Simulating Rail Traffic Safety Systems using HLA 1516

08E-SIW-069

Fred van Lieshout
Ferdinand Cornelissen
Jan Neuteboom
Atos Origin Technical Automation
Papendorpseweg 93
3528 BJ Utrecht, The Netherlands
fred.vanlieshout@atosorigin.com
ferdinand.cornelissen@atosorigin.com
jan.neuteboom@atosorigin.com

Björn Möller
Pitch Technologies
Nygatan 35
SE-582 19 Linköping, Sweden
+46 13 13 45 45
bjorn.moller@pitch.se

Keywords:
HLA, IEEE 1516, Rail Traffic Safety Systems, Rail Infrastructure, Simulation

ASTRACT: *The High Level Architecture (HLA [1]) has its origins in the defense sector and was primarily used to simulate vehicle and troops movements. However, the HLA is not limited to this area. It is very suitable for simulating rail traffic, in a simulated rail infrastructure, and the rail traffic safety systems that control that same infrastructure. By doing so, one is able to test the rail traffic management systems that act at the operator level.*

Although it is not the first time that a simulator is built for the rail traffic domain, the simulator described in this paper is unique in its modularity, scalability and flexibility, thanks to the use of HLA. It is flexible because each safety system is implemented as a federate. This means that large rail traffic areas can be simulated by simply instantiating the required number of federates, each protecting their own part of the rail infrastructure. Just as in reality. That said, plus the fact that using HLA allows distribution over multiple PC's, makes this simulator scalable.

This paper also describes which objects are simulated, which object attributes are published to the HLA Run Time Infrastructure (RTI) and what kind of design decisions were made during the development of the simulator.

1 Introduction

As in any other engineering area, modeling and simulation is an important tool in the railroad industry. And just like in many other areas there exist a large number of models within different industries and departments, but the power of making several simulations interoperate hasn't been taken to its full potential. Some examples of where modeling and simulation is used are for training of train operators, for evaluation of different designs and for test and verification purposes. This paper focuses on a simulator that is used by the Dutch railways to test their applications before being installed in the field. It describes how HLA has been successfully applied to create a modular, flexible and scalable architecture for simulation interoperability. As a result, it is now possible to test more complex configurations with combinations of multiple simulated safety systems. The benefit of this is that the test environment resembles the real world situation more closely and the correct operations can be verified to a higher degree than before.

1.1 The safety systems

Rail traffic safety systems ensure that only an authorized train may enter a part of the rail infrastructure at a given time. This is usually achieved by placing signals at the entry of a specific path of the rail infrastructure (see Figure 1). Dividing the path into sections and protecting each section with a signal increases the capacity of the railroad because multiple trains can make use of the available rail infrastructure, without the danger of collisions.

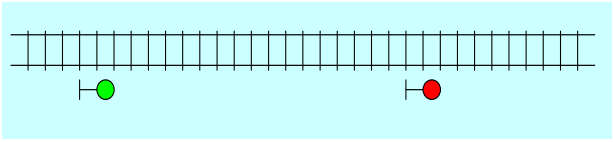

Figure 1: rail infrastructure example

When a train occupies a section, the signal shows red to indicate to another train driver that it is not allowed in that section. Of course this is a very simplistic description, but it is not too hard to imagine because of the resemblance with the traffic lights at ordinary roads.

There are many different rail traffic safety systems, developed in different countries and by various manufacturers. Each country has their specific requirements and each manufacturer developed their system as they thought was most appropriate. It is only until recent times that in Europe various countries have come to an agreement to implement one standard system called ERTMS, which stands for European Rail Traffic Management System. But it will take years before that system will be implemented and until that time there will be a variety of systems that has to be dealt with.

Below is a brief summary of some of the safety systems that are in use by the Dutch railroads.

NX: The Entrance/Exit safety system [2], abbreviated as NX, has been developed in North America and was adopted by the Dutch railways to secure their railroads. As the name implies, only one train can enter a section and the signal will only allow the next train until the first has left (exit) the section. It's a fail-safe system, implemented by electric relays. In The Netherlands a large number of railway yards is still secured by NX safety systems.

VPI: Vital Processor Interlocking [4] can be seen as the successor of the NX safety systems. VPI is a fail-safe, microprocessor-based control system designed to meet the needs of interlocking control for mainline railroads and mass transit applications. VPI essentially executes a program that consists of Boolean formulas that expresses dependencies between objects such as signals and points. Each railway station is supplied with its own set of formulas, based on the particularities of the rail infrastructure is has to secure.

EBS: Elektronische Beveiliging SIMIS, where SIMIS stands for 'Sicheres Mikrocomputersysteem Siemens'. It is a safety system that is functionally equal to NX and VPI. A variety of dialects of this system exists, each with a slightly different mechanism of handling logical safety rules.

ERTMS: European Rail Traffic Management System [3], a pan-European standard to be implemented over the next decades. At this moment only a small amount of the rail infrastructure is secured by ERTMS.

1.2 Automatic train security

On many railroad tracks a system called 'Automatic Train Interference' has been implemented. This system is activated when a train driver ignores a speed signal or red sign. The speed of the train will automatically be reduced and eventually the train will be stopped. For our simulation purposes, it has been decided that this functionality will not be considered.

1.3 Managing the traffic

There are a number of systems, in a layered architecture, to control the flow of trains that enter and leave railway stations (Figure 2).

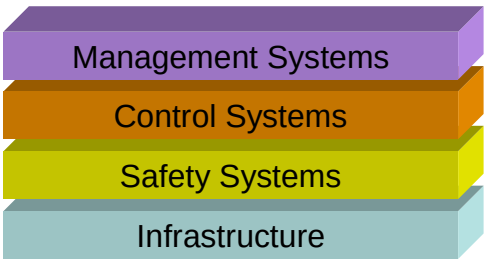

Figure 2: layered architecture

The functionalities captured in the components of this layered architecture range from rail utilization safety, automated train numbering, and train scheduling and planning.

1.4 Evolution of rail traffic management systems

Each of the components contained within the rail traffic management system tends to have a long development history: an increased utilization of the rail infrastructure leads to a continuous development of these components, while still retaining a guaranteed reliability. Furthermore, components are usually built by different manufacturers.

In order to guarantee a certain level of reliability, as well as to allow flexibility in the delivery of new versions of components, each component has to be properly tested before being integrated into the final operational environment. Given the overall complexity of the rail management system, this is not a straightforward task, and using simulation as the basic means of testing may be the only way to cope with it.

2 Modeling the real world

First it has to be decided which of the systems need to be simulated in order to meet the test goals. Since the safety systems themselves are generic, while they are configured with data that is specific for the rail infrastructure that needs to be secured, we decided to simulate the lower two layers, while maintaining the real systems in the upper two layers in the test environment. They are the 'systems under test'.

2.1 Federates

Because we want to have a model of the real world that should be intuitive to both the developers as well as the users of the simulator, we decided to have:

- A federate for each real-life instance of a safety system;
- One federate to simulate the behaviour and state of the rail infrastructure;
- One to simulate the trains; and
- One to control the whole simulation.

The latter also contains the Graphical User Interface, which provides the tester with the means to monitor and manipulate the simulation during execution. This is useful to, for example, introduce a signal or point machine failure and test the behaviour of the control and management systems. It can also be used to teach the traffic controller on incident scenarios.

The control federate is also used to create and remove instances of simulated trains in the federate that controls the simulated trains. All of the above federates are implemented as executables and together they form the HLA federation (Figure 3).

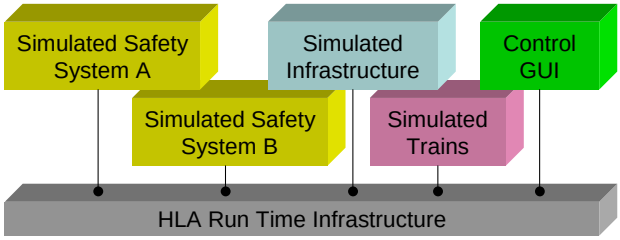

Figure 3: federation

The simulated safety systems connect to the control systems and use the same interface as their real-life counterparts, which usually is a protocol over TCP/IP and Ethernet.

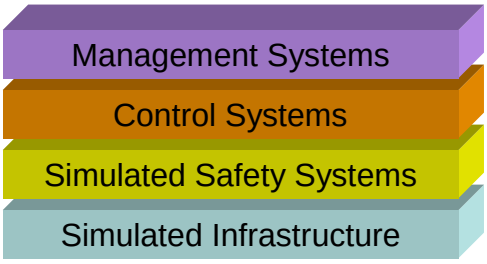

Figure 4: real world + simulated systems

2.2 Simulated Objects

It is quite straightforward to find the objects that have to be simulated, when using an Object Oriented approach. While keeping the requirements in mind, we came up with the following main objects (and their attributes):

- **Trains** (head and tail position, length, speed, target speed, number);
- **Routes** (id, from signal, to signal);
- **Points** (id, position, required position, failure status);
- **Sections** (id, occupied status, failure status);
- **Speed signs** (allowed speed);
- **Crossings** (id, status, failure status); and
- **Power supplies** (failure status).

By using an object hierarchy, we were able to structure approximately 20 objects into a logical class diagram. The next step was to translate this object hierarchy into the HLA Federation Object Model (FOM). For this purpose we used the 'Visual OMT' editor from Pitch Technologies (Figure 6), which helped in visually defining the object tree.

The spatial representation of trains is somewhat different from what is usually seen in for example aircraft simulation. A train has a considerable linear extension, which means that while the front end has left the platform the rear end may still be at the platform. This is somewhat similar to convoy models used in defense logistic simulations. The environment where trains move is somewhat easier to model than traditional environments. The railroad system is described as a set of interconnected links. In fact, the infra federate is capable of simulating all rail infrastructure in the Netherlands, which consists of more than 2 800 km of rails, more than 8 000 switches and roughly 10 000 signals.

2.3 Interactions

All communication between federates is either via object reflection, or via interactions. The latter are used to implement communication between simulation components, such as in commanding signals or in querying the modeled infrastructure.

The implemented interactions can be divided into three categories. In the first category we find all interactions that are related to command & control, e.g. instructing trains, infrastructure or safety systems, as well as monitoring the situation of the simulated trains and infrastructure. Examples of such interactions are:

- **ElementCommand:** issues a command to an element of the infrastructure such as a point machine or a signal;
- **TrainCommand:** issues a command to a train / driver.

Most interactions are usually a result of, or are in direct relation to, commands that are issued from the top two layers of the rail management system.

The common denominator of the second category of interactions is the fact that they are all related to driving trains. These interactions provide creation, destruction, speed, position and status modifications based on interaction between the trains schedule, the infrastructure, and the safety systems. The following interactions are exemplary for this category:

- **AllowedSpeed:** an interaction used in determining the maximum speed a train is allowed to travel on a track, given the signs, signals and order speed dictating elements;
- **TrackOccupied:** specifies if a certain section of the rail tracks is occupied or not. This triggers the safety systems;
- **NextTrainStop:** provides the location and distance to the next stop the train should stop;
- **NextRailTrack:** used in providing the next part of the railtrack the train is driving on, given the current position and status of the infrastructure.

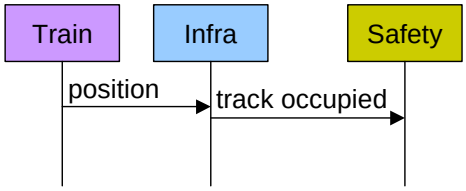

Figure 5: interactions between train schedule, infrastructure and safety system

A third category is related to interactions for simulation control such as introducing system malfunctions, status changes or adaptations to the infrastructure, train schedule or mechanisms of the safety systems:

- **PlaceTrain:** places a new train on the simulated rail tracks;
- **RemoveTrain:** removes the train from the simulation;
- **ModifyElement:** changes the characteristics of a certain infrastructure element.

3 Design decisions

3.1 Code generation

To speed up the development of the simulator, and to quickly adapt to alterations in the FOM, a code generator was developed that reads the FOM file – created by the Visual OMT tool – to generate C++ objects and a 'generic framework' of object reflection and interaction handling. It is one of our major design decisions made early on in the project to extend the level of generality of HLA into the communication layer of each federate. This resulted in the aforementioned generic framework. This has resulted in a major speed increase in the development of each of the federates. Furthermore, it provides a high flexibility in adapting to changes in the FOM during the whole project lifecycle.

A further advantage of the code generator and the generality of the FOM-editor is that it can be used in future HLA projects as well, since no domain-specific information is needed to generate the code.

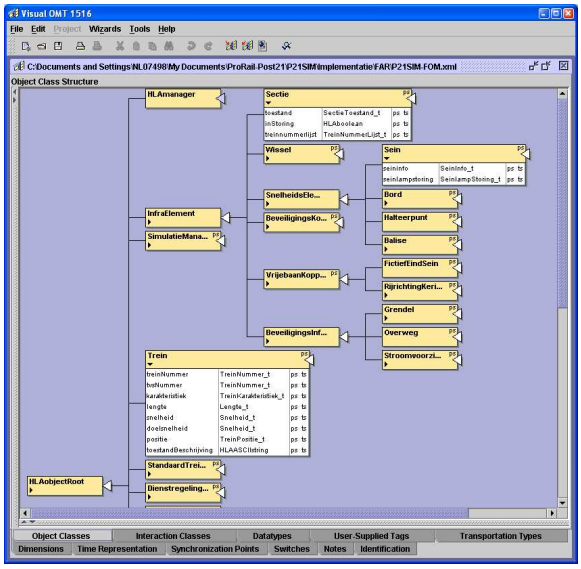

Figure 6: screenshot of FOM-editor

3.2 Federation time management

Because the simulator connects to real world systems, it was decided to use a fixed time step that is synchronized to the wall clock time. Because of the flexibility offered by HLA, it is always possible to change this in a future version and support, for instance, *as fast as possible* simulation.

The Control GUI is a separate federate and responsible for starting the other federates, based on a central configuration file. The Control GUI offers the tester functions to start trains (typically just one or two trains, with the option to run a complete schedule with f.i. twenty trains), introduce system failures and pause, resume and restart the simulation. The latter functions are implemented by using HLA synchronization points.

3.3 Transfer of ownership and DDM

At this moment, the simulator does not make use of features like transfer of ownership and dynamic data management. However, the design for a future version could be modified so that f.i. a signal's *state* is owned by the appropriate simulated safety system rather than the infra simulator. In the current version the safety system sends an interaction to control the state of the signal object.

4 Future

4.1 Visualization

The Control federate provides an event-logging window on the graphical user interface. In this window all significant events that appear during the simulation are logged. The user can define the level of detail to be shown per federate type, by selecting and enabling the available modes: errors, warnings, information, debug details. Another window shows the actual status of the simulation objects, such as trains and point machines.

An additional federate is the visualization module. This provides a dynamic graphical view of the status of the simulation, which is more intuitive to the end user. One or multiple segments of the railway infrastructure are shown, including tracks, signals and safety sections. Sections that are occupied by a train light up yellow. This functionality facilitates the tester to compare the train controller view from the control center monitors with the situation in the simulated environment, at a simple glance.

The visualization has been extended with a command option. By clicking at a particular signal it's state can be changed, and a track can be set to left- or right leading.
This can be used by the tester to interfere in the system and to adjust settings to simulate malfunctioning and disturbances. This also shows the power of HLA, since it makes no difference whether the interaction originates from the simulation controller or from another federate.

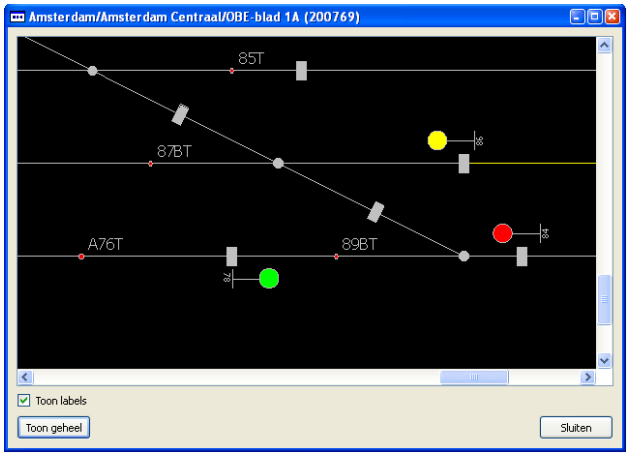

Figure 7: screenshot of visualization federate

4.2 Simulating Control and Management Systems

A subsequent step will be to simulate the control layer and the management layer as well. This will provide the user with a complete train traffic control system "in a

box". In other words: a complete simulation environment that can be run on a single laptop. Such a system can be used for training purposes for train traffic controllers to get familiar with the system or to exercise incident scenarios. It can also be used to study future changes in infrastructure topology, to measure capacity of a station yard or to prepare for major maintenance activities.

4.3 Integration of third-party simulators

The HLA allows us to integrate software modules from another manufacturer into the federation. This way it is possible to reuse proven functionality for a specific safety system simulation in our federation environment.

In order to simulate the behaviour of a particular safety system, a stand-alone simulator had already been developed. As a proof of concept we wanted to integrate this simulator without modifying the existing code. Fortunately, this existing simulator has a TCP/IP command and logging interface. So we developed an HLA adapter to connect and integrate the external software package into the existing federation. Control of the new federate is provided by the general graphical user interface of the simulation system. This whole exercise turned out to work just fine.

5 Conclusion

The HLA is a powerful means to develop a modular and scalable simulator in an efficient way.

The decoupled federates make it easy to construct independent building blocks. This allowed the project to do a number of phased deliveries, each phase providing additional functionality in a new federate.

The decoupling also allowed the engineers to work in parallel, without interference between the multiple building blocks. The Federate Object Model can be considered as the glue.

For a particular test goal, the set-up of the environment is scalable because the type and number of simulated safety systems can be adjusted during the preparation phase.

It can be determined by the user which part of the system layers will be simulated and which part consists of the real control systems. This boundary may be shifted by the user, making the system extremely flexible for testing in different configurations.

6 References

[1] IEEE 1516 Standard for Modeling and Simulation (M&S) High Level Architecture (HLA), ISBN 0-7381-2620-9.

[2] ENTRANCE-EXIT ROUTE INTER-LOCKING CONTROL APPARATUS, US Patent 2567887.

[3] European Rail Traffic Management System, http://www.ertms.com

[4] Vital Processor Interlocking Control System, Product Overview Manual, P2511G, ALSTOM Signaling Inc.

Author Biographies

FRED VAN LIESHOUT is a technical specialist at the Technical Automation (TA) department of Atos Origin Nederland BV. He is responsible for the general architecture of the simulator. Fred has over twenty years of experience in professional software development. Together with Ferdinand Cornelissen, he gives direction to a group of people specializing in simulation, visualization and training systems (SVTS).

FERDINAND CORNELISSEN is a technical specialist at the Technical Automation (TA) department of Atos Origin Nederland BV. Ferdinand is in charge of the design for the rail infrastructure simulation federate. Ferdinand holds an MSc degree in Cognitive Artificial Intelligence.

JAN NEUTEBOOM is project manager at the Technical Automation (TA) department of Atos Origin Nederland BV, an international information technology services company. He is in charge of the project that delivers the simulator described in this paper. He is experienced in managing multi-disciplinary projects and coaching.

BJÖRN MÖLLER is the Vice President and co-founder of Pitch, the leading supplier of tools for HLA 1516 and HLA 1.3. He leads the strategic development of Pitch HLA products. He serves on several HLA standards and working groups and has a wide international contact network in simulation interoperability. He has twenty years of experience in high-tech R&D companies with an international profile in areas such as modelling and simulation, artificial intelligence and Web based collaboration. Björn Möller holds an MSc in Computer Science and Technology after studies at Linköping University, Sweden and Imperial College, London.