Using HLA Ownership Management in Distributed Material Flow Simulations

Steffen Straßburger
Andreas Hamm
Günter Schmidgall
Siegmar Haasis
DaimlerChrysler AG - Research & Technology
Wilhelm-Runge-Str. 11
89013 Ulm, Germany
+49-(0)731-505-2443 (-2377,-2330)
{steffen.strassburger, andreas.hamm, guenter.schmidgall, siegmar.haasis}@daimlerchrysler.com

Keywords:
Ownership Management, Material Flow, Engineering and Production Simulation

ABSTRACT: *This article describes the application of distributed simulation under the leitmotiv of the Digital Factory (DF). The objective of the DF is that no production facility shall be planned, constructed, and operated without a complete prior digital coverage. In order to have a detailed digital representation of the real factory which covers all relevant causal relationships it is necessary to combine several independently developed simulation and planning models which each represent an isolated part of the factory. This requires a suitable IT architecture which is capable of connecting the individual simulation models representing a process chain. HLA seams to be an appropriate choice for this purpose.*

In this article we focus on the application area of material flow simulations. For this area the HLA ownership management services play a very important role. Our research has shown that HLA Ownership Management is generally capable of being helpful in this area, but not sufficient. It is especially unfortunate that there are
a) no time managed versions of these services
b) no possibilities of specifying a recipient of a transferred object/attribute,
c) no possibility to transfer the final state of the object regarding attribute values.
Related work has already been performed to analyze these deficiencies [1,2,3], but to date no solution has been implemented in the HLA interface specification.

Therefore this article picks up the discussion about the topic and suggests a solution especially suited for material flow applications. The presented solution is not intended as a fixed and final solution to the problem. Rather, it is intended as a basis for discussion among industry partners and standardization boards like OMG or IEEE about the standardization of a solution regarding the aforementioned problems.

The paper describes the introduction of "connection points" (CPs) as a way of specifying sender/receiver relationships for transferring entire object instances with their attributes. CPs are especially useful in material flow scenarios, but also applicable for general logistical problems.

1. Introduction

The paper focuses on the U.S. Department of Defense (DoD) High Level Architecture (HLA) Ownership Management Services and its application for material flow scenarios within the Digital Factory (DF).

The objective of the DF is that no production facility shall be planned, constructed, and operated without a complete prior digital coverage. Among other aspects this objective also requires the heavy use of simulations to plan and analyze the factory operation.

For providing the connectivity between different tools and systems applied within the DF HLA seems to be an appropriate choice. Some of the limitations of HLA in the area of ownership management are discussed in this article. We propose a solution which provides the missing functionality for scenarios within the DF and suggest to discuss about standardization efforts.

2. Distributed Simulation in the Digital Factory

With the idea of the Digital Factory the requirements regarding the application of simulations increase: The objective is to have a detailed digital representation of the real factory which covers all relevant causal relationships. Furthermore, the simulations shall not only be applied in the planning phase of the factory, but also during its actual operation. Towards this objective it is necessary to combine several independently developed simulation models which each represent an isolated part of the factory. Also, it may be required to integrate operative IT systems into distributed simulations, e.g., to run simulations based on the real data from the current state of the factory.

2.1 The tools and the need to connect them

Simulation systems which need to be integrated into a complex distributed simulation in the DF may include typical material flow simulators like QUEST from DELMIA and eM-Plant from Tecnomatix. When tools like these (which operate on the same level of detail) are connected, we call this is a *horizontal integration*.

However, it may also be required, that tools modeling at different levels of detail are integrated for investigating a specific problem. In a DF scenario, it may be required to include robot simulations like IGRIP and ROBCAD from the same respective vendors. The latter is called a *vertical integration*, since tools operating at different levels of detail need to be connected (Figure 1).

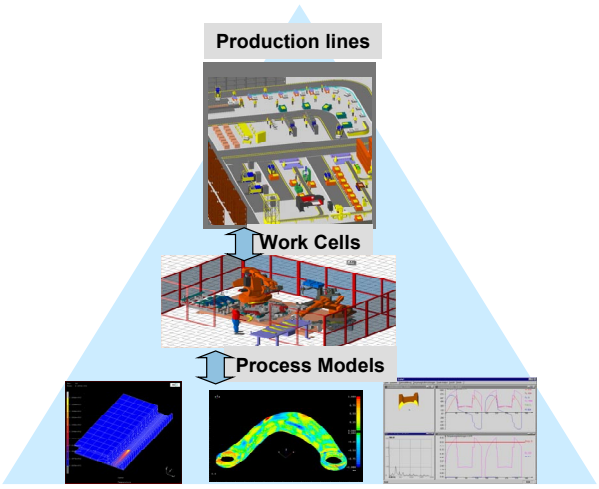

Figure 1: Vertical Integration

2.2 The concept of a universal federate adapter (UFA)

Connecting simulation systems and other IT systems to the RTI is a non-trivial task which requires in-depth knowledge of the HLA Federate Interface Specification and a significant amount of development time [4].

Taken the fact, that much of the code for connecting a system to the RTI is almost the same for each application, certain functionality could be packaged and easily reused.

For providing this capability of reusing the low level functionality to connect a system to HLA and for simplifying the communication of applications with the RTI DaimlerChrysler has developed a software package called *universal federate adapter* (UFA).

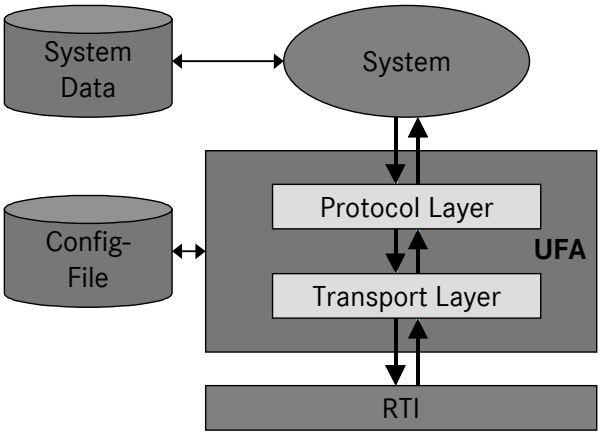

Figure 2: Architecture of the Universal Federate Adapter

The UFA is based on a layer design pattern. Currently two layers are included:

- A transport layer, which encapsulates all relevant RTI functionality (method calls to RTI as well as federate ambassador)
- A protocol layer, which needs to be adapted to the system which shall be connected to the RTI.

The transport layer is implemented as a library with a lean interface to the protocol layer (15 required functions and 16 optional functions for providing additional information). This reduces the complexity involved when connecting a system to the RTI and frees the programmer from the need of becoming an HLA expert.

It is especially important to note, that the UFA has been very carefully designed under the aspect that no

proprietary protocol is introduced with the UFA, since this would be contra-productive to the aims of HLA. Our objective was that systems which are connected to HLA via the UFA can easily interact with non-UFA federates.

Other solutions, e.g., the IMS Mission Architecture [6,7], have taken a different approach. They have developed an adapter for distributed manufacturing simulations (DMS) which is based on HLA's RTI as communication backbone. For adjusting to the needs of globally distributed enterprises, this architecture makes application specific enhancements and assumptions about the participants and the way they interact with each other. This works fine as long as all federates agree on being based on DMS as a de-facto standard above HLA. One drawback of the approach is that it becomes difficult for federates not built for DMS to participate in a DMS simulation.

Although the results obtained within the mission project are significant (several simulation systems have been connected with DMS), we felt that in our research and development efforts we should try to strictly stick to HLA and make no proprietary enhancements (e.g., by introducing specific object models, handshake mechanisms, startup-procedures, etc.).

The only area where this attempt has failed is the area of ownership management/transfer. Without agreeing on a certain protocol above HLA it is not possible to implement a synchronized and directed ownership transfer. In the section 3 we shall report on our findings in this area and suggest a protocol to overcome the problems in our application area.

3. Ownership Transfer

The current HLA interface specification is generally capable of transferring object attribute ownership. Ownership can be transferred for single attributes or sets of attributes of an object instance. There is no special functionality for transferring entire object instances between federates. The closest to migrating entire objects is to transfer all attributes of the instance and the special attribute "PRIVILEDGE TO DELETE". With the latter it is possible to transfer the right to delete an object instance from its creator federate to a different one.

3.1 The Need for Ownership Transfers

In many logistical scenarios it is very important to model entities which are moved from one simulation to another. In military simulations the entity may represent a tank which is transported to the battlefield. In industrial material flow applications such an entity may be a package, a manufacturing product (e.g. a car) or something similar.

In almost all material flow scenarios sinks and sources are very important simulation elements. Sinks produce items according to some statistical distribution or real order data from the factory. Afterwards the items move through the system according to route and processing logics or strategies. When the item reaches a sink element it is removed from the system.

This is the basis for almost all material flow simulations. With this source-sink concept in mind, it is very easy to partition a material flow simulation into several individual models. One can insert a special sink-source pair at the desired system boundary which implements the task of transferring the items between the submodels (Figure 3).

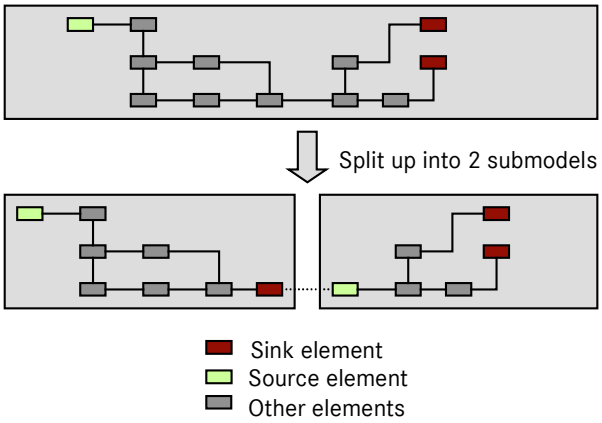

Figure 3: Creating a distributed material flow simulation

From the HLA point of view it is necessary to transfer the ownership of the object instance representing the item. The side conditions typically are that the transfer needs to take place at a specific time and must be directed to a specific receiver (since it is usually a 1:1 relationship between the sink-source pair).

Furthermore it would be beneficial, if the receiver could get the latest values of all instance attributes with the transfer, since there is no need for him to know them before he owns the object.

3.2 What's missing in the current version of the services

Several deficiencies of the ownership management services exist and have been reported [2,3,8]
Here is a short summary of the critics:

1) There is no directed ownership transfer.
2) There is no possibility to explicitly reject transfer of ownership.
3) There is no possibility to include the last valid attribute values in the transfer.
4) The ownership management services are not time managed.

The last item has the consequence that there is no guaranty from the RTI at what federation time the transfer will take place.

This is especially unfortunate for all as-fast-as-possible simulations operating with logical simulation as it is the case in our application area. Comparable problems in the area of data distribution management have been discussed in [1].

3.3 Workarounds and the need for standardization

It is acknowledged that although these problems exist, there are ways to work around these deficiencies. One could, for example, introduce federation specific protocols to include the recipient ID in the userSuppliedTag field when initiating an ownership transfer. This would provide a mechanism for the directed ownership transfer.

For solving the problem of having no mechanism to reject an ownership transfer one could use time out mechanisms, as it is done in our solution with the UFA.

For our core problem (the non-time-managed character of the ownership management services), on could, for instance, specify the desired time stamp of ownership transfers using a special interaction message which is sent before initiating the transfer via ownership management services. With this one could introduce a proprietary protocol for the ownership transfer which is directed and time-synchronized.

Other workarounds address the problem that the RTI does not store any information about released object instances, and thus that it is not easily possible to transfer the "best and final state of object instances"[3,8].

The problem with all these solutions and workarounds is that they introduce federation specific protocols which are counter productive towards the goal of interoperability based on a common standard. A federate from a federation using a certain workaround for the ownership transfer problems will most likely not be interoperable with a federate from a federation using a different workaround.

The next section describes our solution to the problems and is intended as a suggestion for standardizing a comparable solution among industry partners.

3.4 Protocol for a directed, time-stamped ownership transfer

As mentioned in section 3.1 virtually all material flow scenarios can apply a simple sink-source-concept to connect different material flow federates.

From the modeling point of view each two federates can be connected by one or more sink-source connections which we call *connection point*s.

A connection point (CP) is a user defined directional connection for transferring items from one exit element (e.g., a sink) of a simulation to an input element (e.g. a source) of another simulation.

Each such connection point has a federation unique identifier which is used in the ownership transfer to identify the receiver of the transfer request. **This solves the problem that ownership management services are not directed to a specific recipient**.

By our definition a CP always implies a 1:1 connection between elements of two simulations. In scenarios, where two federates can deliver items for one recipient, one would in fact have to introduce two connection points and potentially an additional input element in the recipient's simulation.

The process of connecting multiple federates by creating connection points can be supported by tools like Federation Builder (FB), which is currently under development at DaimlerChrysler[1].

For solving the problem that the ownership management services are not time managed we have introduced a special protocol using ownership management services in conjunction with an interaction message which is time stamped. The ownership

[1] FB provides all required functionality of a simulation and object model repository, like storing and retrieving simulation models, merging (i.e., combining) SOMs and FOMs, connection simulations using connection points, and exporting the whole federation with all federates.

management services use the userSuppliedTag for transmitting the CP identifier. The interaction also carries the CP identifier as well as the instance name and the desired time stamp.

Figure 4 shows the sequence diagram for the federate initiating the transfer. Please note that the UFA encapsulates all required functionality.

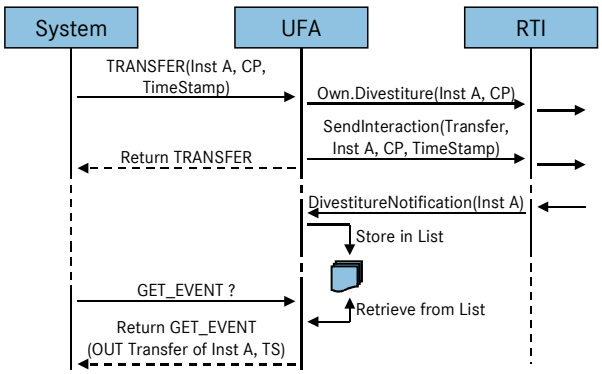

Figure 4: Sequence diagram for a federate transferring an object instance

A federate simply calls a function called "Transfer" to initiate the transfer. During subsequent calls of "Get_Event"[2], the federate will be informed about the success of the ownership transfer.

Inside the transfer call the UFA initiates the ownership transfer and sends an special interaction with a time stamp, the name of the instance to be transferred, and the connection point identifier.

The sequence diagram for the receiver side is shown in Figure 5. The receiver will usually first receive an ownership assumption request indicating the availability of the instance. It then checks whether it is the correct recipient (i.e., if the CP is defined as incoming for the simulation). After that the UFA delays the acquisition until the receipt of the transfer interaction tells it the right time stamp at which the transfer shall take place.

In some occasions, the transfer interaction may be received before the ownership request. The working principle in this case is very similar, i.e., the UFA will wait until the ownership assumption request is received.

[2] The Get_Event function is the mechanism for retrieving all incoming information which the UFA has received via its Federate Ambassador and stored internally. Get_Event frees the simulation from dealing with tick() and related methods.

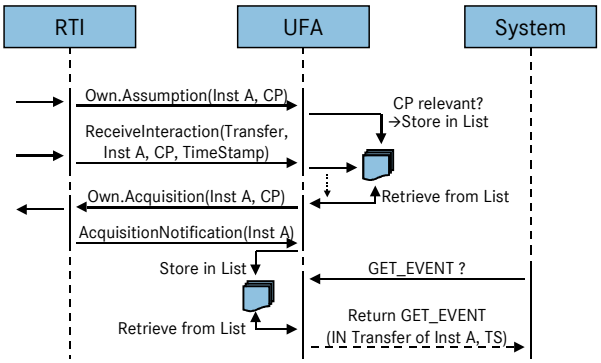

Figure 5: Sequence diagram for a federate receiving an object instance

In any case the UFA knows the right time stamp at which the ownership shall logically be processed by the system and can deliver it to the system when the GetEvent call is issued.

This protocol solves the problem that ownership management services are not time managed. The drawback is that it introduces a proprietary arrangement which each federate should follow to avoid compatibility problems.

The problem that it is not possibility to include the last valid attribute values in the ownership transfer is also solved internally in the UFA. The protocol layer of the UFA keeps a list of all relevant object instances (i.e., all instances of subscribed classes which may be transferred to it at a later time) and their current attribute values. When an instance is transferred to the federate the UFA informs the federate about the transfer and also about the attribute values with which the object instance needs to be initialized.

We have chosen this approach because of its general applicability. Another way to deal with the problem would have been to include the latest attribute values in the transfer interaction. We have not chosen this approach because it would introduce another proprietary agreement.

Please note that our approach is necessary since our material flow simulations usually do not need to keep a copy of remote objects in their internal representation. Therefore the UFA stores it until it is needed. Although doing so, our approach is also applicable for training simulations which do need to represent the state of remote objects in their virtual environments, because the UFA can also be told to forward any update to the simulation immediately.

This implementation agreement circumvents the problem that it is not possible to include the last valid attribute values in the ownership transfer.

The last mentioned problem from section 3.2 was that there is no possibility to explicitly reject an ownership management transfer. This problem was reported in [2]. In our case this problem does not apply since all instances are transferred in push-mode. The only case were an ownership transfer can fail is if the recipient federate is not able to accept ownership (e.g., because it has not joined the federation). In this case the UFA applies a time out mechanism which aborts the transfer if it was not successful for a certain amount of processing time.

4. Conclusions

The paper has described the introduction of "connection points" (CPs) as a way of specifying sender/receiver relationships for transferring entire object instances with their attributes. CPs are especially useful in material flow scenarios, but also applicable for general logistical problems.

Furthermore a time-synchronized ownership management protocol has been described, which allows the directed transfer of object instances.

It is essential to stress the need of standardization in this area. We are not suggesting that our solution is best suited in any case. Rather, it can be seen as one possibility for solving the problems. This could be the basis for finding an standardized solution or at least an agreement among industry partners from the automotive sector and its suppliers. Other input for standardization can be taken from [5].

The straight forward way of dealing with the problem would of course be an enhancement of the HLA Interface Specification. Unfortunately this is not very likely to happen.

5. References

[1] R. Fujimoto, I. Tacic: "Time Management of Unsynchronized HLA Services", 1999 Fall Simulation Interoperability Workshop, Sept. 8-12, 1999. Paper Number 99F-SIW-165.

[2] G. Sauerborn et. al: "HLA Ownerhsip Management Services: We Almost Got it Right", 2000 Fall Simulation Interoperability Workshop, Sept. 17-22, 2000. Paper Number 00F-SIW-076.

[3] M. Myjak, S. Sharp, T. Lake, K. Briggs. "Object Transfer in HLA". 1999 Spring Simulation Interoperability Workshop, Mar. 14-19, 1999. Paper Number 99S-SIW-140.

[4] S. Straßburger: "Distributed Simulation Based on the High Level Architecture in Civilian Application Domains". Ghent: SCS-Europe BVBA, 2001. ISBN 1-565552180.

[5] F.-W. Jaeckel, M. Rabe: "Input to Standardization Working Groups", Deliverable D21. Result of the IMS Mission Project. Publicly available at http://www.ims-mission.de.

[6] C. McLean, F. Riddick, S. Leong: "Architecture for Modeling and Simulation of Global Distributed Enterprises". In: Proceedings of the 9th ASIM Dedicated Conference, eds. K. Mertins and M. Rabe. ISBN 3-8167-5537-2.

[7] C. McLean, F. Riddick: "The IMS MISSION Architecture for Distributed Manufacturing Simulation". In: Proceedings of the 2000 Winter Simulation Conference, eds. J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, pp. 1539-1548.

[8] M. Myjak, T. Lake, D. Roberts, B. Worthington: "Timing: Mechanisms for Ownership Transfer". 2000 Spring Simulation Interoperability Workshop, Mar. 26-31, 2000. Paper Number 00S-SIW-140.

Author Biographies

STEFFEN STRASSBURGER is a research assistant at the DaimlerChrysler Research Center in Ulm, Germany. He holds a PhD and a Master's degree in Computer Science from the Otto-von-Guericke University in Magdeburg, Germany. His international experience includes a one-year stay at the University of Wisconsin, Stevens Point and a stay at the Georgia Institute of Technology, Atlanta. He actively participates in several international conferences. His main research interests lie in distributed and web-based simulation and the High Level Architecture.

ANDREAS HAMM is has studied Computer Science at the Technical University in the Ilmenau, Germany. Main emphasis during his studies were software technology and mathematics. His Diploma Thesis was carried out at the DaimlerChrysler Research Center in Ulm, Germany, and dealt with the development of an plug-and-play-capable interface for distributed simulations. In 2001 Mr. Hamm joined the Knowledge Transfer Group of DaimlerChrysler.

GÜNTER SCHMIDGALL has studied Mechanical Engineering at the Technical College Heilbronn. From 1982 to 1992 he carried out different occupations in the areas of applied CAx technologies and optimization of product development chains. From 1992 to 1998 he worked as a freelancer in the areas IT for engineering and new technologies for the product development. He also participated in several research projects. Since 1998 Mr. Schmidgall is responsible for the Distributed Simulation group at the DaimlerChrysler Research Center in Ulm.

SIEGMAR HAASIS holds a PhD degree in Mechanical Engineering (CAD/CAM, Expert Systems, Feature Technology) and a Diploma in Mechanical Engineering and Computer Science. In 1995 he joined the Daimler-Benz International Management Associate Program. He has worked as a Project Leader within Technology Development at Untertürkheim. Since 1999 he is the Senior Manager of the Research and Technology Department "Product, Process, Resource Integration" within the Lab "IT for Engineering".