Different Forms of Interoperability for Harbor Models

Marco Schumann
Fraunhofer Institute for Factory Operation and Automation
Sandtorstrasse 22
39106 Magdeburg, Germany
+49-391-4090-110, +49-391-4090-158
schuma@iff.fhg.de

Thomas Schulze, Steffen Straßburger
Department of Computer Science
„Otto-von-Guericke“ University, Magdeburg
Universitätsplatz 2
39106 Magdeburg, Germany
+49-391-67-12017
tom@isg.cs.uni-magdeburg.de, strassbu@isg.cs.uni-magdeburg.de

KEYWORDS

HLA, Interoperability, SLX™, Web-based Simulation.

ABSTRACT

This paper discusses the current trends of interoperability and its impact on simulation. The special focus is on the application of interoperability techniques within the area of maritime modeling and simulation. The paper uses a traditional harbor model developed for the Riga Baltic Container terminal to demonstrate the potential of different kinds of interoperability. Technologies which are discussed include different web-based techniques for interoperability and the High Level Architecture for Modeling and Simulation (HLA), as the latest standard in the area of distributed simulation. The simulation system of choice which serves as the demonstration system is SLX, since SLX offers excellent extensibility mechanisms and is best suited for demonstrating different kinds of interoperability.

1 INTRODUCTION

Harbors are traditional objects for simulation studies. The complexity of the harbor world can not be mapped into analytical models. Simulation models are used for solving different problems. (See Bruzzone, Giribone and Revetria 1999 and Merkurjev, Kampermann, Tolujev 99). Permanent changes in the harbor reality lead to new requirements for harbor models. One of the new challenges is the increasing requirement for more interoperability of the harbor models. Theses models have to operate in new environments strongly influenced by new information technologies. Harbor simulation models will overcome their classical stand-alone behavior, they will operate together with other applications.

The paper describes in the following parts different kinds of application-independent interoperability levels. Implementations of the interoperability levels for harbor models are described in section 3 and 4. The section 2 points out the classical basic harbor model.

Interoperability is the ability of a system or a product to work with other systems or products without special effort on the part of the customer. Interoperability becomes a quality of increasing importance for modern information technology products.

We adopt this general definition of interoperability for simulation models. The interoperability of a simulation model is the ability to provide services to, and accept services from other simulation models or simulation model related components with the goal to operate effectively together. Interoperability requires the ability both to

exchange data and to interpret it. This includes effective data sharing and consistent data interpretation. Three different levels of interoperability for simulation models can be distinguished:

Level 0: No interoperability between the simulation models and any other systems exists, i.e., there is no need for data exchange.

Level 1: The simulation model is capable of interoperating with distributed information sources during the initialization phase and during the shutdown phase of a simulation run.. This could also be described as off-line interoperability, i.e., the interoperability does not occur at the runtime of the system. In a simulation environment this could apply to simulation models which are connected with additional components via the WWW. The communication takes place during the startup phase (e.g., by initializing the model with input data) and during the shutdown phase (e.g., by sending the results back to the client).

Level 2: This is the highest level of interoperability. Systems are connected on-line and communicate with each other at runtime. In relation with simulation models one can distinguish between two sub-categories: In the internal case the simulation model can be divided into several simulation components communicating during the simulation run. In the external case simulation model components communicate with other (external) components like visualization and training devices during the simulation run.

2 Traditional Harbor Models

Within the EU-funded INCO-COPERNICUS project DAMAC-HP (1998-2000), a set of generic models for the application in the Riga Baltic Container Terminal (BCT) has been constructed.

These models have been implemented in different languages, e.g., Arena, GPSS/H, and SLX. Initially all of the models were developed as traditional models, i.e., they have a classical monolithic character.

With the deployment of SLX it became possible to investigate the application of advanced technologies for interoperability in the maritime environment.

SLX itself is a discrete event simulation tool for the Windows 95/98/NT/2000 operating systems developed by the Wolverine Software Corporation (Henriksen 1995, 1996, 1997a). SLX is a classical simulation language-oriented stand-alone tool that includes a programming language with a C-like syntax. SLX is the designated successor of the well-known GPSS/H.

Since SLX offers excellent extensibility mechanisms, it was well suited for experiments with different interoperability techniques (see sections 2 and 3).

This section gives a brief overview about the SLX simulation model of the Riga BCT and its development.

Porting GPSS/H Models to SLX

The SLX simulation model discussed here is based on an existing implementation of the model in GPSS/H.

When transferring an existing GPSS/H simulation model to SLX, the user has two general options for doing this:

a) The user can use the SLX-hosted implementation of GPSS, contained in the H5/H6.SLX modules. This implementation was built to support the adoption process of old GPSS users converting to SLX. At the same time it demonstrates one of the main strengths of SLX, its extensibility. Within the SLX language the users can define their own statements and thus built simulation packages onto of the actual SLX language. This has been done with the GPSS language. The SLX-hosted implementation of GPSS supports the GPSS language to a large extend, but not completely. Some re-writing of GPSS models may be necessary to adopt them to make them run under SLX.

b) The other choice is to completely re-develop an existing GPSS/H model with the native SLX constructs and approaches.

While at the first view this seems to be the more time-consuming approach it also has several advantages. In a lot of cases, techniques used for producing good GPSS/H models result in bad SLX models, if models are transliterated from GPSS/H to SLX, or if new SLX models are developed imitating GPSS/H style. Probably the biggest offending technique is using integer indices to access entities. In GPSS/H, this is the only way to access a facility in the single collection of facilities available at run-time.

In SLX, one can have arrays of facilities, facilities as sub-objects, dynamically created facilities, etc. Failure to take advantage of these improved ways of organizing data and objects is resulting in badly readable SLX models.

Implementation of the SLX Model for the Baltic Container Terminal

In the development of an SLX based simulation model of the Riga Baltic Container Terminal (BCT) it also had to be decided which strategy was to be used. A GPSS/H model of the harbor was existing (at least in major parts). In first experiments it was tried to use approach a) described above. Soon the disadvantages of this approach were discovered and the development efforts were redirected to use approach b). One extra-advantage of this approach is that native SLX models can easily fit into the HLA world view of objects and interactions. This tremendously facilitates the interoperability aspects, e.g., if the BCT model should be coupled with other components or programs.

The implementation of the Riga BCT in SLX has been structured in the following way: For all moving entities of the model, active SLX object classes have been defined. SLX distinguishes between active and passive objects. While passive objects merely function as complex data types, i.e., a collection of attributes without own functionality, active objects also define the properties, i.e., the behavior, of a class.

The active objects contained in the model include the following classes:

- cl_Ship models the arriving ships, their unloading and loading processes, and their departure.
- cl_Trailer models the trailers moving through the container yard
- cl_Crane models the different cranes located in the harbor. Different sub-types exist for instance for the quay cranes and the railway cranes.
- cl_Train models arriving and departing trains and their loading and unloading.
- cl_Truck models truck sets which arrive and deliver/receive containers.

The following figures depict the relationships between the most important classes of the model and also shows their attributes and properties.

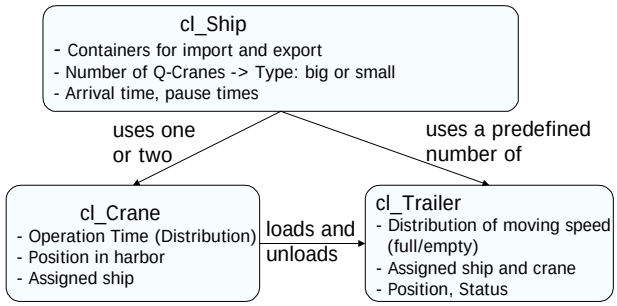

Figure 1a: Relationship of Important Classes of the SLX Model and their attributes

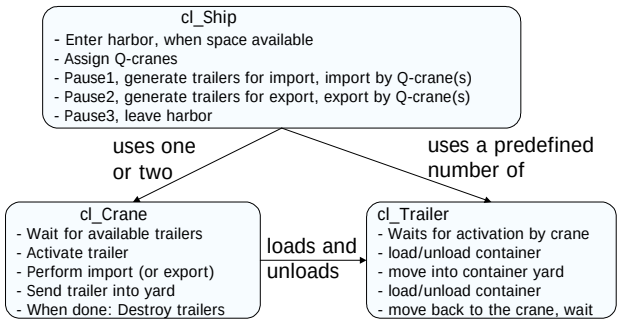

Figure 1b: Relationship of Important Classes of the SLX Model and their behavior

The management of resources in the SLX model is performed using sets. Sets are a common way of modeling the management of resources in the SLX language. In the model, all quay cranes are, for instance, part of the set

"AllCranes" which marks its members as available. When cranes are assigned to a ship they are removed from this set and place into the ship's set "MyCranes". Thus they are marked as currently busy serving a certain ship. A similar strategy is used for trailers. Trailers are initially created for a certain ship, place into the ship's set of available trailers, and removed by a crane once the crane is ready to load/unload the trailer.

Visualization Options for the BCT Model

It was chosen to use Proof Animation for providing an visualization of the SLX model.

Proof Animation is a general-purpose animator designed for use with the widest possible variety of simulation tools. Every Proof animation requires two ASCII input streams, (1) a layout stream, describing static characteristics of an animation, e.g., the background drawing over which objects move, and (2) a trace stream, which is a time-ordered sequence of commands which create, destroy, move, and otherwise change objects displayed on the layout, portraying events in a simulation. Both of these streams are free-format ASCII text, with well documented ("open") architectures which can be generated easily in a variety of ways (Henriksen 1997b, 1998b).

Proof can be used in post-processing mode or directly driven by another program. When Proof is directly coupled to simulation software, input streams are transmitted to Proof one line at a time via subroutine calls. Proof can be directly driven by any program which is capable of constructing C-compatible Dynamic Link Library (DLL) calls; i.e., the directly driven version of Proof is packaged as a Windows DLL.

The simulator SLX is tightly coupled with the DLL version of Proof and could thus be easily used for performing on-line visualization of the BCT model (Henriksen 1998a). Since one of the target scenarios which was to be achieved within our project has SLX run in a distributed environment on a web-server, the traditional post-processed animation option was chosen as the default visualization. Optionally it is possible to switch to an local on-line animation, if SLX and Proof is available locally. This can be done easily with minor modifications of the model, i.e., by exchanging the default include file "proof3.slx" (for post-processed animation) to "p4dll.slx" (for on-line visualization).

The implementation of the visualization of the SLX-BCT model makes extensive use of the path concept of Proof. All important points in the layout, e.g., position of cranes, loading and unloading stations for ships and trains, positions of container yards, etc. are assigned a unique number in the layout. Between each of the points paths following a certain naming convention exist. A path connecting position 17 (the left most quay crane in Figure 2) and a position 3 (one of the container yards) would be named "pa1703". In the opposite direction, a path named "pa0317" exists. All movements of vehicles, e.g., trailers and trucks, takes place on these paths.

The SLX model dynamically constructs the paths which a trailer uses. It analyzes the input files for the distances between certain points and adjusts the speed of trailers accordingly.

Figure 2 : Screenshot obtained from the Proof Animation of the Riga BCT

3 Web-based Harbor Models

A new simulation subarea has been titled web-based simulation by Fishwick and Hill (Fishwick and Hill 1999). This new area combines general web-based technologies with simulation. Page (Page 1998) identifies five

different areas of focus: simulation as hypermedia, simulation research methodology, web-based access to simulation programs, distributed modeling and simulation, simulation of the WWW.

We define web-based simulation as an instance of interoperability level 1. The simulation model does not interact with other components during the simulation run.

Our focus in this section will be on architectures for web-based access to simulation programs. General application independent architectures will be explained in the first part. The second part describes possible applications in the world of harbor models.

3.1 General architecture forms

The client-server structure is the basic architecture of the web. Architectures for web-based simulation are derived from this basic structure. The simulation user is the client which interacts with different forms of servers.

- *Remote execution of existing simulation models (form A)*

The client invokes a web browser, opens a special HTML-document, specifies values for predefined input parameters of the simulation model, submits this document to the server and starts the simulation machine via the common gateway interface (CGI) or similar mechanisms on the server. Output data will be send back from the server to the client after the termination of the simulation. In these cases the servers operate as an application server.

This form offers a small model flexibility for the user. Time consuming simulation runs can be executed on high-performance simulation server. The user is exempted from model maintenance.

- *Local execution of a downloaded simulation models (form B)*

In this case the server operates as an applet server. The complete simulation model is downloaded to the client site and will be executed locally. There are special requirements for the simulation software used in this scenario. The simulation program must be executable in the heterogeneous computer world in the web. Java applet based simulation programs are possible solutions.

- *Execution of a modifiable downloaded simulation model (form C)*

This case can be considered as an extension of both previous case. The server has to operate as a model repository or a file server that offers the source code of the simulation model. The client can modify the source code. The execution of the model can be done in two ways. If the appropriate simulation system for processing the source code of the model is available at the clients site, the execution can be done locally. In the other case an additional (application) server has to be used for model execution. This can be an extension to form A, in which not only the input data, but also the source code of the model itself is sent to the server.

- *Download of input data (form D)*

The client owns the simulation model and the server operates as a data server. The server offers special input data for the simulation run. This can be for example stored data about the state of the real system or data about weather forecasts.

3.2 Application-oriented architectures for harbor models

Based on the general architecture forms described above, we introduce three possible application oriented architectures for harbor models.

- *Simulation–based information system*

The harbor management offers an information system where customers can get answers to questions. Not all queries can be answered on the base of analytical methods. For theses kinds of queries a simulation based information system will be used. Core of such a system is a simulation model of the internal harbor processes. Depending on the forecast horizon the simulation model will be initialized with

the current or an predicted harbor state. The information system offers a special interface for definition of the query. The simulation model is encapsulated, it is hidden from the customer. That means the model can not be changed or modified by customers.

This application oriented architecture is based on the described forms A and B. No differences exist between both forms related to the functionality for a simulation user. He specifies the input data for the simulation run, and after termination of the simulation he will get back the answer. It is not important for the results where the simulation process is located.

Figure 3 shows a possible scenario. A ship which is on the way to the harbor wants to get time-dependent information about its load and unload processes. This query will be created inside a web browser on the ship and submitted via wireless communication to the harbor information system. The information system initializes the simulation model with the concrete query parameters and the needed information of the future harbor state. The results of the simulation run will be prepared. The ship can use the answer data for its own scheduling tasks.

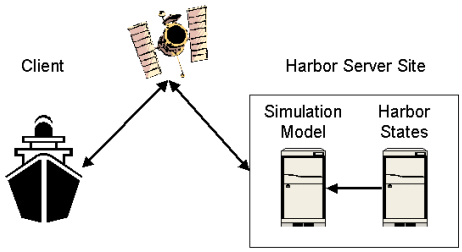

Figure 3: Simulation-based information system

Similar scenarios could be developed for other carriers like trains and trucks.

- *Internal model adaptation*

An increasing number of simulation models will be used in the day-to-day operation of harbor facilities. These models help management to evaluate the system capacity for new orders, for changes in the operator team and for changes in operating conditions. They support the management of harbors for analysis of throughput and detection of bottlenecks. Management can evaluate operating decisions relative to the performance of the system. The needed models have to be adapted to the new conditions. Often the modification of the model can not be done by changes of the input parameter values. It is necessary to alter the model on the source code level.

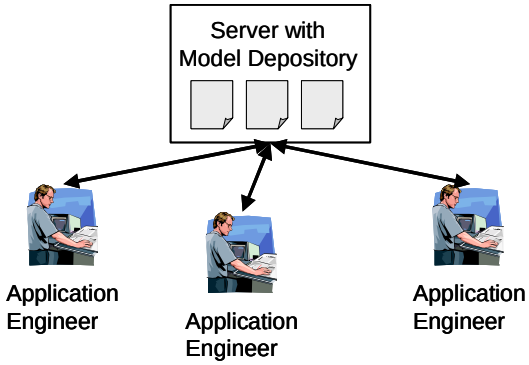

Figure 4: Model depository based model adaptation

If the harbor model is well structured and it consists of exchangeable modules this job can be done by non-simulation experts (See Figure 4). The simulation user chooses different modules from a model depository for building the adapted model. The model depository is located on a server and will be maintained by simulation experts. Different internal users can access the depository via the harbor specific intranet.

- *Preparing external data*

There are situations where the simulation model has to be supplied with information stored on an external server. A typical example in the harbor area is the use of weather and tide forecasts from external servers. The simulation models operate as a client which requires information from a server. In the case of weather forecasts the received information has to be prepared for being used as values for input parameters. With increasing forecasting horizon the predicted values will be more blurred. The use of preprocessors is necessary to transform the fuzzy values into acceptable values for the simulation model.

4 HLA-based Harbor Models

Whereas the approach of web-based simulation is suitable to transfer data during the initialization and shutdown phase of a simulation, it does not provide mechanisms of synchronizing distributed simulation modules at run-time.

Exchanging data at run-time is a quality that characterizes the simulation models belonging into the interoperability level 2. Several standards for distributed simulation, such as DIS or ALSP, have been developed in the past. One of the most recent and sophisticated standards is the High Level Architecture (HLA). This chapter presents ideas how harbor models can benefit from an HLA-based implementation of simulation models.

The creation of HLA-based simulation models can be significantly accelerated by using an HLA-compliant simulator instead of using the DoD provided C++ libraries. Approaches for the integration of existing simulation tools into the HLA are presented in (Straßburger et. al 1998).

As pointed out in the introduction chapter, one can distinguish between internal and external interoperability.

4.1 Internal interoperability

In case the simulation model is divided into several sub-models (federates) that typically run on different machines, we speak of internal interoperability. Splitting a complex model into a set of sub-models provides the flexibility of configuring the simulation for different purposes and levels of detail.

Regarding the application of a harbor simulation, a distinction into the following modules might be useful:

- *Simulation of railroad station*

The railroad and motor vehicle transportation business might be run by separate companies that maintain their own data. For the purpose of simulation, the railroad transportation company might have implemented a simulation model of their business that can be included in a logistical simulation. The advantage is that when the simulation is executed, it can always access the current railroad station data.

A relevant simulation problem could be the question if it is possible for the railroad station to handle a certain amount of cargo. The amount of cargo is communicated through the HLA interface by other federates. In response, the railroad station federate can report its current state to other federates.

- *Simulation of the motor carrier*

The same applies to the simulation of the motor carrier. According to the current order load of the motor carrier, a computer simulation could determine if the carrier is able to process a given amount of cargo.

- *Simulation of fork lifts*

Another federate could simulate the forklifts used to move the containers. This federate could rely on current maintenance data from the service shop in order to determine the number of available fork lifts and their current status.

- *Simulation of cargo ships*

A last federate might be established to simulate incoming and outgoing ships with their characteristic data.

4.2 External interoperability

In addition to several simulation federates, an HLA federation can also include non-simulation federates that do not directly contribute to the simulation. This is called external interoperability. The following section describes how non-simulation components can be utilized to further enhance the functionality of the system.

- *Visualization federate(s)*

Visualization federates can be used to observe an on-line simulation of the harbor model. Using the HLA enables different departments of the harbor administration to have different views on the specific data of their interest.

A typical problem of complex systems of this kind is that dynamic data is only gathered at certain locations in the system. For instance, the position of a specific container is not continuously tracked but more likely only when the container is unloaded from the ship and the next time when it is placed in the container storing area.

A simulation model can be employed to interpolate between those system states and the visualization federated can be used accordingly to enhanced the transparence of the system.

This approach has been applied in a simulation of the streetcar traffic described in (Schulze et. al 1999).

- *Real-time operator training federate(s)*

Equipped with a more sophisticated visualization, such as a 3D Virtual Reality World, the forklift simulation federate can be used to train operators of forklifts using authentic data from the logistical simulation.

- *Decision support federate(s)*

The staff involved in logistical decision-making can also be simulated as an HLA federate. Instead of a programmed algorithm, a human decides manually between given alternatives (e.g. shipping routes), and can evaluate the quality of his/her decision by simulating the different consequences. For this purpose, an abstract 2D visualization from a bird's eye view might be sufficient, but a much higher simulation speed than real-time is desired. This can be accomplished by simply changing the visualization component and not including a real-time synchronization federate.

5 Conclusions and Outlook

The increase of the interoperability of simulation models is one of the trends in the future of modeling technology. The need for this fact is based on the overcome of stand-alone solutions. Simulation models have to be integrated into different forms of distributed information technologies. Web-based simulation and HLA offer possibilities to insert and extend the interoperability of simulation models. First steps have been made in the area of harbor simulation and results are shown in this paper. The success of theses steps circumstantiates the truth of this way.

6 References

Bruzzone, A., P. Giribone and R. Revetria.1999. *Operative Requirements and Advances for the new Generation Simulators in Multimodal Container Terminals*. In Proceedings of the 1999 Winter Simulation Conference, eds. P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, pp. 1243-1252. SCS, Phoenix.

Fishwick, P. A. and D. R.C. Hill. 1999. *Introduction to the Special Issue on Web-Based Simulation.* SIMULATION, volume 73: Number1: July 1999. page 4

Henriksen, J.O. 1995. *An Introduction to SLX*. In Proceedings of the 1995 Winter Simulation Conference, eds. C. Alexopoulos, K. Kang, W.R. Lilegdon, D. Goldsman, pp. 502-507.

Henriksen, J.O. 1996. *An Introduction to SLX*. In Proceedings of the 1996 Winter Simulation Conference, eds. J.M. Charnes, D.M. Morrice, D.T. Brunner, J.J. Swain, pp. 468-475.

Henriksen, J.O. 1997a. *An Introduction to SLX*. In Proceedings of the 1997 Winter Simulation Conference, eds. S. Andradóttir, K.J. Healy, D.H. Withers, B.L. Nelson, pp. 559-566.

Henriksen, J.O. 1997b. *SLX and Proof Animation.* In Deussen, O. and P. Lorenz (Ed.), Proceedings of the Simulation and Animation Conference Magdeburg, March 6.-7., 1997. SCS European Publishing House San Diego / Erlangen / Ghent / Budapest 1997, pp. 287-294.

Henriksen, J.O. 1998a. *Windows-Based Animation with Proof™.* In Proceedings of the 1998 Winter Simulation Conference, eds. Medeiros, D., E. Watson, J. Carson,

and M. Manivannan, pp. 241-247. SCS, Washington.

Henriksen, J.O. 1998b. *General-Purpose Concurrent and Prost-Processed Animation with Proof™*. In Proceedings of the 1999 Winter Simulation Conference, eds. P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, pp. 176-181. SCS, Phoenix.

Page, E. 1998. *The Rise of Web-Based Simulation: Implications for the High Level Architecture*. In Proceedings of the 1998 Winter Simulation Conference, eds. D.J. Medeiros, E.F. Watson, J.S. Carson and M.S. Manivannan, pp.1663-1668. SCS, Washington.

Merkuryev, Y., F. Kapermann, J. Tolujew, G. Merkuryeva, A. Smits and I. Demyanov. 1999. *Simulation of container processing at the Baltic Container Terminal*. In The International Workshop on Harbor, Maritime & Industrial Logistics Modelling and Simulation, eds. A. G. Bruzzone, Y. A. Merkuriev, and R. Mosca. SCS International, pp.9-14

Schulze, T., S. Straßburger, U. Klein. 1999. *Migration of HLA into Civil Domains: Solutions and Prototypes for Transportation Applications*. In: SIMULATION, Vol. 73, No. 5, pp. 296-303, November 1999.

Straßburger, S., T. Schulze, U. Klein, J.O. Henriksen. 1998. *Internet-based Simulation using off-the-shelf Simulation Tools and HLA*. In Proceedings of the 1998 Winter Simulation Conference, eds. Medeiros, D.J. and E. Watson, pp. 1669-1676. SCS, Washington, D.C.

7 Author Biographies

THOMAS SCHULZE is an Associate Professor in the Department for Computer Sciences at the Otto-von-Guericke-University in Magdeburg. His research interests include modeling methodology, public systems modeling, traffic simulation, and distributed simulation with HLA. He is an active member in the ASIM, an organization for computer simulation in Gemany.

MARCO SCHUMANN is an employee at the Fraunhofer Institute in Magdeburg. He holds a Master's degree in Computer Science from the Otto-von-Guericke University in Magdeburg. His experiences in developing simulations and applications for the Internet include a one-year stay at the University of Wisconsin – Stevens Point. His main research interest lies in application of simulation methods in the field of factory planning and optimization.

STEFFEN STRASSBURGER holds a Master's degree in Computer Science from the Otto-von-Guericke University in Magdeburg, Germany. He is currently working towards his PhD at the Institute for Simulation and Graphics at the same university. His experience with inter-networking and simulation includes a one-year stay at the University of Wisconsin, Stevens Point and a stay at the Georgia Institute of Technology, Atlanta. His main research interests lie in distributed simulation and the High Level Architecture.