# HLA- und Web-basierte Simulation: Ziele, Unterschiede und Synergien

Steffen Straßburger
Institut für Simulation und Graphik
Fakultät für Informatik
Otto-von-Guericke-Universität Magdeburg
Email: strassbu@isg.cs.uni-magdeburg.de
WWW: http://wwwisg.cs.uni-magdeburg.de/isg/strassbu.html

### *Zusammenfassung*

Der Beitrag untersucht aktuelle Trends im Bereich der Simulation. Hierbei geht es zum einen um die High Level Architecture (HLA) als neuen de-facto Standard auf dem Gebiet der verteilten Simulation und zum anderen um den Bereich der Web-basierten Simulation. Die Ziele und Grundideen beider Gebiete werden miteinander verglichen und mögliche Synergieeffekte herausgearbeitet.

## 1 Einleitung

Im Bereich der verteilten Simulation kann die *High Level Architecture for Modeling and Simulation* (*HLA*) als der derzeit gültige State-of-the-Art Standard angesehen werden. HLA wurde durch das Institute of Electrical and Electronics Engineers (IEEE) standardisiert und erhält aus dem militärischen Simulationsbereich kommend inzwischen auch vermehrt in zivilen Simulationsanwendungen Einzug.

Neben HLA ist seit einiger Zeit der Bereich der Web-basierten Simulation (und Animation) ein aktuelles Forschungsthema. Die Web-basierte Simulation versucht, die Vorteile des WWW (z.B. Plattformunabhängigkeit, Ortsunabhängigkeit) auf Simulationsanwendungen zu übertragen und nutzbar zu machen.

Der vorliegende Beitrag untersucht die aktuellen Tendenzen in beiden Bereichen, erläutert die Unterschiede und wie sich mögliche Synergieeffekte erzielen lassen.

## 2 HLA im zivilen Bereich

### *Was ist HLA?*

HLA ist ein moderner Interoperabilitätsstandard für Simulationsanwendungen, der den Anspruch hat, nicht auf eine bestimmte Kategorie von Simulationen (z.B. echtzeitorientiert, ereignisorientiert) beschränkt zu sein. HLA wird definiert durch

- Regeln, die das Verhalten von individuellen Simulationsanwendungen (Federates) und ihrer Kombination (Federation) festlegen,
- Objektmodellen, die zur Beschreibung der Modellierungsfähigkeiten von Federates und Federations genutzt werden, und

- einer Interface Spezifikation, welche die Schnittstelle zwischen Federates und einer Infrastruktursoftware (Runtime Infrastruktur, RTI) festlegt.

Weitere Detailinformationen über HLA sind z.B. in [1] zu finden.

HLA unterscheidet sich von anderen Interoperabilitätsstandards für allgemeine Applikationen (wie z.B. CORBA, DCOM) durch spezielle Dienste, die für Simulationsanwendungen unbedingt erforderlich sind. Das in HLA integrierte Zeitmanagement, welches die Synchronisation von Federates zur Laufzeit erlaubt, sei hier als ein Alleinstellungsmerkmal genannt.

## *Was kann HLA?*

Vorhergehende Untersuchungen haben gezeigt, dass HLA auch im zivilen Simulationsbereich als Interoperabilitätsstandard geeignet ist [1,2,3]. Obwohl es merkliche Unterschiede zwischen den Methoden und Werkzeugen gibt, die in zivilen und militärischen Simulationen zum Einsatz kommen, ist es möglich, HLA in die Simulationssysteme der zivilen Welt zu integrieren. Hierfür lassen sich zwei grundsätzliche Ansätze wählen:

1) Expliziter Ansatz: Im Simulationssystem wird eine HLA-Schnittstelle bereitgestellt, die vom Simulationsmodell aus angesteuert werden muss. Bei der Verteilung eines Simulationsmodells auf mehrere eigenständige Teilmodelle entsteht hierdurch ein gewisser Overhead sowohl bei der Modellerstellung (zeitlicher Mehraufwand) als auch bei der Ausführung der Simulationsexperimente (größere Komplexität und häufig höhere Laufzeit im Vergleich zur stand-alone-Lösung).
2) Impliziter Ansatz: Das Simulationssystem selbst besitzt die Fähigkeit, die notwendigen Aufrufe an die HLA-Schnittstelle zu generieren (z.B. zur Synchronisation mit anderen Simulationen oder zum Austausch von Daten). Der Nutzer muss lediglich spezifizieren, wie das Mapping zwischen lokalen und externen Daten auszusehen hat. Hierdurch wird der Overhead bzgl. der verteilten Modellerstellung minimiert. Ein gewisser Overhead bzgl. der Ausführung der Simulationsexperimente ist auch hier nicht zu vermeiden.

Weiterhin sind hybride Ansätze denkbar, die einen Mittelweg zwischen implizitem und explizitem Ansatz gehen. Hierdurch können diverse Probleme durch Inflexibilitäten des impliziten Ansatzes umgangen werden, ohne auf dessen Komfort verzichten zu müssen [4].

Es läßt sich feststellen, dass HLA für den Einsatz im zivilen Bereich geeignet ist. Verschiedene HLA-Schnittstellen für Simulatoren wurden innerhalb von Forschungsprojekten entwickelt (z.B. SLX, SIMPLEX). Einige Hersteller von Simulatoren haben ebenfalls reagiert. So verfügt MODISM seit einiger Zeit über eine HLA-Schnittstelle. Für ACSL ist eine solche Schnittstelle für die nächste Version (Release 12) angekündigt.

## *Was fehlt?*

Aus Sicht des diskreten Simulationisten gibt es durchaus diverse Mängel in der Interface Spezifikation von HLA. So ist es für diskret-ereignisgesteuerte Simulationen aus Kausalitätsgründen z.B. unerläßlich, den genauen Zeitpunkt eines Dienstaufrufs vorhersagen

bzw. bestimmen zu können. Für die wichtigsten Dienste bietet HLA selbstverständlich eine solche Funktionalität, da HLA sonst für derartige Simulationsanwendungen nicht geeignet wäre. So ist es z.B. bei Datenübertragungen (u.a. Attribut Updates, Interaktionen) immens wichtig, den Zeitpunkt spezifizieren zu können, zu dem die Übertragung von anderen Simulationen empfangen bzw. in die Simulation einbezogen werden soll. Um diese Eigenschaften zu gewährleisten, sind die entsprechenden Dienste(-gruppen) mit den Diensten des HLA-Zeitmanagements abgestimmt. Man spricht von *time-managed services*.

Bei diversen Diensten fehl jedoch eine solche Abstimmung, sie sind somit *non-time-managed* bzw. *unsynchronized* [5]. Die Dienste des Ownership Managements gehören zu dieser Kategorie und sind daher für Simulationen, die strikte Kausalität erfordern, nur bedingt nutzbar.

Weiterhin gibt es Argumente gegen eine Nutzung von HLA, da viele Entwickler von dem umfassenden Diensteangebot von HLA abgeschreckt werden. Es sind Argumente für ein HLA-Lite, d.h. ein reduziertes Diensteangebot und eine reduzierte Funktionalität zu finden. Teilweise wird auch gegen das in HLA integrierte Zeitmanagement argumentiert (da dieses für diverse Echtzeitanwendungen nicht erforderlich ist) und eine auf CORBA basierende Schnittstelle propagiert [6].

Trotz der oben erwähnten Mängel in der Interface Spezifikation wird HLA aus Sicht des Autors den grundlegenden Interoperabilitätsansprüchen der zivilen Welt gerecht und ist der derzeitig beste am Markt verfügbare Standard für verteilte Simulationsanwendungen.

Aus Anwendersicht sind Verbesserungen und Ergänzungen der HLA hauptsächlich bzgl. der Handhabbarkeit verteilter Anwendungen im allgemeinen notwendig. Die grundsätzliche Tatsache, dass die Komplexität verteilter Anwendungen im Vergleich zu stand-alone Anwendungen wesentlich höher ist, trifft natürlich auch auf verteilte Simulationen unter HLA zu. Die Entwicklung, der Test und das Debuggen sind Tätigkeiten, die im Vorfeld der Ausführung notwendig sind und eine gesteigerte Komplexität aufweisen. Auch die Ausführung einer verteilten Simulation erfordert einen höheren Koordinierungsausfwand. Hier wären zentrale Command&Control-Werkzeuge zur Ausführung, Konfiguration und Verwaltung von Federation Executions wünschenswert. Im Idealfall könnten hier Web-basierte Techniken zum Einsatz kommen, die dem Nutzer Applets offerieren, von dem aus Federates gestartet und beendet werden können und der Lauf einer Federation überwacht werden kann. Existierende Lösungen beschränken sich auf stand-alone Applikationen, die das Überwachen einer Federation zur Laufzeit unterstützen (z.B. FedDirector von AEgis Technologies)

Es stellt sich die Frage, ob in Verbindung mit den Techniken der Web-basierten Simulation Synergieeffekte erzielt werden können.

# 3 Web-Basierte Simulation

Keines der derzeitig aktuellen Forschungsgebiete im Simulationssektor ist so weitläufig und divers wie das Gebiet der Web-basierten Simulation. Verschiedene Auffassungen und Begriffsdefinitionen existieren für den Begriff „Web-basierte Simulation“.

Laut Page [7] werden vielfach auch Web-basierte Modell-Datenbanken, Web-gestützte Modellierung, und die Bereitstellung von Texten und Anleitungen zur Simulationsentwicklung zum erweiterten Gebiet der Web-basierten Simulation gezählt.

Unter Web-basierter Simulation im engeren Sinne soll hier

a) die Ausführung von Simulationsmodellen auf einem Web-Server (oder einem über einen Web-Server verfügbar gemachten Anwendungsserver), und

b) die Ausführung von Simulationsmodellen als Applet in einem Web-Browser verstanden werden.

Auf die Betrachtung der Web-basierten Simulation im Sinne der Web-gestützten Simulation und Modellierung durch umfassende Web-basierte Simulationsumgebungen soll hier verzichtet werden. Eine kritische Analyse dieser Gebiete ist in [8] zu finden.

In Variante a) der oben vorgestellten Ansätze wird ein Web-basierter Zugriff zu Simulationsprogrammen, die dann auf einem entfernten Rechner (remote) ausgeführt werden, ermöglicht. Abbildung 1 veranschaulicht das Arbeitsprinzip dieser Lösung.

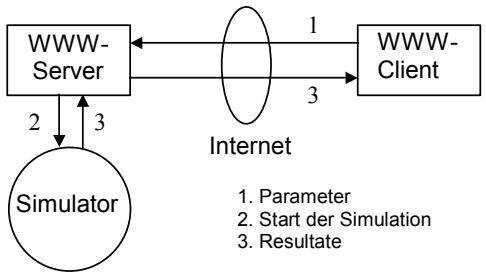
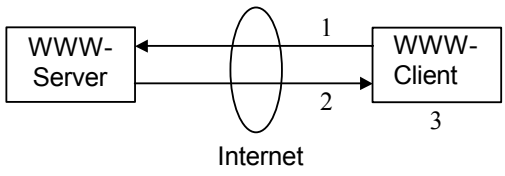



*Abbildung 1: Web-basierte Simulation mit Ausführung der Simulation auf einem (Web-)Server*

Eine auf diesem Ansatz basierende Lösung wurde für den Simulator GPSS/H entwickelt und wird erfolgreich für Forschung und Lehre eingesetzt [9,10].

Variante b) wird auch als „mobile Code“ Lösung [11] charakterisiert und basiert auf Applets, die auf der Seite des Clients ausgeführt werden. Abbildung 2 veranschaulicht diese Lösung.




1. Abruf eines Applets
2. Download des Applets
3. Ausführung des Applets und Auswertung der Resultate

*Abbildung 2: Web-basierte Simulation mit der Ausführung der Simulation innerhalb des Web-Browsers*

Lösungen, die auf dieser Variante basieren, sind seltener zu finden, da hierfür geeignete Simulationsentwicklungsumgebungen häufig noch nicht existieren. Die auf Java basierende Simulationssprache SILK stellt hier eine Ausnahme dar [12]. SILK ist eine Klassenbibliothek mit vordefinierten Simulationskonstrukten für die Programmiersprache Java und erlaubt u.a. die Erstellung von Java-Applets zur Simulation.

Eine weitere auf Ansatz b) basierende Lösung ist das in Java implementierte Animationssystem Skopeo, welches u.a. Post-Run Animation innerhalb von Web-Browsern ermöglicht.

# 4 HLA- und Web-basierte Simulation

## *Gemeinsamkeiten und Unterschiede*

Sowohl Web-basierte Simulationen als auch verteilte Simulationen gemäß HLA stellen im Normalfall verteilte Anwendungen dar. Im Falle der Web-basierten Simulation basiert die „Verteilung“ auf einer klassischen Client-Server-Architektur.

In verteilten Simulationen unter HLA liegt eine Verteilung im Sinne einer Peer-to-Peer Architektur vor: Anwendungen (Federates), die auf verschiedenen Rechnern laufen, kommunizieren direkt miteinander. Zur Koordinierung ist lediglich ein Rechner notwendig, der einen Server-Dienst bereitstellt, über den sich die Federates finden und eine gemeinsame verteilte Simulation aushandeln.

Der Grundgedanke der Web-basierten Simulation besteht darin, die Vorteile des WWW (z.B. Orts- und Plattformunabhängigkeit) auch für die Simulation nutzbar zu machen. Folglich wurden Lösungen entwickelt, die Simulationen und Animationen vom Web-Browser aus zugänglich machen.

Bei der verteilten Simulation unter HLA liegt die Grundidee darin, Interoperabilität zwischen heterogenen Simulationsanwendungen zu schaffen. Die Heterogenität bezieht sich hierbei sowohl auf die genutzte Software, als auch auf die zugrunde liegende Hardwareplattform.

Obwohl derzeitige Implementierungen der HLA-Basissoftware (RTI) als Kommunikationsmedium die Standard-Netzwerkprotokolle des Internet benutzen (UDP, TCP/IP), wäre es falsch, eine verteilte HLA-basierte Simulation als „Web-basierte Simulation“ zu bezeichnen. HLA-basierte Simulationen können jedoch in Kombination mit Web-basierten Techniken durchaus Synergieeffekte erzielen. Im folgenden sollen einige Möglichkeiten in dieser Richtung diskutiert werden.

## *Synergien*

Um Synergien zwischen HLA- und Web-basierten Simulationen und Techniken zu erzielen, sind zwei generelle Fragestellungen zu untersuchen:

1) In welcher Form können HLA-Federates „web-fähig“ im Sinne der oben gegebenen Definition Web-basierter Simulation gemacht werden?

2) Können Web-basierte Applikationen (d.h. nicht Simulationen im eigentlichen Sinne) entwickelt werden, die es ermöglichen, den gesamten Lebenszyklus einer Federation Execution zu steuern bzw. zu unterstützen?

Zur Beantwortung der ersten Fragestellung liegt es nahe, die bekannter Ansätze der Web-basierten Simulation auf ihre Anwendbarkeit für HLA-basierte Simulationen zu untersuchen. In Analogie zu diesen Ansätzen sind folgende Lösungsmöglichkeiten denkbar:

1a) HLA-Federates, die auf dedizierten Web- bzw. Applikationsservern ausgeführt werden und aus dem Web-Browser gestartet werden

1b) HLA-fähige Java-Applets (bzw. Anwendungen innerhalb des Web-Browsers)

Ansatz 1a) erscheint als grundsätzlich durchführbar und wird derzeit anhand des Simulationssystems SLX untersucht (siehe auch Abschnitt 5).

Das Probleme bei Ansatz 1b) bestehen darin, dass Java-Applets diversen Sicherheitsbeschränkungen unterliegen. So ist es für normale Applets nicht möglich, direkt mit Rechnern zu kommunizieren, die verschieden von dem Web-Server sind, von dem sie heruntergeladen wurden. Dies ist im Normalfall jedoch notwendige Voraussetzung für eine über mehrere Rechner verteilte Simulation. Es stellt sich die Frage, ob die HLA Interface Spezifikation überhaupt für Anwendungsszenarien, in denen Applets direkt als Federates agieren, geeignet ist. Eine zu untersuchende Lösungsmöglichkeit stellen sogenannte „Trusted Applets“ dar. Dies sind Applets, denen der Nutzer vertraut und zusätzliche Rechte auf seinem Rechner einräumt.

In existierenden Lösungen für diese Problematik werden Applets mit Relais-Funktionalität eingesetzt. Diese Applets können unter Nutzung eines Relais (z.B. implementiert unter Verwendung von CORBA), welches auf dem Web-Server des Applets residiert, die Sicherheitsbeschränkungen von Java umgehen. In einer existierenden Lösung für das Animationssystem Skopeo wird von einem CORBA-Relais, welches auf dem Web-Server residiert, eine Java-Applikation gestartet, die anstelle des Applets Mitglied in der HLA-Federation wird. Diese Java-Applikation läuft auf dem Web-Server und unterliegt wegen ihres Applikationscharakters nicht den Java-typischen Restriktionen und kann somit vollwertiges Mitglied einer HLA-Federation werden. Der Skopeo-Client fungiert also lediglich als Front-End, welches mit dem eigentlichen Federate auf dem Skopeo-Server kommuniziert und die für eine Animation der HLA-Federation relevanten Daten empfängt.

Bei näherer Betrachtung der zweiten eingangs erwähnten Fragestellung, ob Web-basierte Applikationen entwickelt werden können, die zur Steuerung von Federation Executions genutzt werden können, wird klar, dass hier weitergehende Probleme zu erwarten sind.

Zu einer umfassenden Command&Control Lösung für Federation Executions würden Aufgaben wie das Bereitstellen von Eingangsdaten, die Verteilung auf der Daten auf die Rechner der beteiligten Federates, das Starten der Federates, die Steuerung und Überwachung zur Laufzeit und das Herunterfahren der Federation Execution gehören. Neben den bekannten Sicherheitslimitationen von Java-Applets sind weitere Einschränkungen der Betriebssysteme selbst zu überwinden. Hierzu gehören z.B. das Starten und Beenden von Anwendungen auf entfernten Rechnern und deren Fernsteuerung.

# 5 Prototypen

Im Rahmen des von der EU geförderten Projektes DAMAC-HP wurde eine Web-basierte Simulationslösung für den Rigaer Hafen (das Riga Baltic Container Terminal, BCT) geschaffen [13]. Die implementierte Lösung stellt eine „traditionelle“ Web-basierte Simulation dar. Auf der Client-Seite steht ein ActiveX-Control bereit, welches innerhalb des Web-Browsers[1] die Generierung von Input-Parametern für ein Simulationsmodell des BCT erlaubt (Abbildung 3).

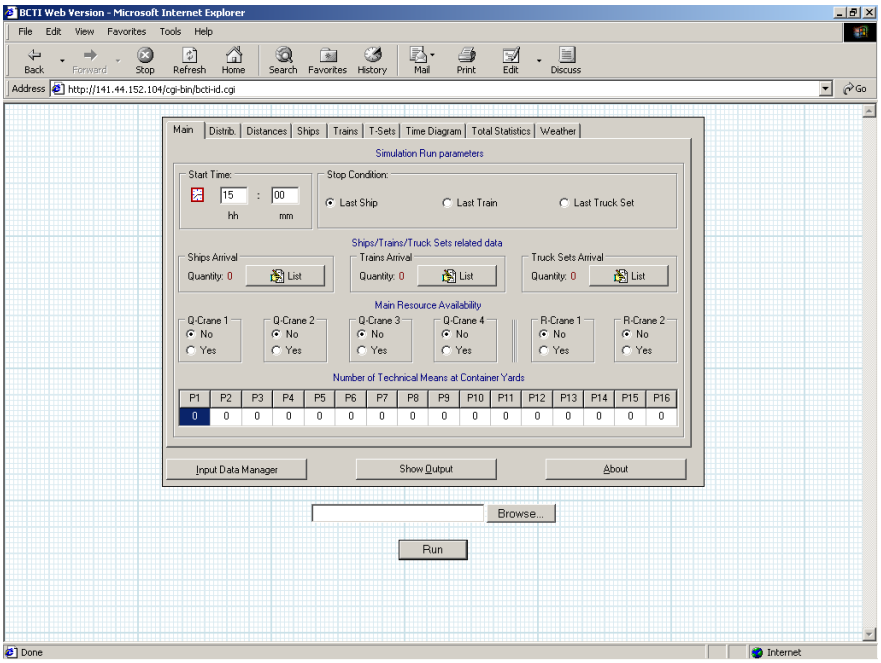



*Abbildung 3: Screenshot des ActiveX-Controls zur Eingabedatengenerierung*

Auf der Server-Seite dient ein Perl-Skript zum Empfang der Input-Daten für das Simulationsmodell und zum Start des Simulationssystems. Das in diesem Fall genutzt Simulationssystem ist SLX [14], welches die nötigen Anforderungen für eine Web-basierte Lösung (u.a. eine Kommandozeilen-Option zur automatischen Ausführung eines Modells) erfüllt. Die Resultatdaten werden in Form mehrerer ASCII-Dateien bereitgestellt und zum Client zurücktransferiert. Dieser bereitet die Daten innerhalb des Web-Browsers grafisch auf.

In einer zukünftigen Entwicklung ist vorgesehen, dass SLX-Modell unter Nutzung der SLX-HLA-Schnittstelle [15] um HLA-Fähigkeiten zu erweitern und auf dem Web-Server

---

[1] Man beachte, dass die Wahl einer ActiveX-basierten Lösung den Internet Explorer von Microsoft als Web-Browser erfordert. Obwohl diese Wahl bezüglich des Ziels der Plattformunabhängigkeit kontraproduktiv ist, sprachen pragmatische Gründe (hier: das Vorhandensein einer Delphi-basierten stand-alone-Lösung, die leicht zu einem ActiveX-Control ausgebaut werden konnte) für diesen Lösungsweg.

als HLA-Federate auszuführen. Die grundsätzliche Durchführbarkeit dieses Ansatzes wurde bereits getestet. Als Anwendungsszenario sind die Kooperation mit anderen Simulationen (z.B. Kopplung mit Federates, die aktuelle Wetterdaten bereitstellen, Online-Visualisierung beim Client durch ein eigenständiges HLA-Federates, Kopplung mit Leitstandssystemen der den Hafen ansteuernden Schiffe etc.) denkbar.

# 6 Zusammenfassung

HLA kann als der derzeitig gültige Interoperabilitätsstandard im Bereich der verteilten Simulation angesehen werden. HLA ist, obwohl aus dem militärischen Bereich kommend, auch in der zivilen Simualtionswelt anwendbar und wird von einer wachsenden Anzahl von Anwendern akzeptiert. Lösungen zur Integration von HLA-Schnittstellen in verschiedene Simulationssysteme existieren und sind anwendbar. Die Frage, ob es einen wirklichen Bedarf für HLA von Seiten großer Industrieprojekte geben wird, kann noch nicht beantwortet werden. Die Mittel und Methoden stehen hierfür jedoch zur Verfügung.

Web-basierte Lösungen zur Simulation bestechen durch die Übertragung der Vorteile des WWW (Plattform- und Ortsunabhängigkeit) auf den Bereich der Simulation. Es stellt sich die Frage nach den Kombinationsmöglichkeiten beider Technologien. Synergieeffekte zwischen Web-basierten Simulationstechniken und HLA sind nach Meinung des Autors möglich und wünschenswert.

Als besonderer Mangel im Umfeld von HLA kann z.Zt. die fehlende Unterstützung bei der Ausführung und beim Management von HLA-Federations zur Laufzeit betrachtet werden. Es gibt z.B. keine Unterstützung für das kontrollierte Hoch- und Herunterfahren einer Federation Execution mit den dazugehörigen Federates. Hier erscheint es besonders vorteilhaft, web-basierte Techniken einzusetzen, die es ermöglichen würden, Federates von einem beliebigen Ort aus zu starten und auch wieder zu beenden. Die Kombination von Web-basierten Techniken, HLA und CORBA erscheint hier als ein vielversprechender Ansatz.

# Referenzen


[1] Straßburger, S. *Distributed Simulation Based on the High Level Architecture in Civilian Application Domains*. Dissertationsschrift. Otto-von-Guericke-Universität Magdeburg. Eingereicht am 31. Oktober 2000.

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

[3] Straßburger, S. *On the HLA-based Coupling of Simulation Tools*. In: Proceedings of the 1999 European Simulation Multiconference, ed. H. Szczerbicka, pp. 45-51 (Vol. 1). SCS, Warsaw, Poland.

[4] Straßburger, S., T. Schulze, G. Lantzsch. *Simplex 3 und SLX - gemeinsam unter HLA*. In: Proceedings 13. Symposium Simulationstechnik ASIM 99, 21.09.-24.09.1999, Weimar, (Ed.) G. Hohman, SCS International, pp. 331-336.

[5] Fujimoto, R., I. Tacic. *Time Management of Unsynchronized HLA Services*. 1999 Fall Simulation Interoperability Workshop, September 12-17, 1999, Orlando. Paper Number 99F-SIW-165.

[6] Herzog, R., T. Usländer, K. Pixius, H.-P. Menzler. *A CORBA Infrastructure plugged into the German pSISA Architecture*. 2000 Fall Simulation Interoperability Workshop, September 17-22, 2000, Orlando. Paper Number 00F-SIW-040.

[7] Page, E. *The Rise of Web-based Simulation: Implications for the High Level Architecture*. In: Proceedings of the 1998 Winter Simulation Conference, eds. Medeiros, D.J. and E. Watson, pp. 1663-1668. SCS, Washington, D.C.

[8] Kuljis, J., R.J. Paul. *A Review of Web Based Simulation: Whither We Wander?* In: Proceedings of the 2000 Winter Simulation Conference, eds. J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, pp.1872-1881.

[9] Dorwarth, H., P. Lorenz, K.-C. Ritter and Th. J. Schriber. *Towards a Web Based Simulation Environment*. 1997 Winter Simulation Conference Proceedings pp.1338-1344

[10] Lorenz, P., T. Schriber. *Teaching Introductory Simulation in 1996: From the First Assignment to the Final Presentation*. 1996 Winter Simulation Conference Proceedings pp. 1379-1386.

[11] Page, E. *Beyond Speedup: PADS, the HLA and Web-Based Simulation.* In: Proceedings of the 13th Workshop on Parallel and Distributed Simulation (PADS'99), eds. R. Fujimoto, S. Turner, pp. 2-9.

[12] Healy, K., R. Kilgore. *Introduction to Silk and Java-Based Simulation*. In: Proceedings of the 1998 Winter Simulation Conference, eds. Medeiros, D.J. and E. Watson, pp. 327-334. SCS, Washington, D.C.

[13] Straßburger, S., T. Schulze, J. Tolujev. *SLX-Model and Proof Animation Based Visualization of the Riga Baltic Container Terminal*. In: Proceedings of the Second International Conference "Simulation, Gaming, Training and Business Process Reengineering in Operations", pp. 332-335. September 8-9, 2000, Riga, Latvia.

[14] Henriksen, J.O., *An Introduction to SLX*. In: Proceedings of the 1995 Winter Simulation Conference, ed. C. Alexopolous, pp. 502-507.

[15] Straßburger, S., U. Klein. *Integration des Simulators SLX in die High Level Architecture*. In: Proceedings of the Conference "Simulation und Visualisierung 1998", eds. P. Lorenz and B. Preim, SCS Europe Publishing House, Delft, Erlangen, Ghent, San Diego, pp. 32-40. Magdeburg 1998.