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.

APPLYING CSPI REFERENCE MODELS FOR FACTORY PLANNING

Steffen Strassburger
School of Economic Sciences
Technical University of Ilmenau
Helmholtzplatz 3
98693 Ilmenau, GERMANY

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

Marco Lemessi
Industrial Engineering, Simulation
Group, Deere & Co.
One John Deere Place
Moline, IL 61265, U.S.A.

ABSTRACT

This paper investigates the applicability of the CSPI reference models in different factory planning scenarios. These scenarios are taken from real industrial use cases. The CSPI reference models are put forward by the CSPI Product Development Group within the Simulation Interoperability Standards Organization (SISO). The objective of this group is to facilitate commercial-off-the-shelf (COTS) simulation package interoperability (CSPI). The approach to do this is to define and standardize use patterns of the High Level Architecture (HLA) which is the state-of-the-art standard for distributed simulation. An intermediate step towards this goal is the definition of the interoperability reference models discussed here. They describe typical interoperability problems encountered when connecting different COTS simulation packages. This paper focuses on the first two of these reference models and reports on experiences drawn for their implementation.

1 INTRODUCTION

This paper is based on work conducted within the commercial-off-the-shelf simulation package interoperability (CSPI) product development group (PDG) within SISO. Taylor et al. (2006) introduce the work of this group in detail.

The motivation for the work of this PDG is given by the existence of a good standard for simulation interoperability, namely the HLA (Strassburger 2006b), and the lack of standardized ways to use it for enabling interoperability between different COTS simulation packages from different vendors. The results of this conflict are often incompatible solutions in using the HLA in different packages. Strassburger (2006a) reports in detail on these problems. In essence, as there are multiple ways of using HLA to achieve interoperability, only tools which provide flexible HLA use patterns can achieve interoperability in all cases. Simulation tools which provide only specific HLA use patterns to the package user may have problems interoperating with tools which can not support the same specific HLA use pattern.

To come close to a plug-and-play interoperability and to provide package vendors with guidance on how they should include HLA in their packages specific HLA use patters must therefore be standardized. This is the objective of the PDG. This objective is addressed by developing

- interoperability reference models (IRM) – they provide a scheme for classification of differing interoperability needs,
- HLA use patterns – they provide guidance on how to use HLA to implement a solution for an interoperability reference model.

The current activities of the PDG are still focused on releasing the standard for the IRMs. However, as work in the group progresses, also the first solutions for formally defining use patterns for implementing the IRMs have been published and are intensively discussed. This paper contributes to this discussion by introducing a solution for implementing the Type A.2 IRM which deals with a bounded receiving buffer. A real industrial test application serves as background for this discussion.

The remainder of the paper is structured as follows. Section 2 explains motivations why the application of distributed simulation is useful in the context of factory planning and which added benefits can be gained from it. Section 3 briefly introduces the interoperability reference models relevant for this paper. Section 4 introduces the real industrial use case which has served as the test application for the solutions discussed in this paper. Section 5 reports on the implementation of this application in terms of how the solution for the IRM is implemented. Section 6 provides recommendations and reports on best practices. Section 7 summarizes and introduces future work.

2 BENEFITS OF DISTRIBUTED SIMULATION

The traditional objective of parallel and distributed simulation has almost always been to gain speedup in executing a simulation. The general idea was to split-up a complex simulation problem into different logical processes in order to run them on multiple processors.

The motivation of the work presented here is quite different: Typically, individual existing simulation models (often implemented in different simulation packages) need to be brought together in order to form a larger simulation model.

The motivations for doing this can be quite different. On the technical side, there is currently no other viable way of re-using and combining models developed in the different discrete-event COTS simulation packages discussed here than coupling them at runtime.

This is simply because no general standard interchange format exists or is supported by any of these tools which would be capable of really translating complete models between the packages. Only partial solutions like the Simulation Data Exchange (SDX) format exists (Sly and Moorthy 2001).

The SDX format is a big step forward for bridging the gap between factory CAD systems and simulation tools, but it is still far from being a real simulation model interchange format bridging the gap between different simulation packages. It is questionable if such a complete format can ever be created as discrete event simulation packages differ quite significantly in their modeling constructs and capabilities.

For these reasons, only the coupling of CSPs at runtime is capable of integrating heterogeneous discrete event simulation packages.

On the application side, the coupling of simulation models is required and useful for a wide variety of purposes. Speaking for the area of factory planning today's typical application of simulation is limited to certain sections of the factory. Typical examples can be assembly lines, paint systems, body-in-white, etc.

For each of these factory sections, modeling and simulation is applied individually. This is good and constitutes a big advantage over non-simulation based design decisions. However, in order to investigate the interdependencies between the different sections of the factory, it is not sufficient to look at the individual sections separately.

In order to harmonize the production across all sections and to investigate existing dependencies (e.g. in the treatment of rework, buffer sizes, transport strategies, scheduling strategies, shift regimes etc.) distributed simulation is the suggested tool of choice and the perfect setting to demonstrate the advantages of a standard like HLA.

The only alternative to distributed simulation for the stated objectives would be to create a new monolithic simulation model from scratch that includes all relevant production sections. If this new simulation model is supposed to have the same level of detail the development of such a model is time consuming and costly. If a lower resolution for the overall model is chosen, it is questionable if it can serve its purpose of investigating the interferences between production sections.

Generalizing this setting the general prerequisites for a successful and economically reasonable application of distributed simulation are:

- the existence of complex simulation models,
- the existence of a problem or question which can only be solved or answered if the models are combined,
- time and money constraints which prevent the redevelopment of a monolithic model with the required level of detail.

In these cases it is very likely that an economic benefit and a real business case for distributed simulation exists.

The questions to be answered in our industrial pilot application (compare section 4) related to the best sizing of buffers between production sections, the implications of different shift models used throughout the factory and questions on where to introduce the line-up desired as an output of the final assembly line.

3 CSPI REFERENCE MODELS

Detailed and complete introductions to the interoperability reference models (IRM) developed within the CSPI PDG can be found in Taylor et al. (2006) as well as on the groups web page at <www.sisostds.org>. The latest version of the definitions referred to at the time of writing can be found in these proceedings (Taylor et al. 2007).

For the discussions in this paper, only IRMs Type A.1 and A.2 are relevant. They are both focused on entity passing between different models. While the Type A.1 Interoperability Reference Model deals with general entity transfer, Type A.2 covers the synchronous case where a bounded buffer occurs.

Please note that the differentiation between approaches with unbounded and bounded buffers is of significant importance, because it has major implications on its potential implementations.

Figure 1 outlines the basic idea behind both reference models. Both represent models that interact on the basis of entities, i.e. models are linked together so that one model may pass an entity to another. The term "Entity" here refers to the dynamically created elements that move

through a simulation. They may be called differently in different simulation systems.

In the Type A.1 IRM there is no necessity for direct feedback when an entity is passed between models. It is assumed that the entity can be accepted by the receiving model in any case.

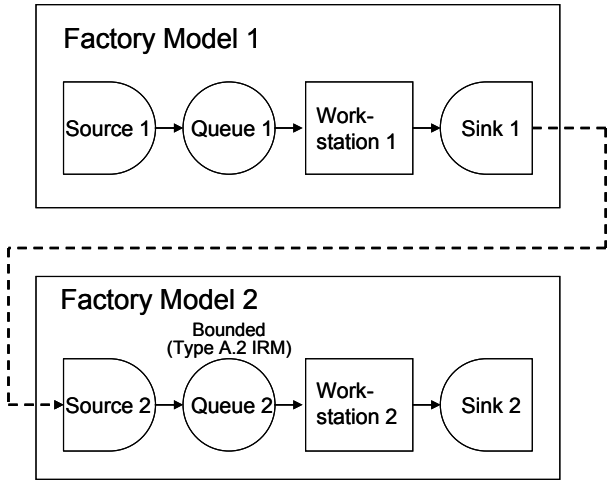

Figure 1: Type A.1 and A.2 IRMs.

The major difference in the Type A.2 IRM is that the receiving simulation may be in a state unable to accept the entity being passed, e.g., because of a bounded queue (depicted in Figure 1). This constitutes a very realistic condition, esp. in the area of factory simulation. If buffer space between production sections fills up, the sending section of the production may become blocked. This is a major issue which typically can not be investigated if production sections are simulated individually.

The application of Type A.1 IRMs with the simulation package SLX has been discussed in Strassburger (2006a). In essence, this paper has reported that it was possible to encapsulate an implementation of this IRM into two SLX statements and one additional method which need to be inserted into the SLX code.

The Entity Transfer Specification (ETS) 1.1.1 (Taylor et al. 2003), a draft standard of the CSPI PDG for defining a use pattern for Type A.1 IRMs had been applied in this scenario. Some modifications had to be made to this protocol relating to the hierarchy of transfer interactions and the definition of the sender-receiver relationship.

The same specification with the suggested modifications builds the base for the implementation of the solution for the Type A.2 IRM reported on in section 5. More technical detail will be given in that section.

4 INDUSTRIAL TEST APPLICATION

A planned tractor factory in South America was chosen as a real industrial application to test the synchronous entity passing approach mentioned above.

With a target production of 40 tractors per day, Monday through Friday, the focus system consists of six components, as shown in Figure 2: two pre-paint asynchronous assembly lines for two different chassis models, two post-paint asynchronous assembly lines for cabs and tractors, a wet-on-wet paint system, and a fabrication area for welding and machining of cab components (fenders, frames, etc.). Each system component consists of multiple manned work stations.

The two pre-paint assembly lines and the fabrication area feed the paint system, and the latter feeds the two post-paint assembly lines. Assembly lines and paint system are operated two shifts per day, while the fabrication area runs three shifts per day. In such a scenario it is critical to properly size the pre-paint chassis and component buffers, and the post-paint buffer for final products (see Figure 2).

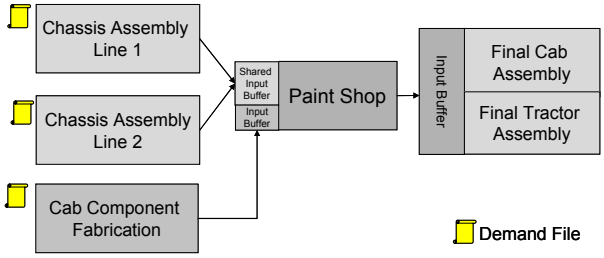

Figure 2: Scenario of the industrial test application.

Two simulation frameworks (one for assembly and one for paint shop simulation) have been used in this industrial test application. Both frameworks are based on the simulation system SLX. The frameworks consist of complex code structures inside SLX and user interfaces based on Excel or Access, respectively.

Users of the packages do not typically modify the SLX code themselves. Models are generated based on the user input in the Excel or Access interfaces. This is done dynamically at simulation startup based on the precursor module of SLX.

The objective for the distributed simulation was to integrate all distributed simulation capabilities into the code base of both frameworks in a manner transparent to the user, i.e., no manual code interactions should be required if operating in distributed or standalone mode. The implementation of the solution is based on the SLX-HLA-Interface (Strassburger et al. 1998) and uses conservative synchronization.

Conservative synchronization is typically chosen when connecting different CSPs as none of them supports any sophisticated state-saving and rollback capabilities.

Details about the implementation of the industrial test application are discussed in the following section.

5 IMPLEMENTATION

As the sizing of input buffers and the potential effects of blocking production sections due to full buffers were subject of the investigations, the interoperability requirements to connect the models fall into the category of a Type A.2 IRM.

The implementation had to enable a synchronous entity transfer between the production sections. The selected solution is based on the Entity Transfer Specification (ETS) 1.1.1 mentioned in section 3 but incorporates the same modifications required for the Type A.1 IRM introduced in Strassburger (2006a).

5.1 Usage of the Entity Transfer Specification

The Entity Transfer Specification (ETS) standardizes a certain usage of HLA interactions for transferring entities between two models. The sender-receiver relationship is represented by building a special hierarchy of interaction classes and sub-classes. As the root class, an interaction class called "TransferEntity" is specified. It has subclasses which correspond to all potential recipient models, e.g. "TransferEntityToFactoryModel1," "TransferEntityToFactoryModel2," etc.

Further to this subclass identifying the target federate, individual interaction classes are introduced for each connection between a sending model and a target model. In our example (compare Figure 1), there would be a single subclass to "TransferEntityToFactoryModel2" called "TransferEntityFactoryModel1ToFactoryModel2."

This interaction class is published by the sending model (Factory Model 1). The receiving federate subscribes to that same interaction class. Please note that this is an approach different from the original ETS specification, reasons being explained in Strassburger (2006a).

For transmitting the state of a transferred entity, the interaction classes have a single parameter named "Entity." The type of this parameter is a complex data type (record) identifying the name of the entity, identifiers for the source and destination, and any simulation dependent attributes. In our industrial test application, simulation depended attributes to be transferred included a unique *sequence number* of the entity and a set of *option codes*.

The source and destination tags used in our implementation identify the sink of the sending model and the targeted source of the receiving model.

5.2 Enhancements for Synchronous Entity Passing

To enable the synchronous entity transfer some mechanism for checking if there is enough space in the input buffer/queue of the receiving model had to be implemented in addition to the solution for the Type A.1 IRM. Taylor et al. (2006) have suggested a protocol with synchronous "accept" and "block" messages for doing this. Other solutions could be based on a query protocol. In that case a sending model would first query the recipient if its input queue had enough capacity to accept the incoming entity. Depending on the answer from the receiving model, the entity would either be transferred or delayed.

Both suggestions have been considered in our implementation. However, due to several reasons a different approach was chosen. The most important reason is performance. Both suggestions required zero-lookahead interactions between the models, i.e. the lookahead of *both* models goes down to zero. This has tremendous performance implications in our scenario, as it was bound to use a conservative synchronization protocol. For a discussion of lookahead and synchronization algorithms for distributed simulation, the reader is referred to Fujimoto (2000).

The idea of our implementation is that the buffer (i.e. the input queue) is modeled as a persistent object class instance in the HLA sense. This is also attractive from a modeling perspective. We also suggest to use a hierarchical definition of these object classes similar to the one used for the transfer interaction classes in the ETS. Our implementation uses the hierarchy shown in Figure 3.

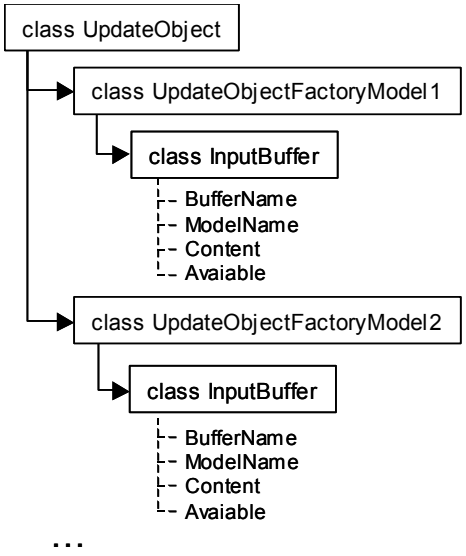

Figure 3: Class hierarchy used in the implementation.

Please note that in this hierarchy all attributes are defined at the leaf level. This may or may not be the most efficient approach. The hierarchy as presented has the advantage of being open to extensions needed for other reference models. For these models it is anticipated that other persistent object classes are needed (e.g., for implementing shared resources etc.). In that case all these

classes can be grouped under the super-class for the respective model.

An alternative hierarchy which could be considered would group the object classes by type rather than by model. Figure 4 shows this approach.

The approach has the advantage that attributes can be defined at the level of the respective class type, in our case "InputBuffer". Models which would be interested in all input buffers would only need to subscribe to the super-class "InputBuffer" and be able to receive all attribute information. If attributes were defined at the sub-class level this was not be possible.

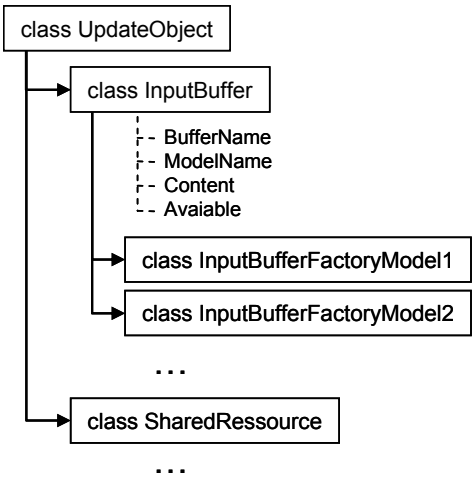

Figure 4: Alternative class hierarchy.

With these two alternative suggestions the authors hope to stimulate a discussion within the CSPI PDG about which approach should find its way into the standard definition.

Now, independent from which approach will prevail, each sending model needs to subscribe to the respective input buffer of the receiving model into which it wants to transfer entities. The receiving model updates this buffer whenever its contents is modified. Before transferring entities the sending models checks its local copy of the input buffer. Only if it has the required capacity, the entity is transferred, otherwise the transfer is delayed.

The overall working principle of the implemented solution is shown in Figure 5.

5.3 Lookahead and Performance Considerations

Now, how does the described approach help us to relax the zero-lookahead requirement? Theoretically all modifications of the buffer size need to be communicated to the sending model immediately, i.e. with zero lookahead. However, it can be assumed that the sending model is only interested in the fact whether or not the buffer has sufficient capacity. It is not interested in the exact number of its content.

With this knowledge it becomes possible to relax the zero lookahead constraint as follows: As long as the content of the input buffer stays below a certain percentage, e.g., 70 %, it is not required to send updates about the buffer with zero lookahead. The exact percentage depends on additional knowledge about the model, e.g., the minimum time interval between two entities leaving the system. This can, for instance, be derived from the minimum work time at the last station.

Only when a defined threshold in the input buffer is exceeded, the models must interact with zero lookahead for correct results. In this case the models in our implementation can dynamically change their lookahead values. Once the buffer frees up below a threshold, the lookahead can again be increased. The exact sizing of thresholds and performance measurements are subject of ongoing investigations.

It should also be noted that it really depends on the needed accuracy of the results if zero lookahead is required for the communication of the buffer size. Even in the reality of our test application this is not always the case. It may very well happen that some more entities are unloaded from the last station of assembly line 1 or 2 than there is space in the input buffer area of the next section, i.e. the paint shop. Such a behavior can happen because the controller of the assembly line is not informed in time or because of the actual travel time of the information (sometimes a manual process).

In our opinion this justifies our course of implementing the input buffer as object instances, simply because it gives more flexibility in the treatment of lookahead.

On the other hand, there are also some limitations which are not covered by our current solution. One limitation is that the solution only performs correctly if only one sending model transfers entities into one recipient buffer. If there was more than one model checking the same input buffer problems with simultaneous entity transfers from both models can occur, e.g., if only one unit in the buffer is available.

Again, in the practical application it must be asked what would happen in reality in this case. In our application it was very well acceptable and realistic that both systems would transfer their entities to the recipient section of the production.

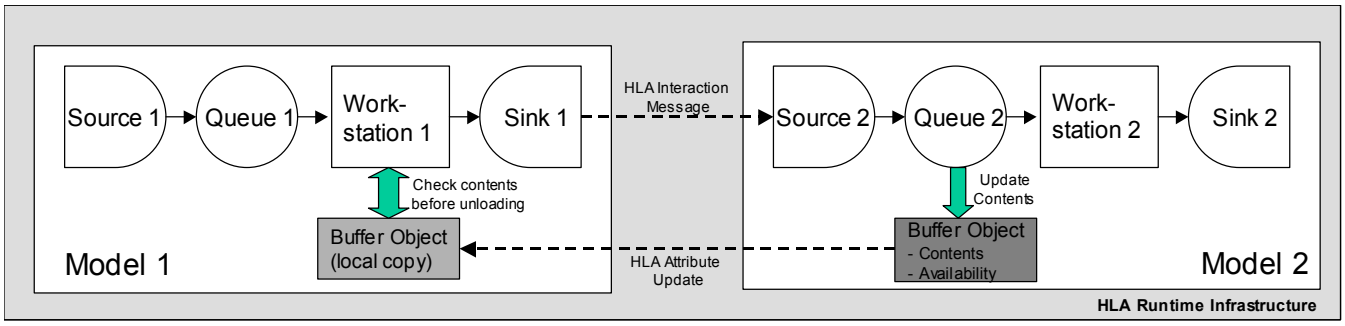

Figure 5: IRM Type II implementation based on buffer object classes.

6 BEST PRACTICES

In our test application we have coupled up to five different models. It can generally be stated that it is crucial to the performance of the federation to avoid zero lookahead. This can be achieved by either relaxing the accuracy requirements on the results of the distributed simulation, or by dynamically changing lookahead whenever possible. The latter requires obviously more effort and also depends on the real system being modeled.

In the context of the ETS it is crucial for performance to specify a travel time when passing entities in either Type A.1 or A.2 IRMs. When no travel time is used, both the departure event and the arrival event are in essence shared events which have to be processed by both models at the same time. The result is a zero lookahead requirement. It is crucial to avoid these shared events. In the Type A.1 and A.2 IRMs this is quite often easily doable because there is often a natural travel time between the models being coupled.

Performance of the federation also depends on the way a synchronization protocol is implemented. In our experience, it is worth trying not to synchronize each internal event with HLA NextEventRequest (NER) service. Rather, we have used a protocol which combines the usage of the HLA service QueryMinNextEventTime (QMT) and the NER service.

The protocol in essence avoids calling NER for each internal event. Rather, it calls QMT first. The QMT service asks for the lowest possible timestamp of the next external event which will be delivered to a federate. If the return value of QMT is large enough to process the internal event, this event and all other events equal or below the QMT return value are executed without any additional NER or QMT service invocation. Only if QMT returns a value smaller that the time stamp of the next internal event NER is called.

This simple mechanism is especially suitable for federate with larger than zero lookahead values because only in that case it can significantly reduce the number of NER invocations. In our experiments, the usage of the protocol yielded a performance improvement of the overall federation runtime of 33%. In absolute numbers, the runtime of a federation with four federates was reduced from 134 seconds to 90 seconds. This was achieved by reducing the total number of NER calls in the federation from 197088 (value without QMT usage) to 19966 NER calls in the version with QMT usage. The QMT service was called 10210 times to achieve this.

7 SUMMARY AND OUTLOOK

In the presented paper we have focussed on the investigation of the practical applicability of the Type A.2 IRM suggested by the SISO CSPI PDG.

A real industrial application has served as the background of this investigation. It should be noted that the project backing this investigation is continuing beyond the time of writing and further analysis with the distributed models will be conducted.

From the technical point of view the usage of the IRM and the suggested reference implementation was very successful. It is envisioned that the different IRMs currently being standardized will in future be helpful for classifying an interoperability problem and for selecting the right solution.

From the application side it was certainly the case that the reference application had the perfect setting to be solved by distributed simulation: the customer already had the different models of the production sections established. They only needed to be combined in order to investigate the overall system. Besides some technical issues not discussed here (e.g. different time representations) the implementation of the reference solution based on the SLX-HLA-Interface was straight forwarded and yielded no surprises which could not be solved.

Future work will include the conception and development of a HLA controller tool. The practical work with starting and configuring multiple federates has shown that such a tool is in fact needed if distributed simulation shall be used in the daily work. For our industrial test application, this controller tool would provide a unified access to the current user interfaces of the simulation packages and allow the user to configure, start and monitor the distributed simulation from a central point.

Future work may also include the investigation of web-service technologies as a way of remotely providing simulation services. In that case, the (distributed) simulation could be executed on a powerful multi-processor server machine while the user interface is located on the client side.

On the theoretical side more work towards quantifying any potential inaccuracies introduced by relaxing the lookahead restrictions would be really interesting and beneficial.

REFERENCES

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

Sly, D. and S. Moorthy. 2001. Simulation data exchange (SDX) implementation and use. In *Proceedings of the 2001 Winter Simulation Conference*, ed. B. A. Peters, J. S. Smith, D. J. Medeiros, and M. W. Rohrer, 1473-1477. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Straßburger, S. 2006a. The road to COTS-interoperability: from generic HLA-interfaces towards plug-And-play capabilities. In *Proceedings of the 2006 Winter Simulation Conference*, ed. L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, 1111-1118. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Straßburger, S. 2006b. Overview about the high level architecture for modelling and simulation and recent developments. In *Simulation News Europe* 16(2): 5-14.

Strassburger, S., T. Schulze, U. Klein and J. O. Henriksen. 1998. Internet-based simulation using off-the-shelf Simulation tools and HLA. In *Proceedings of the 1998 Winter Simulation Conference*, ed. D. J. Medeiros and E. Watson, 1669-1676. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Taylor, S. et al. 2003. HLA-CSPIF discussion document entity transfer specification. version 1.1.1. Available via <http://people.brunel.ac.uk/~csstsjt/HLA-CSPIF/ets1_1_1.doc>.

Taylor, S., S. Turner, M. Low, X. Wang, S. Straßburger, and J. Ladbrook. 2006. Developing interoperability standards for distributed simulation and COTS simulation packages with the CSPI PDG. In *Proceedings of the 2006 Winter Simulation Conference*, ed. L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, 1001-1110. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Taylor, S., N. Mustafee, S. Strassburger, S. Turner, M. Low, J. Ladbrook. 2007. The SISO CSPI PDG standard for commercial off-the-shelf simulation package interoperability reference models. In *Proceedings of the 2007 Winter Simulation Conference*, ed. S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

AUTHOR BIOGRAPHIES

STEFFEN STRASSBURGER is a professor at the Technical University of Ilmenau 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 Master's 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>.

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>.

MARCO LEMESSI is a member of the Simulation Group in the Industrial Engineering Department of Deere and Company, the worldwide corporate headquarters of John Deere. He received his Ph.D. (2002) in Traffic Engineering and M.S. (1998) in Civil Engineering from the University of Rome "La Sapienza," Italy. Before joining Deere and Company in 2002, he has been an expert consultant in several EU transportation research projects, and has lectured in the Department of Environmental Engineering of the University of Perugia and in the Transportation Department of the University of Rome "La Sapienza." He can be reached by email via <LemessiMarco@JohnDeere.com>.