

HLA on top of CORBA Common Object Services

Reinhard Herzog

John Mulder

Fraunhofer IITB

Fraunhoferstr. 1

D-76131 Karlsruhe, Germany

+49-721-6091 294, +49-721-6091 480

hzg@iitb.fhg.de, mul@iitb.fhg.de

Kay Pixius

Hans-Peter Menzler

Wehrtechnische Dienststelle der Bundeswehr

für Fernmeldewesen und Elektronik (WTD 81)

Kalvarienberg

D- 91171 Greding, Germany

+49-8463 652-546; +49-8463 652-599

KayPixius@bwb.org, HPMenzler@bwb.org

Keywords:

Object Modeling, CORBA, HLA, pSISA, Interoperability

ABSTRACT The simulation oriented world of the HLA and the industrial oriented world of CORBA are moving closer together. Many CORBA-based applications have the requirements to provide a time-synchronised service quality and also a model-based communication architecture. This is supported by an approach of the OMG to define the HLA as a part of the standard services. On the other side, the typical service level for average industrial CORBA application is the "Common Object Service (COS)" level.

The paper gives an overview of a study based on a prototype implementation of mapping pSISA to CORBA, by using the concept of mapping the HLA services onto standard COS services. It presents a component based HLA service organisation by utilising the standard common object services defined by the OMG. This allows the building-block usage of the HLA functionality, which is important in order to provide a smooth migration to the HLA service world ("light-weight HLA"). It also demonstrates ways to connect the HLA world with non-simulation CORBA applications.

1 Introduction

In mid 1999 the Federal Armed Forces Technical Center for Communications and Electronics (WTD 81) began developing a slim API to the RTI, hereafter referred to as pSISA – the *proposed Standard Interface for Simulation Applications*.

Details of pSISA were presented at the SIWs since then [1 - 3].

The main benefits of pSISA for distributed simulations underneath the IEEE 1516 are:

- To hide the peculiarities of the HLA interface from the application programmer, thus automating many routine work;

- To become independent from an individual IEEE 1516.2 (I/F-Spec) conform RTI implementation;
- To support *semantic interoperability* by introducing an auto-generated, intuitive API based on the specified SOM. That API contains pure virtual object-oriented methods and objects, respectively.

The present work focuses on the second bullet, i.e. establishing an individual communications infrastructure, thereby replacing DMSO's freeware RTI implementation as used in our former projects. This prototype of a CORBA-based infrastructure merged with pSISA is named by the nickname "GERTICO¹".

¹ German RTI Is CORBA

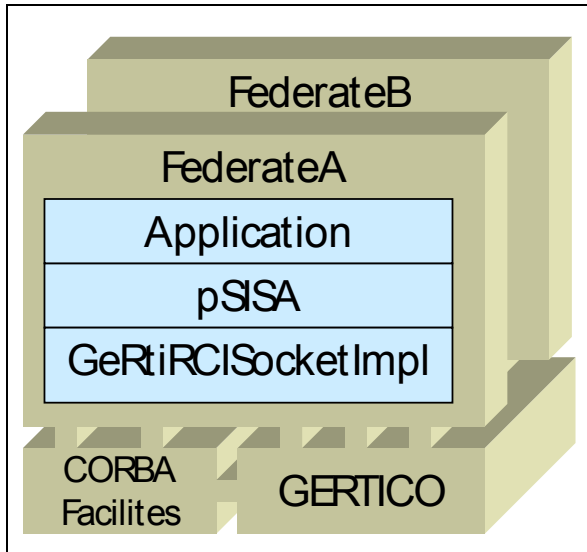


Figure 1: The concept of pSISA and the underlying CORBA based communication infrastructure GERTICO.

2 On HLA interoperability

As pointed out in the report of the study group [5] the topic of interoperability departs in final consequence into four categories. Among these, only one is relevant to our research: interoperability among federates based on a homogeneous FOM and interacting through a homogeneous RTI.

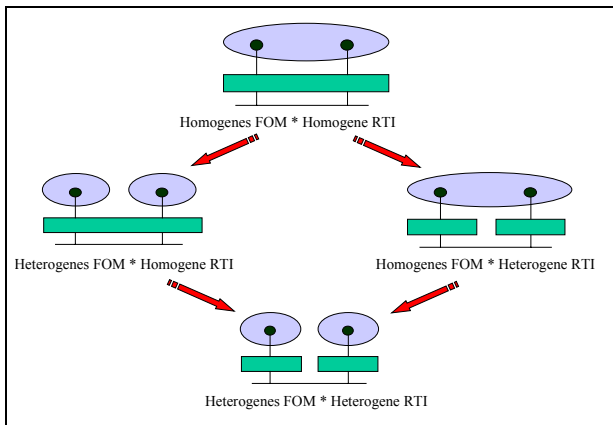


Figure 2: Issues of interoperability.

(after RTI-Study Group[5])

Our research on the pSISA approach revealed that it covers at least those aspects of interoperability as delineated as follows:

Portability: Components of our CORBA infrastructure can be addressed via pSISA's RCI interface. Thus, an exchange of the RTI underneath the RCI layer does not affect simulation applications running under pSISA.

IOP-S²: GERTICO may use services from other RTI implementations as well as vice versa. This is because pSISA shields the application from services of specific RTI versions.

IOP-MW³: As GERTICO exploits the CORBA standard, it may enable bridging to other state-of-the-art middleware technologies, like DCOM, J2EE.

3 Design Requirements

There are several design requirements for the GERTICO architecture. The most important ones are:

- GERTICO should be a CORBA native application;
- GERTICO should be component-based (non monolithic, exchangable modules, individually usable modules);
- GERTICO should be scalable;

As mentioned earlier, GERTICO can be seen as a architecture study. It should be investigated at which degree HLA services can be organized in a scalable and distributed component system.

3.1 What does CORBA native mean?

The term CORBA (Common Object Request Broker Architecture) it often used as a synonym for the Object Management Architecture (OMA), shown in **Figure 3**.

² IOP-S: Interoperability on service level.

³ IOP-MW: Interoperability on middleware level

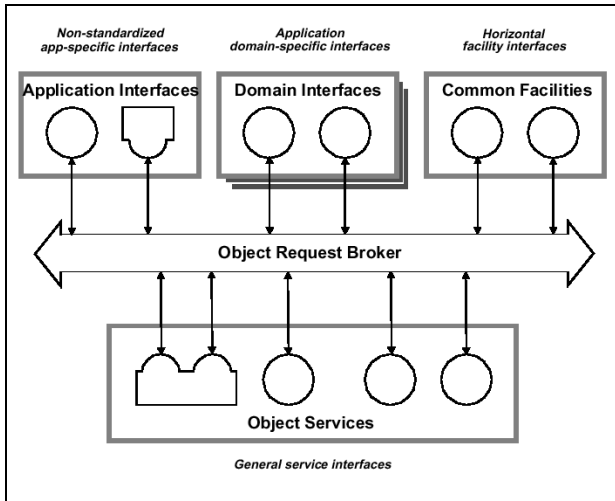


Figure 3: Object Management Architecture
(after [8] OMG)

The OMA contains an important concept – the CORBA-Services. These can be seen as a collection of design patterns and implementations of services for common functionality's. A application which behaves and looks like a typical CORBA program, should respect these standardizes and widely accepted CORBA-Services.

In figure 4 the HLA and the basic CORBA Services are shown. The vertical arrangement of the services demonstrates a rough similarity of the two services groups. E.g. HLA *Federation Management* has something to do with defining names (*CORBA Name Service*) with creating and deleting federations (*CORBA Life Cycle Service*) and also with managing information about federate properties (*CORBA Property Service*).

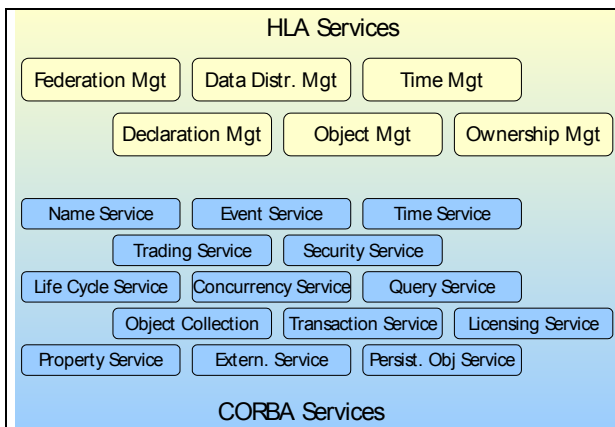


Figure 4: Relevance of CORBA Services for HLA Services

These similarities should not be over rated. It is surely not the case, that HLA services can be realized by just assembling CORBA services. So for example the *HLA Time Service* could possibly be implemented with some kind of *CORBA Transaction Service*, but this would most likely not end up in a good solution.

It is always a tradeoff between re-using an existing and widely accepted solution versa defining a new, and maybe optimal solution.

3.2 A component based Architecture

The design of GERTICO is driven by the idea of a component framework. In the very contrast to existing RTI implementations available from DMSO, our approach is to avoid a monolithic design, but to establish separate components with well-defined interfaces. This idea is similar to the framework of the J2EE: it enables a tailored communication infrastructure, which includes only the necessarily requested services.

With respect to practical experiences with the HLA, only a small fraction of the available RTI services are needed for running a “regular” simulation; e.g. a time-driven message exchange mechanism is hardly ever used for live-simulations. However, in that special case a very “slim” RTI is needed, which is freed from overhead as much as possible.

Components of GETICO should consist of exchangeable and individually usable modules. This allows a better adoption to specific environments by providing modules with a required service quality (best performance or deterministic real-time or lowest resource usage or ...). It also improves the re-usability of containing modules, because modules with specific functionality can be used without the complete framework.

Our architectural approach of GERTICO is depicted in Figure 5: in the smallest configuration (left/yellow), CORBA Name and Event Services are used for the data exchange among arbitrary CORBA applications; in a more mature stage (middle/blue), class and interaction patterns from GERTIFactory and GERTIMarket are included to support HLA specific functionality, i.e. including semantic aspects as defined in the FOM.

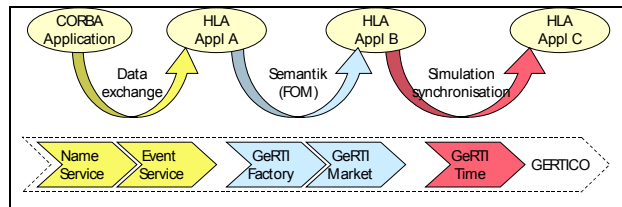


Figure 5: Component based Architecture of GERTICO

Other services can be included as optional services, adding functionality only required for some simulation applications, for example the synchronisation service (right/red).

3.3 Scalability

Future applications which can be envisaged for simulations in virtual environments will most likely not just consist of a handful of federates. A centralized RTI architecture with a message-based communication strategy, has a conceptual restriction according to the number of connectable federates. Bandwidth limitations and the bottleneck of a single-process RTI do not allow the vision of a large scale federation.

4 CORBA as Runtime Communication Infrastructure

The Common Object Request Broker Architecture (CORBA) is the vital center of the Object Management Architecture (OMA) and is hence widely accepted and employed for distributed applications in general.

Obviously, many problems one encounters in the field of distributed applications were already solved by CORBA and therefore it appears at least reasonable to have a closer look to the peculiarities of CORBA.

Strategies for Implementing

Within our project several options were checked. A summary is shown in Figure 6.

After reviewing the CORBACap solution it turned out that implementing GERTICO as described within this paper was the easiest solution. For details refer to [4].

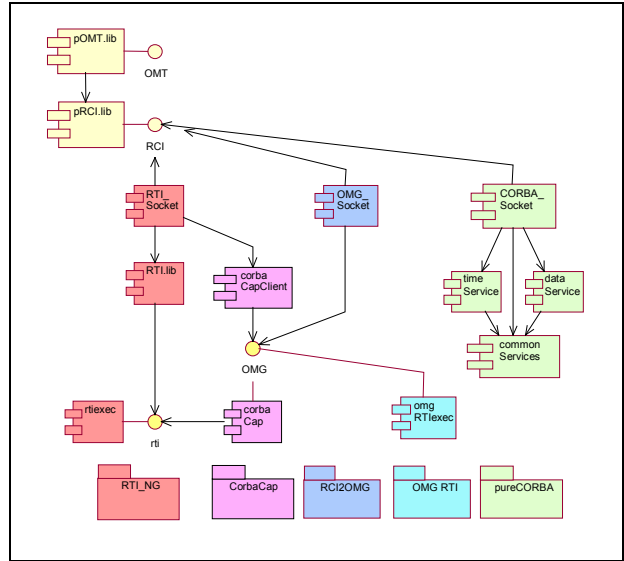


Figure 6: Sketch of several options to “plug” CORBA underneath HLA. Finally, we decided to implement a CORBA based RTI (right/light green).

GERTICO Architecture

Entry to GERTICO is via a module named *CosNaming*, i.e. the standard name service for resolving references to objects.

The responsibility for administering names is with the *GeRtiFactory*: federates can identify relevant information (i.e. class vs. interaction) via a name tree.

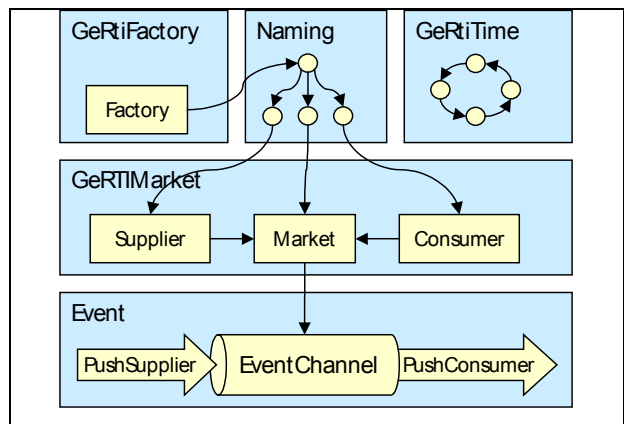


Figure 7: The basic GERTICO modules.

GeRtiMarket is a GERTICO specific implementation of the CORBA component *CosEvent* and is used for data exchange.

The runtime environment structure is as follows: the executable *GeRti.exe* as the vital part is started once in a

federation; each federate appears as an executable itself: note, that it is structured conform to pSISA, i.e. it consists of an application specific part, the pSISA library and an implementation of the RCISocket as specific to GERTICO. Besides an instance of the Naming service is required as indicated in Figure 1 as *NS.exe*.

In other words: a HLA federation is set-up by the NameServiceServer, the GERTICO Server and the federates.

In more detail, GERTICO consists presently (May 2001) of three modules:

- GeRtiFactory;
- GeRtiEventMarket;
- GeRtiTime (projected);

plus the *CosNaming* standard CORBA module.

Scalability

The data exchange in GERTICO is based on the concept of *Event Channels*. Basically an event channel receives data from a group of *Publishers* and delivers these data to a group of *Subscribers*. Optionally a subscriber may set up filters, in order to received only data which he is really interested, e.g. a subset of attributes out of a set of class attributes. In GERTICO one class is assigned to one *Event Channel Object* (see figure 8). The default situation is that all event channel objects are realized in one server process. However, it is foreseen to organize the mapping of the CORBA objects to server processes according to the actual workload situation. In CORBA the concept of a “smart proxy” supports such dynamic organizations.

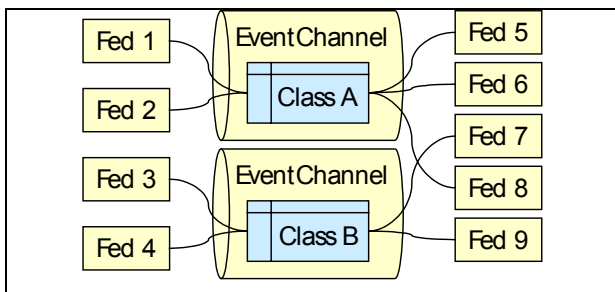


Figure 8: Building sub-domains with event channels

As a result of such dynamic reorganizations the run-time infrastructure can build local “sub-domains” where the work load and the communication paths are equally distributed.

Covering the HLA Services

GERTICO maps the HLA services onto the three above mentioned modules as depicted in Figure 9.

This mapping arrangement is due to the similarity to the definition of standard CORBA services.

It has to be stressed that presently only those HLA services are covered by GERTICO, which are already covered by pSISA. Unfortunately, the time-management is excluded thus far. However, as time-management is necessary for more complex distributed simulation applications, this gap will be filled in the near future by us.

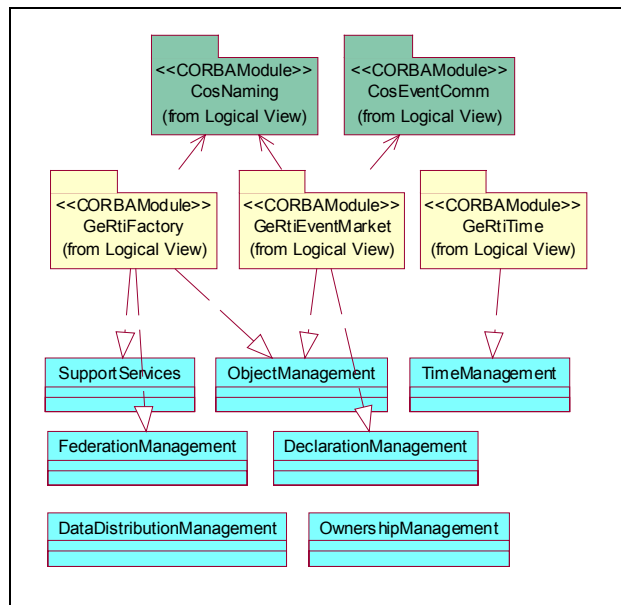


Figure 9: Mapping of HLA services to GERTICO.

5 A proof-of concept

In it's current version GERTICO is not a fully grown HLA run time infrastructure. It should be seen as a case study for a component based RTI architecture.

The implementation of services following the IEEE 1516 as required by pSISA's RCI layer was validated by running a reference application called “Billiard”.

That applications simulates simply a distributed billiard game, with the queue and the holes modeled by one federate and the balls are modeled by another federate.

Objects in this federation are the balls and the queue. As interaction – in terms of an HLA interaction – there is only the trapping of a ball in a hole.

That application used to be run on DMSO's RTI1.3NG. We were able to replace DMSO's RTI successfully by GERTICO.

GERTICO is not yet optimized in terms of performance. However, benchmarks have shown, that for a small number of attribute and interaction updates (up to about 100 updates per second) GERTICO is comparable to the RTI1.3NG. With more updates the RTI1.3NG shows unsystematic oscillation, but in general a better performance than the GERTICO.

The reason for this is, that in GERTICO the *Tick* concept was not implemented. The method *Tick* is used to control the interaction between a federated and the RTI executable. This method is not defined in the IEEE standard, but realized in the DMSO RTI's. It is clear that also GERTICO needs a similar concept, but it is not yet decided how to implement this in a HLA conformant way.

6 Conclusion & Outlook

This paper has complemented our previous work on the intercorrelation of CORBA and the HLA.

It turned out that the Federal Armed Forces' pSISA approach is well suited to support the application programmer to have its simulation application working in a federation compliant to the IEEE 1516.

Furthermore, a selection of RTI services were re-implemented either by use of existing CORBA services or by introducing our own source code, in order to provide a proof-of-concept.

It could be demonstrated, that CORBA is well suited to support a flexible component-based and scalable RTI implementation.

7 References

- [1] H.-P. MENZLER, U. KROSTA, K. PIXIUS: "HLA in an Nutshell: *ψ-SA Proposed Standard Interface for Simulation Applications*";(00S-SIW-026)
in: Proc. Simulation Interoperability Workshop SIW Spring 2000, 26.- 31. March 2000, Orlando, FL, U.S.A.
- [2] H.-P. MENZLER: „*A Proposed Standard Interface to Improve Interoperability of Simulation Applications*“; (01S-SIW-022)
in: Proc. Simulation Interoperability Workshop SIW Spring 2001, 25.- 30. March 2001, Orlando, FL, U.S.A.
- [3] T. WEYRATH, A. TOLK, K. PIXIUS, H.-P. MENZLER, S. KRUSCHE, A. DABELSTEIN, D. KUNDE: „*Decision Support Tools II – Insights from a German R&D Study*“; (00F-SIW-038)
in: Proc. Simulation Interoperability Workshop SIW Fall 2000, 17.- 22. September 2000, Orlando, FL, U.S.A.
- [4] T. USLÄNDER, R. HERZOG, J. MULDER, K. PIXIUS, H.-P. MENZLER: „*A CORBA Infrastructure plugged into the German pSISA Architecture*“; (00F-SIW-040)
in: Proc. Simulation Interoperability Workshop SIW Fall 2000, 17.- 22. September 2000, Orlando, FL, U.S.A.
- [5] RTI INTEROPERABILITY STUDY GROUP: *Final Report*; 99F-SIW-001: April 1999.
- [6] S. REILY, H. WILLIAMS: "IDL4HLA: Implementing CORBA IDL Middleware for HLA;" (01S-065)
in: Proc. Simulation Interoperability Workshop SIW Spring 2001, 25.- 30. March 2001, Orlando, FL, U.S.A.
- [7] OBJECT MANAGEMENT GROUP, INC. (OMG): "Facility for Distributed Simulation Systems, Proposed Request for Comments"; mfg/98-06-06, <http://www.omg.org>.
- [8] OBJECT MANAGEMENT GROUP, INC. (OMG): "CORBAServices: Common Object Services Specification"; Updated Edition December 1998, formal/98-12-11, <http://www.omg.org>.
- [9] CorbaCap; HLA Software Distribution Center supported by DMSO; available at http://hla.dms0.mil/sdc/hla_soft.htm
- [10] ACE/TAO: ACE Version OCI-5.0a and TAO Version OCI1.0a; OCI's Distribution of TAO; released Mon Aug 30 1999.
- [11] P. HOARE, G. MAGEE, I. MOODY: „*The development of a prototype HLA RTI (RTI-Lite) using CORBA*“, *in:* Proc. 1997 Summer Computer Simulation Conference - Simulation and Modeling Technology for the Twenty-First Century; San Diego USA 1997

Author Biographies

REINHARD HERZOG studied computer science at the University of Karlsruhe and received his masters degree in 1989. After then he joined the Fraunhofer Institute IITB where he is currently leading the group “simulation and training”. The main topic of the group is the development of distributed simulation-based training platforms.

JOHN MULDER graduated from York University Toronto Canada. He has been at Fraunhofer IITB since 1986 in the field of computer communications and is currently working on tailored CORBA-based integration platforms and client/server information systems.

KAY PIXIUS joined the Simulation Infrastructure Department of WTD 81 in late 1998, after his trainee program with the Federal Office for Defense Technology and Procurement BWB. He graduated from University of Cologne with a masters degree in semiconductor physics and earned his PhD in Material Sciences from RWTH Aachen. He has worked as a research scientist at the Institute of Materials Research of the German Aerospace Research Establishment DLR; where he finally was Assistant to the Board of Directors prior to joining the BWB.

HANS-PETER MENZLER studied physics at the University of Osnabrueck, Germany, and earned his PhD in applied mathematical physics in 1989. He worked for three years as a scientist at the Max-Planck Institute for Plasmaphysics and then became a system engineer at Competence Center Informatik GmbH (CCI). In April 1999 he became head of the Simulation Infrastructure Department at WTD 81, Greding.