OurWork/Safety/Navigation/Pages/LRIT.aspx

[6] VMS - Vessel Monitoring System, available at: http://ec.europa.eu/fisheries/cfp/control/technologies/vms/index_en.htm

[7] AIS - Automatic Identification System, available at: https://www.itu.int/rec/R-REC-M.1371/en

[8] M. Clune, P. Mosterman, and C. Cassandras. Discrete Event and Hybrid System Simulation with SimEvents, in Proc. of the *8th International Workshop on Discrete Event Systems*, Jul. 2006, pp. 386-387.

[9] H. Zheng. Operational semantics of hybrid systems. *Ph.D. dissertation*, Berkeley, CA, USA, 2007, adviser-Edward A. Lee.

[10] A. Borshchev, Y. Karpov, and V. Kharitonov. Distributed simulation of hybrid systems with AnyLogic and HLA, in *Future Gener. Comput. Syst.*, vol. 18, no. 6, pp. 829-839, 2002.

[11] E. Kofman, M. Lapadula, and E. Pagliero. PowerDEVS: A DEVSBased Environment for Hybrid System Modeling and Simulation, School of Electronic Engineering, Universidad Nacional de Rosario, Tech. Rep. LSD0306, 2003.

[12] F. Bouchhima, G. Nicolescu, E. M. Aboulhamid, and M. Abid. Generic discrete-continuous simulation model for accurate validation in heterogeneous systems design, in *Journal of Microelectronicic*, vol. 38, no. 6-7, 2007, pp. 805-815.

[13] M. Wetter and P. Haves. A Modular Building Controls Virtual Test Bed for The Integration of Heterogeneous Systems. in Proc. of the *3rd National Conference of IBPS (SimBuild)*, Berkeley, USA, Jul. 2008, pp. 69-76.

[14] A. Borshchev and A. Filippov. From system dynamics and discrete event to practical agent based modeling: Reasons, techniques, tools, in Proc. of the *22nd International Conference on the System Dynamics Society*, 2004, pp. 25-29.

[15] A. Agogino and K. Tumer. Regulating air traffic flow with coupled agents, in Proc. of the *7th international joint conference on Autonomous agents and multiagent systems*, vol. 2, 2008, pp. 535-542.

[16] J. Krozel and N. Doble. Simulation of the National Airspace System in Inclement Weather, in *Modeling and Simulation Technologies Conference*, 2007, pp. 20-23.

[17] V. Gorodetsky, O. Karsaev, V. Samoylov, and V. Skormin. Multi-Agent Technology for Air Traffic Control and Incident Management in Airport Airspace. in Proc. of the *International Workshop on Agents in Traffic and Transportation*, 2008, pp. 119-125.

[18] I. Hwang, J. Kim, and C. Tomlin. Protocol-based conflict resolution for air traffic control. in *Air Traffic Control Quarterly*, vol. 15, no. 1, 2007, pp. 134.

[19] D. ilk, P. Volf, M. Jakob, and M. Pchouek. Agents for Games and Simulations, in *Distributed Platform for Large-Scale Agent-Based Simulations*, Lecture Notes in Computer Science LNCS, vol. 5920, 2009, pp 16-32.

[20] S. Y. Lim and T. G. Kim. Hybrid Modeling and Simulation Methodology based on DEVS Formalism, in *Sunner Computer Simulation*, Jul. 2001, pp. 188193.

[21] A. Borshchev, Y. Karpov, and V. Kharitonov. Distributed simulation of hybrid systems with AnyLogic and HLA, in *Future Generation Computing Systems*, vol. 18, no. 6, 2002, pp. 829-839.

[22] R. Aversa, B. Di Martino, M. Ficco, and S. Venticinque. A simulation model for localization of pervasive objects using heterogeneous wireless networks, in *Simulation Modelling Practice and Theory*, vol. 19, no. 8, 2011, pp. 1758-1772.

[23] Ding-Yong Hong, Fang-Ping Pai, Shih-Hsiang Lo and Yeh-Ching Chung. A Scalable HLA RTI System based on Multiple-FedServ Architecture, in Proc. of the *12th International Conference on Computer Modelling and Simulation*, 2010, pp. 527-532.

[24] G. Gigante, F. Gargiulo, and M. Ficco. A semantic driven approach for requirements verification, in Proc. of the *8th International Symposium on Intelligent Distributed Computing (IDC'14)*, Madrid, Spain, 2014.

[25] G. Zazzaro, G. Gigante, E. Zaccariello, M. Ficco, and B. Di Martino. Supporting development of certified aeronautical components by applying text analysis techniques, in Proc. of the *8th International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS'14)*, Birmingham, UK, 2014.

[26] SESAR - Single European Sky ATM Research, available at http://www.sesarju.eu/

[27] [NISTnet - National Institute of Standards and Technology, available at http://snad.ncsl.nist.gov/nistnet/

[28] KAUNET - Deterministic Network Emulation, available at http://www.kau.se/en/kaunet

[29] [Dummynet- Network emulation tool, available at http://info.iet.unipi.it/ luigi/dummynet/

[30] CORE - Common Open Research Emulator, available at http://www.nrl.navy.mil/itd/ncs/products/core

[31] J. Ahrenholz, T. Goff, and B. Adamson. Integration of the CORE and EMANE Network Emulators, in Proc. of the *IEEE Military Communications Conference*, Nov. 2011, pp. 1870-1875.

[32] W. Barth. Nagios System and Netwok Monitoring. William Pollock Editor. 2008. ISBN 978-1-59327-179-4

[33] OpenVswitch - An Open Virtual Switch, available at http://openvswitch.org/

[34] Mckoewn, Nick, et al. OpenFlow: enabling innovation in campus networks. In *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, 2008, pp. 69-74.

[35] OpenNebula, An user-driven cloud management platform for sysadmins and devops, http://opennebula.org/