Proceedings of the 2008 Winter Simulation Conference
S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler eds.

MANAGEMENT OF HLA-BASED DISTRIBUTED LEGACY SLX-MODELS

Michael Raab

Fraunhofer Institute for Factory
Operation and Automation IFF
Sandtorstr. 22
39106 Magdeburg, GERMANY

Thomas Schulze

School of Computer Science
University of Magdeburg
Universitätsplatz 2
39106 Magdeburg, GERMANY

Steffen Strassburger

School of Economic Sciences
Ilmenau University of Technology
Helmholtzplatz 3
98693 Ilmenau, GERMANY

ABSTRACT

This article discusses management tasks that have to be supported for an efficient re-use of simulation models within HLA-based distributed simulation environments. After a review of the controversial discussions on the deployment of distributed simulation in the industrial domain, this article derives the need of support for such basic management tasks. Based on the practical experiences from an industrial project from the manufacturing domain, in which several legacy SLX models had to be integrated, we demonstrate how these management tasks have been supported and which tools were needed for this purpose. We furthermore discuss the results of the distributed simulation and show the necessity and the added benefit provided through its usage in industry.

1 INTRODUCTION

Distributed simulation uses technologies that enable different simulation models running on a computer system containing multiple processors that are interconnected by a communication network (Fujimoto 2000). Technologies originally have been developed by impulses from the military domain. Goals of distributed simulation in the early developing phase were the need for more effective means to train personnel in a distributed virtual environment that simulate combat situations. (Fujimoto 1998; Lendermann 2006). Today distributed simulation is accepted in the military domain.

There were a lot of attempts to apply distributed simulation in industrial domains for example (Boer 2005; Rabe and Jaekel 2004; Strassburger et al. 2003; Gan et al. 2000). Additional work to foster interoperability between different COTS simulation packages is done by the CSPI product development group (PDG) within SISO. Its objective is to develop interoperability reference models (IRMs) and to suggest reference solution for their implementations (SISO 2008; Strassburger et al. 2007).

Although successful in the military domain the ideas of distributed simulation have not reached the state of commercial use until today. (Lendermann 2006; Lendermann et al. 2007).

Nevertheless, a study conducted recently on future trends in distributed simulation and distributed virtual environments (Strassburger et al. 2008) confirms that

- Distributed simulation and distributed virtual environments are attributed a high economic potential
- A broader usage of these technologies needs improvements in the areas of distributed simulation middleware and solutions for semantic interoperability.

Furthermore the study identifies the creation of true plug-and-play-simulation capabilities as an important challenge. Even an agreement on automatic or semi-automatic interoperability mechanisms between (or even within) domains would already constitute an important improvement.

The objectives of the application of distributed simulation in the industrial context can be divided into the categories of **Speedup** (or performance in a more general sense) and **Interoperability**. The category speedup aims at reducing the execution time of simulations through parallelism. A model is typically split up into several logical processes which are distributed to and executed on different processors.

A different approach in this category is the parallel execution of copies of the original model with different experimental parameters.

The objective of the second category (interoperability) is to provide and facilitate interoperability between and reusability of heterogeneous simulation models. This category plays an important role in the context of the digital factory.

Here, the planner is often confronted with the task to combine different simulation models for production sections into a single overall model of the factory.

Although these tasks can be controversially discussed (Lendermann et al. 2007) they are not simple fantasies

978-1-4244-2708-6/08/$25.00 ©2008 IEEE

coming from the ivory tower research. There are real-world necessities behind them. Customers in the industrial domain are willing to sponsor projects in the distributed simulation area. Of course, they are expecting a return on invest, too. The discussions in this article are linked to a real industrial project from the digital factory context.

When implementing such a distributed simulation project one has to consider syntactical and technical aspects, as well as semantic aspects. Furthermore there is a challenge in the efficient execution, control, and management of the distributed simulation and its models.

For supporting the syntactical interoperability standards like the HLA can be used. For supporting the semantic interoperability there are no ready-to-use industrial standards available yet. The objective of this paper is to address these questions in more detail based on experiences from the industrial project already mentioned.

The remainder of this paper is structured as follows. Section 2 explains basic management tasks that have to be supported for an efficient re-use of simulation models within HLA-based distributed simulation environments

Section 3 introduces the industrial project which has been successfully completed. Section 4 reports on the implementation approach for customizing legacy SLX-models that have been used in the industrial project. Section 5 discusses some aspects of user support for preparing and executing distributed simulation in industrial environment. Section 6 discuss the influence of lookahead on the modeling features and simulation run time. Section 7 presents selected outcomes from the federation to show responses that can be delivered only from complex distributed models. Section 8 introduces future work.

2 BASIC MANAGEMENT TASKS

Basic management tasks relate to the tasks of creating and executing distributed simulation models. We especially focus on the tasks that constitute additional tasks compared to the creation and execution of monolithic simulation models.

The first subsection discusses the tasks which must be performed before the actual simulation execution. The second subsection focuses on user support during the execution of the distributed simulation.

2.1 Maintaining consistency of the monolithic legacy model and the federated model versions

A distributed simulation model interacts with other models at defined interaction points.

The CSPI PDG defines Interoperability Reference Models (IRMs) which help in defining the interoperability requirements within a certain distributed simulation (SISO 2008). These IRMs help the modeller to identify which interoperability problem he must solve within a given simulation problem. He can then identify the required interaction points or elements within the legacy simulation models.

At these interaction points, certain modifications have to be made to the models. These modifications are not required in the monolithic models and shall therefore only be effective in the federated model versions. A single code base for both model versions is a typical requirement towards the model implementation. The user shall be able to switch between using the model in standalone mode or in connected mode.

Let us consider the following example: A monolithic model MM describes the production of three part types A, B und C. The parts are each created in their own source and destroyed at the end of the manufacturing chain. Let us now consider a federation consisting of two distributed models DM1 and DM2 (see Figure 1). Distributed Model DM2 has been derived from the monolithic model MM and represents the same production processes. The only difference is that the parts of type B are now generated within the distributed model DM1 and passed to DM2. So DM2 does not create the parts of type B itself. This represents a typical connection between two production sections.

The interoperability problem stated here can be classified as a IRM Type A.1 problem. The interoperability requirement is the passing of entities from model DM1 to DM2

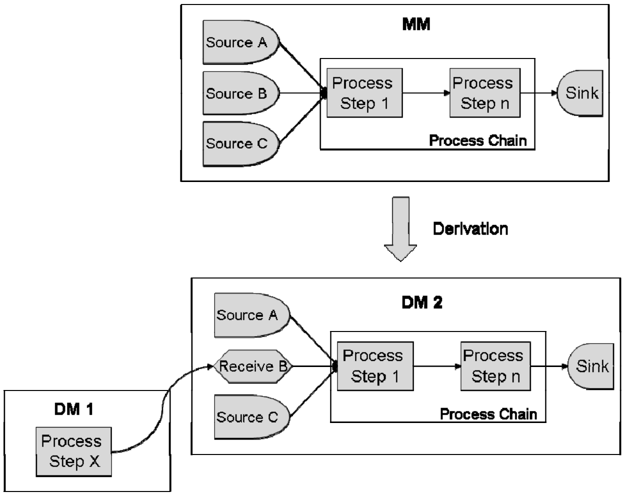


Figure 1: Federation F1 with Two Federates

Let us now consider a federation F2 which consists of the distributed models DM1, DM2* and DM3. The distributed model DM2* was again derived from the monolithic model MM. The only difference is that now parts A and C are generated from the other models. Thus the models DM2 and DM2* differ regarding their acceptance of external parts and the creation of own parts. All other parts of the models are identical (See Figure 2).

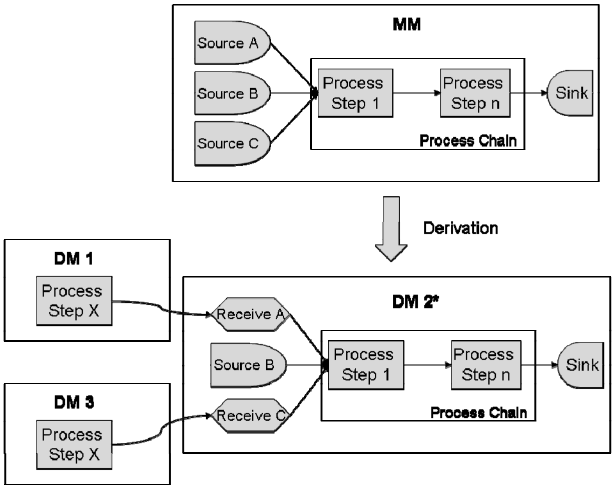


Figure 2: Federation F2 with Three Federates

These examples are meant to demonstrate that re-using of monolithic simulation models not only involves changes at the syntactical level, but also requires interoperability at the semantic level. A typical example for interoperability on semantic level is the use of different scales for simulation time units. A simulation time unit in one model may represent seconds of real time while in the other model it may represent minutes. An additional typical interoperability problem is the use of days. One model may simulate working days only while the other model includes all days of the week, i.e. also non-working days.

When creating a common semantics for the models it has to be verified that semantic changes of the legacy models can be performed at all and at the same time that consistency between monolithic and distributed model versions is maintained.

For a typical simulation project it is necessary to simulate different scenarios with different models and compare their results. A manual creation of the individual model versions is often not practicable. This leads us to the following questions:

1. How can the consistency between the individual model versions including the monolithic and the federated ones be maintained?
2. Which mechanisms can ensure a consistent maintainability of the model parts which are not affected be the modifications for the distributed simulation?

A solution practiced within our project is the usage of a generic base version of the model from which a distributed model can be generated. The generation process involves the usage of additional configuration data which specifies the connections and interoperability requirements of the federated model versions.

2.2 User Support during the simulation experiments

The execution of experiments with distributed simulations in industry will not only be conducted by simulation experts. More and more also non-simulation experts are required to use simulation and consequently also distributed simulation in the future.

The user therefore has to be supported when dealing with the execution of distributed simulations. This includes the activities described below. For these activities it is advantageous to provide a tool to the user that automates all otherwise manual activities dealing with low-level HLA services and that supports the experiment configuration process.

One of the required **activities** is the **selection of the simulation models and their composition to a distributed simulation (federation)**.

Assuming the suggested usage of configuration data for generating the distributed models the generic base models have to be selected. In the following, the distributed models will be generated using the configuration information specified by the user.

For the generation on the models both the syntactical and the semantic interoperability have to be ensured. Depending on the context, the latter may even be the more challenging task.

Decisions have to be made regarding the flow of parts through the distributed models, esp. regarding the question which parts have to be passed to the following model.

Furthermore the FED-File (i.e. the HLA object model version needed for execution of the distributed simulation) has to be generated.

Another **activity** which typically needs to be supported is the **process of monitoring the execution** of the distributed simulation. The **generation of an observer federate** and its integration into the federation is a beneficial way to facilitate this process. The necessity of such an observer federate is explained in the following.

The objective of a simulation with distributed models is the derivation of results from a complex distributed model, as only this complex model allows an investigation of the causal relationships between the individual models.

The generated distributed models produce their predefined statistics as originally intended by their modellers for the monolithic model version. In the context of the distributed simulation, they also collect special statistics about the behaviour of each individual model in its interplay with the other models of the distributed simulation.

The latter includes, for instance, delay times when one model has to wait for parts from another model. Furthermore information is collected about states and conditions of individual model components such as buffers.

The observer federate is a suggested way of collecting this data at a central point. The observer federate has the additional advantage that it can extract certain types of information directly from the communication between the individual models (e.g., numbers of entities passed) without having to modify the models themselves for this purpose.

An additional **activity** to be supported is the **execution and control of the distributed simulation.** This requires the actual start-up of the individual models and the simulators executing them and joining them into the federation.

The location and the required start-up procedures of the individual models are known as a result of the selection and combination process of the models. This can be on a local or remote computer.

The federation control has to ensure that all models delay their time advancement until all selected models have joined the federation. Finally, a controlled shut-down of all federates must be supported.

3 EXAMPLE DESCRIPTION

Within design and construction of a vehicle production factory, independent simulation models were developed to simulate the behaviour of seven interrelated assembly and paint sections. These models are used after the design phase to support ongoing factory operations. The material flow inside the factory is presented in figure 3. The aim of our industrial project was to connect the pre-existing sub section models of a factory and to integrate them into a complex distributed simulation model. This complex model should be used to investigate the interrelationships between the different sub sections. Therefore the dimensioning of buffers between sections and the coordination of production schedules were investigated in order to assess bottlenecks in the complex system. In our experience this scenario of having to couple different independently developed simulation models represents a typical use case in industrial applications.

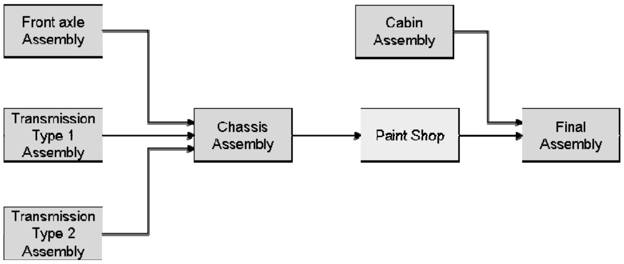


Figure 3: Schematic View on the Production System

Differences which typically need to be solved concern the implementation of the simulation models in varying simulation software, levels of detail in the models, time advance mechanisms and time dependencies. Our models can be divided into two application categories: assembly and paint shop models. Each category uses its own generic SLX simulation model, which is used in combination with an appropriate configuration file to generate an executable SLX simulation model suitable for one production area.

Due to existence of a generic HLA interface for the SLX simulator, HLA with RTI 1.3NG was selected to perform as middleware for distributed simulation (Strassburger 2001).

4 CUSTOMIZATION OF LEGACY SLX MODELS

Section 2 has outlined basic support services for the model management and execution within distributed simulations. A key service in this context was the preparation of existing legacy simulation models for their reuse in a complex distributed model. A potential solution which was outlined in that section was the generation of a distributed model from a generic base model. This approach has the advantage that it preserves the consistency of the models (distributed vs. monolithic).

In our specific project, there were two different types of generic monolithic legacy simulation models. One was for simulation of assembly processes and the other for simulation of painting processes. Both monolithic model types differ in modelling approaches; they are specialized for their application area. Still, they both use the implementation language SLX and generic modelling concepts which allow parameterized model generations.

For the distributed simulation, both types of generic models had to be extended to become distributed simulation "aware". In specific, both generic simulation models had to be extended by:

- HLA-services for joining to and resigning from a federation
- HLA-services for sending and receiving of object attributes and interactions
- HLA-services for time management
- Integration of external events into their local event list and responding on this external events and
- Modelling of part movement between coupled models.

A major challenge was the adoption of the pre-existing generic monolithic simulation models to be able to participate in the distributed simulation in a generic way. A mandatory requirement was the avoidance of different model versions for the monolithic and distributed applications. The generic model had to be able to generate both a monolithic and a distributed version of the simulation model to ensure the consistency between both models.

A solution to this problem was a feature of the simulation language SLX that allows the extension of model source code during compilation. By using this SLX feature different model versions can be generated smartly. Figure 4 shows the approach to customize the pre-existing generic models.

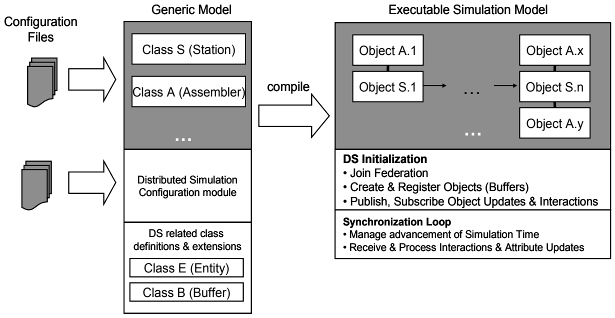

Configuration Files
Generic Model
Class S (Station)
Class A (Assembler)
...
Distributed Simulation Configuration module
DS related class definitions & extensions
Class E (Entity)
Class B (Buffer)
compile
Executable Simulation Model
Object A.1
Object A.x
Object S.1
...
Object S.n
Object A.y
...
DS Initialization
• Join Federation
• Create & Register Objects (Buffers)
• Publish, Subscribe Object Updates & Interactions
Synchronization Loop
• Manage advancement of Simulation Time
• Receive & Process Interactions & Attribute Updates

Figure 4: Customizing Pre-existing Generic Simulation Models for Use in Distributed Simulations

Monolithic simulation models usually destroy their flowing material parts when they reach the end of the modeled process chain. In a distributed simulation it becomes necessary to hand over subsets of these parts to successive models. The transfer conditions of our industrial pilot project include the transfer of material to a bounded buffer of a successor model. Enhancements to the simulation code had to be made to allow this behaviour.

Regarding their interoperability requirements our distributed simulation models are compliant with a Type A.2 Interoperability Reference Model. The classification used here is derived from the Draft Standard for Commercial-Off-The-Shelf Simulation Package Interoperability Reference Models (SISO 2008). The A.2 reference model type describes the synchronized transfer of entities into a buffer with limited capacity. In our solution implementing this transfer and its conditions, we use an HLA interaction for the entity transfer and HLA attribute updates to communicate the current buffer content of the respective input buffers. Figure 5 shows a principle view and some important data structures.

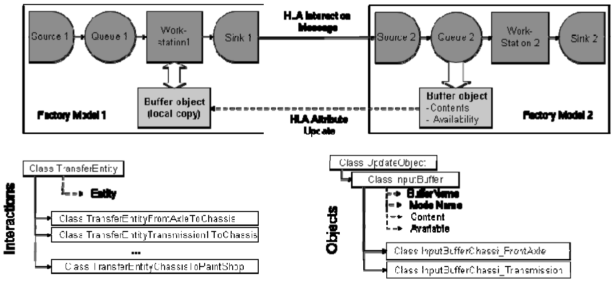


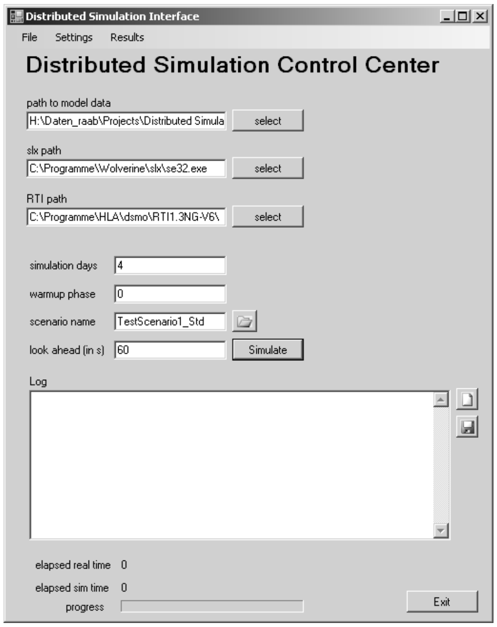
Figure 5: Modelling the Material and Information Flow beyond the Borders

For allowing the operation in a distributed simulation certain data about local objects and its attributes (which are of interest to the other models) has to be communicated regularly to other simulation models.

The Factory Model 1 in Figure 3 has to know the availability of the input buffer (Queue 2) in Factory Model2. Only if this buffer is available Factory Model 1 can transfer a part to Factory Model 2. The buffer object is published to all other models with the attributes Contents and Availability. Factory Model 2 publishes all changes of the attributes to other simulation models. A local copy of the buffer from Factory Model 2 is used inside Factory Model 1. If an attribute update from the original buffer is sent then the local copy will be updated automatically. So consistency between original and copy is ensured and Factory Model 1 decides about part transfer depending on the local copy of the buffer in Factory Model 2. For the publication and subscription as well as the sending and receiving of such data certain HLA functionalities had to be used. This behaviour was implemented in a manner transparent to the end user.

5 USER SUPPORT

For real industrial usage of distributed simulation it is necessary to support the end user in his operational work with the distributed simulation. The existing features and user interfaces of commercial simulators do not support the work with models in distributed environments. During the implementation of the industrial pilot project (which included up to seven individual models) it became very soon clear that the existing manual control features for starting, closing and controlling the distributed simulation were insufficient. For this reason the development of a user-interface (called commander) was initiated.



Distributed Simulation Control Center

path to model data: H:\Daten_raab\Projects\Distributed Simula

slx path: C:\Programme\Wolverine\slx\se32.exe

RTI path: C:\Programme\HLA\dsmo\RTI1.3NG-V6\

simulation days: 4

warmup phase: 0

scenario name: TestScenario1_Std

look ahead (in s): 60

elapsed real time 0

elapsed sim time 0

progress

Figure 6: Form for the User-Interface

The version of the user-interface currently developed provides the following services:

- Encapsulation of RTI-Services against the user
- Supporting the user in creating the composition of the complex model
- Creation of model visualization and calculation of simulation statistic
- Starting, controlling and closing down entire executions of the distributed simulation

The commander provides the opportunity to compose and manage distributed simulation models on an abstract layer without having the need to access the HLA API. Such a layer is mandatory to achieve high acceptance of distributed simulation in operational environments. The commander was implemented in C++ to be able to access the RTI-services provided by the HLA API. To store the commander configuration, an XML-file is used. Figure 6 shows an example of the Commander user interface.

6 LOOKAHEAD AND RUNTIME

In distributed simulation the lookahead describes a period of time in which a sub model will not create external events for other sub models. Using a lookahead greater zero conservative synchronisation methods can calculate a time limit for other sub models. In general an increasing lookahead enables a greater parallelism of the sub model execution and leads to a decrease of execution time of distributed simulation models. Results of experiments with different sizes of lookahead are shown in Figure 7.

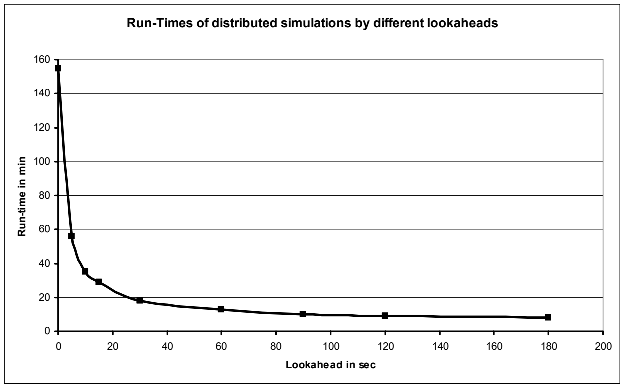


Figure 7: Run-Times of Distributed Simulations by Different Lookaheads

To use a large lookahead is useful from the point of performance. But forcefully using a lookahaed greater than 0 (i.e. a lookahead which is not derived from the model logic) is influencing the logic inside execution of events. This is shown using one example (Compare Figure 5). Factory Model 1 can not deliver parts if the buffer of Factory Model 2 is not available. Assume that the buffer attribute Availability has been updated inside Factory Model 2 at time t1. The other models will be notified about the new value will at a timestamp t2 = t1 + lookahead. The value of the local copy of the buffer object inside Factory Model 1 will therefore not be updated until time t2. As a result the logic inside Factory Model 2 uses an incorrect value about availability until time t2. This can lead to undesired reactions inside Factory Model 2. For example, if the buffer is not available and the Factory Model 2 doesn't know this fact and then factory Model 2 could send a part into the full (not available) input buffer.

No general solutions for this problem have been found inside this project. Special application-depending approaches were used to avoid such problems.

7 OUTCOME OF DISTRIBUTED MODELS

Besides the standard numerical result preparation within the pre-existing monolithic models, an "Observer-Federate" was used to collect data regarding the distributed simulation including the flow of material between the connected models and the history of buffer content changes. These data can be collected from the regular communication traffic between the models. For further characterization of dependencies between the different factory sections, each model collects additional statistics on waiting times due to full buffers and waiting times due to missing materials. This information relieves the search for bottlenecks in the complex system.

Table 1 shows results from model M4 for waiting times. Models M1, M2 and M3 an deliver parts to model M4. The production in model M4 has to stop, if needed parts from provider models are not available. Statistic for the duration of this models state is gathered.

Table 1: Delay Times for Model M4 Due Missing Incoming Parts from Provider Models

Delays due missing incoming sub-models			
Model	Provider Models		
M4	M1	M2	M3
# of delays	0	10	4
Minimum delay	0s	30m	4m
Average delay	0s	1h 45m	1h9m
Maximum delay	0s	7h 34m	2h23m
Sum delay	0s	17h 30m	4h34m

Figure 8 shows an example about demand and simulated production output of a sub model. The sub model depends on other sub models which deliver parts. The production output, depending on the behaviour of provider models can be calculated only in a complex model.

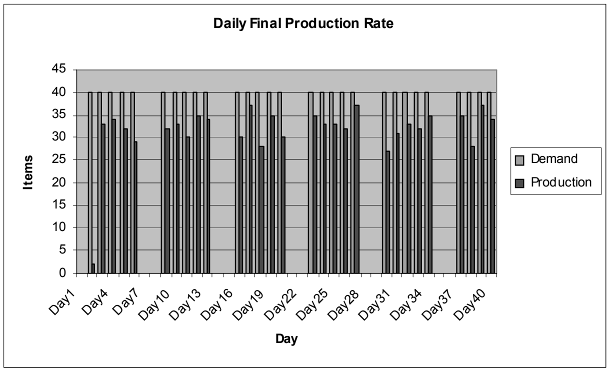


Figure 8: Daily Demand and Simulated Production Output for a sub model

For each of the monolithic simulation models in our industrial project an individual visualization model had been created. These models were inappropriate for the visualization of the distributed simulation, as for this use case only the material flow and the buffer fill levels *between* the individual models were of interest (i.e., the internal actions *within* a model were not to be visualized). Therefore a new visualization including statistic features for the interesting outputs had to be created (compare Figure 9).

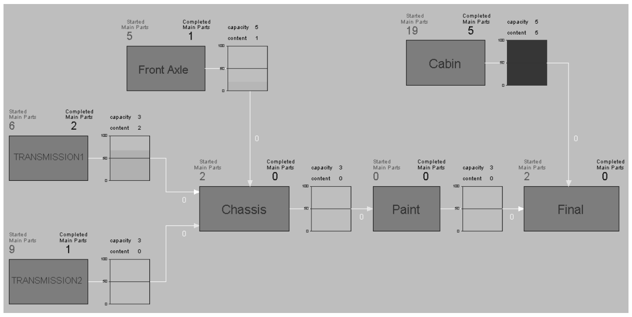


Figure 9: Visualisation of Material Flow and Buffer Contents

8 OULOOK

The goal of this industrial project to use legacy monolithic simulation models in a distributed simulation models has been achieved. Additional work will be done in two directions: The development of more general solutions for the look ahead problem and the improvement of management services for more simply use an application of distributed simulation models.

REFERENCE

Boer, C. A. 2005. Distributed Simulation in Industry. PhD Thesis Erasmus University Rotterdam.

Fujimoto, R. M. 1998. Parallel and Distributed Simulation. In *Handbook of Simulation: Principles, Methodology,Advance, and Practice*. Ed. J. Banks. John Wiley &Sons Inc. New York NY, USA. pp. 429-464

Fujimoto, R. M. 2000. *Parallel and Distributed Simulation Systems*, Wiley Interscience, 2000.

Gan, B. P., L. Liu, S. Jain, S. J. Turner, W. T. Cai, and W. J. Hsu. 2000. Distributed Supply Chain Simulation Across the Enterprise Boundaries. *Proceedings of the 2000 Winter Simulation Conference*. J.A. Joines, R. R. Barton, K. Kang and P. A. Fishwick, eds., 1245-1251. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Lendermann, P. 2006. About the Need for Distributed Simulation Technology for the Resolution of Real-World Manufacturing and Logistics Problems. In *Proceedings of the 2006 Winter Simulation Conference*, L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds., 1119-1128. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Lendermann, P, L. F. McGinnis, C. R. McLean, S. Taylor, M. Heinicke, and S. Strassburger. 2007. Distributed Simulation in Industry - a Real-world Necessity or Ivory Tower Fancy? In *Proceedings of the 2007 Winter Simulation Conference,* S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, eds., 1053-1062. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Rabe, M. and F.-W. Jaekel. 2004. The MISSION project – demonstration of distributed supply chain simulation. In *Enterprise inter- and intra-organizational integration*, 235-242. Boston, Dordrecht, London: Kluwer.

SISO. 2008. Standard for Commercial-off-the-shelf Simulation Package Interoperability Reference Models. SISO-STD-006-2007-DRAFT-1.0. Available via <http://www.sisostds.org> [accessed June 2008]

Strassburger, S. (2001). Distributed Simulation based on the High Level Architecture in Civilian Application Domains. Doctoral Thesis, University of Magdeburg, Germany

Strassburger, S., G. Schmidgall, and S. Haasis. 2003. Distributed Manufacturing Simulation as an Enabling Technology for the Digital Factory. *Journal of Advanced Manufacturing Systems (JAMS)*. 2(1), pp.111-126.

Strassburger, S., Schulze, T., Lemessi, M. 2007. Applying CPSI Reference Models for Factory Planing. In *Proceedings of the 2007 Winter Simulation Conference,* S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, eds. 603-609. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Strassburger, S., Schulze, T., Fujimoto, R. 2008. *Future Trends in Distributed Simulation and Distributed Virtual Environments*. Peer Study Final Report. Version 1.0. Ilmenau, Magdeburg, Atlanta – January 17, 2008.

AUTHOR BIOGRAPHIES

MICHAEL RAAB is an employee at the Fraunhofer Institute in Magdeburg, Germany. He holds a Diploma degree in Computer Science from the Otto-von-Guericke-University Magdeburg. His research interests include manufacturing simulation, distributed simulation and virtual reality applications.

THOMAS SCHULZE is a professor in the School of Computer Science at the Otto-von-Guericke-University, Magdeburg, Germany. He received the Ph.D. degree in civil engineering in 1979 and his habil. Degree for computer science in 1991 from the University of Magdeburg. His research interests include modeling methodology, public systems modeling, manufacturing simulation, distributed simulation with HLA and online simulation. He is an active member in the ASIM, the German organization of simulation. His web page can be found via <www-wi.cs.uni-magdeburg.de>.

STEFFEN STRASSBURGER is a professor at the Ilmenau University of Technology in the School of Economic Sciences. In previous positions he was working as head of the “Virtual Development” department at the Fraunhofer Institute in Magdeburg, Germany and as researcher at the DaimlerChrysler Research Center in Ulm, Germany. He holds a Ph.D. and a Diploma degree in Computer Science from the University of Magdeburg, Germany. His research interests include the topics simulation and distributed simulation as well as general interoperability topics within the digital factory context. Mr. Strassburger is also the Vice Chair of SISO’s COTS Simulation Package Interoperability Product Development Group. His web page can be found via <www.tu-ilmenau.de/fakww>