REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	25 Mar 1998	Final Report

4. TITLE AND SUBTITLE
Advanced Distributed Simulation Technology II (ADST II)
High Level Architecture Support Experiments Final Report

5. FUNDING NUMBERS
N61339-96-D-0002

6. AUTHOR(S)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Lockheed Martin Corporation
Lockheed Martin Information Systems
ADST II
P.O. Box 780217
Orlando, FL 32878-0217

8. PERFORMING ORGANIZATION REPORT NUMBER
ADST-II-CDRL-011R-9600166

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
NAWCTSD/STRICOM
12350 Research Parkway
Orlando, FL 32826-3324

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

19990104 046

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT
A - Approved for public release; Distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 Words)***

The purpose of the High Level Architecture (HLA) Support Experiments (HSE) project was to perform experimentation and research in HLA technology to support the evolution and further implementation of the HLA specifications. This project was performed as Delivery Order (DO) #0041 under the Lockheed Martin Advanced Distributed Simulation, Training and Instrumentation Command (STRICOM). The project work was sponsored by the Defense Modeling and Simulation Office (DMSO).

14. SUBJECT TERMS
ADST-II, STRICOM, MODSAF

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

DTIC QUALITY INSPECTED 3

ADST-II-CDRL-HLA-9800039
March 25, 1998

ADVANCED DISTRIBUTED SIMULATION TECHNOLOGY II (ADST II)

HIGH LEVEL ARCHITECTURE SUPPORT EXPERIMENTS

(DO #0041)

CDRL AB01

FINAL REPORT

FOR: NAWCTSD/STRICOM
12350 Research Parkway
Orlando, FL 32826-3224
N61339-96-D-0002
DI-MISC-80711

BY: Lockheed Martin Corporation
Lockheed Martin Information Systems
ADST II
P.O. Box 780217
Orlando, FL 32878-0217

Approved for public release; distribution is unlimited

Document Control Information

Revision	Revision History	Date
	Initial Release	03/25/98

Table of Context

EXECUTIVE SUMMARY x
1. INTRODUCTION 1
1.1 PURPOSE 1
1.2 CONTRACT OVERVIEW 1
1.3 TASKING OVERVIEW 1
1.4 TECHNICAL OVERVIEW 2
2. APPLICABLE DOCUMENTS 3
2.1 GOVERNMENT 3
2.2 NON-GOVERNMENT 3
3. BRIDGE FEDERATE 6
3.1 HIGH LEVEL ARCHITECTURE SUPPORT EXPERIMENTS PHASE I REPORT OCTOBER 1997 6
3.1.1 Introduction 6
3.1.2 Bridge Federate Description 6
3.1.3 Bridge Federate Development Approach 7
3.1.3.1 Specification Analysis 7
3.1.3.2 Documentation 7
3.1.3.3 Prototype Implementation 7
3.1.3.4 Application Level Experiment 8
3.1.3.5 Phased Development 8
3.1.4 Phase I Capabilities 8
3.1.4.1 Specification Analysis 8
3.1.4.2 Documentation 8
3.1.4.3 Prototype Development 9
3.1.4.3.1 Phase I – HLA Services 9
3.1.4.3.2 Phase I – Bridge Federate Services 12
3.1.4.4 Application Level Experiment 13
3.1.5 Phase I Challenges and Lessons Learned 13
3.1.5.1 Software Development Approach 13
3.1.5.2 General Lessons Learned 13

3.1.5.3 Initial Capability Demonstration 14
3.1.5.4 Application Demonstration 14
3.1.5.4.1 Purpose 15
3.1.5.4.2 Capabilities 15
3.1.5.4.3 Federation Required Execution Details (FRED) 15
3.1.6 Conclusions 16
3.2 HIGH LEVEL ARCHITECTURE SUPPORT EXPERIMENTS PHASE II REPORT JANUARY 1998 17
3.2.1 Introduction 17
3.2.2 Phase II Capabilities 17
3.2.2.1 Documentation 17
3.2.2.2 Prototype Development 17
3.2.2.2.1 Phase II – HLA Services 17
3.2.2.2.2 Phase II – Bridge Federate Services 17
3.2.2.3 Application Level Experiment 18
3.2.3 Phase II Challenges and Lessons Learned 18
3.2.3.1 Software Development Approach 18
3.2.3.2 General Lessons Learned 18
3.2.3.3 Application Demonstration 20
3.2.3.3.1 Purpose 24
3.2.3.3.2 Capabilities 24
3.2.3.3.3 Developers Notebook ..... **Error! Bookmark not defined.**
3.2.4 Conclusions 24
4. MULTI FEDERATION EXECUTION EXPERIMENT 25
4.1 MULTIPLE FEDERATION EXECUTION EXPERIMENTATION 25
4.1.1 Level 1 – Multiple Fedex Support 25
4.1.1.1 Approach 25
4.1.1.2 Experiment Scenario 26
4.1.1.3 Experiment Results 26
4.1.2 Level 2: Performance Impact 26
4.1.2.1 Timing Experiment 27
4.1.2.2 Timing Experiment Results 27

4.1.3 SEOD Implementation **29**
4.1.3.1 Approach 29
4.1.3.2 Results 29
5. HLA TEST VICTIM **32**
6. FOM FLEXIBILITY ANALYSIS **32**
6.1 FOM FLEXIBILITY REPORT **32**
6.1.1 Introduction **32**
6.1.2 Background **33**
6.1.3 Distributed Simulation Software Components **33**
6.1.3.1 Model / Simulation (Internal) 34
6.1.3.2 Middleware 34
6.1.3.3 Run Time Infrastructure 35
6.1.4 FOM Flexibility in Distributed Simulation **35**
6.1.5 HLA Interface Development Approaches **37**
6.1.6 FOM Flexibility Approaches **37**
6.1.6.1 Specific Case Studies 38
6.1.6.1.1 CASE 1: CCTT Semi-Automated Forces (SAF) 38
6.1.6.1.2 CASE 2: ModSAF Federation Common Software 39
6.1.6.1.3 CASE 3: Air Combat Environment Test and Evaluation Facility (ACETEF) 40
6.1.6.1.4 CASE 4: Joint Modeling and Simulation System (JMASS) 42
6.1.6.1.5 CASE 5: Eagle 43
6.1.6.1.6 CASE 6: IST Gateway 45
6.1.6.1.7 CASE 7: Simulation Middleware Object Classes (SMOC) 47
6.1.7 Lessons Learned – Developer Guidance **48**
6.1.8 Remaining Issues **48**
6.1.9 Additional Information Resources **49**
7. OBJECT MODEL DATA DICTIONARY SYSTEM (OMDDS) EXPERIMENT **50**
7.1 OBJECT MODEL DATA DICTIONARY SYSTEM (OMDDS) EXPERIMENT REPORT **50**
7.1.1 OMDD Experiment Plan **51**
7.1.1.1 Introduction 51
7.1.1.2 Purpose and Goals 51

7.1.1.3 Experiment Approach....51
7.1.1.3.1 General Approach....51
7.1.1.3.2 Technical Approach....52
7.1.1.4 Products....52
7.1.1.5 Experiment Participants....52
7.1.1.6 Experiment Conduct – Assumptions....53
7.1.1.7 Approximate Schedule....53
7.1.2 OMDD Experiment Execution....53
7.1.2.1 Introduction....53
7.1.2.2 FOM Development and The Federation Development and Execution Process 53
7.1.2.2.1 FEDEP Process....53
7.1.2.2.2 Description of the Federation Object Model Development Process....54
7.1.2.2.2.1 Summary of FOM Development Approaches....54
7.1.2.2.2.2 FOM Development Process....**Error! Bookmark not defined.**
7.1.2.3 Object Model Tool Suite: Description of the Tools and Their Use....2
7.1.2.4 Object Model Development Experiment....4
7.1.2.4.1 FEDEP Preliminaries....4
7.1.2.4.1.1 Requirements Definition....4
7.1.2.4.1.2 Conceptual Model Development....5
7.1.2.4.1.3 Federation Design....8
7.1.2.4.2 FOM Prototype Development....8
7.1.2.4.2.1 Bottoms-Up Approach....9
7.1.2.4.2.2 Merge / Single SOM Approach....12
7.1.2.4.2.3 Existing FOM / Reference FOM Approach....13
7.1.3 OMDD Experiment Results....13
7.1.3.1 Experiment Summary....13
7.1.3.1.1 Process Summary....13
7.1.3.1.2 Object Model Tool Suite Implementation....14
7.1.3.2 Guidance for Federation Developers....15
7.1.3.3 Conclusions....16
7.1.3.4 Problems and Recommended Enhancements....16
7.1.3.4.1 OMDT....17

7.1.3.4.1.1 Problems 17
7.1.3.4.1.2 Recommended Enhancements 17
7.1.3.4.2 OMDDS 18
7.1.3.4.2.1 Problems 18
7.1.3.4.2.2 Recommended Enhancements 19
7.1.3.4.3 OML 20
7.1.3.4.3.1 Problems 20
7.1.3.4.3.2 Recommended Enhancements 20
7.1.4 Federation Developers Workbook 20
Appendix A – REQUIREMENTS ANALYSIS USE CASE A-1
Appendix B – BRIDGE FEDERATE SYSTEM REQUIREMENTS B-1
Appendix C – DESIGN REVIEW PRESENTATION C-1
Appendix D – PHASE I FEDERATION WORKBOOK D-1
Appendix E – PHASE II FEDERATION WORKBOOK E-1
Appendix F – OMDDS WORKBOOK F-1
Appendix G – SIMULATION INTEROPERABILITY WORKSHOP (SIW) PAPERS G-1
Appendix H – FOM FLEXIBILITY QUESTIONNAIRES H-1
Appendix I – ACRONYMS I-1

Figures

FIGURE 1 BRIDGE FEDERATE COMPONENTS....6

FIGURE 2 ITERATIVE DEVELOPMENT PROCESS....8

FIGURE 3 APPLICATION DEMO HARDWARE CONFIGURATION....14

FIGURE 4 FEDERATION EXECUTION CONFIGURATION....15

FIGURE 5 BRIDGE FEDERATE FUNCTIONAL DIAGRAM....19

FIGURE 6 APPLICATION DEMO HARDWARE CONFIGURATION....21

FIGURE 7 APPLICATION DEMO FUNCTIONAL DIAGRAM....21

FIGURE 8 APPLICATION DEMO USE CASE....23

FIGURE 9 MULTIPLE FEDERATION EXECUTION EXPERIMENT CONFIGURATION....25

FIGURE 10 LEVEL 1 EXPERIMENT CONFIGURATION....26

FIGURE 11 SINGLE JOINT FEDERATION EXECUTION....30

FIGURE 12 TWO DISTINCT FEDERATIONS....31

FIGURE 13 DISTRIBUTED SIMULATION SOFTWARE COMPONENTS....34

FIGURE 14 CCTT SAF RTI INTEGRATED ARCHITECTURE....39

FIGURE 15 FEDERATION COMMON SOFTWARE....40

FIGURE 16 ACETEF HLA SYSTEM ARCHITECTURE....41

FIGURE 17 ENTITY FILE LAYOUT....41

FIGURE 18 JMASS HLA ARCHITECTURE....43

FIGURE 19 EAGLE ARCHITECTURE....44

FIGURE 20 EAGLE HLA INTERFACE ARCHITECTURE....45

FIGURE 21 IST GATEWAY RTI INTERFACE DATA FLOW....46

FIGURE 22 SMOC ARCHITECTURE 47

FIGURE 23 FIVE-STEP PROCESS 54

FIGURE 24 FOM DEVELOPMENT PROCESS 1

FIGURE 25 OBJECT MODEL TOOLS SUITE 2

Tables

TABLE 1 PHASE I HLA SERVICES 10

TABLE 2 PHASE I BRIDGE FEDERATE SERVICES 12

TABLE 3 PHASE II HLA SERVICES 17

TABLE 4 PHASE II BRIDGE FEDERATE SERVICES 17

TABLE 5 TIMING EXPERIMENT STEPS 27

TABLE 6 TIMING EXPERIMENT RESULTS 29

TABLE 8 FEDERATE SUMMARY 36

TABLE 9 EQUIPMENT & FACILITIES 4

TABLE 10 PLANNED CLASSES & INTERACTIONS 6

TABLE 12 INTERACTION MAPPING BETWEEN FEDERATES AND CONCEPTUAL MODEL 9

TABLE 13 SUMMARY OF OBJECT MODEL TOOL SUITE USE 14

TABLE 14 FOM DEVELOPMENT APPROACH GUIDANCE AND TOOLS 15

EXECUTIVE SUMMARY

The purpose of the High Level Architecture (HLA) Support Experiments (HSE) project was to perform experimentation and research in HLA technology to support the evolution and further implementation of the HLA specifications. This project was performed as Delivery Order (DO) #0041 under the Lockheed Martin Advanced Distributed Simulation Technology II (ADST II) Contract administered by the U.S. Army Simulation, Training and Instrumentation Command (STRICOM). The project work was sponsored by the Defense Modeling and Simulation Office (DMSO)

Research for this project focused on several task areas:

1. Bridge Federate: The HSE project defined the concept and developed the specification for a Bridge Federate. To verify these requirements, a prototype was developed and several application experiments conducted using the Bridge Federate.
2. Multiple Federation Execution Experiments: A series of experiments were conducted to explore the Run-Time Infrastructure's (RTI) ability to support multiple federation executions at the same time. Some non-rigorous timing experiments were executed to determine the impact of using two versus one federation execution.
3. HLA Test Victim: The project's CCTT implementation of HLA was used as a "test victim" to support GTRI's HLA test tool development.
4. FOM Flexibility Analysis: The project team examined the interfaces of the seven HLA implementations to explore how each interface achieved FOM flexibility. This information was used to define a generic description of FOM flexible software components.
5. Object Model Data Dictionary (OMDD) Experiment: The purpose of the OMDD experiment was to exercise the OMDD and associated tools, including a new version of the Object Model Development Tool (OMDT) within the federation development process. It also served to explore numerous ways to develop FOMs.

Analysis and software development were conducted at Science Applications International Corporation's (SAIC) facility in Orlando, Florida from 22 May 1997 to 20 March 1998.

In accordance with the Government SOW, this Final Report includes a description of the tasks / experiments, their conditions and conduct, and lessons learned as applicable.

INTRODUCTION

1.1 PURPOSE

The purpose of this final report is to document the ADST II effort that supported HSE. This report includes a full description of the tasks and experiments conducted, their conditions, and lessons learned.

1.2 CONTRACT OVERVIEW

HSE was performed as DO#0041 under under the Lockheed Martin Corporation (LMC) ADST II contract with STRICOM. The contract requirements included exploration of federation management issues and bridge federate definition and prototyping. Other tasking was added later in the contract period to include FOM flexibility analysis, HLA testing support and the OMDD experiment.

1.3 TASKING OVERVIEW

A number of tasks were performed to support the objectives of this project. An overview of each task is included below:

Bridge Federate: The HSE project defined the concept and developed the specification for a Bridge Federate. The Bridge Federate serves to "bridge" multiple federation executions together into a single "virtual" federation. Each federation may have a different Federation Object Model (FOM) for its definition. The Bridge Federate requirements were defined for the following HLA services: Federation Management, Declaration Management, Object Management, Ownership Management and Time Management. Future work will include the definition of the Bridge Federate requirements for Data Distribution Management. To verify these requirements, a prototype was developed and several application experiments conducted using the Bridge Federate.

Multiple Federation Execution Experiments: A series of experiments were conducted to explore the Run-Time Infrastructure's (RTI) ability to support multiple federation executions at the same time. This capability was explored to see if the RTI could support multiple federation executions within the Close Combat Tactical Trainer (CCTT) environment. It was envisioned that the CCTT Semi-Automated Forces (SAF) SAF Entity Object Database (SEOD) data could utilize a separate federation execution. Some non-rigorous timing experiments were executed to determine the impact of using two verses one federation execution.

HLA Test Victim: The project's CCTT implementation of HLA was used as a "test victim" to support Georgia Tech Research Institute's (GTRI) HLA test tool development. This task involved following developed procedures, working with the GTRI team, and providing feedback to GTRI and the Simulation Interoperability Workshop (SIW) Testing Forum.

FOM Flexibility Analysis: Beginning with an SIW paper describing five cases studies in FOM Flexibility, our project team examined the interfaces of the five cases and two additional HLA implementations to explore how each interface achieved FOM flexibility. This information was

used to define a generic description of FOM flexible software components. FOM flexibility in this context referred to the ease with which an application was able to "adjust" to the use of a new FOM. After an initial draft report on this activity, the definition of FOM flexibility was refined and an analysis initiated to explore how FOM flexibility could be implemented within CCTT.

Object Model Data Dictionary (OMDD) Experiment: The purpose of the OMDD experiment was to exercise the OMDD and associated tools, including a new version of the Object Model Development Tool (OMDT) within the federation development process. It also served to explore numerous ways to develop FOMs. The work of this experiment was documented in a paper presented at the Spring 1998 SIW.

1.4 TECHNICAL OVERVIEW

The technical approach to HSE tasking was based on the individual task to be performed. When a task or experiment was identified by the customer, the project team defined the technical approach for the particular task. The approach for each of the tasks is described later in this report with the information on the respective tasks.

2.

APPLICABLE DOCUMENTS

2.1 GOVERNMENT

Department of Defense, "High Level Architecture Interface Specification, Version 1.0", DMSO, August 15, 1996.

Department of Defense, "High Level Architecture Interface Specification, Version 1.2", February 1997.

Department of Defense, "High Level Architecture Interface Specification, Version 1.2 (DRAFT 6)", 1 August 1997.

Department of Defense, "High Level Architecture, Federation Development and Execution Process (FEDEP) Model, Version 1.1", 21 November 1997.

Defense Modeling and Simulation Office, "HLA Object Model Template, Version 1.2," 13 August 1997.

Defense Modeling and Simulation Office, "HLA Federation Development and Execution Process (FEDEP) Model, Version 1.1 (DRAFT)", 22 November 1997.

Defense Modeling and Simulation Office, "HLA Compliance Checklist, Version 1.1", 26 March 1997.

Defense Modeling and Simulation Office, "Engineering Proto-Federation Evaluation of the High Level Architecture", January 1997.

Defense Modeling and Simulation Office, "HLA Glossary".

Defense Modeling and Simulation Office, "Management Object Model", 17 October 1996.

2.2 NON-GOVERNMENT

Rothenberg, Jeff: "The Nature of Modeling", Rand Corporation, Santa Monica, California.

Braudaway, Wesley; Little, Reed: "The High Level Architecture Bridge Federate", Simulation Interoperability Workshop, Spring 1997.

Beebe, B., Bouwens, C., Braudaway, W., Harkrider, S., Ogren, J., Paterson, D., Richardson, R., and Zimmerman, P., "Building HLA Interfaces for FOM Flexibility: Five Case Studies," *Proceedings of the 1997 Fall Simulation Interoperability Workshop* (97F-SIW-172), Orlando FL, September 1997.

Braudaway, W. and Harkrider, S., "Implementation of the High Level Architecture into DIS-Based Legacy Systems," *Proceedings of the 1997 Spring Simulation Interoperability Workshop*, Orlando FL, March 1997.

Fullford, Deb; Hoxie, Sue; and Lubetsky, Ben: "Transitioning Your DIS Simulator to HLA"

Lewis, Charles E.: "Evaluation Final Report, Engineering Proto-Federation Evaluation of the High Level Architecture Developed By the Defense Modeling and Simulation Office (DMSO)", January, 1997

Briggs, R. and Miller, G.: "The JPSD Experiment Federation Common Software", *Summary Report – 15th Workshop on the Interoperability of Distributed Interactive Simulation (96-15-095)*, pp. 625 – 629, Orlando FL, September 1996.

Ogren, J.: "The Analytical Community and the High Level Architecture: A Happy Marriage", *Proceedings of the 1997 Spring Simulation Interoperability Workshop (97S-SIW-017),* pp. 129 – 138, Orlando FL, March 1997.

Ogren, J.: "Eagle Time Management," *Proceedings of the 1997 Fall Simulation Interoperability Workshop (97F-SIW-073),* pp. 469 – 476, Orlando FL, September 1997.

Wood, D., Petty, M., Cox, A., Hofer, R., and Harkrider, S.: "HLA Gateway Status and Future Plans," *Proceedings of the 1997 Spring Simulation Interoperability Workshop,* pp. 807 – 814, Orlando FL, March 1997.

Wood, D.: "An Object Oriented RTI Interface," *Proceedings of the 1997 Fall Simulation Interoperability Workshop,* pp. 477 – 486, Orlando FL, September 1997.

Paterson, D., Anschuetz, E., Biddle, M., Kotick, D., and Nguyen, T. "Architecture Issues for DIS-TO-HLA Conversion," NAWC-TSD Technical Report 97-014, Orlando FL, December 1997.

Petty, M. D. and Cox, A. (1997). "HLA Interoperability for Existing Simulations", Proceedings of the 1997 Summer Computer Simulation Conference, Arlington VA, July 13-17 1997, pp. 437-442.

Lutz, R., "HLA Object Model Development: A Process View," 1997 Spring Simulation Interoperability Workshop, March 1997.

Lutz, R.; Hooks, M.; and Hunt, K., "Automation in the HLA FOM Development Process," 15th DIS Workshop, 16-20 September 1996.

Scrudder, R. and Lutz R., "High Level Architecture (HLA) Object Model Tool Development," Simulation Interoperability Workshop, 8-12 September 1997.

Dahmann, J.; Olszewski, J.; Briggs, R.; Richardson, R.; Weatherly, R.; Calvin, J.; and Zimmerman, P., "High Level Architecture (HLA) Performance Framework"

3.

BRIDGE FEDERATE

3.1 HIGH LEVEL ARCHITECTURE SUPPORT EXPERIMENTS PHASE I REPORT OCTOBER 1997

3.1.1 Introduction

The High Level Architecture (HLA) Support Experiments (HSE) project has been performing a number of tasks to support the evolution and application of the HLA. One task that the project has been focusing on has been the concept development and prototyping of a Bridge Federate. This report provides a summary of the progress made during the first phase of development for the Bridge Federate.

3.1.2 Bridge Federate Description

A Bridge Federate is a federate that participates in multiple federations. It has several internal components: surrogates (one for each federation it participates in) and a Transformation Manager that coordinates the surrogates. The Bridge Federate serves to combine multiple federation executions to provide the means for their interoperation.

The combined federation execution as shown in **Figure 1** shows two federation executions interoperating through the support of a Bridge Federate. The communication of this combined federation execution is achieved without requiring any one federate to understand each and every Federation Object Model (FOM). Each federation execution (fedex) can communicate with other fedexs using its own FOM and the Surrogate Federate provided by the Bridge Federate. Each federate has no knowledge that it is, in fact, interoperating within multiple federation executions. The specification for how objects, attributes, and interaction are mapped throughout the combined federation execution is defined by the FOM Mappings/Transformations specification (FOMAT in **Figure 1**).

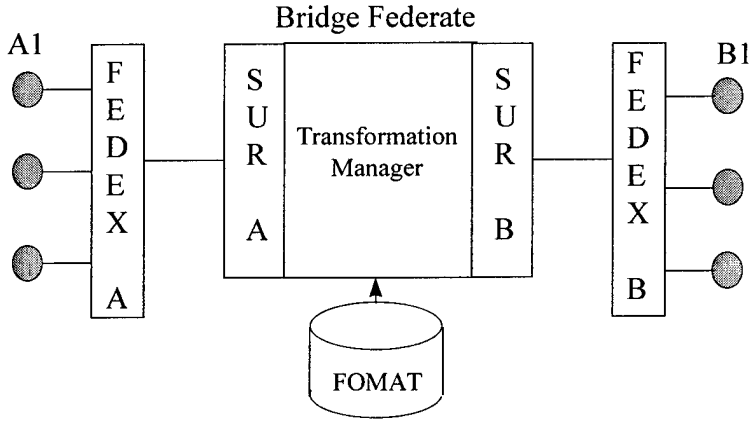


Figure 1 Bridge Federate Components

The Bridge Federate is composed of multiple surrogate federates which behave as federates within their respective federation executions. The figure illustrates two federations labeled FEDEX A and FEDEX B and a surrogate for each federation execution labeled SUR A and SUR

B, respectively. Each surrogate participates within its joined FEDEX as a federate on behalf of the other federation executions. The Transformation Manager provides a message conduit and translator between these multiple surrogate federates; thus linking the surrogates' federations. Its transformation functionality is responsible for implementing all HLA services across the bridge and for transforming object, attributes, and interaction communication between the federation executions as specified by the FOMAT.

A Bridge Federate can be as simple or complex as needed to support the needs of linking multiple federation executions. It is envisioned that the Surrogate components will provide the common functionality that will be needed by any Bridge Federate implementation. Also it is envisioned that the Transformation Manager will be tailorable to support the actual requirements of the Bridge Federate implementation. However, any Transformation Manager will provide, at a minimum, the common functionality that can be expanded if needed to support additional requirements (e.g., security)

3.1.3 Bridge Federate Development Approach

The Bridge Federate concept development and prototyping effort features an iterative approach for defining system requirements, prototype implementation and application experimentation (**Figure 2**).

3.1.3.1 Specification Analysis

An initial analysis is performed to examine use cases for the bridge for a particular configuration / application of the bridge. The use cases are generated to determine the nature of the interface between components of the bridge. This has been documented in stylized English.

3.1.3.2 Documentation

The use cases resulting from the specification analysis are next used to generate the requirements for the prototype development. The concept of execution for key event sequences is described during this stage using event traces. A requirements specification is generated.

3.1.3.3 Prototype Implementation

The purpose of the prototype is to validate the specified functionality described in the requirements specification. It also provides a single example of an implementation of the Bridge Federate. Later iterations provide additional functionality to the prototype.

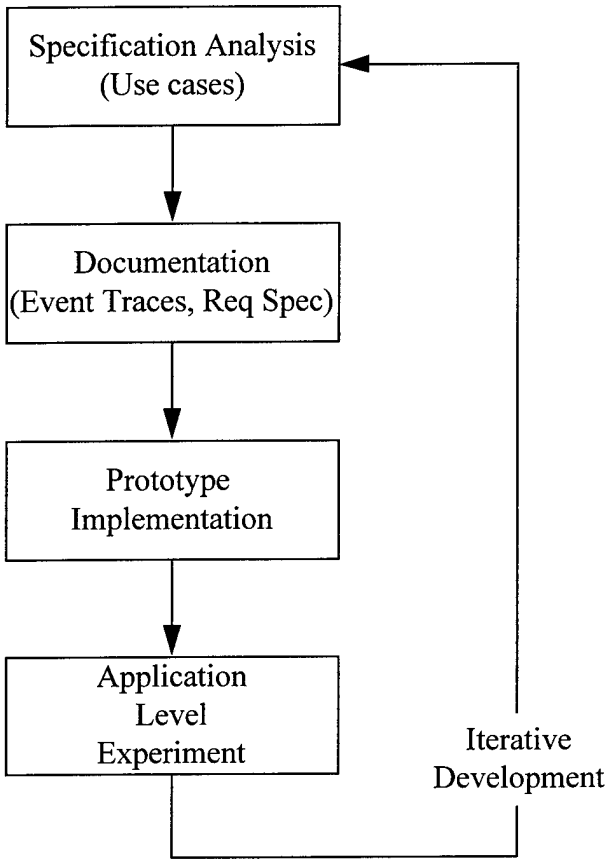


Figure 2 Iterative Development Process

3.1.3.4 Application Level Experiment

An application level experiment is developed to provide an operational context for the prototype bridge. This experiment focuses the use of the bridge for a particular application. The purpose of the experiment is to exercise the bridge's functionality for a particular situation and to further verify the defined required capabilities.

3.1.3.5 Phased Development

The four steps described above are carried out in an iterative manner in order to examine the feasibility of the various functions of the bridge in manageable parts.

3.1.4 Phase I Capabilities

3.1.4.1 Specification Analysis

Specification Analysis for Phase I includes the initial Use Cases defined in conjunction with Carnegie Mellon University. The results of this analysis is found in the Requirements Analysis Use Cases (Appendix A).

3.1.4.2 Documentation

Documentation for Phase I currently includes a Requirements Specification which records the functional requirements for the Bridge Federate. This document is the primary product of this research effort and has been slightly modified to reflect the results of our Phase I development

efforts. The final revised document is included as Appendix B. A design review presentation (Appendix C) records the software design developed to address the requirements in the requirements document. A summary of the requirements and design has been included in a paper developed for the 1997 Fall Simulation Interoperability Workshop entitled "The High Level Architecture's Bridge Federate" (Appendix G).

3.1.4.3 Prototype Development

Phase I prototype development addressed the requirements and design documented for the Bridge Federate. A subset of the HLA Services was implemented during the first phase, which includes Federation Management, Declaration Management and Object Management. In addition, particular services of the Bridge Federate were also developed. Services for HLA and the Bridge provided by Phase I are included in the tables that follow.

3.1.4.3.1 Phase I – HLA Services

Unless noted in the "Details" column, the Bridge Federate will implement the following services as specified in the Interface Specification (version 1.1).

Table 1 Phase I HLA Services

SERVICE GROUP	SERVICE TYPE	SERVICE	Additional Details
FEDERATION MANAGEMENT	Create/Destroy	Create Federation Execution	Provided by Federation Driver software.
		Destroy Federation Execution	Provided by Federation Driver software.
	Join/Resign	Join Federation Execution	
		Resign Federation Execution	
	Pause/Resume	Request Pause	
		Initiate Pause†	
		Paused Achieved	
		Request Resume	
		Initiate Resume†	
		Resume Achieved	
	Save/Restore	Request Federation Save	Two iterations – 1st w/o time and 2nd w/ time
		Initiate Federate Save†	Two iterations – 1st w/o time and 2nd w/ time
		Federation Save Begun	Two iterations – 1st w/o time and 2nd w/ time
		Federation Save Achieved	Two iterations – 1st w/o time and 2nd w/ time
		Request Restore	
		Initiate Restore†	
		Restore Achieved	

Table 1 Phase I HLA Services (cont)

SERVICE GROUP	SERVICE TYPE	SERVICE	Additional Details
DECLARATION MANAGEMENT	**Publication and Subscription**	Publish Object Class	
		Subscribe Object Class Attributes	
		Publish Interaction	
		Subscribe Interaction	
	Flow Control	Control Updates†	If a federate receives a Stop_Updates for attributes, this will not cause the invocation of unsubscribe in the other federation executions.
		Control Interactions†	If a federate receives a Stop_Updates for interactions, this will not cause the invocation of unsubscribe in the other federation executions.
OBJECT MANAGEMENT	**Object Representation**	Request ID	
		Register Object	
		Update Attribute Values	
		Discover Object†	
		Reflect Attribute Values†	
		Delete Object	
		Remove Object†	
	Object Interactions	Send Interaction	
		Receive Interaction†	
	Request Attribute Values	Request Attribute Value Update	
		Provide Attribute Value Update†	
	Event Retraction	Retract	
		Reflect Retract†	

3.1.4.3.2 Phase I – Bridge Federate Services

Table 2 Phase I Bridge Federate Services

Bridge Federate Components	Component Objects	Object Functions	Implementation Details
SURROGATE	Surrogate Federate Object	Provide Transformation Mgr methods which invoke Surrogate's RTI Ambassador	
		Maintain Save/ Restore, Pause/ Resume state	
		Saves / Restores own state	Implemented during second iteration of Phase I.
		Get event / Request RTI Event Tick	
	Surrogate Ambassador	Implements Federate Ambassador for Surrogate	
		Get Next RTI Event	
TRANSFORMA-TION MANAGER	Transformation Manager Object	Determining which FEDEX to bridge	
		Determine how to map transactions between FEDEXs	
		Provide methods (Join, Publish, Tick and Resign) to Main of Bridge Federate	
		Provide methods to support the Tick Method	
	Mapping Manager Object	Maintains all translations and transformations	
		Maintains FOMAT Mapping / Transformation specification	Supports 1 to many mappings.
		Maintains object / attribute ID and ownership association	
		Value transformation process	Supports all defineable transformations.

3.1.4.4

Application Level Experiment

To provide a proof-of-principle for the functional requirements and design in Phase I of the Bridge Federate, an application level experiment was performed. This experiment featured CCTT SAF interoperating with ModSAF. Further details for this experiment are included in Section 0 and have been documented using a Federation Workbook (Appendix D).

3.1.5 Phase I Challenges and Lessons Learned

3.1.5.1 Software Development Approach

All Bridge Federate software was developed using Ada 95 (Power Ada 2.2.2) running under AIX 4.2. The design for the software was specified through the development of the Ada Specifications (Appendix C). These specs were then used to develop the Ada body of the software.

All operations were tested by using a pair of user interactive drivers. The driver allowed a user to direct the execution of a number of HLA services. For example, a user can request that an object be created and updated, or an interaction be sent. When the driver is invoked by the RTI, text I/O indicates the invocation and its parameters. These drivers allowed the developer to verify that operations which were invoked in one federation were properly bridged to the other participating federation execution. In addition to this check-list oriented testing, the bridge federate was also exercised/tested by its use to bridge a CCTT federation execution and a ModSAF (used with the Federation Common Software – FCS) federation execution.

3.1.5.2 General Lessons Learned

The implementation of the Bridge Federate requirements with two different types of federations and FOMs afforded us some interesting observations:

1. **Mapping Classes to Attributes:** The class hierarchy of the FOM implemented in CCTT SAF differed significantly from the hierarchy in the ModSAF FCS FOM. Because of the single object class in the ModSAF FCS FOM, mapping to the class hierarchy in the CCTT SAF FOM required the bridge federate to be able to map from a single object class to multiple classes based on attribute values. The requirements document needs to be refined to address this type of situation. One approach for incorporating this requirement is to allow for predicates to be specified in the FOMAT file as part of the mapping definitions. These predicates would define the attributes and values which must be received in order to map to a particular object class. In addition, this particular mapping would require the bridge to wait for attribute updates before the object can be created in the other federation execution (in order to identify the class to which the object belongs in the other federation execution).
2. **Defining Transforms:** Defining the transforms is a tedious process which requires detailed knowledge of the systems being bridged. For example, a great deal of time was spent to determine exactly what kind of data is needed in order for the federate models to react correctly to some of the interactions.
3. **Object ID Mappings:** Many FOMs in use today contain ID related attributes such as entity site/application/entity ID. This leads to two challenges for the bridge federate:

(F) The bridge federate may need to generate these attributes if they are present in one fedex and not in the other, and thus, must somehow be informed of the id generation algorithm used as well as any input parameters for the algorithm (e.g. for this exercise use site=9304).

b) The bridge federate must maintain the following kind of mapping: (fedex1, id attributes, object id) -> (fedex2, id attributes, object id)

3.1.5.3 Initial Capability Demonstration

To demonstrate the initial capabilities of the Bridge Federate, a demonstration was conducted using the test drivers to exercise the bridge and show its ability to perform as required.

3.1.5.4 Application Demonstration

The application demonstration of the Bridge Federate was conducted to show the capability of the Bridge to work with two similar simulations using very different FOMs. These applications, ModSAF FCS and CCTT SAF were demonstrated to work together in a previous demonstration of the Ada API and binding code in March 1997. For the earlier demonstration, a common FOM was developed and the software of the federates were modified to allow them to work to the common FOM. For the Bridge Federate application demonstration, the two federates were allowed to use their native FOMs and the translation between the two was provided by the Bridge Federate.

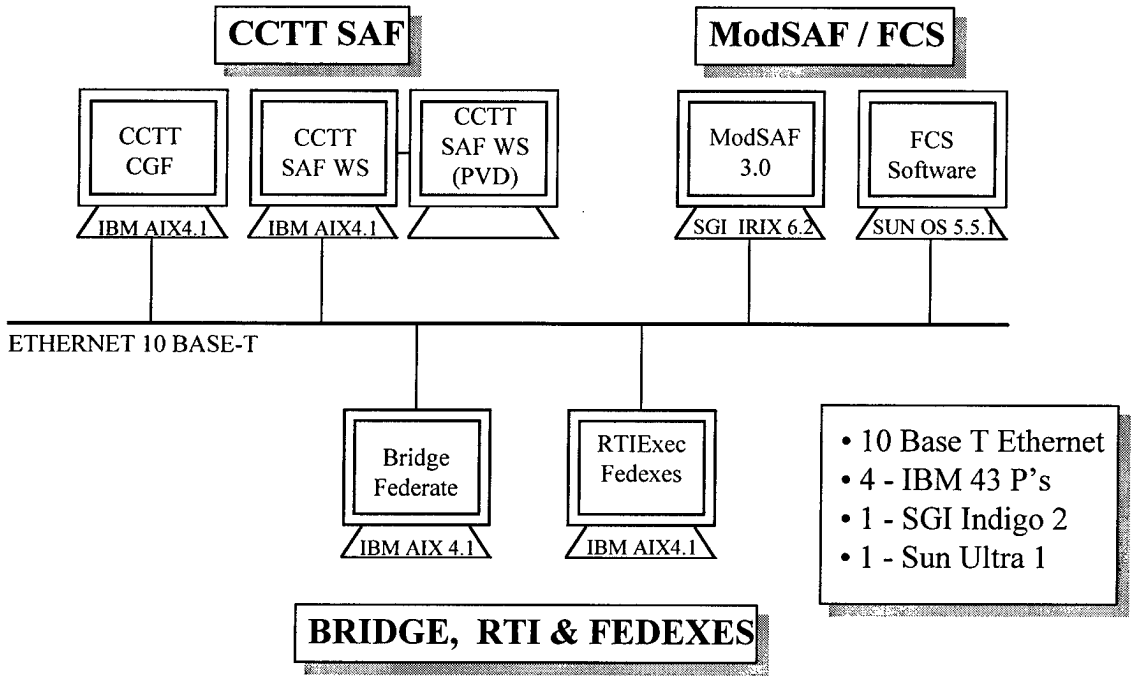


Figure 3 Application Demo Hardware Configuration

The application demo was conducted using a variety of hardware and operating systems (**Figure 3**). ModSAF was executed on a SGI system under IRIX 6.2. The FCS software executed on a Sun Ultra using Sun OS 5.5.1. CCTT SAF, the Bridge Federate and the RTI & Fedexes were all executed on IBM 43P's using AIX 4.1. These systems were linked together on an Ethernet network. CCTT SAF used FDDI for management traffic (not utilizing HLA).

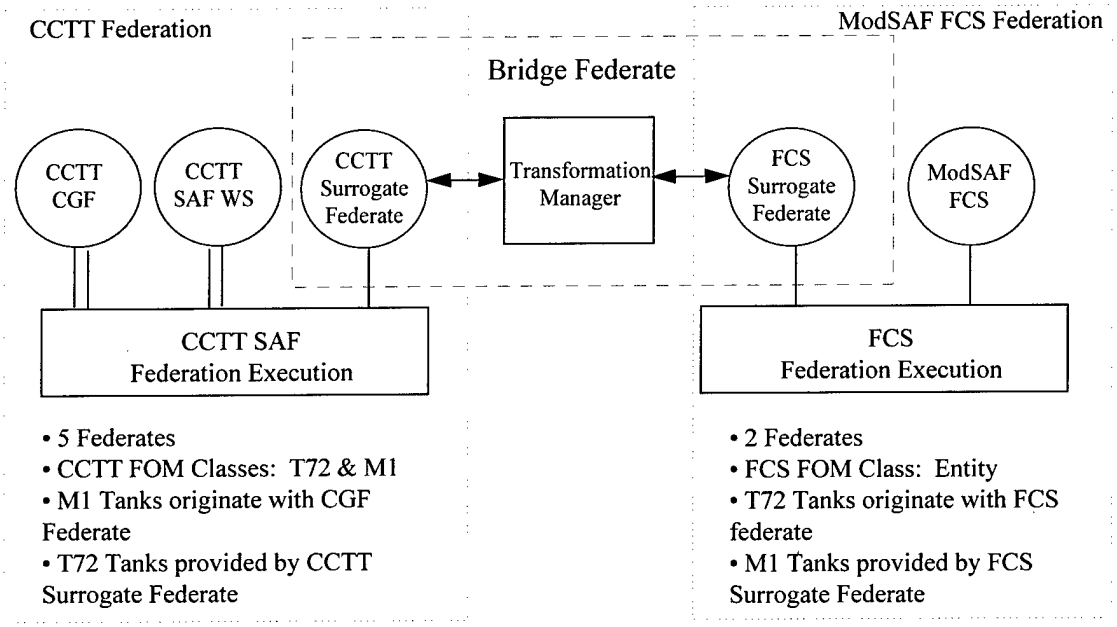


Figure 4 Federation Execution Configuration

A federation execution view of the application demo is provided in **Figure 4**. This figure shows the CCTT SAF Federation with the CGF and Workstation each connecting to the RTI twice (as is in the CCTT SAF HLA design). The CCTT Surrogate Federate is the part of the Bridge Federate that participates in the CCTT Federation on behalf of the ModSAF FCS Federation. It also collects information from the CCTT SAF Federation and provides it to the Transformation Manager for use by the ModSAF FCS Federation.

Likewise, a FCS Surrogate Federate is the part of the Bridge that participates in the ModSAF Federation on behalf of the CCTT Federation. It collects information from the ModSAF Federation and provides it to the Transformation Manager for use by the CCTT Federation.

3.1.5.4.1 Purpose

The primary purpose of the application demonstration was to demonstrate the initial capabilities of the bridge within a real-life type of application situation. Secondary purposes included basic interoperability between ModSAF and CCTT SAF, examination of bridging two disparate FOMs for similar types of applications and exploration of details that should be included in the Federation Required Execution Details (FRED).

3.1.5.4.2 Capabilities

Because of limitations of the federates, not all Phase I capabilities were demonstrated during the application demonstration. All capabilities were demonstrated except for: Save/Restore, Pause/Resume and Retract. These were tested and demonstrated using the driver demo.

3.1.5.4.3 Federation Required Execution Details (FRED)

As part of our Phase I implementation, the project team kept a notebook of information that would be required for execution of the Federation. These included the follow:

1. Brief description of the federation experiment and the demonstration configuration and scenario.
2. Federate SOMs/FOMs.
3. Federation Developer's Notebook.
4. Draft procedures for startup and execution of demo equipment and software programs.

The contents of the notebook are included in Appendix D.

3.1.6 Conclusions

The conduct of the first Phase of Bridge Federate prototyping accomplished its goal to examine the requirements for the Bridge federate through prototype development and demonstration. Several modifications were made to the Requirements Document as a result of our Phase I research. Plans for Phase II implementation include:

1. Development / revision of Bridge Federate requirements to support Ownership Management. This will be followed by an implementation of Ownership management.
2. Develop detailed description of the FOMAT and plans for implementation.
3. Development and implementation of an application experiment that will examine ownership transfer across the bridge.

As time allows, other brief experiments will be performed to explore the bridge's ability to support:

- More than two federations bridged (3 or more). We would explore the scalability of the bridge.
- The bridging of different versions of the RTI.

The first capability should be supported by the current design for the bridge. The second would require some redesign of the software (currently have a single process design – this would require multi-thread processing with IPC). Neither of the concepts have been tested to date.

3.2

3.2.1 Introduction

The High Level Architecture (HLA) Support Experiments (HSE) project has been performing a number of tasks to support the evolution and application of the HLA. One task that the project has been focusing on has been the requirement definition, concept development and prototyping of a Bridge Federate. This report provides a summary of the progress made during the second phase of analysis and development of the Bridge Federate which focuses on Ownership Management services. The bridge federate description and the bridge federate development approach are documented in the HLA Support Experiments phase I report.

3.2.2 Phase II Capabilities

3.2.2.1 Documentation

Documentation for Phase II includes a Requirements Specification which records the functional requirements for the Bridge Federate. This document is the primary product of this research effort and is originated from the Phase I activity. The Requirement Specification has been revised to reflect the results of the Phase II analysis and development efforts. The final revised document is included as Appendix B. A summary of the requirements and design has been included in a paper developed for the 1998 Spring Simulation Interoperability Workshop entitled " Implementing Ownership Management Services With a Bridge Federate" (Appendix G).

3.2.2.2 Prototype Development

The prototype development addressed the HLA Ownership Management Services. In addition, particular services of the Bridge Federate were also developed. Services for HLA and the Bridge provided by Phase II are included in the tables that follow.

3.2.2.2.1 Phase II – HLA Services

Table 3 Phase II HLA Services

SERVICE GROUP	SERVICE
Ownership Management	Request Attribute Ownership Divestiture
	Request Attribute Ownership Assumption +
	Attribute Ownership Divestiture Notification +
	Attribute Ownership Acquisition Notification +
	Request Attribute Ownership Acquisition
	Request Attribute Ownership Release +

3.2.2.2.2 Phase II – Bridge Federate Services

Table 4 Phase II Bridge Federate Services

Bridge Federate Components	Component Objects	Phase II Functions	Implementation Details
SURROGATE	**Surrogate Federate Object**	Provide methods which invoke Surrogate's RTI Ambassador's Ownership Management services.	
	Surrogate Ambassador	Implements Federate Ambassador for Surrogate	
TRANSFORMA-TION MANAGER	**Translation Manager Interface**	Propagate Ownership Management services to the surrogates of other FEDEXs.	This package is not being implemented as a class due to problem with circular dependency.
	Transform	Transform ids that are necessary for this phase.	

3.2.2.3 Application Level Experiment

To provide a proof-of-principle for the ownership management requirements and design in Phase II of the Bridge Federate, an application level experiment was performed. This experiment featured a damage assessment federate which can acquire the appearance attributes from CCTT SAF and perform damage assessment on its behalf. Further details for this experiment are included in section 0 and have been documented using a Federation Workbook (Appendix E).

3.2.3 Phase II Challenges and Lessons Learned

3.2.3.1 Software Development Approach

All Bridge Federate Software was developed using Ada 95 under AIX 4.2. The compiler used was Power Ada 2.2.2. The phase II of the bridge federate uses the code from phase I as the baseline and adds ownership management services to phase I implementation.

All the new operations were tested by using a pair of user interactive drivers before integrating the bridge federate with the applications. The drivers allowed the divestiture or acquisition of the attributes of certain objects from one driver to the other.

3.2.3.2 General Lessons Learned

The analysis and implementation of the phase II Bridge Federate requirements afforded us some interesting observations :

(F) **Negotiation of the Ownership of the Attributes:**
From the requirement analysis, it was discovered that when negotiation has to take place among multiple federations, there are potential problems that one has to be aware of. These potential problems are mainly due to the fact that RTI conducts the negotiation among multiple federates wishing to take ownership of the same attributes. The functional diagram in **Figure 5** and subsequent paragraphs describe the potential problems in further detail.

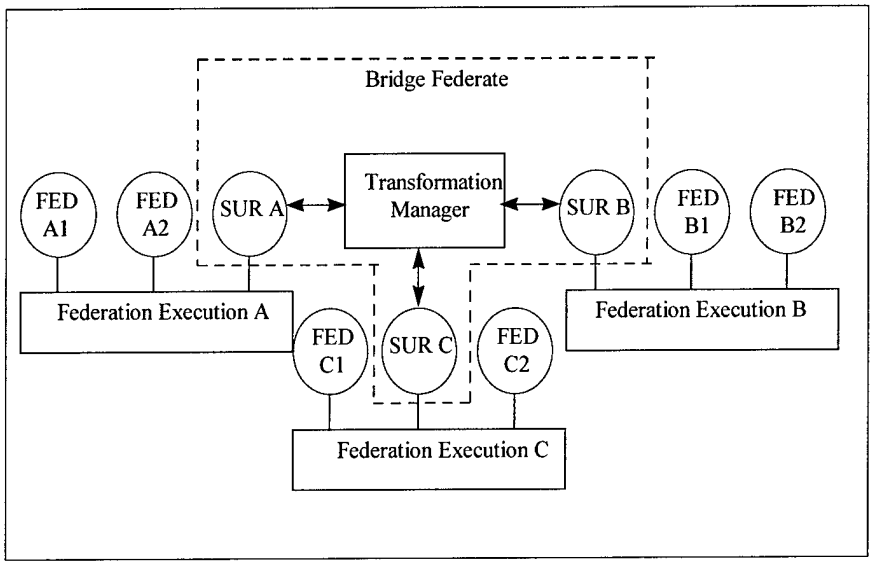


Figure 5 Bridge Federate Functional Diagram

Take for instance, federate A1 (Fed A1) issues a Request Attribute Ownership Divestiture of a specific FEDEX A object instance and attributes. Through the Bridge Federate, this Request Attribute Ownership Divestiture message is sent to Surrogate Federate B (SUR B) and SUR C. FEDEX A, B and C send a Request Attribute Ownership Assumption to their own federates, i.e. FEDEX A to FED A2 and SUR A; FEDEX B to B1 and B2; FEDEX C to C1 and C2. The problem occurs when there are more than one federate from different federation executions that want to take over the ownership of the offered attributes. The following paragraphs examine two possible cases:

Case 1: If both Fed C1 and Fed B1 want to take over the ownership, Fed C1 will send an Attribute Ownership Acknowledge to FEDEX C, FEDEX C will grant that ownership to Fed C1 with an Attribute Ownership Acquisition Notification. In the same time, Fed B1 will also send a Attribute Ownership Acknowledge to FEDEX B, and the ownership will be granted to Fed B1 by FEDEX B. Although the Fed C1 and Fed B1 have the ownership of the attributes in their respective federation executions, but the same set of attributes is owned by both federates in the combined federation executions.

Case 2: If Fed A2 and Fed B2 both wants to take over the ownership, Fed B2 will send an Attribute Ownership Acknowledge to FEDEX B, and FEDEX B will grant the ownership to Fed B2. Surrogate B will get the Attribute Ownership Divestiture Notification from FEDEX B and passes the message to the Transformation Manager and to Surrogate A. Surrogate A then sends an Attribute Ownership Acknowledge to FEDEX A to ask for the ownership. However, Fed A2 also has informed FEDEX A that it wants to take over the ownership. FEDEX A now conducts negotiation and suppose FEDEX A granted the ownership to Fed A2. Therefore the same set of attributes is once again owned by both Fed A2 and Fed B2.

Possible solution for Case 1 is to have Transformation Manager propagate the Request Attribute Ownership Assumption to the other Federations one at a time. For example, the Transformation Manager will propagate the Request Attribute Ownership Assumption to Federation B first, wait for the result, if no federate in Federation B wants the ownership, then the Transformation Manager will propagate the Request Attribute Ownership Assumption to Federation C. If a federate in Federation B wants to assume the ownership of the attributes, the Transformation

Manager will not propagate the request to Federation C. The short fall for this solution is this can take a long period of time if a large number of Federations are involved in the negotiation.

For Case 2, one possible solution is to use RTI interactions to perform pre-ownership transfer negotiation before the actual ownership transfer takes place. This method will require a common ownership negotiation protocol that is understood by all the participating federates from all the bridged federation executions.

(F) **FOMAT file :**

The generation of the FOMAT file requires extreme care. If a parenthesis is missing or an extra parenthesis is mistakenly added to the FOMAT file, an error will occur during runtime when the file is being parsed. The error message does not indicate which line the error occurs, so it can be a very tedious exercise to find out where the error is. The best debugging method is to build the FOMAT file incrementally and run the bridge software every time to make sure there is no parsing error before adding more lines to the file.

(F) **Two-way blocking call from RTI:**

The Attribute Ownership Assumption service, which features is a two-way blocking call initiated by the RTI had created some challenges for the design and implementation of the bridge federate. The Tick method was used to implement the phase I bridge federate. When the federate ambassador for the surrogate object receives a call from the RTI, it put that event into a queue, the transformation manager object Tick will request the next event from a surrogate object. The surrogate will in turn request the next event from the federate ambassador for that surrogate. However, with the two-way blocking call, it is not possible to put the Attribute Ownership Assumption service in the event queue.

Therefore, this iteration of the bridge federate does not put Ownership Management services into event queue. Rather, direct calls as shown in the Requirement Analysis documentation use case was implemented. This can be changed when the future version of the RTI breaks the two-way blocking call to two one-way calls (a feature to be included in RTI 1.3). Another interesting implementation fact is when the surrogate objects send a Request Attribute Ownership Divestiture to their respective FEDEX after they received a message from the translation manager, the surrogate objects have to stay in the Request Attribute Ownership Divestiture routine till the federate ambassador for the surrogates received a Attribute Divest Notification from their FEDEXs. This has to be done so the control does not go back to the original Request Attribute Ownership Assumption two-way blocking call prematurely.

3.2.3.3 Application Demonstration

The application demonstration of the phase II of the Bridge Federate was conducted to show the capability of the Bridge to work with ownership management. Two identical FOMs were used for the two federations that were bridged to minimize delays during transformation. The demo was conducted using Motorola Power Stacks running AIX4.2. The hardware configuration is shown in **Figure 6**.

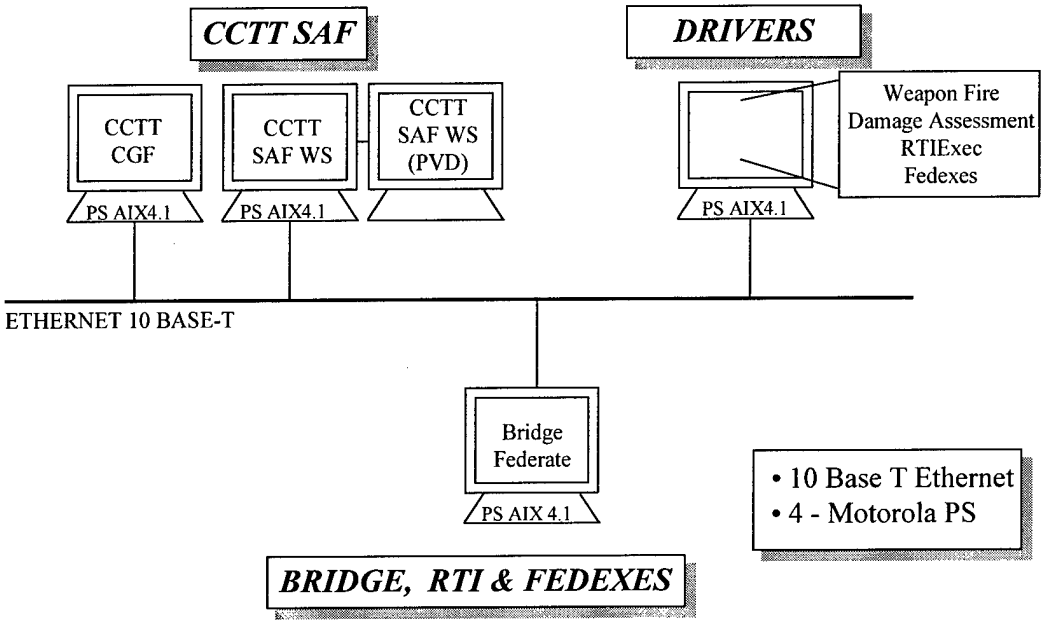


Figure 6 Application Demo Hardware Configuration

The scenario for the ownership transfer application experiment involves two federation executions as shown in **Figure 7**. One is the CCTT federation where the CCTT system is simulating a platoon of M1 tanks moving along a designated route. The CCTT system joins that federation through its Network Process and Entity Database federates. The same federation is also joined by its federate surrogate from the bridge federate. The other federation is the Specialty federation joined by the Damage Assessment federate, the Weapon Fire federate and its surrogate from the bridge federate.

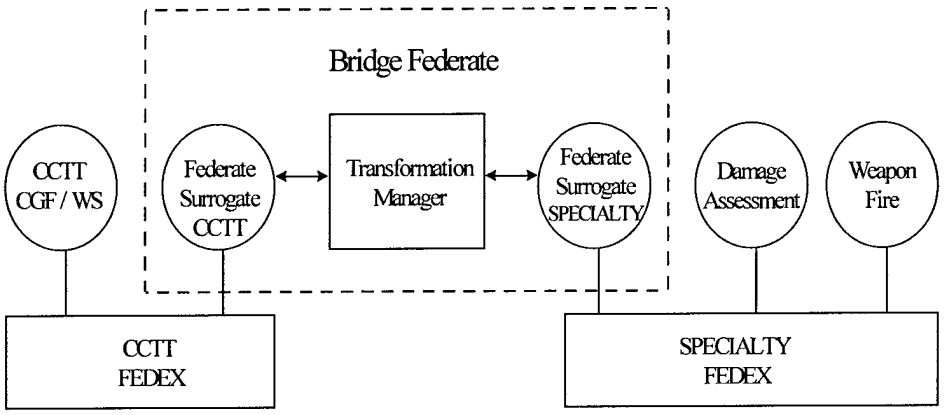


Figure 7 Application Demo Functional Diagram

For the purpose of this experiment, the two federation executions have the same federation object model (FOM), although federation executions using different FOMs can be bridged. The difference between them is only notional in the sense that the information flow from one federation execution to another is through the bridge federate. In this experiment, the bridge federate performs a minimum of information transformation to map the object identifications

from one federation to another, all other information is propagated without any data transformation.

The Weapon Fire federate's firing action is controlled through user inputs. The Weapon Fire federate plays the role of a weapon system whose damage effects on its targets is only known to the Damage Assessment federate. It is, therefore, the responsibility of the Damage Assessment federate in the Specialty federation execution to assess any damage on the M1 tanks caused by the Fire and Detonation interactions originated from the Weapon Fire federate on behalf of the CCTT federate.

The application experiment is composed of the following players:

1. A blufor M1 tank platoon.
2. A weapon system firing "special" munitions at the M1 tanks.
3. A damage assessment system to assess the damage suffered by the M1 tanks caused by the munitions fired by the weapon fire system.

The scenario for the experiment unfolds as follows:

The M1 tank platoon will march along a designated route while its tanks will be fired at by the Weapon Fire system. The tank platoon will continue to follow the route until it is completely disabled, meaning that all of its tanks have suffered catastrophic kills. The Weapon Fire system will be depicted as a stationary T72 tank, which will not suffer any damage. The Damage Assessment system will not be depicted visually because it is not a battlefield entity.

For this experiment, the Appearance attributes were chosen to portray the state of the M1 tanks on the battlefield as to their damage level. There are three attributes that describe the damage state of a vehicles, namely, the Damage State Appearance, the Damage State Mobility and the Damage State Fire Power attributes. The transfer of these attributes signifies the transfer of the damage assessment responsibilities from one federate to another.

A typical sequence of events for the appearance attribute ownership transfer is described in **Figure 8**:

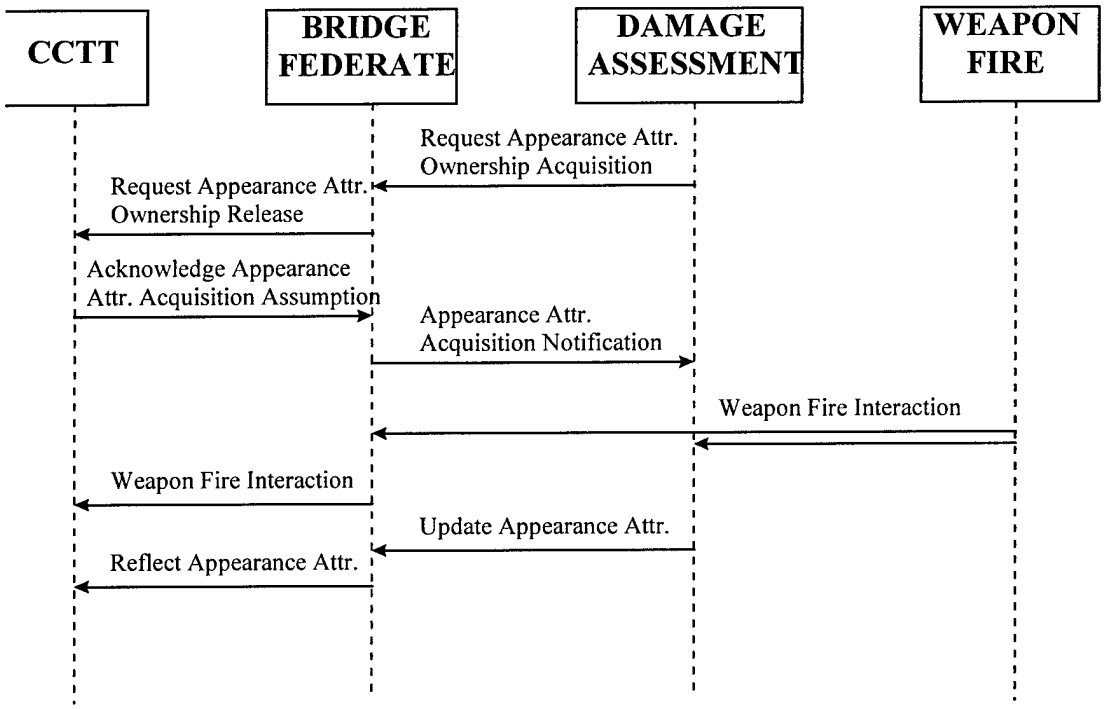


Figure 8 Application Demo Use Case

1. The Damage Assessment federate invokes the Request Attribute Ownership Acquisition service to obtain the ownership of the appearance attributes from the CCTT federate with regard to the M1 tanks.
2. The CCTT federate acknowledges the request for divestiture release invocation and makes the appearance attributes ownership available.
3. The Damage Assessment federate is informed of the ownership acquisition notification which concludes the appearance attribute ownership transfer from the CCTT federate to the Damage Assessment federate.
4. The user commands the Weapon Fire federate to fire "special" munitions at a specific M1 tank.
5. Both the CCTT federate and the Damage Assessment federate receive the interaction corresponding to the fire of the "special" munitions.
6. The CCTT federate ignores the interaction because it no longer owns the appearance attributes.
7. The Damage Assessment federate, verifies that it owns the appearance attributes, assesses the damage, updates its internal state of the M1 tank and updates the M1 tank's appearance attributes on behalf of the CCTT federate.
8. The CCTT federate receives the reflection of the appearance attributes from the Damage Assessment federate and updates its internal state of the M1 tank with regard to its damage level.

In the scenario above, the transfer of appearance attributes ownership can occur at any time at the Damage Assessment federate at the user's discretion. The damage assessment responsibilities is initially assigned to the CCTT federate. The transfer of the appearance attributes ownership is

initiated by the Damage Assessment federate using the Request Attribute Ownership Acquisition service. These same attributes are transferred back to the CCTT federate from the Damage Assessment federate using the Request Attribute Ownership Divestiture service.

3.2.3.3.1 Purpose

The primary purpose of the application demonstration was to demonstrate the ownership management capabilities of the bridge federate within a real simulation application.

3.2.3.3.2 Capabilities

The Phase II capabilities, namely, the acquisition and divestiture of the attributes ownership were demonstrated during the application demonstration.

3.2.3.3.3 Capabilities

As part of the Phase II implementation, the project team kept a notebook of information that would be required for execution of the Federation. These included the following items in (Appendix E):

1. Brief description of the federation experiment and the demonstration configuration and scenario.
2. Federate SOMs/FOMs.
3. Federation Developer's Notebook.
4. Draft procedures for startup and execution of demo equipment and software programs.

3.2.4 Conclusions

The conduct of the second phase of the Bridge Federate accomplished its goal to examine the ownership management services requirements for the Bridge Federate through analysis and prototype development and demonstration. Plans for upcoming phases for the Bridge Federate are:

1. Development / revision of the Bridge Federate requirements to support Time Management services.
2. Development / revision of the Bridge Federate requirements to support Data Distribution Management.
3. If necessary, development and implementation of an application experiment that will examine the Time Management and/or Data Distribution Management across the bridge.

4. MULTI FEDERATION EXECUTION EXPERIMENT

4.1 MULTIPLE FEDERATION EXECUTION EXPERIMENTATION

The purpose of this experiment effort was to explore RTI F.0 support for multiple federation executions operating simultaneously. The motivation for this experiment was to explore the use of a separate federation execution to support SEOD traffic for CCTT SAF.

There were two levels of experiments. The first level sought to demonstrate RTI F.0 support for a single federate participating in multiple federation executions (fedex) operating simultaneously. The second was to generate some preliminary data on the performance impact of using two federation executions verses a single federation execution.

4.1.1 Level 1 – Multiple Fedex Support

The configuration for level 1 of the experiment is shown in **Figure 9**. Fedex A represents a typical federation execution such as the PPF. Fedex B represents internal use of HLA for data not required by federates outside of CCTT (such as SEOD or simulation management). For the experiment, a single federate joins both fedexes which are operating simultaneously.

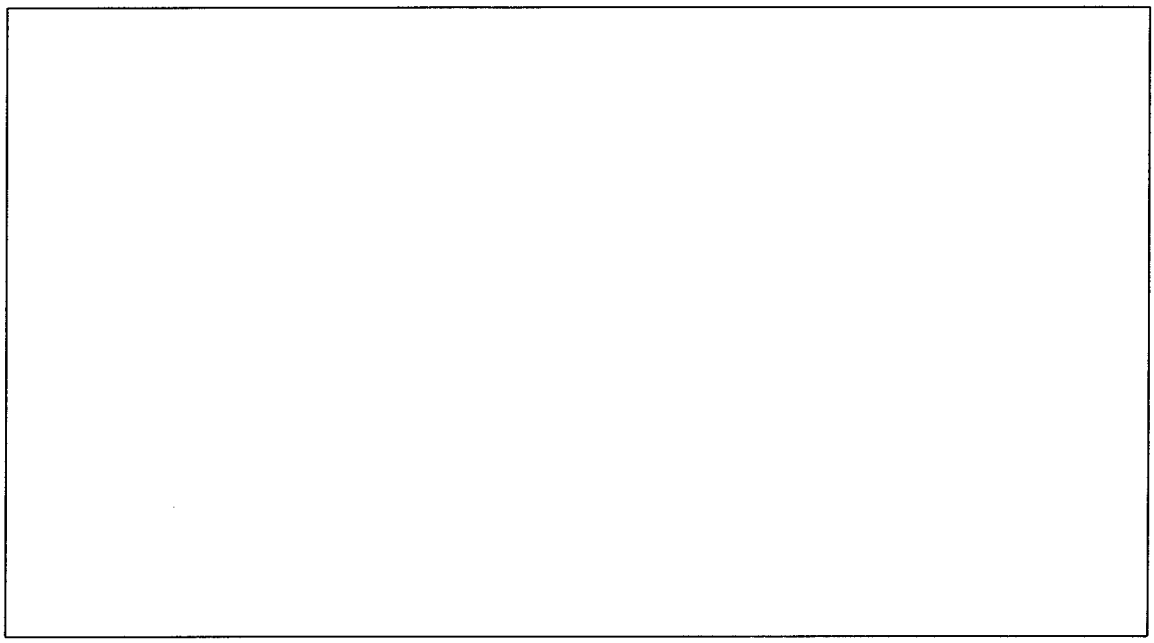


Figure 9 Multiple Federation Execution Experiment Configuration

4.1.1.1 Approach

To perform this experiment, a special driver was developed which created and joined the two fedexes. The driver also allows the user to perform publish, subscribe, update attribute values and send interactions for a specific fedex.

In addition, two pairs of FOMs were used. One FOM had no overlapping data other than the MOM data (e.g. CCTT with PPF Vs CCTT with SEOD). The other FOM had overlapping data (same FOM was used). This allowed us to examine whether overlapping data was delivered to the correct fedex.

4.1.1.2 Experiment Scenario

The experiment was conducted using the driver as shown in **Figure 10**. For this configuration, the sequence of events was as follows:

1. Execute the RTI Exec
2. Execute the 1st driver. Then create fedex A and fedex B. Join both fedexes.
3. Execute the 2nd driver. Join both fedexes.
4. Perform a variety of operations: publish, subscribe, update attribute values, send interactions.

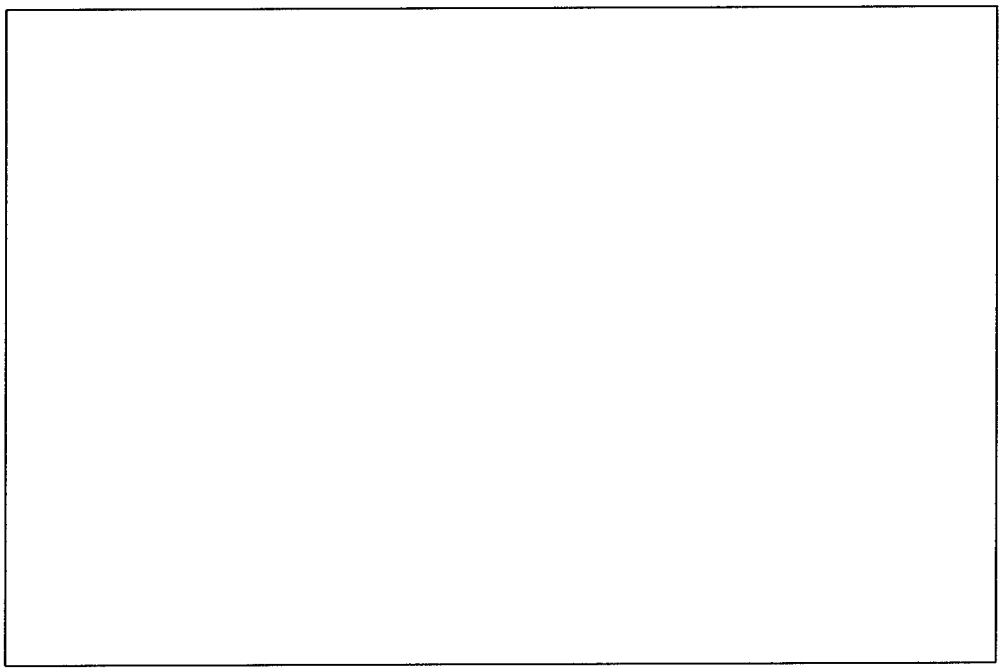


Figure 10 Level 1 Experiment Configuration

4.1.1.3 Experiment Results

Our experiment found that the RTI F.0 did correctly allow a single federate to join multiple federation executions operating simultaneously when Reliable Transport delivery service was utilized. Our experiment uncovered a problem with the IBM/AIX configuration we were using when Best Effort Transport was employed. This problem allowed cross delivery when identical FOMs were used and caused an error with the MOM data when non-overlapping FOMs were used.

4.1.2 Level 2: Performance Impact

The level 2 experiment sought to address the problem: What is the performance impact (if any) of executing two fedexes simultaneously versus a single fedex? Our approach to address this

problem was to do some preliminary timing experiments with a driver for some initial results, then to implement CCTT SEOD as a second fedex and run further timing experiments.

4.1.2.1 Timing Experiment

For comparison purposes, we ran two copies of the driver with a single fedex, then running with two fedexes. In each case, both drivers send / receive 1000 updates and 1000 interactions. The steps used for the configuration is shown in **Table 5**. A driver was written for each setup to carry out the steps in the table. Two federate drivers are then executed and communicate with one another.

Table 5 Timing Experiment Steps

Single Fedex	Two Fedex

create fedex1
join fedex1
publish objects, interactions for fedex1
subscribe objects, interactions for fedex1
for I in 1..N loop
  update object for fedex1
  tick the RTI
  update object for fedex1
  tick the RTI
  send interaction for fedex1
  tick the RTI
  send interaction for fedex1
  tick the RTI
end loop
resign fedex1
	create fedex1
	create fedex2
	join fedex1
	join fedex2
	publish objects, interactions for fedex1
	publish objects, interactions for fedex2
	subscribe objects, interactions for fedex1
	subscribe objects, interactions for fedex2
	for I in 1..N loop
	  update object for fedex1
	  tick the RTI
	  update object for fedex2
	  tick the RTI
	  send interaction for fedex1
	  tick the RTI
	  send interaction for fedex2
	  tick the RTI
	end loop
	resign fedex1
	resign fedex2

4.1.2.2 Timing Experiment Results

Each of the configurations was run an average of 5 times. During each run, both federate drivers send/receive 1000 updates and 1000 interactions.

Table 6 shows the results of the timing experiment measurements. In general, execution memory is about 50% more for two fedexes.

Table 6 Timing Experiment Results

Action	1 Fedex Avg time (sec)	2 Fedexes Avg time (sec)	Percent Increase
Update Objects	9.537	13.803	45%
Send Interactions	2.043	3.407	67%
RTI Tick	2.120	1.941	-8%
Create	0.434	0.740	71%
Join	2.742	3.873	41%
Publish	0.077	0.192	148%
Subscribe	0.018	0.033	79%
Resign	0.226	1.279	466%

The results of this experiment were presented at the Federation Management Technical Exchange on 4/29/97.

4.1.3 SEOD Implementation

After the multiple fedexes were implemented using drivers, the technique was implemented using the CCTT SAF SEOD as the second federation. This was to see the multiple fedex support in action with a real application and to test the theory of using the approach within CCTT.

4.1.3.1 Approach

SEOD information is very different from the types of information that is exchanged with other federates for a particular warfare type scenario. SEOD information is used to set up the CCTT SAF and to send information between the CCTT CGF and the CCTT SAF workstation. The data tends to be variable in length and quite long, something allowed by DIS. Although some information would logically call for the use of Update messages (for corresponding PDUs), they were not well suited for our data needs so Interactions were used for SEOD data.

To address the problem of large pieces of data, which could be as large as 4 kbytes, the SEOD data was segmented into smaller pieces that form the parameters of the HLA interactions. In CCTT SAF's DIS mode, there are seven types of CGF PDUs that are used to carry the SEOD data across the network. For our HLA implementation, each type of CGF PDU corresponds to a class of HLA interaction. Each interaction is defined to have as many parameters as needed to accommodate the maximum CGF PDU size for that interaction.

4.1.3.2 Results

The configuration for our experiment is shown in **Figure 11** and **Figure 12**. The CCTT SAF system (SAF workstation and CGF) was used to assess the performance of RTI in these two different scenarios by timing the Update_Attribute_Value invocations and Send_Interation invocations during a simulation exercise. The Update_Attribute_Value operation updates of object attributes and the Send_Interaction issue the new events. The conditions of the experiments are described below:

1) The scenarios were executed using one CGF system and two SAF workstations being one for the friendly force and one for the opposing force. The CGF system simulated the vehicles for both forces. All three systems were connected to a FDDI network.
2) The simulated friendly force composed of a tank platoon of four M1s followed a route where it would later come in contact with the opposing force and engage it in a battle.
3) The simulated opposing force composed of a tank platoon of three T72s followed a route where it would later come in contact with the friendly force and engage it in a battle.
4) The experiment was conducted in the SAF lab at the time where the general network traffic was low.
5) The first scenario was executed by running the systems joining two federation executions, CCTT and CGF fedexes, where the DIS related information was routed through the CCTT fedex and the SEOD information was routed through the CGF fedex.
6) The second scenario was executed by running the systems joining one federation execution whose FOM combines the information from the individual CCTT and CGF FOMs.

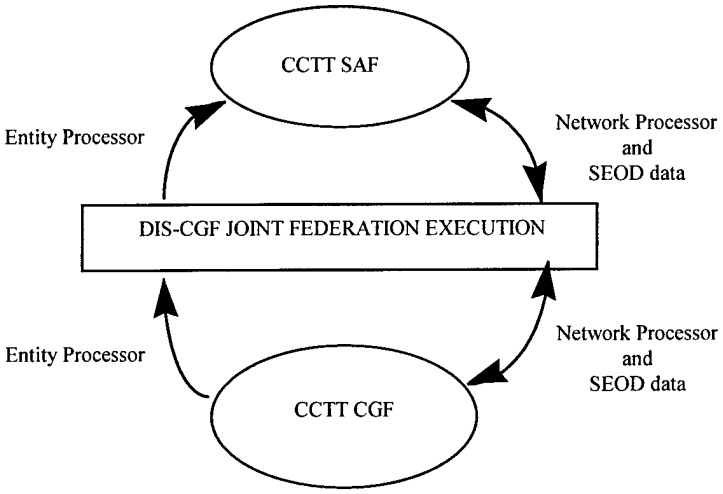


Figure 11 Single Joint Federation Execution

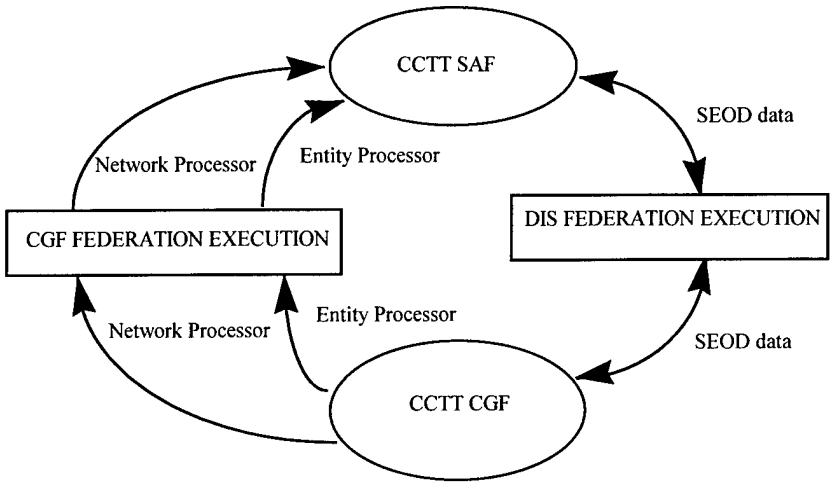


Figure 12 Two Distinct Federations

The experiment produced the following results:

1) The Update_Attribute_Value invocations for the vehicle status updates were made in a high frequency for this particular implementation and produced over 17700 sampling points.
2) The Send_Interaction invocations were made sparsely with more active periods at the simulation start and during the engagement. The total sampling points collected was 390.

The timing was collected twice under the same conditions yielding the following results:

MEAN VALUE (SEC)	TIMING 1		TIMING 2	
	Two Feds	Joint Fed	Two Feds	Joint Fed
Update_Attribute_Value	0.00098	0.00098	0.00684	0.00092
Send_Interaction	0.00281	0.00256	0.0025	0.00262

Final Remarks:

1. The Send_Interaction timing results were consistent throughout the experiments.
2. Although most of the Update_Attribute_Value invocations (90%) executed in less than 0.001 seconds, there were few instances where that time was around 0.1 second, which caused the mean time to vary from experiment to experiment.
3. Based on the results, there is some indication that the RTI performance may not be influenced by using two federation executions or a combined single federation execution; however, the results were weakened by the conditions under which they were obtained that reflects the arbitrary nature of the simulation. More studies is recommended where the conditions should be better controlled to arrive at more stable results.

5. HLA TEST VICTIM

To support HLA testing capability development, the HSE project team supported GTRI test development by having our CCTT SAF participate as a "friendly test victim." In this capacity, we performed the following tasks:

1. Reviewed testing documentation to prepare for testing with GTRI.
2. Obtained access to our facility (across our firewall) for GTRI.
3. Performed SOM testing as required and sent results to GTRI.
4. Performed "live" federate testing with GTRI and AB Technologies.
5. Supported SIW Testing Forum panel discussions and AMG discussions on testing results.

As part of our participation we were able to help identify several issues with testing. One issue was the practice of testing over the internet. Although testing could be performed on-site in the case of a classified facility, the procedures did not take into account corporate firewalls. Although we were able to overcome our firewall issue, this needs to be part of testing considerations in the future.

Another issue was the ability to test a federate (system) that consisted of multiple federates. In our case, CCTT SAF, considered at a system level as a federate, implemented HLA where the CGF and SAF workstations both "join" the federation twice, thus representing two federates. The testing tools sought compliance from a single federate, not the sum of several federates. The testing folks were able to determine that the sum of our federate actions were compliant, there was a need for the tools to accommodate this type of situation.

The results of our testing were positive. We were able to provide valuable feedback to the HLA testing community while determining that our HLA implementation with CCTT SAF was HLA compliant.

6. FOM FLEXIBILITY ANALYSIS

6.1 FOM FLEXIBILITY REPORT

6.1.1 Introduction

The introduction of the High Level Architecture (HLA) has provided the modeling and simulation (M&S) community with a well-defined, flexible framework for developing distributed simulation applications. Migration from existing distributed technologies to HLA requires careful thought to ensure that the flexibility of HLA is maintained for maximum reuse of the application.

The purpose of this report is to provide simulation developers with guidance and lessons learned concerning the development of HLA interfaces that are flexible enough to participate in multiple federations with minimal code or interface changes. This report presents information representing the HLA development process underway by the Defense Modeling and Simulation Office (DMSO) and the Department of Defense (DOD) Architecture Management Group (AMG).

The report begins with some general background information about components of a distributed simulation and where federation flexibility is found. The report then explores several case studies where multiple federations were supported to identify what areas of the implementations were sensitive to Federation Object Model (FOM) changes. Several of these case studies are examined in additional detail. The report concludes with a summary of lessons learned and resources available to the simulation developer to support their HLA development activity.

6.1.2 Background

The successful implementation of HLA requires the creation of conditions under which optimal reuse is a natural and assured outcome. For HLA, reuse means the ability to utilize a particular federate in multiple federations. A critical consideration during HLA interface development is the need to provide an interface flexible enough to allow the federate to participate in different federations, which use different Federation Object Models (FOM).

The development of a Federation Object Model (FOM) is a critical activity within the Federation Development and Execution Process (FEDEP). The contents of the FOM determine the capabilities that must be supported by the participating federates. For federates targeted for reuse in multiple federations, FOM flexibility is essential. Experience has shown that changes to the FOM result in changes to the federates software. The type of changes and the extent to which they are required depends on the manner in which the federate developer implemented the FOM specified capabilities in their High Level Architecture (HLA) interface and associated simulation software.

6.1.3 Distributed Simulation Software Components

The software components of a distributed simulation, which allow it to participate in a federation, may be as shown in **Figure 13**. This representation provides a framework for discussing the software components of a distributed simulation that may be impacted by changes in the FOM under which it is operating. Each of the components is described in the subparagraphs that follow.

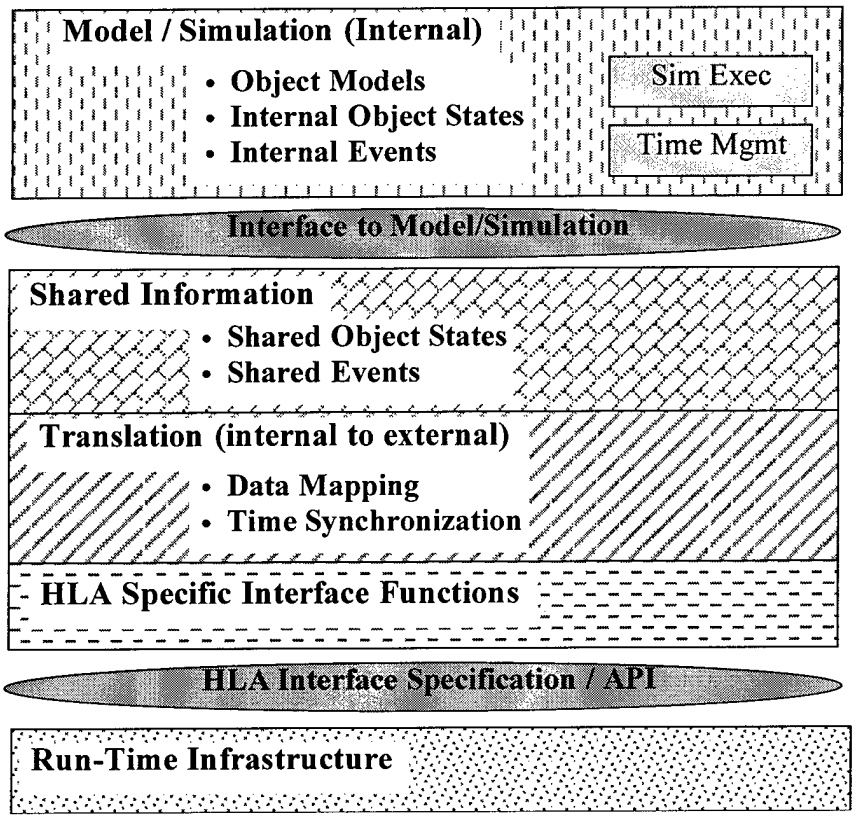


Figure 13 Distributed Simulation Software Components

6.1.3.1 Model / Simulation (Internal)

The Model/Simulation component is the distributed simulation's primary capability. It is the part of the simulation that operates independent of the rest of the federation. The Model/Simulation component contains, but is not limited to, the object models, object states, events, time management, and the simulator's executive. Object models, states and events all represent internally generated objects and events that may or may not be communicated to others in the federation. It represents that particular simulation's view of the world. The time management component manages the simulator's internal simulation time mechanism and the simulation executive provides general operational capability for the federate.

6.1.3.2 Middleware

This distributed simulation component has three layers to it: Shared Information, Data Management / Translation, and HLA Interface functions.

Shared Information

The shared information layer features object and event information that either needs to go from the Model/Simulation to the rest of the federation or, is coming from the federation and is needed by the Model/Simulation. The interface to the Model/Simulation component included here allows the middleware to "speak" to the model in a manner the model understands.

Data Management / Translation

This layer provides the translation of information between the Model/Simulation's internal data model and the FOM specified data model. This requires mapping and translation of information in both directions.

HLA Specific Interface Functions

This bottom layer of the middleware meets the requirements of the HLA Interface Specification [2].

6.1.3.3 Run Time Infrastructure

This distributed simulation component contains the RTI and its standard functionality to communicate with the rest of the federation.

6.1.4 FOM Flexibility in Distributed Simulation

FOM flexibility in distributed simulation requires that changes in the FOM affect as little of the model as possible. The RTI is a component that must remain completely independent of the rest of the model. FOM changes must have no impact here. It is also a common requirement that little or no changes to the model/simulation are made when implementing the HLA interface, therefore, FOM changes must seek to have no impact here. This leaves FOM flexibility to the middleware layers, more specifically to the object representations and data mapping.

Flexibility in the middleware would mean that the flexibility of an interface will depend on how object state information for federation consumption is maintained, how data is mapped from the FOM to the simulation's data model (usually expressed in its Simulation Object Model (SOM)) and how time management is handled by the federate compared to how it is implemented in the target federation. State information can be maintained a number of ways. Typically in DIS, shared memory or hash tables have been used. Object-oriented simulations maintain object state through its associated attributes. Data mapping methods have not been rigorously explored, though the complexity of such mappings will vary depending on how much the FOM varies from the SOM or from previously implemented FOMs. Time management has not been that much of an issue to date, except in the case of one implementation which had to use a hybrid time management scheme to meet the requirements for its simulation. But even in this case, the time management scheme for the various federations the subject simulation participated in were similar.

Table 7 Federate Summary

	CCTT	ModSAF (FCS)	ACETEF	Eagle	JMASS	IST Gateway	NAWC-TSD SMOC
Model Language And Platform	Ada83 PowerPC AIX 4.2	C Sun and SGI	C++ SGI Onyx IRIX 5.2	LISP Sun & HP	C / C++ SunSparc 20 SunOS	C / C++ SGI Irix 6.2	FORTRAN, C PC, SGI Windows NT 4.0, Irix 6.2
Interface Language and Platform	C++/Ada95 IBM 43P AIX 4.2	C++ Sun	C / C++ SGI 440 IRIX 5.2	C++ / LISP	C++ Sun Sparc SunOS	C++ SGI Irix 6.2	C++ PC Windows NT
Model Type	Virtual	Virtual	Constructive (provides for combination of CV&L)	Constructive (Analysis)	Constructive	Virtual	Virtual
Existing Network interface	DIS	DIS	DIS	ALSP	None	DIS	DIS
Interface Approach	Integrated	Gateway	Integrated	Integrated	Integrated	Gateway	Integrated and Gateway
Time Management	Real-Time (TM not impl)	Real-time (TM not impl)	Event-driven, (TM not impl) time-based	Event-Stepped	Event-driven, (TM not impl) time-based	Real-Time (TM not impl)	Real-Time (TM not impl)
Model Code Modifications	None required.	None required.	None required.	Only to support other federate requirements	None required.	None required.	None Required.
Object State	Shared memory	Hash table	Shared memory	LISP Objects	Objects	Hash table	Hash table
Difference between FOMs	Class structure of PPF was deeper and less driven by DIS – converted some attribute types.	Very little. FOMs were nearly the same.	Initial required update for fixed sites, newer requires higher update rates	Little difference	Little – just a few attributes changes and name of basic entity types.	Class structure, data types and attributes from PPF to RPR FOM.	Implemented only RPR FOM but also supported DIS.
New FOM Impact	Changes for mapping data from generic to specific.	Minimal changes required.	Changes only in configuration files.	Little change. Added CCSIL msgs.	Changes just to middleware code.	Changes in classes only in corresponding C++ object classes.	Additional Object Class definitions.

To explore aspects of FOM flexibility, seven different federate implementations were examined.[1] A summary of the federates and their general characteristics is included in

[1] Detailed information on the federate implementations are found in references 2, 4, 6, 7, 8, 9, 10, 12 and attached questionnaires.

Table 7. Some general observations are included in section 0.

6.1.5 HLA Interface Development Approaches

Types of Federates

There was nearly an even split between the number of constructive and virtual type simulations examined for this report. The constructive models varied from analysis / force-on-force to engineering type simulations. The constructive simulations were event-driven with one that required the use of time management. All of the virtual simulations did not use time management and were real-time (wall clock time) types of simulations.

Programming Languages and Hardware

The participating federates were developed in several different languages: LISP, C++, C and Ada. The LISP and Ada simulations required special "binding" code within their interface to allow the native language of the federate to work with the C++ language of the HLA API. Most interfaces were written in C++ with the exception of CCTT SAF that developed its interface in Ada95. Hardware was generally UNIX-based with a mixture of Sun, SGI and IBM equipment. One application was developed under Windows NT running on a 200 MHz Pentium PC.

Pre-existing Interface

Most of the participating federates were DIS-based before implementing HLA. One simulation did not have an interface at all (was actually built from scratch) and the other was ALSP-based.

Object Representation

Each federate had its own method for modeling and updating both internal and external objects. These methods varied from object-oriented representation to shared memory and hash tables. The particular method chosen usually depended on the manner in which the previous DIS or ALSP interface had been implemented.

Implementation Approach

Several papers have discussed various approaches for implementing HLA ([2], [3] and [13]). In general these include an integrated approach (where the existing interface is replaced with an HLA interface) or a gateway approach (where the existing interface and application is not changed, but the gateway provides the translation between the different interfaces). Of the federates explored, two used a gateway approach, four used an integrated approach and one could be used as a gateway or part of an integrated approach. The particular approach chosen was usually due to the constraints for the particular federate's system. Constraints were mainly driven by the federates hardware platform, costs & time (most straightforward approach), and RTI platform support (primarily C++, Sun and SGI).

6.1.6 FOM Flexibility Approaches

After examining the seven cases it was discovered that FOM flexibility was achieved in several ways:

1. **Data driven approach**: In this approach most FOM changes were isolated to changes in configuration data files that did not require recompilation of code. If the changes were severe enough, code changes were required but were localized to make their implementation relatively straightforward.

2. **Object-Oriented FOM Implementation:** For this approach, the FOM objects and interactions were defined in the interface code as objects. Changes in the FOM represented changes to these objects. This approach also allowed for multiple FOM representations to be supported simultaneously.
3. **Isolated interface code /functionality:** All of the examined cases isolated the interface code from both the simulation and the RTI, allowing changes to the FOM to be localized to a few code routines.

Each of the case studies is discussed in the context of these approaches.

6.1.6.1 Specific Case Studies

6.1.6.1.1 CASE 1: CCTT Semi-Automated Forces (SAF)

The STRICOM/PM CATT Close Combat Tactical Trainer (CCTT) is the first fully DIS compliant training system that will train armor, cavalry, and mechanized infantry platoons through battalion/task forces on their doctrinal mission training plan collective tasks. The system consists of networked vehicle simulator manned-modules in combination with Semi-Automated Forces (SAF), and other workstation systems. The SAF system has the capacity to create a wide variety of friendly or opposing force units and vehicles that exhibit highly realistic behaviors to support the CCTT training objectives. CCTT SAF was modified as a case study within a federation of three other DIS, platform-level simulation systems to demonstrate and evaluate the HLA approach.

CCTT SAF HLA Implementation

CCTT utilizes a DIS data model within the simulation. Information is represented internally in terms of Entity state and other PDUs. Components of the DIS network interface were modified to transmit HLA compliant information instead, thus the approach represents an "Integrated" approach.

The CCTT SAF implementation has an architecture as shown in **Figure 14**. Some number of simulation or workstation application processes on a single processor access an entity database and PDU input/output queues contained in shared memory. A separate Entity Processor process maintains the dynamic state of the entity database and the DIS Network Manager process manages the flow of DIS PDUs to and from a FDDI Network. The Entity Processor was modified from its original DIS form to support the HLA interface. For the HLA implementation, it collects, translates, and communicates object updates to and from the RTI. For each locally simulated entity, this process monitors the entity updates in the Entity Database, collects those attribute values that change beyond their dead reckoning thresholds, and sends the entity's attributes as an RTI object update. For remote entity object updates, this process receives the update and modifies the remote entity's attributes in the Entity Database.

The Network Manager was also modified to support HLA. In the HLA implementation, the Network Manager translates certain outgoing DIS PDUs (Fire, Collision, and Detonate PDUs) into FOM interactions and issues these interactions using the RTI. Conversely, the Network Manager receives interactions from the RTI, translates them into corresponding DIS PDUs, and processes them as if they were received from the FDDI network.

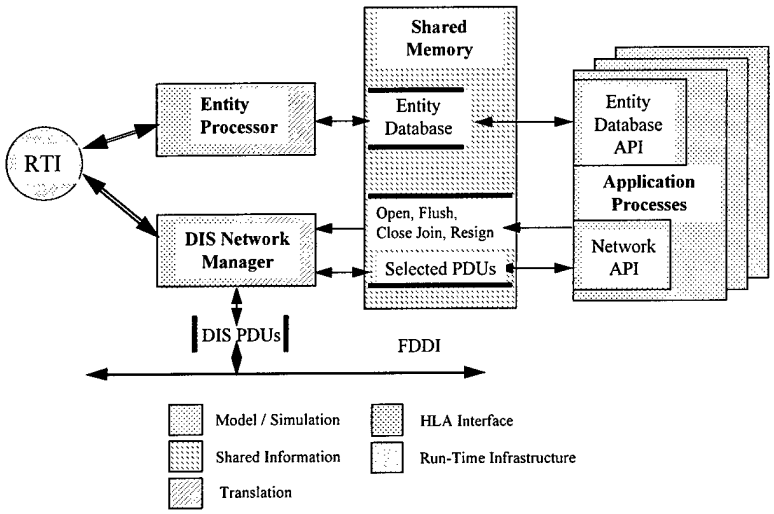


Figure 14 CCTT SAF RTI Integrated Architecture

Through this design each CCTT processor, containing some number of simulation applications, acts as two federates to the RTI regardless of the number or type of applications running on that box. The Network Manager is a federate concerned exclusively with RTI interactions while the Entity Processor is a federate concerned exclusively with objects. It should be noted that DIS PDUs were still used by CCTT SAF for CCTT internal communications (e.g. Simulation management information).

Minor modifications were also made to the applications to request that the Network Manager and Entity Processor join and resign the federation execution.

The HLA implementation for CCTT SAF [2] involved the application, entity database, and PDU queues operating much as they did in their DIS configuration, with minimal changes. To support HLA, a translation between the DIS data model of CCTT and the FOM defined data model was required. Translated data was sent to other federates through the RTI / Federate Interface services to the RTI. Changes to the FOM required changes to the FOM Data Model translation.

FOM Flexibility in the CCTT SAF Implementation

The focus of the CCTT SAF HLA interface development was not on FOM flexibility, so the design was not optimized for this purpose. The CCTT SAF HLA interface used approach 3 (Isolate interface code / functionality). The resulting interface supports one FOM at a time. Future development efforts will look into FOM flexibility as a critical requirement.

6.1.6.1.2 CASE 2: ModSAF Federation Common Software

The Federation Common Software (FCS) was developed to support the objective of the Joint Precision Strike Demonstration (JPSD) experiment to integrate an existing federation of various simulation types using the HLA. This development allowed the integration of the RTI at the output API, but within the same process space of the various federates. Although this approach did not take advantage of the complete set of HLA services, it allowed the verification of the RTI API and its capability to support a complex mixed federate type federation.

FCS Implementation

The FCS (built in C++ using OO design methodologies), shown in **Figure 15**, is composed of several software objects grouped in three functional areas: the RTI API interface services, the federate interface services, and the FOM evaluation services. The software objects were designed to be reusable, i.e., only extensions to base classes are necessary to adapt the software to various federates and FOMs. The FCS provides many functions: encapsulation and automation of services all simulations must exercise; FOM management and RTI services; a framework for translation between simulation and FOM data representation; and a common instrumentation for performance analysis. The JPSD specifically uses simulation translation actors within this FCS to allow for multiple FOMs. The actors are FOM specific.

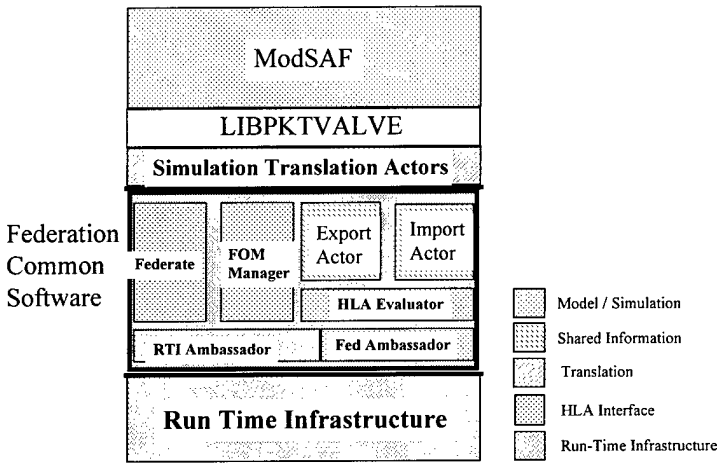


Figure 15 Federation Common Software

FOM Flexibility in ModSAF FCS

FCS was developed with FOM flexibility in mind. It utilized approach 2) through its object-oriented approach to interface development. The Simulation Translation Actors allow for multiple FOM representations. New FOMs required the addition of new actors to this component of the interface and may also require minor changes in the HLA Evaluator and the FOM Manager. Changes are fairly straightforward to make.

6.1.6.1.3 CASE 3: Air Combat Environment Test and Evaluation Facility (ACETEF)

The Air Combat Environment Test and Evaluation Facility (ACETEF) is a fully integrated ground test facility that supports test and evaluation of multi-spectral offensive and defensive aircraft systems in a secure and scientifically controlled engineering environment. ACETEF is located within the Naval Air Warfare Center, Aircraft Division (NAWC-AD) at Patuxent River, Maryland, and is an enterprise team consisting of the Warfare Simulation, Aircraft Simulation, Electronic Warfare Stimulation, and the Electromagnetic Environmental Effects (E3) Stimulation teams. ACETEF led the Engineering Proto-Federation, which successfully completed its mission to "determine the suitability of the HLA to support detailed, high-fidelity simulations in the context of a realistic engineering situation".

ACETEF Implementation

ACETEF's main simulation integrator is the Simulated Warfare Environment Generator (SWEG). The primary function of this software object is to manage a dynamic representation of

the state of the total simulation (much like a stealth observer does for humans). This representation is accessed by all of the computers on the facility's secure network. The ACETEF HLA System Architecture is shown in **Figure 16**.

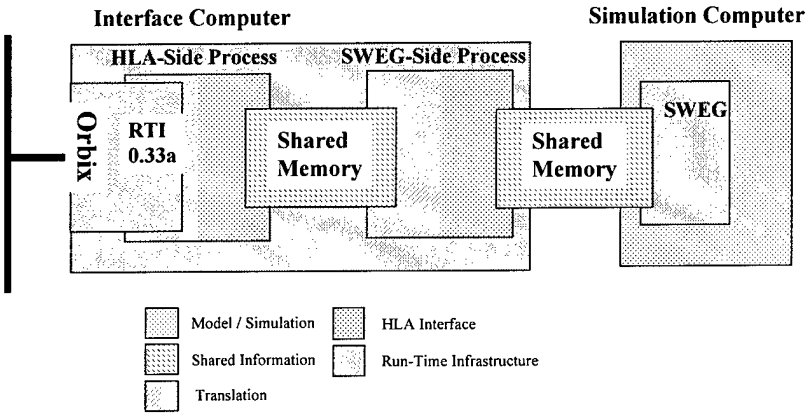


Figure 16 ACETEF HLA System Architecture

The state data for the simulation is called SWEDAT and is physically mapped to shared memory which is reflected in every host through VME-based SCRAMNET® and BIT-3® memory. All ACETEF hosts use an instance of the low-level SWEDAT access routines. These routines are collectively known as the SWEG Interoperability Software Object (SISO). HLA and/or DIS software objects are loaded into their own host and are interfaced, through local shared memory to an instance SISO. If necessary, SISO and the HLA software objects can be loaded into two processors of the same host. The function of the HLA software object is to provide input/output between the SISO-controlled local shared memory and the RTI. Incoming objects are mapped onto SWEG Semantic codes by the HLA software object using the "Entity File" (**Figure 17**). Outgoing objects are mapped onto HLA object classes, or DIS enumeration, using the same file. In order to utilize different FOMs, the "Entity File" must contain all of the classes that ACETEF subscribes to and/or published. When implementing a different FOM, new, previously unknown classes used by the other federation are added to the Entity File. Old object classes would be left in the entity file so that a capability to operate in either (or both) federation(s) is preserved.

SWEG Semantic Code	DIS Enumeration	HLA Object Class
710006	1.2.225.1.1.0.0	Airplane
710007	1.3.225.1.1.1.0	Boat
710006	1.2.225.1.1.0.0	MilitaryPlatform
710007	1.3.225.1.1.1.0	MilitaryPlatform

Figure 17 Entity File Layout

FOM Flexibility in ACETEF

ACETEF had a requirement for a flexible interface because of its need to support a variety of simulation systems. This caused interface functions to be localized and abstracted from the simulations. The ACETEF utilized a combination of approach 1) (Configuration data file) and approach 3) to FOM Flexibility. The way in which the architecture was implemented allows ACETEF to operate simultaneously in multiple federations while preserving support for the pre-HLA DIS protocols currently in use by some of their customers. In the case where new structured attributes or interactions must be implemented, C-language source must be added to both the HLA-side process and the SWEG-side process, and the interface must be re-compiled.

6.1.6.1.4 CASE 4: Joint Modeling and Simulation System (JMASS)

Joint Modeling and Simulation System (JMASS) is a modeling and simulation system that provides a software environment for the development, execution, and post-processing of simulations. JMASS implements standards based set of tools that assist in the development, maintenance, and use of reusable models and model components. JMASS is designed specifically to fulfill the simulation architecture role as defined in the emerging standard test processes from within the Air Force and DDT&E organizations. JMASS implements a single thread of functionality through all the major activities found in modeling and simulation tasks. For the Engineering Protofederation (EPF), the simulation executed under the JMASS architecture provided simulation of threat weapons, a surrogate interface to a defensive avionics suite simulator and the constructive simulation of weapon engagement outcomes. During the JMASS/JPSD experiment, JMASS-based weapon system simulations were integrated into the JPSD Federation in order to explore technologies necessary to enable JMASS simulations to support multiple federation executions.

JMASS Implementation

JMASS used an object-oriented approach to its EPF HLA interface design. Within the JMASS architecture, Port Objects are used to transport data between JMASS models. For the JMASS HLA Engineering prototype, JMASS Port Objects provided the JMASS models with access to the EPF data. The components for the JMASS HLA Engineering Prototype federate implementation are shown in **Figure 18**. The JMASS Simulation Controller, Interconnect Backplane and Team Simulation Engine provide the basis of the JMASS distributed simulation architecture. The JMASS models are linked together with the Team Simulation Engine, and access simulation functions through a standardized Model Applications Program Interface (API). Port Objects, though pictured separately, are technically part of the JMASS model space, providing domain-specific data communications between models. The JMASS Team Simulation Engine provides services for event synchronization, spatial state management, model data management, and model data journalization.

The EPF Middleware presented an standard RTI interface to the EPF federate, while providing federation-wide services to the federates. These federation-wide services included data logging, federation time management, and some standard data conversion routines. Some controls were also placed into the JMASS Team Simulation Engine in order for the JMASS simulation to manage time in accordance with the EPF time management strategy.

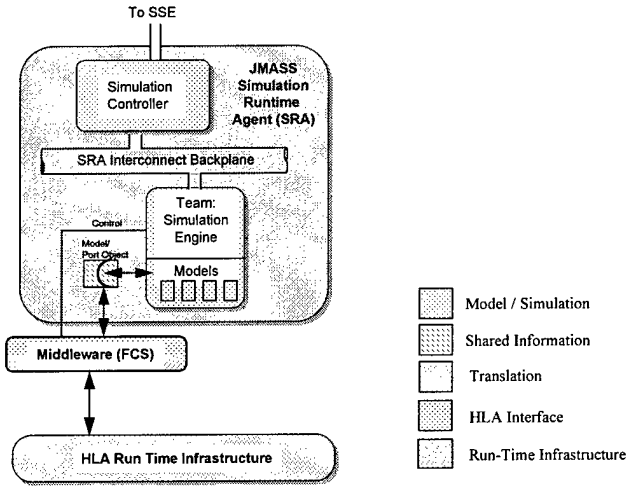


Figure 18 JMASS HLA Architecture

FOM Flexibility in JMASS

Figure 18 shows the JMASS HLA Architecture with the Generic Model overlay. JMASS Port Objects were constructed specifically for the EPF federation data object model, as well as to provide access the EPF Middleware. The Port Objects take care of the sharing of information between the simulation model and the rest of the federation. The EPF Middleware, or Federation Common Software (FCS), served as the interface to the RTI. During the JMASS JPSD experiment, this EPF middleware was modified to provide translation of EPF federation objects into JPSD federation objects. This translation was somewhat straightforward, as the FOMs were very similar. For example, EPF missiles simulated by the JMASS models were translated into JPSD missiles (i.e., the data was largely the same, only the format and units were different).

6.1.6.1.5 CASE 5: Eagle

Distributed Eagle, i.e., Eagle-to-Eagle, was developed to eliminate the single processor constraint of no more than 220 units at a run speed of six to one. Initially, the Aggregate Level Simulation Protocol (ALSP) was used for the Eagle-to-Eagle confederation. Multiple copies of the simulation hosted on multiple processors were connected through the ALSP. Each processor was capable of representing 220 units at six to one run time. Theoretically, an unlimited number of units could be represented using Distributed Eagle. Combat units hosted on each processor appeared as 'ghost units' on all other processors. The 'ghost units' have full interaction with hosted units to include detection, receiving attrition, and causing attrition. In 1996, Distributed Eagle was moved from ALSP and fully implemented under the HLA. The multiple processors now communicate through the RTI. Overhead demand by HLA on the processors is minimal and has almost no effect on run time speed. Current applications include the Joint Precision Strike Demonstration (JPSD). The 1996 version of JPSD ran with 600 units using two processors under HLA. By contrast, the 1995 JPSD was limited to only 200 units using Eagle on a single processor.

EAGLE Implementation

The implementation approach sought to leverage the object oriented design of the Eagle Architecture (**Figure 19**). The Distributed Eagle was modified to use the communication's backbone of the HLA RTI, replacing the existing ALSP interface. The majority of the interface

effort was to modify the Eagle Services, which exist in the Eagle Framework. When the Eagle model is executing as part of a federation, these Services inherit new functionality, or in the case of the Eagle controller, new events. This gives the Services the ability to recognize the need to interface appropriately with the RTI [7]. The HLA Manager (a new Eagle Service) is the key to this interface, providing Eagle with the capability to communicate with outside federates through its interactions with the RTI.

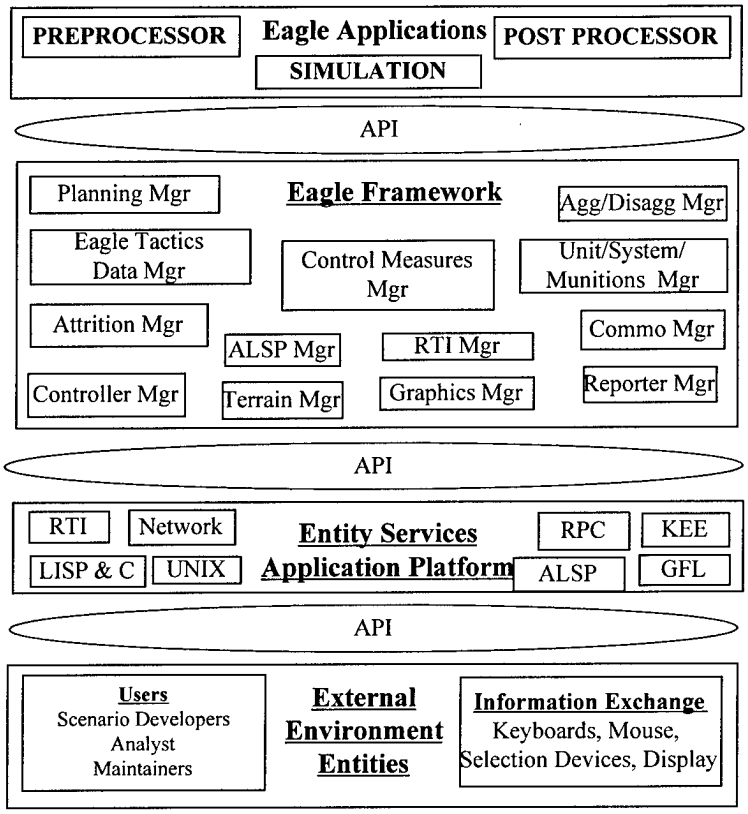


Figure 19 Eagle Architecture

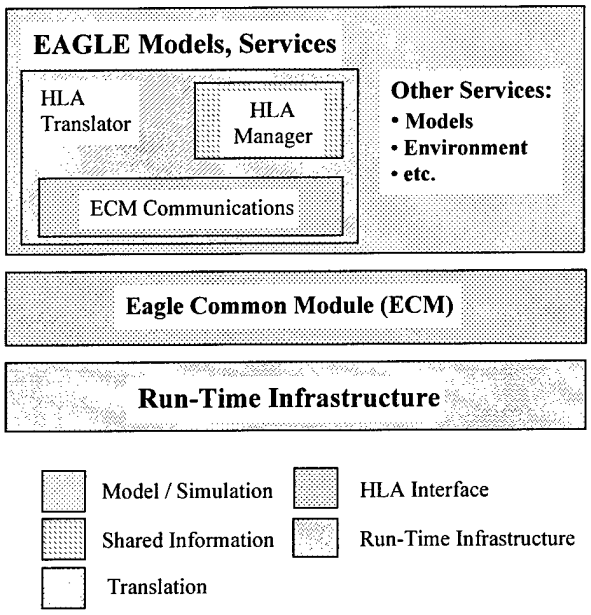


Figure 20 Eagle HLA Interface Architecture

In order to allow Eagle (written in LISP) to work with the RTI code (written in C++), a separate interface was developed to act as a broker between the model's code and the RTI's code (a similar interface was developed for the CCTT interface between CCTT's Ada code and the RTI). For Eagle's interface to the RTI, a separate executable was created written in C++ that brokers the functionality between the RTI and Eagle (**Figure 20**). The interface is called the Eagle Common Module (ECM), which is a separate process running on the same machine as its attached federate.

Eagle requires that causality between events be maintained so that subsequent analysis of the battle outcomes can be validated and verified. To meet this requirement, Eagle implements a Hybrid Time Management scheme which relies on both the notion of continuous time and the projecting of discrete events within a limited look ahead time (for more information please see [8]). To support HLA, Eagle subordinated its time mechanism to that provided by the RTI.

FOM Flexibility in Eagle

Eagle had FOM flexibility in mind when it developed its interface. It was natural that the interface followed Approach 2) because of Eagle's existing object oriented framework. Eagle is able to support multiple federations simultaneously. FOM related changes are localized to the HLA Translator. If new objects need to be represented, new services may need to be added as well, but this requires minimal effort.

6.1.6.1.6 CASE 6: IST Gateway

IST Gateway Implementation

The IST Gateway was originally developed to allow a BDS-D M1 Crewed tank simulator to participate in the HLA Platform Proto-Federation (PPF) experiments. The gateway provided translation capability between DIS or SIMNET and the HLA RTI service calls. The gateway has been redesigned to support DIS legacy simulations in their migration to HLA compliance. This

newly designed gateway is more flexible in its design and currently supports the Real-Time Platform Reference FOM (RPR FOM).

"The Gateway is an adaptation of IST's Computer Generated Forces (CGF) Testbed. The IST CGF Testbed provided a SIMNET and DIS compliant framework for coordinate conversions, dead reckoning, visual displays, and a protocol-independent simulation core. The adaptation of the IST CGF Testbed required the addition of a Gateway Manager and an HLA RTI Interface (**Figure 21**). The Gateway Manager component provides high level support for the translation between the two network simulation architectures. It supports those parts of each network that have no counterpart in the other, such as creating/destroying and joining/resigning federations in HLA or heartbeat PDUs in DIS. The HLA RTI Interface provides the low-level support for the HLA RTI calls in a fashion that insulates the Gateway from the RTI and the HLA FOM implementation details. The RTI Interface follows an object-oriented design that mirrors both the HLA Interface Specification and the target HLA OMT FOM."[2]

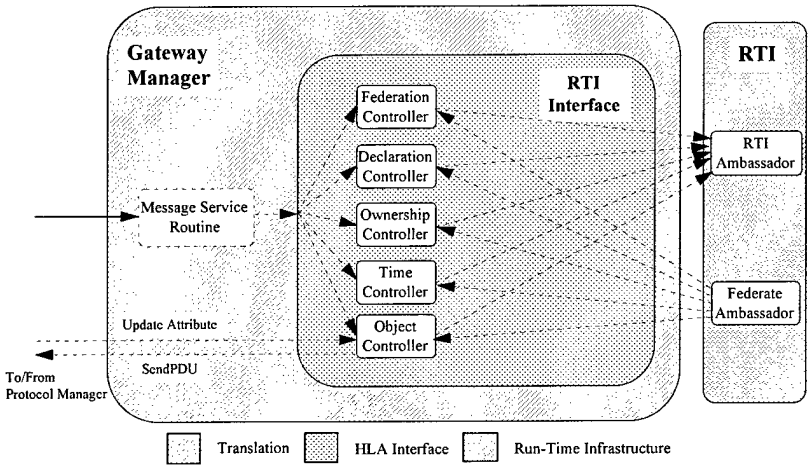


Figure 21 IST Gateway RTI Interface Data Flow

The Gateway's HLA interface essentially has two sides: the RTI Interface and the FOM interface. In the RTI interface, the RTI is represented by an object class with public methods corresponding to the RTI services. That interface does not change at all with FOM changes. In the FOM interface, the FOM is represented by a C++ class hierarchy that corresponds to the object class hierarchy in the FOM.[3] The C++ classes are called "object actors". During a federation execution, when a simulation object (e.g., a tank) is discovered, an instance of the object actor C++ class that corresponds to the tank's FOM class is instantiated. That object actor instance holds the data for the specific instance and either implements or inherits the implementation of the translation routines for the attributes of that class. Changes to the FOM are reflected, almost directly, in the corresponding C++ object actor classes. If a FOM class changes, its corresponding object actor class must be changed in the Gateway code, and no other classes need to be changed (all of this takes inheritance into account).

FOM Flexibility in the IST Gateway

[2] This is an excerpt from reference [10]. Further details can be found there.

[3] This approach is actually an adaptation of the Actor classes used in the ModSAF FCS implementation.

IST's most recent gateway design sought FOM flexibility and achieved it through its object-oriented approach (Approach 2). Adopting the concept of Actor classes introduced in the ModSAF FCS, changes in the FOM would require changes in specific Actor classes. The Gateway also provides the capability to support multiple FOMs. The RTI interface is isolated from such changes.

6.1.6.1.7 CASE 7: Simulation Middleware Object Classes (SMOC)

The Simulation Middleware Object Classes (SMOC) was produced as part of a NAWCTSD project whose objective was to characterize the process of converting DIS-compliant simulators into HLA-compliance. The approach was to utilize a DIS-compliant Pentium-based F-14D simulator running under Windows NT and convert it to HLA. SMOC was developed at NAWCTSD to provide an interface to the RTI. SMOC can be used either as a gateway or as middleware.

SMOC Implementation

The initial SMOC implementation was based on the DIS NIU that was developed under the NAWC-TSD CRDA and used in the NASNET project supporting BFTT EWLAU device as well as the F-14 simulation used as the Testbed for its development. The initial gateway implementation modified the NIU so that it sends and receives DIS PDUs on one side, and HLA attribute updates and interactions on the other side. This implementation was redesigned to capitalize on the benefits of an object oriented design approach while accommodating a degree of reusability, flexibility, and scalability not provided in the initial implementation.

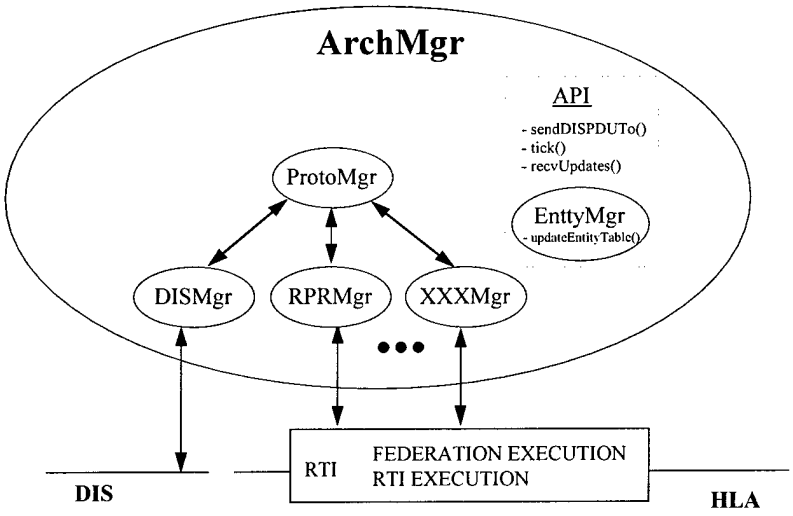


Figure 22 SMOC Architecture

A set of object classes were created that could be configured as necessary to act either as a stand alone gateway or as a layer of middleware for a simulation application to provide seamless integration across DIS and potential variations of HLA FOM environments. The classes were designed to be reusable, adaptable and scaleable. The middleware object classes are show in **Figure 22**. The user interface consists of an architecture configuration data file and an interface that provides the functions such as updateEntityTable (), tick (), sendDISPDU () and recvUpdates ().

The architecture configuration data file provides the user with the ability to specify which "channel" should be used for data transmission. This set up allows the use of multiple channels simultaneously. Operating with a different FOM would only require the development of a new object class for managing the translations for the required FOM type. This new object class would then be added to the configuration data file. In addition, the software can be operated in a gateway mode by adding a line to the configuration data file.

FOM Flexibility in SMOC

SMOC, like the IST gateway, had two tries at its interface. In its current implementation, reusability, flexibility, scalability were a primary concern. The approach combined Approaches 1) and 2). The configuration data file allows the user to specify the FOM or protocol to be used. It also allows the user to select use of the software in either a middleware or gateway mode. The object-oriented design provides the same type of capability as previous implementations; that is, different FOMs require development or modification of object classes.

6.1.7 Lessons Learned – Developer Guidance

Based on the implementations examined in this report, the following guidance can be offered to the simulation developer seeking FOM flexible interfaces:

1. FOM dependent elements of the distributed simulation system should be "grouped" together, organized in such a fashion to facilitate code changes by limiting changes to a minimum set of software modules.
2. FOM dependent functions should be designed to be independent of the Model/Simulation and also of the RTI so that changes in the FOM do not affect the interface to the RTI or Model.
3. Data driven approaches are a simple way to allow for changes without recompiling the software.
4. Object-oriented approaches, which take advantage of inheritance and polymorphism, provide minimal code changes in a localized place and are easily implemented based on a well-defined FOM.

6.1.8 Remaining Issues

A number of issues remain to be considered with respect to FOM flexibility:

1. Many of the applications studied represent DIS legacy systems transitioning to HLA. Other type of systems may have other concerns not addressed here. It does appear that the type of simulation system does not have as much bearing on FOM Flexible interface development, as does the design approach for the HLA interface.
2. Most of the cases examined involved migration from an existing federation to a similar federation. Flexibility here is fairly straightforward. But FOM flexibility may be limited to federations, which are similar in composition and purpose. Federations to be supported which differ significantly in: Domains supported, Levels of simulation resolution, Time Management schemes, etc. may not allow for a flexible approach. The user should examine the feasibility of having such vastly different federations supported by a particular federate.
3. Reconfigurability is an area of interest where FOM flexibleness may be implemented either at run-time (on the fly), code time (limited code changes) or design time (significant redesign/code changes required). Run-time reconfigurabiliy may be difficult and costly (CPU-wise) to achieve but would offer the highest degree of flexibility. Code time

reconfigurability may be approached through data driven (configuration files) or code generation methodologies. This appears to be the most effective approach to date. Design time reconfigurability would only be applicable to new systems and, even then, the design must offer flexibility at code or run time for later reuse.

6.1.9 Additional Information Resources

For more information on FOM flexible interface development please visit the following web-sites:

HLA information – http://hla.dmso.mil
Papers on subject (DIS and SIW archives) – http://www.sisostds.org

POC for the projects featured in this report:

CCTT SAF – Christina Bouwens (Christina.Bouwens@cpmx.saic.com)
ModSAF FCS – Russ Richardson (rrichardson@std.saic.com)
ACETEF – John McMaster (johnm@acetef.nawcad.navy.mil)
JMASS – Brian Beebe (Brian.Beebe@cpmx.saic.com)
Eagle – Jack Ogren (jogren@mitre.org)
IST Gateway – Mikel Petty (mpetty@ist.ucf.edu)
SMOC – Dan Paterson (dpatterson@nawctsd.navy.mil)

~~*Christina Bouwens*~~
~~Science Applications International Corporation~~
~~12479 Research Parkway~~
~~Orlando, Florida 32826-3248~~
~~Christina.Bouwens@cpmx.saic.com~~
~~(407) 207-2742~~

~~*Raymond L. Miller*~~
~~Science Applications International Corporation~~
~~12479 Research Parkway~~
~~Orlando, Florida 32826-3248~~
~~millerr@orl.saic.com~~
~~(407) 207-2785~~

~~*Roy Scrudder*~~
~~Applied Research Laboratory / The University of Texas~~
~~P.O. Box 8029~~
~~Austin, Texas 78713-8029~~
~~rscrudder@arlut.utexas.edu~~
~~(512) 835-3857~~

~~*Robert Lutz*~~
~~John Hopkins University / Applied Physics Laboratory~~
~~Johns Hopkins Road~~
~~Laurel, Maryland 20723~~
~~robert.lutz@jhuapl.edu~~
~~(301) 953-5000~~

7. OBJECT MODEL DATA DICTIONARY SYSTEM (OMDDS) EXPERIMENT

7.1 OBJECT MODEL DATA DICTIONARY SYSTEM (OMDDS) EXPERIMENT REPORT

The development of a Federation Object Model (FOM) is a critical activity within the Federation Development and Execution Process Model (FEDEP) and represents much of the work in assembling a Federation Execution for a particular Sponsor's objectives. To support FOM development, Defense Modeling and Simulation Office (DMSO) has been sponsoring the development of an integrated object model tool suite to support development and management of High Level Architecture (HLA) object models and to provide easy access to the Object Model Data Dictionary (OMDD). The tool suite includes the Object Model Development Tools (OMDTs), Object Model Library (OML), and Object Model Data Dictionary System (OMDDS).

This report is divided into four sections: Section 0 – OMDD Experiment Plan, Section 0 – OMDD Experiment Execution, Section 0 – OMDD Results and Section 0 – Federation Developers Workbook.

1) The OMDD Experiment Plan defines the purpose and goals of this experiment. It discusses the approaches used, the products developed, participants, and assumptions.

2) The OMDD Experiment Execution defines the FOM development according to the Federation Development and Execution Process (FEDEP). It describes the different tools of the object model tool suite used in this experiment. This section contains the details of developing a FOM using the three different approaches, following the guidelines of the FEDEP.

3) The OMDD Experiment Results includes a summary of the experiment, guidance for the federation developer, conclusions, and problems and recommended enhancements to the object model tool suite.

4) The OMDD Federation Workbook contains an execution summary table, host table, LAN table, RTI service table, and object/interaction table for each of the different approaches.

7.1.1 OMDD Experiment Plan

7.1.1.1 Introduction

The methodology for developing object models is an area of particular interest. The initial set of HLA experiments (protofederations) had to develop their own FOMs. Since then, the number of available object models has provided an opportunity for reuse by the rest of the community. The OML provides a repository from which the community can access existing object models. FOM development can be approached in a number of ways. The FEDEP document recommends either the use of existing FOMs or an approach which begins with the Conceptual Model and a standardized Data Dictionary (DD) and allows a federation developer to build their FOM from the "bottom up." Another approach is to utilize "Reference FOMs" which may be characterized as generalized FOMs developed for a particular community or simulation domain but intended to be tailorable for specific simulation needs within that community or domain. This tailoring represents more of a "top-down" approach to FOM development.

Experience has led the developer community to several different approaches to FOM development:

1. Bottom-up: Build the ideal FOM based on the conceptual model and DD as the source, independent of what the individual federates do in their own SOMs.
2. Merge SOM: Get all SOMs from participating federates and merge them, picking out which parts of the SOMs are needed based on the conceptual model.
3. Single SOM: Base case where you begin with a single SOM and add or subtract to obtain your FOM.
4. Similar Federation: Borrow FOM from a previous federation with same characteristics and modify FOM based on federation specific needs.
5. Reference FOM: Begin with a community view for a FOM and add/subtract as needed. Addition should be avoided.

For the purposes of our experiment, approaches 2 and 3 are similar and can be examined by beginning with the SOMs and deriving a common FOM. Approaches 4 and 5 are also similar, except it is assumed that the Reference FOM is more likely to be subsetted where option 4 represents likelihood of adding as well as taking away information.

7.1.1.2 Purpose and Goals

The purpose of this experiment is to examine the FOM development process while exercising the use of the various object model development tools. The experiment will investigate and prototype different approaches to FOM development and their contribution to reuse and interoperability while utilizing the OM tools where appropriate.

7.1.1.3 Experiment Approach

7.1.1.3.1 General Approach

The general approach to this experiment is to step through the FEDEP and OM Process model in the development of a federation execution. We will apply this process in the development of a FOM for a scenario involving CCTT SAF and ModSAF.

7.1.1.3.2 Technical Approach

The technical approach / tasks to perform this experiment include the following:

1. Develop the metrics and methods to address the objectives of the experiment and to report the results.
2. Perform a brief conceptual analysis for the federation to be developed.
3. Develop a scenario that will address the objectives of the experiment, exercise the tools, and assess the use of a common data resource for the development of a FOM.
4. Utilize the OM tools to "prototype" the FOM development process for the various FOM development approaches discussed earlier.
5. Provide an analysis of the use of the FOM development approaches, the utility of the DD and its use with the OMDT.

7.1.1.4 Products

Two products will result from this investigation. The first product will be a briefing that summarizes the findings of our analysis. This briefing will be prepared in time for the February AMG. The second is an informational paper which provides details on the lessons learned and the results of our study. This paper will be prepared in time for the 1998 Spring Simulation Interoperability Workshop. Information in the final paper will include:

Goal	Final Paper
Explore application of FEDEP	Report on lessons learned in use of FEDEP any resulting recommendations
Exercise the FOM development process	Report on lessons learned in the use of the FOM development process and any resulting recommendations
Investigate approaches to FOM development	Report to include: lessons learned in implementation of three approaches and a comparative analysis, and general recommendations concerning top down vs. bottom up approaches
Exercise OM toolset and their use with the Data Dictionary	Report on lessons learned on use of tool and recommendation for future versions

7.1.1.5 Experiment Participants

Participants for this experiment include:

Participant	Role
SAIC Orlando / Chris Bouwens	Experiment lead, CCTT SAF developer
JHAPL / Bob Lutz	OMT Lead
Aegis	OMDT developer
ARL:UT / Roy Scrudder	OM Library / Data Dictionary developer
STRICOM / Susan	COR

Harkrider	
DMSO / Judith Dahmann	Sponsor

7.1.1.6 Experiment Conduct – Assumptions

This experiment plan assumes the following:

1. The Aegis OMDT are available for use by SAIC on 1 November 1997.
2. OMDT developers will provide phone support for use of the tools.
3. Bob Lutz will be available for guidance / advice on the OM development process.
4. ARL:UT will provide phone support for the access and use of the DD and OM Library.

7.1.1.7 Approximate Schedule

Assuming availability of tools and outside schedule constraints, technical tasks listed above would take place from October 15, 1997 – February 28, 1998.

7.1.2 OMDD Experiment Execution

~~*7.1.2.1 1. Introduction*~~

This section describes the execution and results of the OMDD experiment. Included are descriptions of the FEDEP and FOM development processes the tools employed.

~~*1.1.1.1 The development of a FOM is a critical activity within the Federation Development and Execution Process (FEDEP) Model and represents much of the work in assembling a Federation Execution. A good understanding of the development process and available tools provides the federation developer with the means to develop a FOM that is well designed for federation use and reusable for other federation executions.*~~

~~*1.1.1.1*~~

~~*1.1.1.1 This paper discusses the FOM development process, its place within the FEDEP and the tools used throughout. It also provides information on the various tools used during the FOM development process. The OMDD experiment is discussed along with its results. The paper concludes with some developer guidance and references for where to get more information for this process.*~~

7.1.2.2 FOM Development ~~And~~and The Federation Development ~~And~~and Execution Process

The FOM development process is part of a larger process called the FEDEP. The FOM development process receives valuable inputs from the FEDEP and provides critical output to the FEDEP. It is important to understand the place of the FOM development process within the FEDEP.

7.1.2.2.1 FEDEP Process

The FEDEP document [1] describes a recommended process for development and execution of an HLA federation execution. The five~~-step process~~step process recommended in FEDEP v1.1 is shown ~~in~~ in **Figure 23**, and is summarized below:

Step 1~~: The~~: The federation sponsor and federation development team must define and agree on a set of objectives, and document what must be accomplished to achieve those objectives.

Step 2: ~~A:~~ A representation of the real world domain of interest (entities and tasks) is developed, and described in terms of a set of required objects and interactions.

Step 3: ~~Federation:~~ Federation participants are determined (if not previously identified), and a FOM is developed to explicitly document information exchange requirements and responsibilities.

Step 4: ~~All:~~ All necessary federation implementation activities are performed, and testing is conducted to ensure interoperability requirements are being met.

Step 5: ~~The:~~ The federation is executed, outputs analyzed, and feedback provided to the federation sponsor.

It is critical Steps 1 and 2 are accomplished before FOM development begins as part of Federation Design.

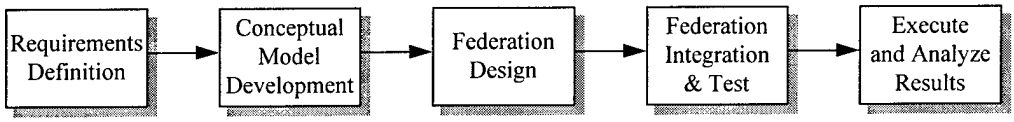


Figure 23 Five-Step Process

7.1.2.2.2 Description of the Federation Object Model Development Process

A full description of the OM development process may be found in [2]. The process is summarized below:

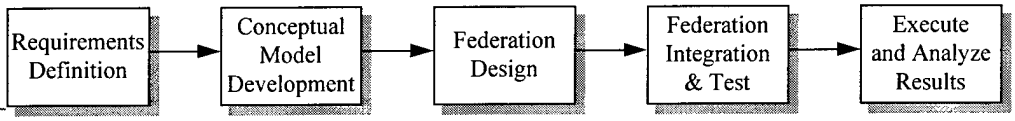
7.1.2.2.2.1 Summary of FOM Development Approaches

There are several different fundamental approaches to the development of HLA FOMs. In addition to bottoms-up approaches, in which one builds the FOM from "darkness" based on the federation conceptual model and the Object Model Data Dictionary (OMDD), other approaches may be appropriate which provide more emphasis on the reuse of existing object models. Possible alternative approaches include:

- Merge together Simulation Object Models (SOMs) of participating federates, removing aspects of SOMs that do not apply to the domain of interest.
- Begin with SOM that most closely aligns with desired FOM, remove aspects of that SOM which do not apply to the domain of interest, and merge-in elements of other SOMs to fully represent domain.
- Begin with FOM from previous, but similar application. Modify and/or augment as required.
- Begin with Reference FOM. Remove elements of Reference FOM that are not required for the immediate application. Modify and/or augment only if necessary.

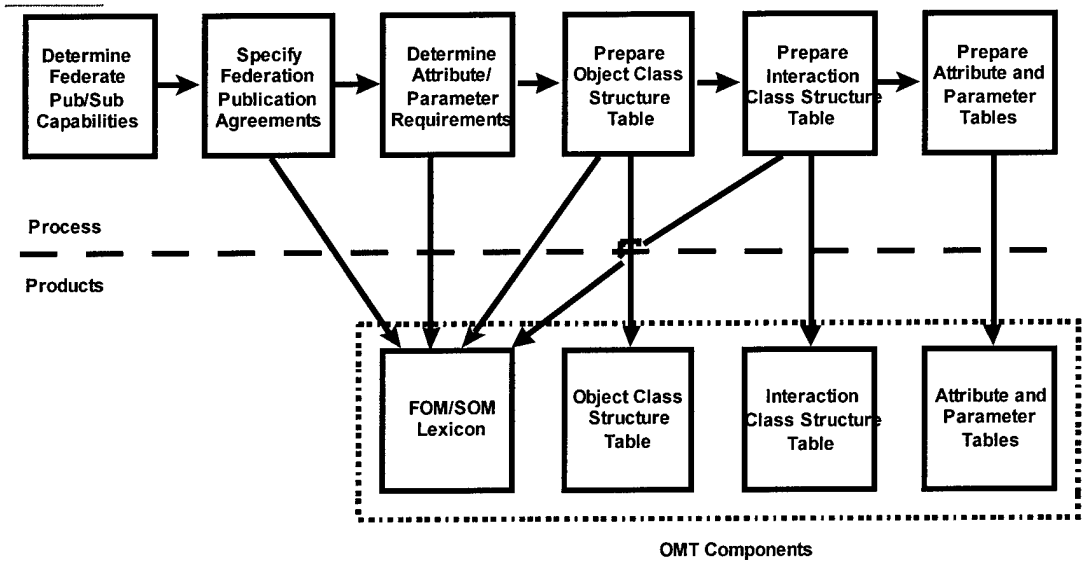
The starting point for the HLA FOM development process is the Federation Development phase of the HLA FEDEP model. Implicit to this assumption is that the federation requirements have been clearly delineated, the scenario(s) have been defined, the real world objects and interactions

that are to be represented explicitly in the federation have been identified (and, where possible, mapped to OMDD entries), and the federation participants have been both identified and fully described according to an HLA SOM.



~~Figure 2-1 Five Step Process~~

~~1.1.1.1.1.1~~



~~1.1.1.1.1.1 Figure 2-2 FOM Development Process~~

~~1.1.1.1.1~~

FOM Development Process

While each of the above FOM development approaches may represent a somewhat more efficient FOM development strategy (relative to starting entirely from scratch) under certain circumstances, all will require some use and appropriate tailoring of the essential activities described in the current HLA Object Model (OM) Development Process [2]. A summary of these activities is provided in **Figure 24**. Federation security personnel must always maintain knowledge of any classified information associated with applicable entries in each federates ~~SOM,~~SOM and the implications when this data is combined into a single FOM.

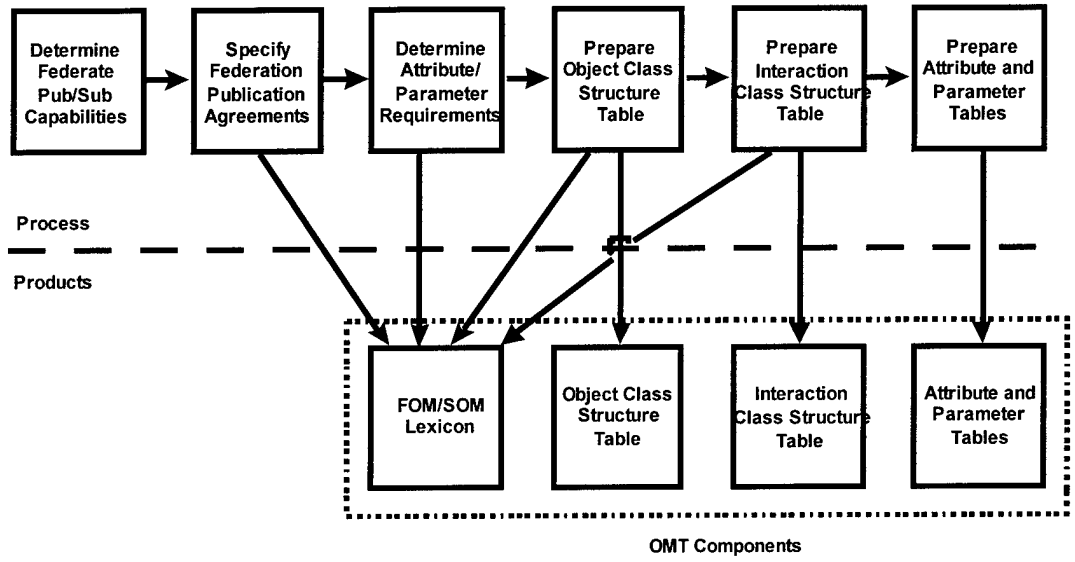


Figure 24 FOM Development Process

The use of automated tools to facilitate the object model development process is strongly encouraged and the subject of this paper. The HLA Object Model Library (OML) provides user access to libraries of reusable object models which can be used either as a starting framework or as individual "piece parts" for a new FOM. In addition, Object Model Development Tools (OMDTs) may be used to modify or extend an existing object model, or to build an entirely new object model from scratch. Other OMDT features include consistency checking, syntax checking, Federation Execution Data (FED) file generation, external interfaces to commercial object model development tools, and an on-line users manual.

In addition to FOM development, there are other important activities that take place in this phase of the FEDEP. Negotiations must take place among federation members and agreements reached regarding supporting resources that will be common across the federation. For instance, decisions must be made regarding any databases (e.g., environmental) or algorithms (e.g., line-of-sight) that must be common across all federation participants. Another important activity is to transition the functional description of the scenario (developed in an earlier phase) to an actual

scenario instance (or set of instances) based on authoritative data sources. The output of this last activity permits federation testing to be conducted directly within the context of the domain of interest, and also drives the execution of the federation later in the FEDEP.

~~*1.1.1.1*~~

~~*7.1.2.3 3*~~***Object Model Tool Suite: Description of the Tools and Their Use***

From the early days of HLA experiments with the protofederations, the need for automated tools to support the Federation Development and Execution Process (FEDEP) has been evident [3]. To meet this need, the Defense Modeling and Simulation Office (DMSO) sponsored the development of the HLA Tools Architecture, which defines a core set of tools to support the FEDEP and interfaces between these tool sets [4]. Also under DMSO sponsorship, an initial suite of HLA support tools has been developed, specifically to support the ~~object-modeling~~object-modeling phase of the FEDEP~~[1]~~. Components of the initial Object Model Tools Suite and their information exchange relationships are shown in **Figure 25**~~3-1~~.

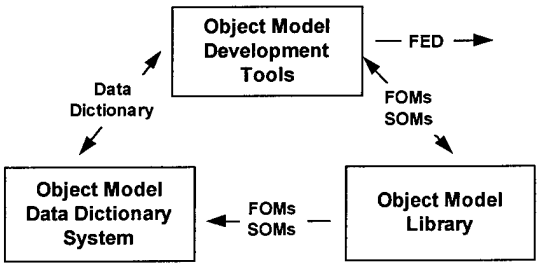


Figure 25 Object Model Tools Suite

The Object Model Development Tool used in the OMDD Experiment was developed by Aegis Research. It is a Microsoft Windows 95/NT 4.0 application which allows users to create and edit FOMs and SOMs compliant with the Object Model Template [5]. The core set of functionality includes data entry and modification, syntax checking, automated access to the Object Model Library and the Object Model Data Dictionary System, and automated generation of Federation Execution Data necessary for the initialization of an HLA Run Time Infrastructure.

The interface between the OMDT and the OMDDS is unique to the version of the OMDT used for the OMDD Experiment. The version of the Aegis OMDT released to the HLA community in October 1997 does not have this feature. The ~~OMDT's~~OMDTs interface to the OMDD allows users to import one or more data dictionary files and then pick from the contents of the data dictionary files when populating or modifying an object model. Data dictionary choices are provided in a "pick list" fashion, within the context of the object model table a user is editing. For example, when a user is editing the object class table of a FOM, only the object classes exported from the OMDD are available for inclusion in this table, not other components such as complex data types or interaction classes.

The Object Model Library is a central repository of reusable FOMs and SOMs. It is available through a Web-based interface and provides mechanisms to search and browse object models in the library, download copies of object models, and submit new object models for inclusion in the library.

The Object Model Data Dictionary System (OMDDS) is the newest component of the Object Model Tools Suite. It is a central repository of reusable components for the construction of FOMs and SOMs and is available through a Web-based interface. The OMDDS provides users with the capability to browse and search OMDD components and select subsets for export and use with the OMDT. User selections for export are maintained persistently, so a user can build up a list of items to export from the OMDD over the span of multiple web access sessions.

The OMDD contents accessible through the OMDDS are separated into five component types: object classes, interaction classes, generic elements, complex data types, and enumerated data types. All of these component types, with the exception of generic elements, map directly to Object Model Template concepts. Generic elements correspond to both attributes and parameters in the Object Model Template. In effect, both attributes and parameters are the same type of component, i.e., a property of a class. The only distinction between attributes and parameters is whether they are a property of an object class or of an interaction class. A generic element from the OMDD (e.g., velocity) could be used as either an attribute or a parameter. Each generic element in the OMDD has one or more representations which specify a unit of measure (if applicable) and a data type. Data types for generic elements may be any of the base data types from the ~~OMT or~~OMT or one of the enumerated or complex data types also contained in the OMDD. For complex data types, the OMDD specifies a list of fields and the corresponding generic element for each field.

OMDD contents available at the time of the OMDD Experiment were developed in a bottom-up fashion based on components found in three widely used FOMs:

- Real-time Platform Reference FOM – A FOM widely used in the transition of legacy systems from the Distributed Interactive Simulation environment to HLA.
- Joint Training Confederation FOM – A FOM developed ~~for the~~for the transition of legacy systems from the Aggregate Level Simulation Protocol environment to HLA.
- Engineering Federation FOM – A FOM evolved from the Engineering Protofederation ~~experiment which~~experiment that has been used in the federation of several engineering-level simulations.

Components from these FOMs were merged into a common set of OMDD components. These OMDD components expanded upon the definitions and representations specified in the FOMs by using information from authoritative sources including the Defense Data Dictionary System and the JCS Pub 1-02, Dictionary of Military and Associated Terms.

7.1.2.4 Object Model Development Experiment

~~1.1.1.1.1~~

~~1.1.1.1.1 The OMDD experiment sought to prototype the FOM development approaches discussed in Section 2.2.1 utilizing the tools in the OM Tool Suite. The experiment approach was to step through the FEDEP and the FOM Process Model, prototype FOMs using several different approaches, and provide feedback and lessons learned concerning the processes and tools employed.~~

~~*7.1.2.4.1 4FEDEP Preliminaries*~~

Before prototyping the FOMs, it is necessary to go through the first two steps of the FEDEP. These define the problem space for which the FOMs will be built.

~~7.1.2.4.1.1~~ ~~4~~Requirements Definition

As part of Requirements Definition the project team developed the following **Problem Statement**:

> With the proliferation of SAF systems, there is a concern in the community that the resulting simulations may not be "fair" due to varying modeling techniques and resolution. The problem is how to determine whether or not the fight is fair. This begins with the determination of whether the results of a particular engagement are the same for two different SAF systems.

For this particular experiment, ModSAF and CCTT SAF will be utilized at platform-level resolution. Proposed measures of merit (MOM): ~~Timing:~~ Timing at various points of the engagement, results of the engagement.

A set of **Objectives** were then developed for the experiment:

1. Develop a scenario that will allow opposing forces to engage in an attack and defense situation. This scenario must be usable by SAF.
2. Run the scenarios in a variety of configurations of the SAF systems to examine the systems individually as well as engaged against one another for comparison.

The **Environmental Context** for the experiment would be the use of the Grafenfels database (normal weather and environment conditions). Hasty Attack and Hasty Defense with M1 and T72 tanks supported by Hinds and Apaches were examined.

There were no **Security Issues** for our experiment. The experiment would be conducted as Unclassified.

The **Equipment, Facilities and Data** to be used ~~is~~are shown in ~~Table 4-0-1~~**Table 8**.

Table 8 Equipment & Facilities

Federates	ModSAF and CCTT SAF
Facilities	SAIC SAFLab utilizing four Motorola machines, one Sun Ultra workstation and one SGI Indy2
Terrain Database	Grafenfels
RTI	Version 1.0, release 2
Other tools to be used	OMDDS, OMDD, OMDT, OML, MSRR, Federation Developers Workbook, HLA documentation

No **VV&A** were to be performed for this experiment. **Test Plans** included unit testing for interfaces and consistency testing for developed object models.

~~7.1.2.4.1.2 4~~Conceptual Model Development

Once the requirements for our federation are identified, the next step is to develop the Conceptual Model. This part of the process includes Scenario Development and the Conceptual Analysis.

~~4.1.2.1~~ Scenario Development

The **Scenario Development** process began with the search for existing scenarios that could be reused for the experiment. The use of scenarios from the A2ATD experiment was explored. The experiment borrowed two scenarios, slightly modified, for the purposes of the experiment. It should also be noted that the problem being addressed by the experiment differed from that of the original experiment. ~~The Modeling~~The Modeling and Simulation Resource Repository (MSRR) was also searched for other possible scenarios, but no specific scenario could be identified for our experiment. The selected scenario was defined as follows:

1. Hasty attack involving a company of BLUFOR (M1s) and a Platoon of OPFOR (T72s) with close air support provided by Apaches (for the BLUFOR) and Hinds (for the OPFOR).
2. Hasty defense involving a company BLUFOR (M1s) and a Bn of OPFOR (T72s).

The scenario requires the use of M1s, T72s, Apaches and ~~Hinds which~~Hinds, which would~~ill~~ need to move, shoot and take damage during the scenario. No filtering requirements (DM) were defined at the time of the scenario development.

Both scenarios will be run using the following conditions: 1) CCTT Blue SAF against CCTT Red SAF, 2) ModSAF Blue SAF against ModSAF Red SAF, 3) CCTT Blue SAF against ModSAF Red SAF, and 4) CCTT Red SAF against ModSAF Blue SAF. To obtain a good sampling, each scenario with each condition will be run five times for a total of forty runs.

~~4.1.2.2~~ Conceptual Analysis

Class Analysis

Based on the scenario described above, the simplest class structure to support our experiment was developed for our Conceptual Model. The Conceptual Model was developed without regard to the federates participating in the federation. The goal was to create a Conceptual Model that met the requirements of the scenario. Federate specific details would be addressed during Federation Design. A rough model was developed which represented tanks and helicopters as primary classes.

Table 9 shows the starting point of Conceptual Analysis.

Table 9 Planned Classes & Interactions

CLASSES

Tank (PS)	M1 (PS)
	T72 (PS)
Aircraft (PS)	Apache (PS)
	Hind (PS)

INTERACTIONS

Fire (IR)
Detonation (IR)
Collision (IR)

Once these elements were identified, the OMDDS was utilized to access the class elements in the OMDD. During this process, elements of the rough model were used to search and browse through the OMDD (using the OMDDS) and selected elements which supported the Conceptual Model. These elements were exported from the OMDD and brought into the OMDT, which was used to document the final Conceptual Model. The resulting OM developed with the OMDT is called the "Ideal FOM." Our Conceptual Analysis yielded the following Class structure:

IDEAL FOM CLASSES

Military Platform (PS)	Military Ground Vehicle (PS)
	Rotary Wing Aircraft (PS)

The type of ground vehicle and RWA would be captured within the attributes through enumerated types.

It should be noted that when browsing the OMDD, multiple elements were identified that could be used to support the Conceptual Model. For example, Apache and Hind aircraft were found under Rotary Wing Aircraft and under Aircraft. Selection of either element would have satisfied our requirements.

Interaction Analysis

Three interactions were identified for our federation: ~~Fire:~~ Fire Weapon, Detonate Munition and Collide (vehicle). These three interactions were found in the OMDD and exported to our Ideal FOM:

INTERACTIONS

Fire (IR)
Detonate (IR)
Collide (IR)

Attribute / Parameter Analysis

Although a minimum set of attributes and parameters had been identified during the conceptual analysis, the final selection of parameters would be related to the particular FOM development approach being explored and some basic assumptions about the systems to drive the tradeoffs. For the purposes of our conceptual model, we included the minimum attribute set as a starting point. Other related attributes / parameters were identified and brought into the Ideal FOM during its development. Analysis of specific requirements would come during the FOM development process.

For the various classes, minimum attributes included: ~~type:~~ type, location, velocity, appearance (state), side (Blue or Red). For the three interactions, the following minimum parameters are defined:

	Minimum Required Parameters
Fire	-who fired -what was fired (F) when was it fired - intended target
Detonate	(F) where detonated

	(G) who was affected (H) type of round (what) (I) corresponding fire event - resulting appearance
Collision	(F) who collided - resulting damage (appearance)

Using the requirements above, elements in the OMDD were identified and exported to our OMDT for inclusion in the Ideal FOM.

The Conceptual Model is now complete and documented using the OMDT in conjunction with the OMDD. Our product is an Ideal ~~FOM which~~FOM, which will be used as a starting point for our bottoms-up approach and reference for other approaches.

~~1.1.1.1.1.1~~

~~7.1.2.4.1.3 4~~Federation Design

Federate Selection

Although the plan was to use CCTT SAF and ModSAF for the experiment, the Object Model Library (OML) was checked to identify any other ~~any~~ federates that might qualify for the experiment. Once the Conceptual Model was completed, the project team logged into the Object Model Library (OML) to other federates (in the list of SOMs) which had the kind of capabilities needed for our experiment. Our two federates' SOMs were found in the OML along with one FOM to be used as our starting point for the Reference FOM approach: ~~CCTT~~: CCTT SAF SOM, ModSAF FCS SOM and ~~the RPR~~the RPR FOM. These Object Models were easily downloaded from the OML to our local PC.

Although the class structures were quite different between the SOMs (and even with the RPR FOM), the leaf nodes proved to be nearly identical. This was due, of course, to the fact that the three object models had their legacy in DIS.

Because of the capabilities of the federates, it was~~we~~ determined that both would subscribe and publish to all the objects identified in the scenario except for the RWA. For the Apache and Hinds, CCTT would simply subscribe and not publish. ModSAF would publish and subscribe.

~~1.1.1.1.1~~

~~7.1.2.4.2 4~~*FOM Prototype Development*

As discussed in Section 0, ~~Summary of FOM Development,~~ Summary of FOM Development Approaches, there are a number of approaches to FOM development. These approaches focus on whether the developer begins from scratch (bottoms-up approach) or begins with an existing FOM or SOM. Since several of the approaches are similar in their implementation, we reduced the number of cases prototyped to three:

1. Bottoms-up Approach
2. Single SOM / Merge SOM Approach
3. Similar FOM / Reference FOM Approach

The following sub-paragraphs discuss the manner in which each approach was implemented along with the results of that approach.

It should be noted that a number of tradeoffs drive the selection of certain attributes, parameters, class structure, etc. Such decisions typically take place during federation design meetings where all involved federate developers coordinate the choices. For the purposes of this experiment, we made some simple assumptions regarding our requirements.

~~7.1.2.4.2.1 4~~Bottoms-Up Approach

The Bottoms-Up approach to FOM development begins with the Ideal FOM developed during Conceptual Analysis. For this approach, the SOMs for the federates to participate would be used to augment the FOM with additional details. When new elements are added to the FOM, the OMDD is consulted and its terminology adopted for the FOM. This approach did not take into consideration the impact that FOM changes ~~that~~ would have on federates in order to support the FOM structure.

Class Structure

This approach involved beginning with the Ideal FOM ~~and modifying~~and modifying it as required to support the experiment with the selected federates. The result was a slightly modified FOM from the Ideal FOM with the addition of one level to the class hierarchy in order to support munitions. This was based on the need to add a class for munition entities to support Fire and Detonation Interactions. We also changed the RWA subclass to Aircraft to be more consistent with the generality of Military Ground Vehicle. The Class hierarchy for developing the Bottoms-up FOM is now:

Physical Entity	Military Platform	Military Ground Vehicle
		Aircraft
	Munition	

Interactions

Utilizing the SOMs / FOMs we had downloaded and selected from the OML, an analysis of the Interactions was performed. Analysis was very simple since both federates supported essentially the same interactions. The results of the analysis are included below:

Table 10 Interaction Mapping between Federates and Conceptual Model

Interaction	Ideal FOM	CCTT SAF	ModSAF FCS	RPR FOM

Fire	Fire	Weapon Fire or Weapon Launch	Fire	Fire
Detonation	Detonate	Detonation	Detonation	Detonation
Collision	Collide	Collision	Collision	Collision

Both participating federates would Interact /React to all three interactions.

Attributes / Parameters

For Attributes selection in this FOM development approach, the project team chose to favor the CCTT specifications but added Angular Velocity. Guise related attributes and Articulated Parts found in the ModSAF SOM would not be supported.

For the most part, the federates and the RPR-FOM agreed on the attributes and parameters that should be included for the objects and interactions to be used since their legacy was DIS. There were some subtle differences. One particular difference is the information included in CCTT SAF's class structure is found in ModSAF FCS attributes. To explore the various FOM approaches, we made some assumptions about the federate information to be used. This is pointed out throughout the FOM development descriptions in this section.

The following observations resulted from this approach:
1. Having the Conceptual Model expressed in the Ideal FOM gave us an excellent starting point for FOM development.
2. Utilizing elements from the OMDD made assembling the FOM easier. All elements of the FOM were found in the OMDD.
3. The resulting FOM did not take into consideration the ~~change~~changes that may be required ~~of the~~of the participating federates.

~~7.1.2.4.2.2 4~~Merge / Single SOM Approach

For the Merge / Single SOM approach, the starting point for FOM development is an existing SOM. The one SOM is modified according to the requirements defined in the Conceptual Model and is augmented using information from other SOMs. In the case that information is not available in the SOMs, elements of the Ideal FOM or the OMDD would be used to augment the FOM. For the experiment this approach was performed twice. The first was based on using CCTT SAF SOM as the baseline with modifications made to accommodate the federation and ModSAF FCS capabilities. The second FOM used the ModSAF FCS SOM as the starting point. Our assumptions for modification were:
- Angular velocity needed to be supported
- Appearance would be expressed as a series of enumerated types (as opposed to a bit encoded value)
- Class hierarchy would match the SOM used as a starting point

This approach proved to be more difficult than other FOM development approaches for this particular experiment because of the differences in SOM hierarchy. Identifying attributes in one SOM that mapped to classes in the other was not ~~straight forward~~straightforward.

The following observations resulted from this approach:
1. The modifications on the base FOM (SOM used for starting point) were few. This can be attributed to the similarities between federates (their DIS legacy).
2. The resulting FOM did not have OMDD elements for its representation since its source consisted of existing SOMs.

3. The FOMs resulting from the two approaches were as different as the SOMs they were based on. No assessment was made as to which FOM might be better or worse to use. As an expression of the Object Model they were equally valid. As a point of departure for federate implementation, a particular FOM may be desirable based on which FOM is more easily supported by the participating federates.

~~7.1.2.4.2.3 4~~Existing FOM / Reference FOM Approach

For the Existing FOM / Reference FOM Approach, the starting point for FOM development is an existing FOM. The FOM is modified according to the requirements of the Conceptual Model, primarily through the removal of elements that are not required for the target federation. Any elements required but not found in the existing FOM ~~are~~were obtained from the Ideal FOM (or the OMDD). The use of an existing FOM which closely matches the federation being assembled represents the easiest approach for FOM development, especially if the participating federates have implemented the existing FOM in past federation executions. If ~~an existing~~ FOM does not exist that is relevant to the target federation, the use of a "reference" FOM has also been explored. The Real-time Platform Reference FOM (RPR FOM) has been developed as a starting point for DIS-based simulation for developing their FOMs. The experiment used the RPR FOM as its starting point for this FOM development approach because of the use of DIS-based systems.

The following observations resulted from this approach:

1. Although the approach resulted in a relatively large number of changes to the baseline FOM to get the target FOM, these changes were primarily ~~deletions which~~deletions, which are quickly and easily accomplished in the OMDT.
2. Because the RPR FOM served as a basis for the OMDD elements, the resulting FOM was largely similar to the Ideal FOM. The Reference FOM approach resulted in a deeper hierarchy, but once again the leaf nodes were the same. This approach, however, may result in FOMs with an artificially deep class hierarchy.
3. The depth of the hierarchy made the removing of the classes and interactions very easy without destroying the integrity of the FOM. This is something that could not be done with a flat hierarchy.

7.1.3 OMDD Experiment Results

7.1.3.1 ~~5.~~ Experiment Summary

The following is a summary of the results of process and tool use for our experiment.

7.1.3.1.1 Process Summary

The FEDEP provided the project team with an easy to follow and complete ~~process which~~process, which provided guidance which allowed for the development of a FOM. The products of the early part of the process were critical in providing FOMs that met the objectives of the federation development.

The Object Model development process provides an easy to ~~follow.~~follow and more detailed approach to FOM development. Although geared for a Bottoms-up approach, the same considerations must be made to support any of the other FOM development processes.

7.1.3.1.2 Object Model Tool Suite Implementation

The HLA OMDD Experiment has provided the first opportunity to test the entire object model tool suite in an integrated fashion. Prior experiments and object modeling experiences concentrated only on the use of the Object Model Library and the Object Model Development Tools. The OMDD Experiment provided the first use case where OMDD contents were used to develop a new FOM.

The OMDD Experiment was also structured to provide early feedback about the OMDT, OMDDS, and OMDD in several key areas. First, the experiment provided a forum for gauging the applicability and usefulness of the OMDD contents. The proportion of elements in the FOMs developed in the experiment that~~which~~ were taken directly from the ~~OMDD~~OMDD is a key performance measure of OMDD usefulness. Second, the experiment provided the first practical use of the OMDDS user interface for identifying and exporting OMDD components for the FOM or SOM creation. The experiment required exercising the full range of OMDDS capabilities—browsing and searching the contents and exporting a set of components for use in FOM creation. Finally, the OMDD Experiment provided valuable information about the integration of the OMDDS with the OMDT and the OMDT user interface for accessing and including OMDD data in an object model.

Table 11 provides a summary of how the tools were used in the FOM development process.

The object model tool suite provided an invaluable resource for development of our FOMs. It reduced the time for FOM development by providing a means to reuse existing SOMs, FOMs and OM elements without the need to manually enter the desired information.

Table 11 Summary of Object Model Tool Suite Use

	Tool		
Process / Experiment	OML	OMDDS/OMDD	OMDT
Conceptual Analysis		√	√
Federation Design	√		
Federation Development	√	√	√
Select Baseline OM	√		

Changes to Baseline		√	√
Addition of elements		√	√
Bottoms-up Approach		√	√
Merge/Single SOM Approach	√	√	√
Existing / Reference FOM Approach	√	√	√

7.1.3.2 Guidance ~~For~~for Federation Developers

This paper explored several methods for development of FOMs utilizing a suite of tools that have been developed for the modeling and simulation community. Based on the results of our work, the following guidance is offered for federation developers:

GENERAL GUIDANCE – TOOL AND PROCESS USE

1. A federation development process such as the FEDEP should be utilized as a guide for ensuring that all relevant information is captured early in the development process. The detail at which such a process should be followed will depend on the particular federation being assembled. In smaller experimental situations (such as that featured in this paper) the process takes only a day or so and allows the developers to completely define the problem space. In the case of a larger federation development, the process could take weeks or even months depending on the requirements of the federation.
2. Use of the OMDT early in the process (during conceptual analysis) provides an easy way to clearly define the Conceptual Model and provides reuse for Bottoms-up FOM development efforts.
3. Consulting various resources documenting past developments as well as the use of the OML could save time and effort in federation development by providing the user with a starting point for federation design and development.
4. A key to reuse is the OMDDS used in conjunction with the OMDD. Utilizing a common resource for OM development (both FOM and SOM) helps to build consistency across a community of users.

FOM DEVELOPMENT - METHOD AND TOOLS

Table 12 provides some guidelines on which FOM Development approach should be employed and which tools are involved:

Table 12 FOM Development Approach Guidance and Tools

FOM Development Method	When to Use	Tools to Use
Bottoms-Up	(F) Capturing the Conceptual Model (G) Development of a Model/Simulation's SOM	OMDDS / OMDD OMDT

	(H) Standardized data use is desired - No existing FOMs/SOMs can be utilized	
Merge SOM	(F) Focus is on meeting needs of a few federates (G) Number of participating federates is small - Federates are similar in purpose and domain	OML OMDDS / OMDD OMDT
Single SOM	(F) One federate dominates the federation (others play minor role) (G) Number of participating federates is small	OML OMDDS / OMDD OMDT
Existing FOM – Similar Application	(F) Federation under construction has similar purpose / objectives to existing federation - Federates participating are same as in previous federation	OML OMDDS / OMDD OMDT
Reference FOM	(F) Participating federates are similar in function and domain supported - Large number of federates participating	OML OMDT (rarely – OMDDS / OMDD)

7.1.3.3 Conclusions

The FEDEP provided a useful guideline for capturing the information relevant to federation design. Its use did not require an excessive amount of time for the development of the federation in the subject experiment. The object model tool suite was a critical resource for quickly developing our FOMs and served to be very useful in Conceptual Analysis and in Federation Design. These easy to use tools saved a tremendous amount of time in the definition and design of our federation.

Many issues remain with regard to FOM development. These include:

1. Reference FOMs
2. Building federations with disparate federates.
3. The availability of existing FOMs/SOMs and information on how they have been used to assist the developer on identifying similarity of objectives.
4. FOM flexible federates and how FOM development time may be reduced.

~~1.~~

7.1.3.4 Problems and Recommended Enhancements

This section describes some technical problems encountered along with some recommended changes to enhance the tool or to make the tool more user friendly. Recommended enhancements are not problems that need to be fixed but changes that would allow the tool to be more user friendly and/or to better assist the user.

7.1.3.4.1 OMDT

7.1.3.4.1.1 Problems

The OMDT was used to develop the FOMs and SOMs in this experiment. The tool is very useful in collecting, organizing and displaying, the classes and interaction along with their associated attributes and parameters. The OMDT successfully imported data from both the OMDDS and the OML. During the course of the experiment, the following system problems were encountered with the OMDT:

1. When trying to copy attributes from an OMD attribute table to a different OMD attribute table, the following system errors occurred:
   OMDT caused an invalid page fault in module FOM.DLL at 014f:10007d89
   OMDT caused an invalid page fault in module MFC42.DLL at 014f:5f40129c
   This error will cause all windows to close and any changes made to be lost.

2. While trying to copy attribute(s) or parameter(s) from one class to another class on the same OMD attribute or parameter table, the following system error occurred:
   OMDT caused an invalid page fault in module KERNEL32.DLL at 014f:bff858f1
   Further investigation revealed that inserting data at the top of a list will NOT cause this error to occur, but inserting data anywhere else in the list will cause this error to occur. This error will cause all windows to close and any changes made to be lost.

3. Trying to load the RPR FOM (from the OML) into the OMDT (BETA version 1.1 build 8) the RPR FOM did NOT load and the following error message was received in the OMDT Status window:
   Error 5001: Syntax error…Line 2676: Unexpected token = "X"
   But the same RPR FOM loaded successfully using the old OMDT (BETA version 1.1 build 5).
   Further investigation revealed that the complex data type called POSITIONSTRUCT has field names of X, Y and Z. Changing the field names to two characters or larger solved the problem, for example Pos_X, Pos_Y, and Pos_Z. It seems that the field names for complex data types must be two or more characters.

4. When the attribute table is closed before deleting attributes or classes from the class table, an attempt to reopen the attribute table results in the following system error:
   OMDT caused a stack fault in module <unknown> at 0000:00000003
   Further investigation revealed that if the attribute table is left open while deleting any attributes or classes from the class table no system error occurs. This error will cause all windows to close and any changes made to be lost.

7.1.3.4.1.2 Recommended Enhancements

It is recommended that the duplicate items be removed from the different data dictionary windows (CLASS, INTERACTIONS, etc.), this would make it less confusing to the user.

7.1.3.4.2 OMDDS

7.1.3.4.2.1 Problems

The OMDDS is a very useful tool. The browse function was outstanding but the search function was lacking. The data was successfully exported from the OMDDS and imported into the OMDT. During the course of extracting component elements from the OMDDS the following problems were encountered:

1. It was noticed when moving from one search listing to another (for example: A-E to F-L) all previous checked items to export were lost.

2. When the system is slow and the export of data takes more than 10 minutes, the message 'Document contains no data' is displayed and the export aborts.

3. The help button on the export area does not work.

4. It was discovered that the generic elements identify the associated enumerated data types but the enumerated data types did not identify the associated generic elements. For example, the Apache helicopter was found, using the search routine, in the enumeration data type called aircraft_type_enum. To find the associated generic name for aircraft_type_enum a search was performed with the generic element, enumerated data type, and name boxes checked and the word 'type' entered into the search field. This displayed the enumerated data type, aircraft_type_enum and its associated generic element, aircraft_type. This works in most cases but not all. For example, the M1 tank was found in the enumerated data type, ground_equipment_enum. The type search did not display a ground_equipment_type as expected, but the associated generic element was found to be equipment_type. It is recommended that all associated generic elements for a enumerated data type be listed.

5. While searching for helicopter, the tool found and displayed Rotary Wing Aircraft (RWA). This may have been correct, but a wider selection was expected, for example, Military Aircraft, Civil Aircraft, Military Air Vehicle, Civil Air Vehicle, Aircraft, Military Platform, and Civil Platform to name a few. While browsing the classes, Aircraft was found and the note indicates that it includes RWA, but RWA was not included in the synonyms. Civil Platform was also discovered and the note contained aircraft, radar, etc. but did not include such items in the synonyms.

7.1.3.4.2.2 Recommended Enhancements

1. It is recommended, as a friendly reminder, that upon completion of exporting the component file from the export area a message be displayed informing the user that the newly exported file must be added to the OMDT data dictionary before it can be used.

2. It is recommended that adding additional useful names to the synonym section would greatly help in the search function.

3. Due to the fact that the exporting of data takes some time the following recommendations are made:

    a) On the Export Area frame, a way be provided for the user to identify only the items to be exported. Comment: Why not use the check boxes for both EXPORT and DELETE, if we do this please include a double check message (for example: 'Do you wish to DELETE the identified component name(s)') for the DELETE button.
    b) On the Export Area frame indicate which items on the list are newly picked items this will assist the user in determining which items to export (goes along with 3a).
    c) During the export of OMDDS data from the Export Area frame, place all the CLASS items into a class file, all the INTERACTIONS into an interaction file, all the GENERIC ELEMENTS into a generic element file, all the COMPLEX DATA TYPE into a complex data type file, and so on. The user can accomplish this if 3a is incorporated and 3b and 6 would be a great help.

4. It is recommend that on the Browse and Search frame the COPY, RESET and HELP buttons be frozen. It is an inconvenience to have to return to the top of the list for the needed button.

5. When picking a generic element, it was not known that the associated enumerated data type was also copied to the export area. It is recommend that the Export Area should display all items both chosen and associated. For example, if a user selects a generic element to be copied to the export area and that generic element has an associated enumerated data type and it is also copied to the export area the export area should display both the generic element and the associated enumerated data type.

6. It is recommended that on the Search frame and Export Area frame that there be a way for the user to be able to sort by component types or component names.

7. It is recommended that multiple export areas be provided. This would allow a single developer to work on multiple federations without having multiple logons and passwords.

7.1.3.4.3 OML

7.1.3.4.3.1 Problems

The RPR FOM was used in the reference FOM approach of this experiment. The following problems were discovered after extracting the RPR FOM from the OML and loaded into the OMDT. The consistency checker of the OMDT identified the following problems:

1. A consistency check in the OMDT caused model invalid errors for the RPR FOM exported from the OML. The cause of the consistency check errors is duplicate names in enumeration type values. The RPR-FOM caused the following model invalid errors:
   (2) TOW_MSL in MUNITION_TYPE_ENUM
   (2) L5_TORPEDO in MUNITION_TYPE_ENUM
   (2) M16 in MUNITION_TYPE_ENUM
   (2) M940_MPTSD in MUNITION_TYPE_ENUM
   (4) HARBIN_AIRCRAFT_MANUFACTURING_CORP_Y in AIRCRAFT_TYPE_ENUM
   The () indicates the number of times the item shows up in the enumeration list, for example there are 2 TOW_MSL items in the MUNITION_TYPE_ENUM list.

2. A few of the attribute type declaration ('unsignedshort' and 'unsignedlong') in the RPR FOM exported from the OML caused consistency check errors in the OMDT and required changing ('unsigned short' and 'unsigned long'). The RPF-FOM caused the following model invalid errors:
   'unsignedshort' not valid use 'unsigned short' in ARTICULATED_PARAMETERS_COUNT
   'unsignedshort' not valid use 'unsigned short' in FIRE_RATE
   'unsignedlong' not valid use 'unsigned long' in EVENT_IDENTIFIER

7.1.3.4.3.2 Recommended Enhancements

With so few FOMs and SOMs in the library, finding a FOM or SOM that is similar to our conceptual model and scenario was relatively easy. The classes, subclasses, interaction, attributes, and parameters were analyzed for each of the FOMs and SOMs in the OML. If there were fifty or more FOMs and SOMs in the library this task would have become more difficult and time consuming. It is recommended that the metadata information for FOMs and SOMs contain participating federates, purpose of federation, security, etc. This information would allow a user to identify more quickly an applicable FOM for possible reuse without analyzing the whole FOM/SOM.

7.1.4 Federation Developers Workbook

The Federation Developers Workbook is made up of four workbooks (

Appendix F). This section includes a federation workbook for each of the different approaches listed below:

1. the bottom up approach
2. the merge SOMs approach using CCTT as the baseline
3. the merge SOMs approach using ModSAF as the baseline
4. the reference FOM approach using the RPR FOM

Each of the workbooks is made up of five tables as defined in High Level Architecture (HLA) Performance Framework [6] paper:

1. Federation Execution Summary Table- includes the fedex name, number of concurrent fedexes and summary information about each federate in the fedex.
2. Host Table – contains information on the hardware, software and capacity of the hosts which support the federates in the fedex.
3. LAN Table – provides descriptive information on each LAN used in the fedex and the LAN-to-LAN connections.
4. RTI Services Table – lists the suite of RTI services and requests that are used at least once in the fedex.
5. Object/Interaction Table – identifies which attributes of object are updated by each federate, how often and in what groupings. The table also identifies those attributes to which a federate subscribes and latency constraints on updates. Similar data is also identified for interactions.

The Federation Developers Workbook can be used to assist developers of fedex to configure the various components of a fedex to meet the needs of their application. It provides a common mechanism to collect data from a range of HLA users to understand the ways different HLA user communities are applying HLA and the performance needs particularly the RTI services. The workbook can be used to describe the federation execution context for RTI testing and is useful to federate developers in defining and profiling the performance characteristics of their federates. The workbook provides a framework for development of automated tools to support fedex planning and testing.

Appendix A – REQUIREMENTS ANALYSIS USE CASE

REQUIREMENTS ANALYSIS USE CASE

Have the general setup: two executions (Exec-A & Exec-B), each with a set of federates and connected with two surrogate federates (Sur-A & Sur-B).

It is assumed that the Create Federation Execution, Destroy Federation Execution, & Resign Federation Execution services have no impact on the bridge federate.

Pause Execution Use Case

Rate-A3:
invoke Request Pause service (A_LABEL) [this is in Exec-A]

Exec-A RTI invokes Initiate Pause service in the execution, including Sur-A

Sur-A [Initiate Pause (A_LABEL)]
send an echo pause message (A_LABEL) to Sur-B

Sur-B [echo pause message processing (A_LABEL)]
invoke Request Pause (A_LABEL) [this is in Exec-B]

Exec-B RTI invokes Initiate Pause in the execution, including Sur-B

Sur-B [Initiate Pause (A_LABEL)]
invoke Pause Achieved (A_LABEL) [this is in Exec-B]

Sur-B
wait until entire Exec-B is paused {{currently nothing in I/F spec to support this!!}}

send echo-pause achieved message (A_LABEL) to Sur-A

Sur-A [echo-pause achieved message (A_LABEL) processing]
invoke Pause Achieved (A_LABEL) [this is in Exec-A]

What is happening here: the surrogate (domestic) in the execution where the pause was initially requested is the "throttle" for that execution. The domestic surrogate will not invoke Pause Achieved until it is notified (from the foreign surrogate) when the foreign execution has paused.

Also, the domestic surrogate requests a foreign execution pause via the foreign surrogate.

Same steps for resume (w/o label processing)

Save Execution Use Case

Rate-A3:
invoke Request Save (A_LABEL, A_TIME) [this is in Exec-A]

Exec-A RTI invokes Initiate Save service in the execution, including Sur-A

Sur-A [Initiate Save (A_LABEL, A_TIME)]
if there is no time parameter: invoke Federate Save Begun service, send an echo save message (A_LABEL) message to Sur-B, and save immediately

otherwise, at time A_TIME:
invoke Federate Save Begun (A_TIME) service

send an echo save message (A_LABEL) to Sur-B

save everything which the surrogate must save

<<translator might have to save. need to somehow communicate success to save initiator.>>

Sur-B [echo save message processing (A_LABEL)]
invoke Request Save (A_LABEL) service [this is in Exec-B]

Exec-B RTI invokes Initiate Save in the execution, including Sur-B

Sur-B [Initiate Save (A_LABEL) service]
invoke Federate Save Begun service

save everything which the surrogate must save

invoke Save Achieved (A_LABEL) [this is in Exec-B]

Sur-B
wait until entire Exec-B is saved and get the save success/no success status
{{currently nothing in I/F spec to support these!!}}

send echo-save succeeded or echo-save failed message, as appropriate,
to Sur-A

Sur-A [echo-save succeeded/failed message processing]
invoke Save Achieved (success indicator) service [this is in Exec-A]

Restore Execution Use Case

Rate-A3:
invoke Request Restore (A_LABEL) [this is in Exec-A]

Exec-A RTI invokes Initiate Restore in the execution, including Sur-A

Sur-A [Initiate Restore (A_LABEL)]
send an echo restore message (A_LABEL) to Sur-B

do any necessary surrogate restoration

<<translator might have to restore. need to somehow communicate success to restore initiator>>

Sur-B [echo restore message processing (A_LABEL)]
invoke Request Restore (A_LABEL) [this is in Exec-B]

Exec-B RTI invokes Initiate Restore in the execution, including Sur-B

Sur-B [Initiate Restore (A_LABEL)]
do any necessary surrogate restoration

invoke Restore Achieved (A_LABEL, surrogate restoration success
indicator) [this is in Exec-B]

Sur-B
wait until entire Exec-B is restored and get the restore success/no success
status {{currently nothing in I/F spec to support this!!}}

send echo-restore succeeded or echo-restore failed message, as appropriate,
to Sur-A

Sur-A [echo-restore succeeded/failed message processing]
invoke Restore Achieved (A_LABEL, combination of foreign execution success

indicator and surrogate restoration success indicator) [this is in Exec-A]

Have the general setup: two executions (Exec-A & Exec-B), each with a set of federates and connected with two surrogate federates (Sur-A & Sur-B). Assume that Sur-A has subscribed and Sur-B has published appropriately.

Discover Object Use Case

Rate-A3:
Register Object (Exec-A handle for Tank object class, A-243)

Exec-A RTI invokes Discover Object (A-243, Exec-A handle for Tank object class, A-Time, A-Tag-1, A-Event-Retraction-345) in all federates in Exec-A which have subscribed to Tank object class, including Sur-A (after the appropriate Update Attribute service invocation)

Sur-A: [Discover Object (A-243, Exec-A handle for Tank object class, A-Time, A-Tag-1, A-Event-Retraction-345)
convert the Exec-A handle for Tank object class to the Exec-A string for the Tank object class

send a echo discover object SICM (A-243, Exec-A string for the Tank object class, A-Time, A-Tag-1, A-Event-Retraction-345) to Sur-B

Sur-B [echo discover object message processing]:
receive the echo discover object SICM (A-243, Exec-A string for the Tank object class, A-Time, A-Tag-1, A-Event-Retraction-345)

Request ID from Exec-B (B-739)

associate domestic if B-749 with foreign if A-243, A-Time, A-Tag-1

convert the Exec-A string version of the Tank object class to the Exec-B handle version

Register Object (Exec-B handle version of the Tank object class, B-739)

Exec-B RTI invokes Discover Object (B-739, Exec-B handle version of the Tank object class, B-Time, B-Tag-3, B-Event-Retraction-950) in all federates in Exec-B which have subscribed to Tank object class (after the appropriate Update Attribute service invocation)

{{maybe a problem, here. need to associate foreign retraction A-Event-Retraction-345 with domestic retraction B-Event-Retraction-950. how does Sur-B get the B-Event-Retraction-950 handle to associate with the A-Event-Retraction-345 handle so it can echo a retraction of the A event? even if Sur-B subscribes to the Tank object class, RTI will not call Discover Object (according to I/F spec because Sur-B registered the object ID). actually, this might be a problem only for request attribute values update message processing, because the retractions are for update/reflect attribute values, not register/discover object. but if that is the case, why does discover object come with a retraction handle? }}

Update Attribute Values Use Case

Rate-A3:
Update Attribute Values (A-243, (Exec-A handle for speed attribute, 45.6), A-Time-1, A-Tag-89) which returns A-Event-Retraction-6945

Exec-A RTI invokes Reflect Attribute Values (A-243, (Exec-A handle for speed attribute, 45.6), A-Time-1, A-Tag-89, A-Event-Retraction-6945) in all federates in Exec-A which have subscribed to Tank object class and speed attribute, including Sur-A

Sur-A: [Reflect Attribute Values (A-243,
(Exec-A handle for speed attribute, 45.6), A-Time-1, A-Tag-89,
A-Event-Retraction-6945)]
convert the Exec-A handle for speed attribute to the Exec-A string for speed attribute

send a echo attribute values update SICM (A-243, (Exec-A string version of speed attribute, 45.6), A-Time-1, A-Tag-89, A-Event-Retraction-6945) to Sur-B

Sur-B [echo attribute values update message processing]
receive echo attribute values update SICM (A-243, (Exec-A string version of speed attribute, 45.6), A-Time-1, A-Tag-89, A-Event-Retraction-6945)

find domestic ID B-739 associated with foreign ID A-243

convert the Exec-A string version of speed attribute to Exec-B handle version

<<need to do some sort of time sync here>>

Update Attribute Values (B-739,
(Exec-B handle version of speed attribute, 45.6), B-Time-5, A-Tag-89) returns
B-Event-Retraction-7839

associate A-Event-Retraction-6945 with B-Event-Retraction-7839

Exec-B RTI invokes Reflect Attribute Values (B-739, (Exec-B handle version of speed attribute, 45.6), B-Time-5, A-Tag-89, B-Event-Retraction-7839) in all federates in Exec-B which have subscribed to Tank object class and speed attribute.

Send Interaction Use Case

still to be completed ???

Delete Object Use Case

still to be completed ???

Retract Use Case

still to be completed ???

assuming the following don't need uses cases because they don't impact the bridge federate:

Request Attribute Values

Change Attribute/Interaction Transportation/Order Type

Attribute Ownership Acquisition Use case

Have the general setup: two executions (Exec-A & Exec-B), each with a set of federates and connected with two surrogate federates (Sur-A & Sur-B).

The following situation exists:

Farkle attribute of Exec-B object instance B-459 is modeled (model-own) by a federate [Rate-B5] in Exec-B

the object instance A-156 in Exec-A is the echo of the Exec-B object instance B-459

Sur-A echo-owns the Farkle attribute of Exec-A object instance A-156 which echoes the Exec-B attribute in Exec-A

A federate [Rate-A1] in Exec-A wishes to start modeling the Farkle attribute for object instance A-156/B459, implying that it must model-own the attribute.

The following happens for successful ownership acquisition:

Rate-A1:
invoke Request Attribute Ownership Acquisition (A-156,
Exec-A handle for Farkle attribute)

... Rate-A1 can now do other things, waiting for response to RAOQ request; ie, this is not a "blocking call" ...

Exec-A RTI invokes Request Attribute Ownership
Release (A-156, Exec-A handle for Farkle attribute) in Sur-A

Sur-A [Request Attribute Ownership Release (A-156,
Exec-A handle for Farkle attribute)]:
change the Exec-A handle for Farkle attribute to the Exec-A string for the Farkle attribute

since Sur-A is echoing object B-456 via object A-156 it must convert A-156 id to B-456

send a request attribute ownership transfer surrogate intercommunication message [SICM] (B-456, Exec-A string version of Farkle attribute) to Sur-B

{{However, there is a problem here:
due to the attribute transfer protocol,
Sur-A cannot return from the Request Attribute Ownership Release call from the Exec-A RTI until Sur-A knows from Sur-B if the Exec-B will give up ownership of the attribute or not. This could take a while to go over

to the other execution and see if the modeling/owning federate will give up ownership. Does this mean that Sur-A has to poll for Sur-B messages until it receives a response to this request?}}

Sur-B [request attribute ownership transfer message processing]:
receive the request attribute ownership transfer SICM (B-456 Exec-A string version of Farkle attribute) message

convert Exec-A string version of Farkle attribute to Exec-B handle version

invoke Request Attribute Ownership Acquisition (B-459, Exec-B handle for Farkle attribute)

... Sur-B can now do other things, waiting for response to the RAOQ request; ie, this is not a "blocking call" ...

Exec-B RTI invokes request Attribute Ownership Release (B-459, Exec-B handle for Farkle attribute) in Rate-B5

Rate-B5 [Request Attribute Ownership Release (B-459, Exec-B handle for Farkle attribute)]:
determines the attribute ownership may be transferred and returns Exec-B handle for Farkle attribute

Exec-B RTI invokes Attribute Ownership Acquisition Notification (B-459, Exec-B handle for Farkle attribute) in Sur-B

Sur-B: [Attribute Ownership Acquisition Notification (B-459, Exec-B handle for Farkle attribute)]
change the Exec-B handle for Farkle attribute to the Exec-B string for the Farkle attribute

send a transfer attribute ownership approved SICM (B-459, Exec-B string version of Farkle attribute) to Sur-A

Sur-A [transfer attribute ownership approved message processing]:
receive the transfer attribute ownership approved message (B-459, Exec-B string version of Farkle attribute)

convert Exec-B string version of Farkle attribute to Exec-A handle version

convert Exec-B object ID [B-459] to Exec-A object ID [A-156]

Sur-A now gives up ownership in of Farkle for A-156 in Exec-A by returning the Exec-A handle for Farkle attribute [remember, Sur-A is still in the Request Attribute Ownership Release call by the RTI]

Exec-A RTI invokes Attribute Ownership Acquisition Notification (A-156, Exec-A handle for Farkle attribute) in Rate-A1

now if Rate-B5 does not give up the Farkle attribute, every thing is the same up to the Request Attribute Ownership Release call to Rate-B5 and then:

Exec-B RTI invokes Request Attribute Ownership Release (B-459, Exec-B handle for Farkle attribute) in Rate-B5

Rate-B5 [Request Attribute Ownership Release (B-459, Exec-B handle for Farkle attribute)]:
determines the attribute ownership will not be transferred and and returns and empty attribute list

Sur-B:
must somehow time out. {{this is not described in the I/F spec!!}}

change the Exec-B handle for Farkle attribute to the Exec-B string for the Farkle attribute

send a transfer attribute ownership denied SICM to (B-459, exec-B string version of Farkle attribute) to Sur-A

Sur-A [transfer attribute ownership denied message processing]:
receive the transfer attribute ownership denied message (B-459, Exec-B string version of Farkle attribute)

since Exec-B would not give up model-own, Sur-A cannot give up echo-own, so it just returns an empty attribute list [remember, Sur-A is still in the Request Attribute Ownership Release call by the RTI]

Rate-A1:
must somehow time out. {{this is not described in the I/F spec!!}}

Attribute Ownership Divestiture Use Case

still needs to be completed ???

<<An "open" Negotiated Request Attribute Ownership Divestiture service request may be canceled by making Request Attribute Ownership Acquisition call with the attributes the federate was trying to divest. >>

Also, need to support all cases when exceptions happen. Most likely, just fail when an exception happens and hand back the exception to the "calling" entity.

intra bridge federate communication conventions

object ids & event handles are those from the "initiating" execution (the modeling execution, the execution where the object was initially discovered or the event originated) only the echoing surrogate converts object id's and event handles

class/attribute/parameter names passed as domestic strings in SICMs

information needed from the rti which is not currently (i/f spec 1.1) available

individual federate pub/sub info

execution paused/resumed

execution saved/restored (w/success indicator)

surrogate intercommunication messages (SICM)

request attribute ownership transfer
(modeling execution object id, domestic string version of attribute, user tag
)

transfer attribute ownership approved
(modeling execution object id, domestic string version of attribute, user tag
)

transfer attribute ownership denied
(modeling execution object id, domestic string version of attribute, user tag
)

echo pause
(label)

echo-pause achieved
(label)

echo resume

echo-resume achieved

echo save
(label)

echo-save succeeded

echo-save failed

echo restore
(label)

echo-restore succeeded
(label)

echo-restore failed
(label)

echo discover object
(modeling execution object id, domestic string version of object class, time, user_tag, originating execution event retraction handle)

echo attribute values update

(modeling execution object id,
(att name/value pair list with domestic string version of att names),
time, user_tag, originating execution event retraction handle)

use cases left to do

declaration management

still need to determine appropriate cases

object management

send/receive interaction
delete/remove object
retract event

ownership management

att ownership divestiture

time management

still need to determine appropriate cases

data distribution management

still need to determine appropriate cases

Appendix B – BRIDGE FEDERATE SYSTEM REQUIREMENTS

Bridge Federate System Requirements

Engineer: Wesley Braudaway

1.1 Bridge Federate Specification

A Bridge Federate is a federate that participates in multiple federations. It has several internal components: surrogates (one for each federation it participates in) and a transformer that coordinates among the surrogates. The Bridge Federate serves to combine multiple federation executions to provide opportunity for their interoperation.

The combined federation execution as shown in Figure 1 shows two federation executions interoperating through the support of a Bridge Federate. The communication of this combined federation execution is achieved without requiring any one federate to understand each and every FOM. Each federation execution (fedex) can communicate with other fedexs using its own FOM and the Surrogate Federate provided by the Bridge Federate. Each federate has no knowledge that it is, in fact, interoperating within multiple federation executions. The specification for how objects, attributes, and interaction are mapped throughout the combined federation execution is defined by the FOM Mappings/Transformations specification (FOMAT in Figure 1).

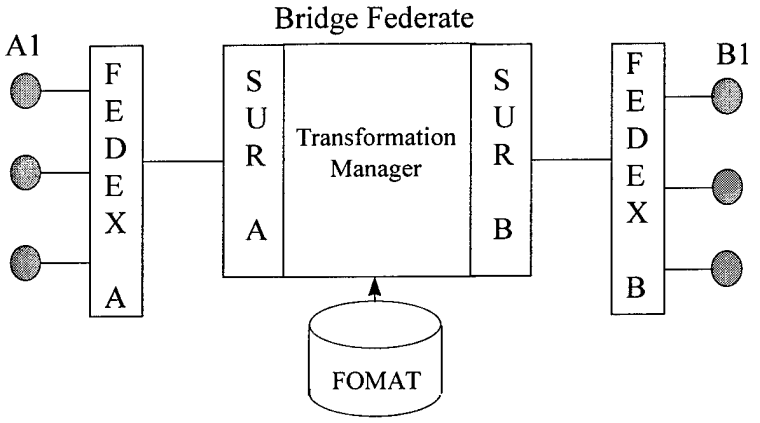


Figure 1: Bridge Federate Components

The Bridge Federate is composed of multiple surrogate federates which behave as Federates within their respective federation executions. The figure illustrates two federations labeled FEDEX A and FEDEX B and a surrogate for each federation execution labeled SUR A and SUR B, respectively. Each surrogate participates within its joined FEDEX as a federate on behalf of the other federation executions. The Transformation Manager provides a message conduit and translator between the these multiple surrogate federates; thus linking the surrogates' federations. Its transformation functionality is responsible for implementing all HLA services across the bridge and for

transforming object, attributes, and interaction communication between the federation executions as specified by the FOMAT.

A Bridge Federate can be as simple or complex as needed to support the needs of linking multiple federation executions. It is expected that the Surrogate components will provide the common functionality that will be needed by any Bridge Federate implementation. Also it is expected that the Transformation Manager will be tailorable to support the actual requirements of the Bridge Federate implementation. However, any Transformation Manager will provide, at a minimum, the core functionality that can be expanded if needed to support additional requirements (e.g., security)

1.2 Concept Of Operations

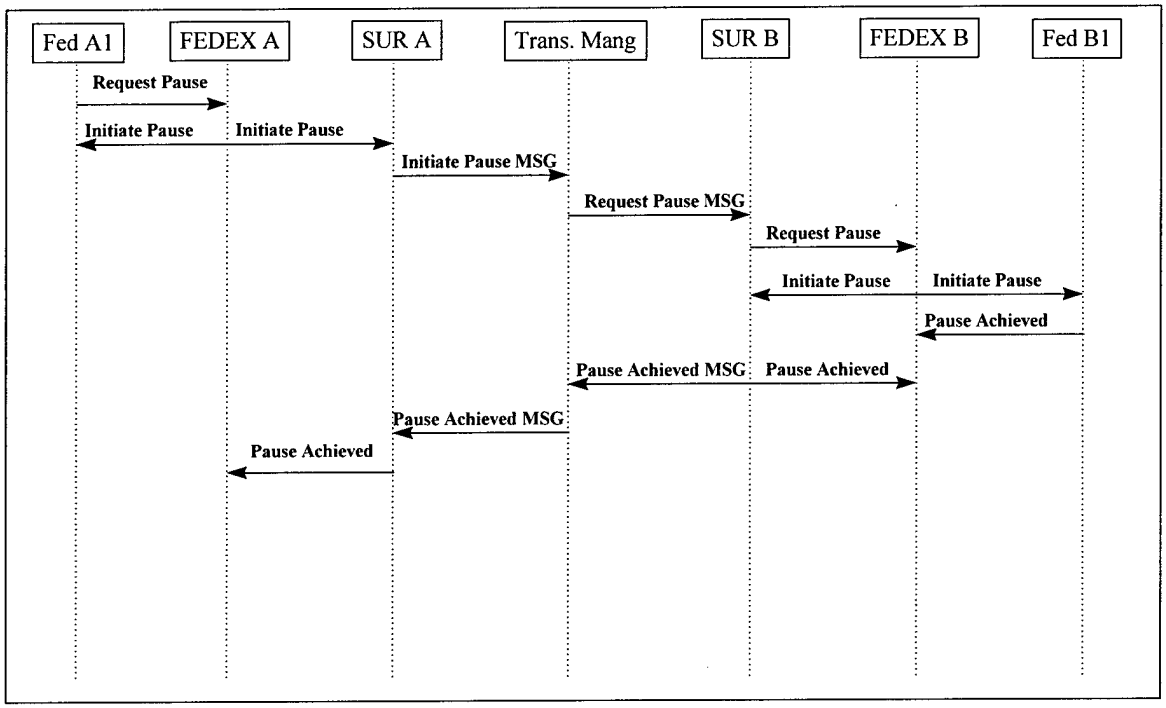
This section contains the event traces that show the critical control flow through the Bridge Federate to support the HLA services and the data transformations across bridge. Each event trace illustrates the concept of execution for key event sequences that will occur during the operation of the Bridge Federate. For each event trace, each vertical line corresponds to a system component that participates in the specific event sequence. The arrows between vertical lines indicate service calls that the source component (at the tail of the arrow) invokes from the target component (at the head of the arrow). The name of the service or message provided by the target component is the label of the arrow. The event traces illustrate these calling event sequences where time advances down the event trace figure.

1.2.1 Federation Management

Assume each federation has its own federation manager that creates and destroys each federation execution. Each surrogate federate will be directed by the Transformation Manager to join its specific federation.

Pause Across the Bridge Federate

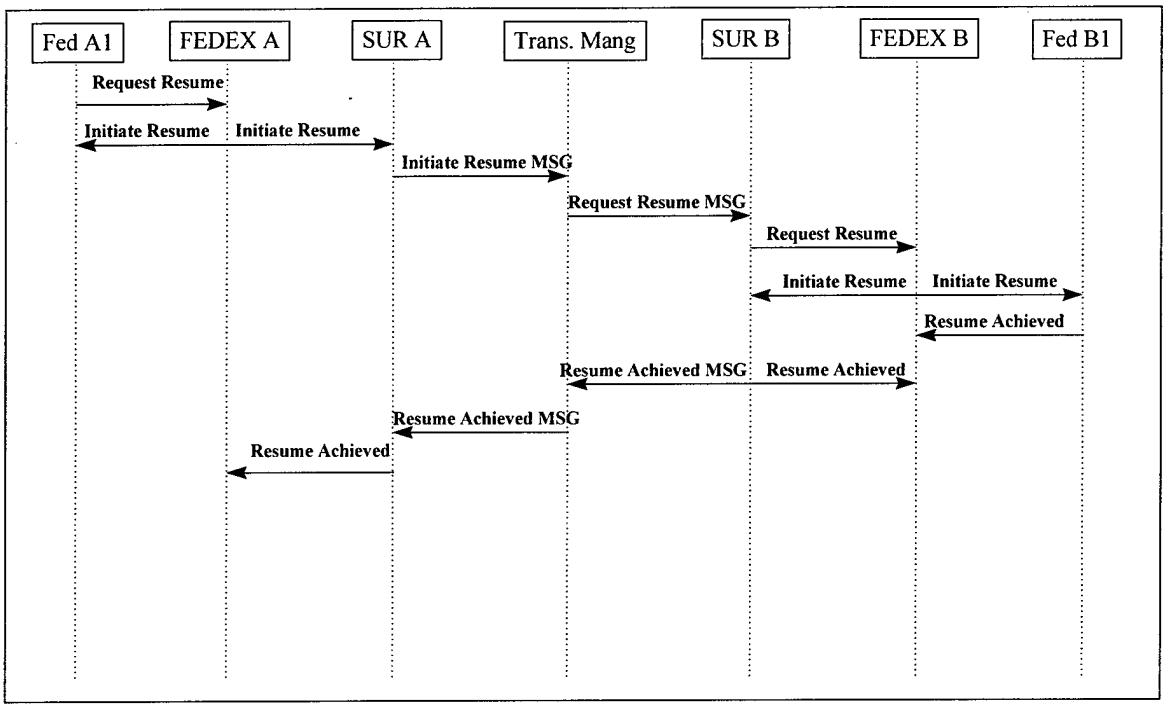
Federate A1 in federation FEDEX A requests a pause which will be transmitted to each federation linked by the Bridge Federate.



1. Federate A1 issues a Request Pause to FEDEX A.
2. SUR A receives an Initiate Pause from FEDEX A.
3. SUR A issues an Initiate Pause message to the Transformation Manager.
4. The Transformation Manager issues a Request Pause message to SUR B.
5. SUR B issues a Request Pause to FEDEX B.
6. Federate B1 receives an Initiate Pause from FEDEX B.
7. Federate B1 issues an Pause Achieved to FEDEX B.
8. SUR B monitors the MOM classes to determine when each and every Federate of FEDEX B reaches the Paused state. Upon this condition, SUR B issues a Pause Achieved to FEDEX B and echoes this message to the Transformation Manager.
9. The Transformation Manager echoes the Pause Achieved message to SUR A once all other surrogates sends a paused achieved message.
10. SUR A issues a Pause Achieved to FEDEX A.

Resume Across the Bridge Federate

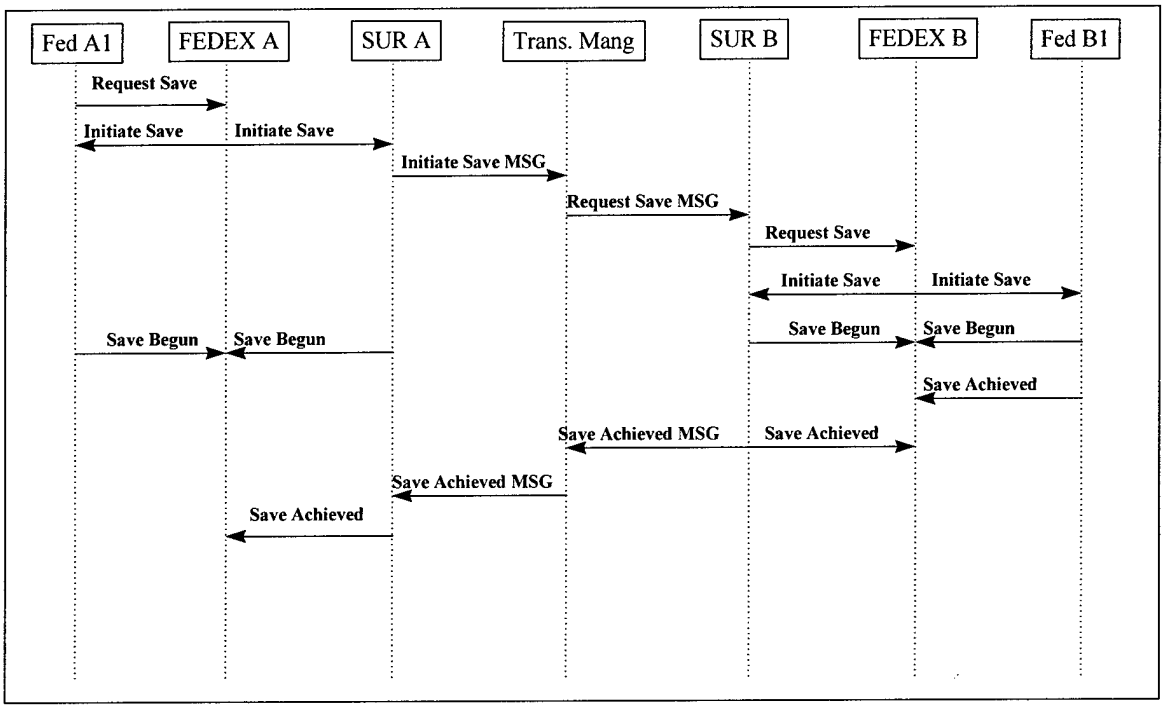
Federate A1 in federation FEDEX A requests a resume which will be transmitted to each federation linked by the Bridge Federate.



1. Federate A1 issues a Request Resume to FEDEX A.
2. SUR A receives an Initiate Resume from FEDEX A.
3. SUR A issues an Initiate Resume message to the Transformation Manager.
4. The Transformation Manager issues a Request Resume message to SUR B.
5. SUR B issues a Request Resume to FEDEX B.
6. Federate B1 receives an Initiate Resume from FEDEX B.
7. Federate B1 issues an Resume Achieved to FEDEX B.
8. SUR B monitors the MOM classes to determine when each and every Federate of FEDEX B reaches the Resumed state. Upon this condition, SUR B issues a Resume Achieved to FEDEX B and echoes this message to the Transformation Manager.
9. The Transformation Manager echoes the Resume Achieved message to SUR A once all other surrogates sends a Resumed achieved message.
10. SUR A issues a Resume Achieved to FEDEX A.

Save Across the Bridge Federate

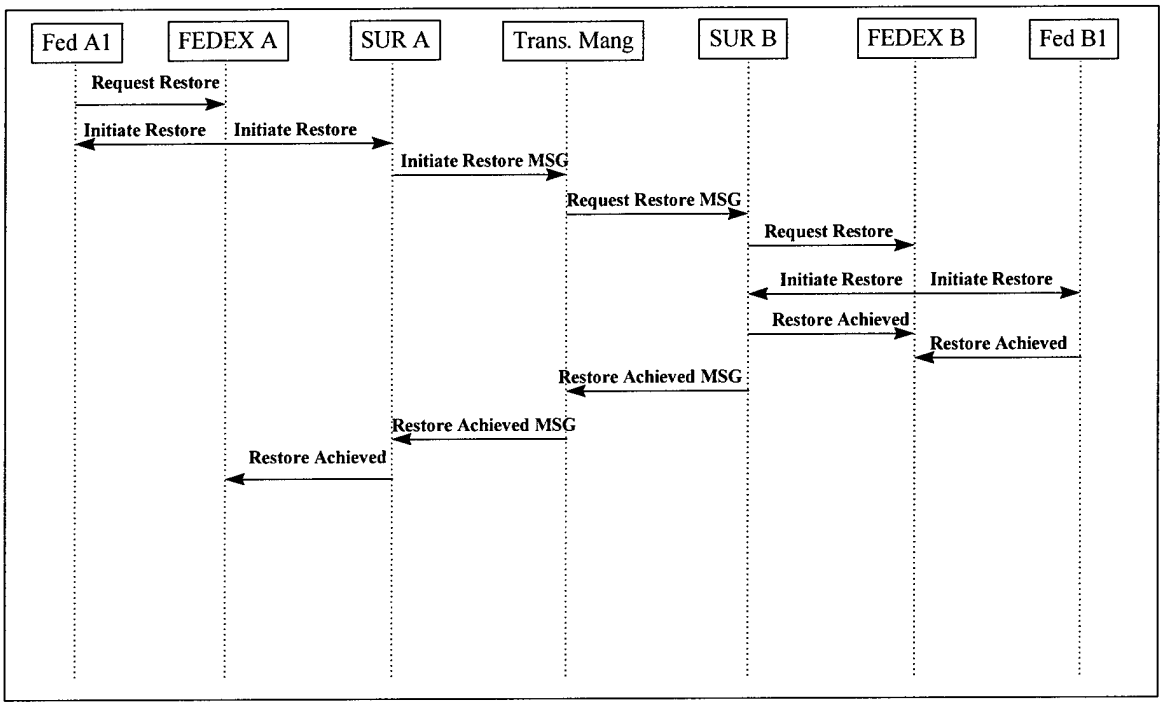
Federate A1 in federation FEDEX A requests a save which will be transmitted to each federation linked by the Bridge Federate.



1. Federate A1 issues a Request Save to FEDEX A.
2. SUR A receives an Initiate Save from FEDEX A.
3. SUR A issues an Initiate Save message to the Transformation Manager.
4. The Transformation Manager issues a Request Save message to SUR B.
5. SUR B issues a Request Save to FEDEX B.
6. Federate B1 receives an Initiate Save from FEDEX B.
7. At the specified time, each federate notifies its respective FEDEX that it has begun the Federation Save.
8. Federate B1 issues an Save Achieved to FEDEX B.
9. SUR B monitors the MOM classes to determine when each and every Federate of FEDEX B reaches the Saved state. Upon this condition, SUR B issues a Save Achieved to FEDEX B and echoes this message to the Transformation Manager.
10. The Transformation Manager echoes the Save Achieved message to SUR A once all other surrogates sends a Saved achieved message.
11. SUR A issues a Save Achieved to FEDEX A.

Restore Across the Bridge Federate

Federate A1 in federation FEDEX A request a restores which will be transmitted to each federation linked by the Bridge Federate.



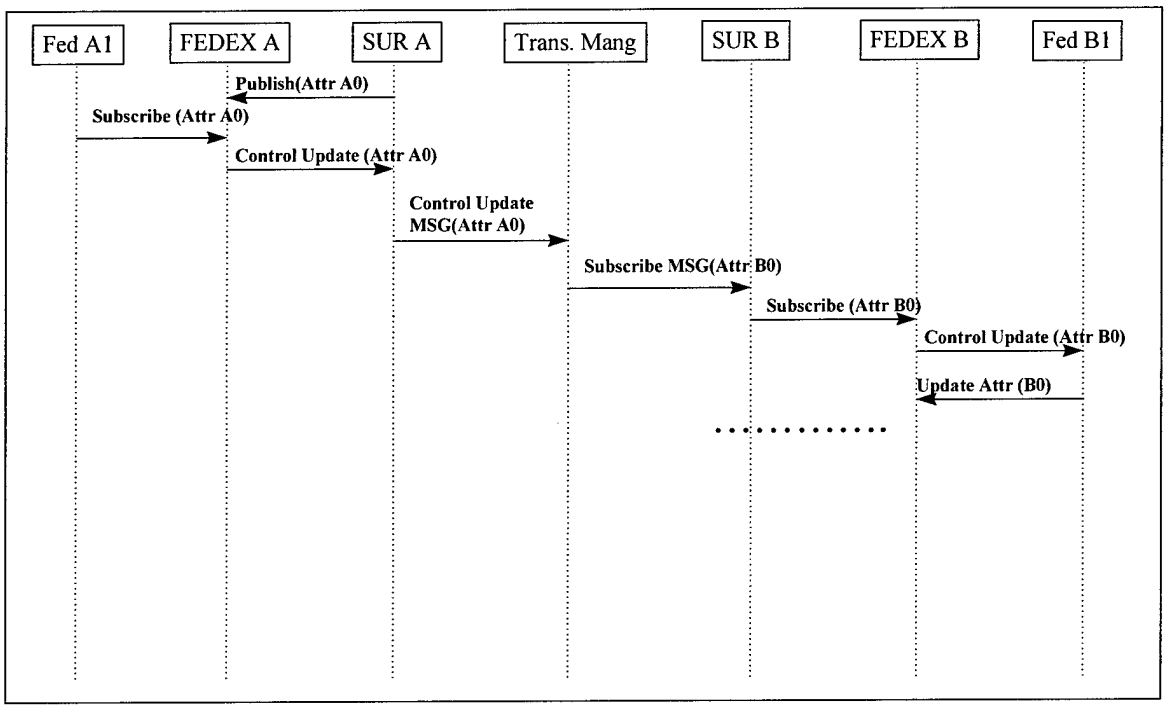
1. Federate A1 issues a Request Restore to FEDEX A.
2. SUR A receives an Initiate Restore from FEDEX A.
3. SUR A issues an Initiate Restore message to the Transformation Manager.
4. The Transformation Manager issues a Request Restore message to SUR B.
5. SUR B issues a Request Restore to FEDEX B, FEDEX B sends initiate Restore to SUR B.
6. Federate B1 receives an Initiate Restore from FEDEX B.
7. Federate B1 issues a Restore Achieved to FEDEX B.
8. SUR B issues a Restore Achieved to FEDEX B then monitors the MOM classes to determine when each and every Federate of FEDEX B reaches the Restored state. Upon this condition, SUR B echoes this message to the Transformation Manager.
9. The Transformation Manager echoes the Restore Achieved message to SUR A once all other surrogates sends a Restored achieved message.
10. SUR A issues a Restore Achieved to FEDEX A.

1.2.2 Declaration Management

Subsequent to joining its federation execution, each surrogate will publish all object attributes and interactions that represent the intersection of the FOMAT's and FOM specification for surrogate's federations. Each surrogate will only issue a subscription for some object attribute or interaction on behalf of some other federation when the other federation scribes interest in an attribute/interaction associated by the FOMAT.

Subscribing Across the Bridge Federate

Federate A1 in federation FEDEX A chooses to subscribe to an object attribute that corresponds, through the FOMAT, to an object attribute that can be generated by Federate B1 in Federation FEDEX B.



1. Surrogate A in Federation execution (FEDEX A) issues a publish object class for each class in the intersection between the Bridge Federate's FOMAT for Federation A and the FOM of Federation A.
2. Federate A1 in Federation execution (FEDEX A) issues a Subscribe Object Class to request any instances of Object Class A0's Attribute A0. Note, that Federate A1 will use the same call to indicate that it no longer wants to subscribe to the specified attribute(s). The following sequence continues the same for either case except that Federate B1 will stop issuing Updates after it receives that Control Update specifying that it stop sending updates for the specified attribute(s). Stop updates are only sent if no other federate is subscribed.
3. The Surrogate Federate (SUR A) receives a control update with the object class A0 Attribute A0.
4. The surrogate federate passes a control update message to the Transformation Manager (Trans Manager) including the object class name and the attribute names.
5. The Transformation Manager passes a Subscribe Object Class to SUR B indicating the FEDEX B object class and attributes corresponding to the given object attributes from Federate A1 as defined by the FOMAT.
6. The SUR B Subscribes to the corresponding Object Class B0 Attribute B0.
7. The Federate B1 which publishes values for Object Class B0 Attribute B0 receives a control update from FEDEX B for that object attribute pair.

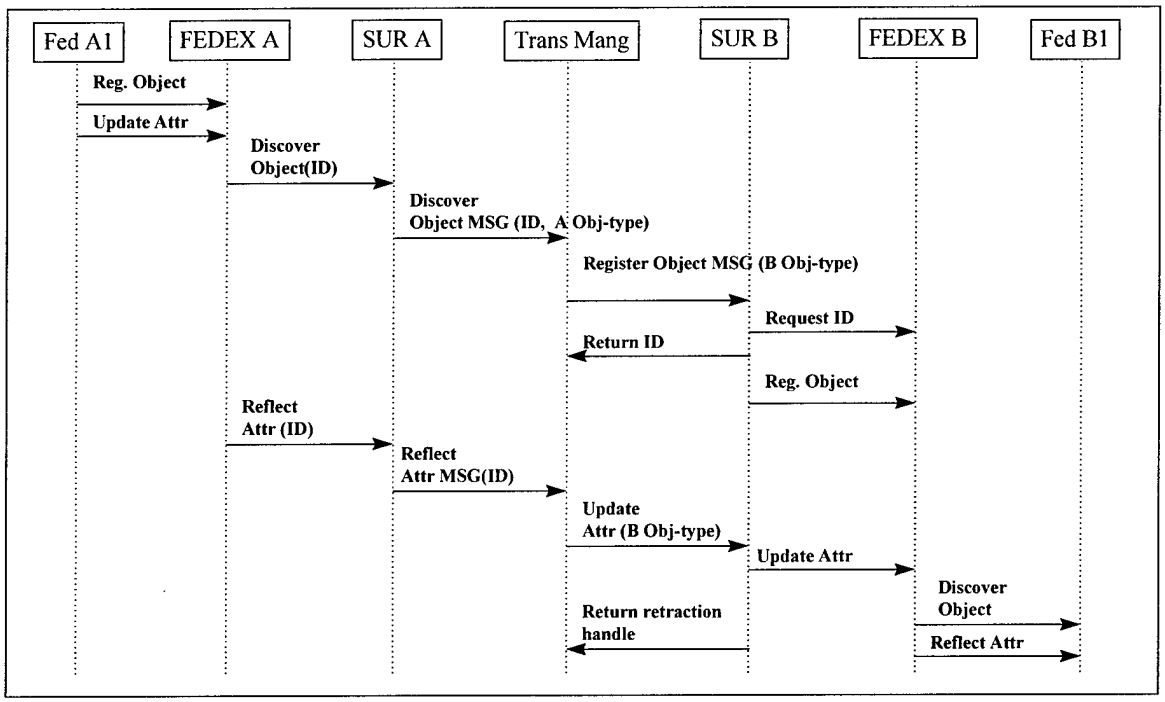
8. Sometime later Federate B1 may send an Update Attribute Value to FEDEX B for Object Class B0 Attribute B0. The remainder of the event trace for update attribute value is described below.

1.2.3 Object Management

The object management group of RTI services deals with creation, modification, and deletion of object and the interactions they produce. All of these services can be implemented by the Bridge Federate with the exception of those services that change the transportation type or ordering type.

Creating and updating a new object

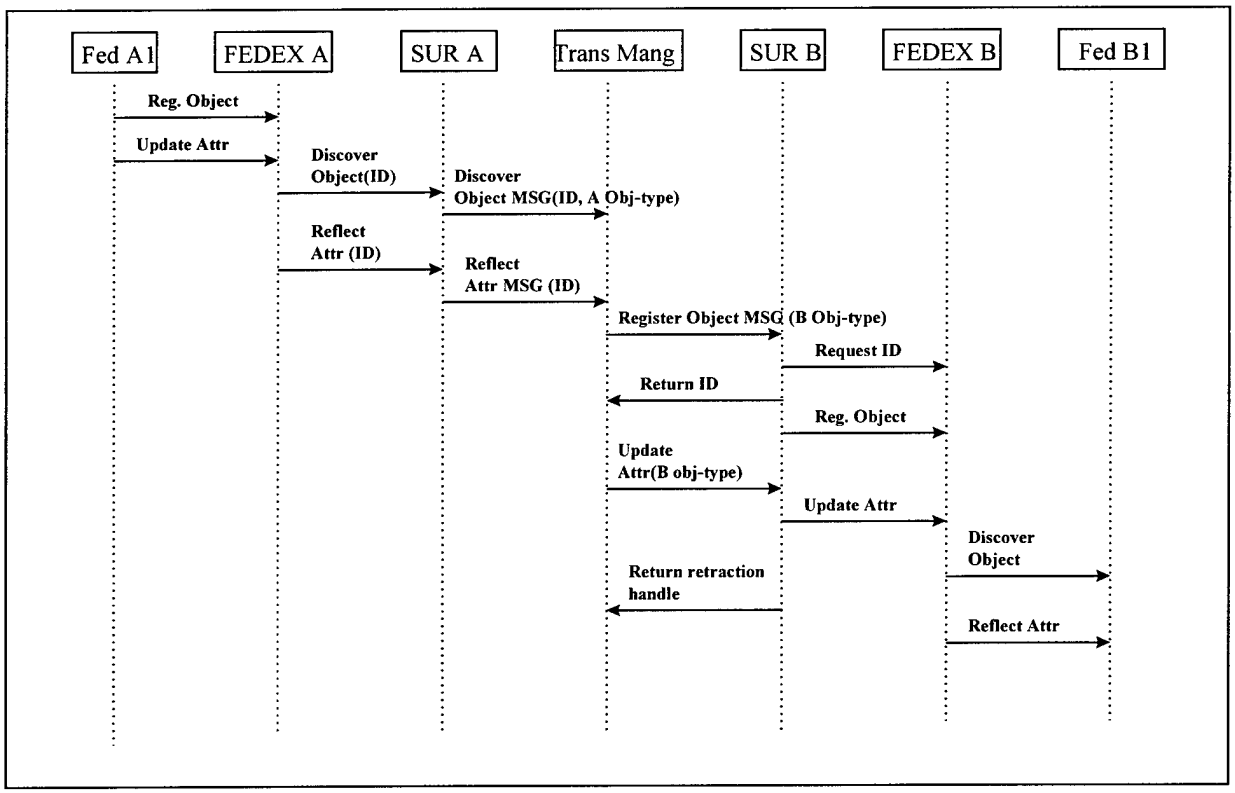
Federate A1 Updates a new instance of an object class that corresponds to an object class in Federation FEDEX B as defined by the FOMAT.



1. Federate A1 in Federation execution (FEDEX A) issues a register object to create a new instance of an object for a particular class.
2. Federate A1 Updates the attributes of the object to the FEDEX A.
3. The Surrogate Federate (SUR A) receives a discover object with the object ID for the registered object.
4. The surrogate federate passes a discover object message to the Transformation Manager (Trans Manager) including the object ID, object class, and a retraction handle for federate A's object. Design Decision: should the Object Class be sent as a string or should the Federation Manager have the object/attribute handle associations which can be constructed during its initialization.
5. The Transformation Manager passes a register object message to SUR B indicating the FEDEX B object type to be created that corresponds to the given object type from Federate A1 as defined by the FOMAT.
6. The SUR B requests an ID for a new object instance.

7. SUR B returns the FEDEX B object ID to the Transformation Manager so it can record the correspondence between the Federate A1 object instance and its new instance in FEDEX B.
8. SUR B registers the new FEXED B object instance with FEDEX B.
9. SUR A receives a reflect attribute value for the object instance from Federate A1.
10. SUR A passes a reflect attribute message to the Transformation Manager including the object ID, attribute value pairs, and retraction handle for federate A's object.
11. The Transformation Manager passes an update attribute value message to SUR B for the corresponding FEDEX B object instance, the translated attribute list (from FEDEX A attributes to FEDEX B attributes), and the transformed attribute values as specified by the FOMAT.
12. SUR B issues an update attribute value to FEDEX B.
13. SUR B returns the retraction handle corresponding to the Update Attribute value event to be associated with corresponding update attribute value retraction handle for FEDEX A.
14. Federate B1 receives an discover object and reflect attribute value call for this object value update.

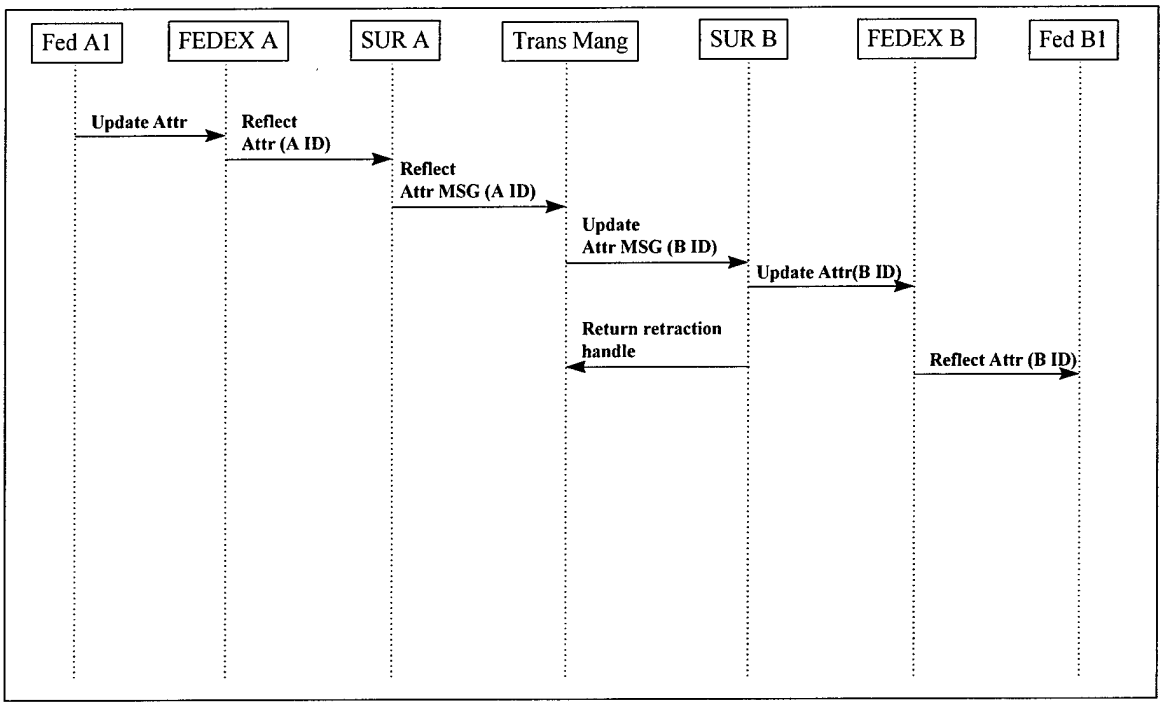
Creating and updating a new object (single class maps to multiple classes case)
Federate A1 Updates a new instance of an object class that corresponds to two or more object classes in Federation FEDEX B as defined by the FOMAT.



1. Federate A1 in Federation execution (FEDEX A) issues a register object to create a new instance of an object for a particular class.
2. Federate A1 Updates the attributes of the object to the FEDEX A.
3. The Surrogate Federate A (SUR A) receives a discover object with the object ID for the registered object.
4. The Surrogate Federate passes a discover object message to the Transformation Manager (Trans Mang) including the object ID, object class, and a retraction handle for federate A's object. At this point, the Transformation Manager cannot pass the discover object message to the Surrogate Federate B because it needs the attributes to determine which FEDEX B object type to transform to.
5. SUR A receives a reflect attribute value for the object instance for Federate A1.
6. SUR A passes a reflect attribute message to the Transformation Manager including the object ID, attribute value pairs, and retraction handle for federate A's object.
7. The Transformation Manager now has enough information to pass the discover object message to SUR B indicating the FEDEX B object type to be created that corresponds to the given object type from Federate A1 as defined by the FOMAT.
8. The SUR B requests an ID for a new object instance.
9. SUR B returns the FEDEX B object ID to the Transformation Manager so it can record the correspondence between the Federate A1 object instance and its new instance in FEDEX B.
10. SUR B registers the new FEDEX B object instance with FEDEX B.
11. The Transformation Manager passes an update attribute value message to SUR B for the corresponding FEDEX B object instance, the tranlated list (from FEDEX A attributes to FEDEX B attributes), and the transformed attribute values as specified by the FOMAT.
12. SUR B issues an update attribute value to FEDEX B.
13. SUR B returns the retraction handle corresponding to the Update Attribute value event to be associated with corresponding update attribute value retraction handle for FEDEX A.
14. Federate B1 receives an discover object and reflect attribute value call for this object value update.

Updating an existing object

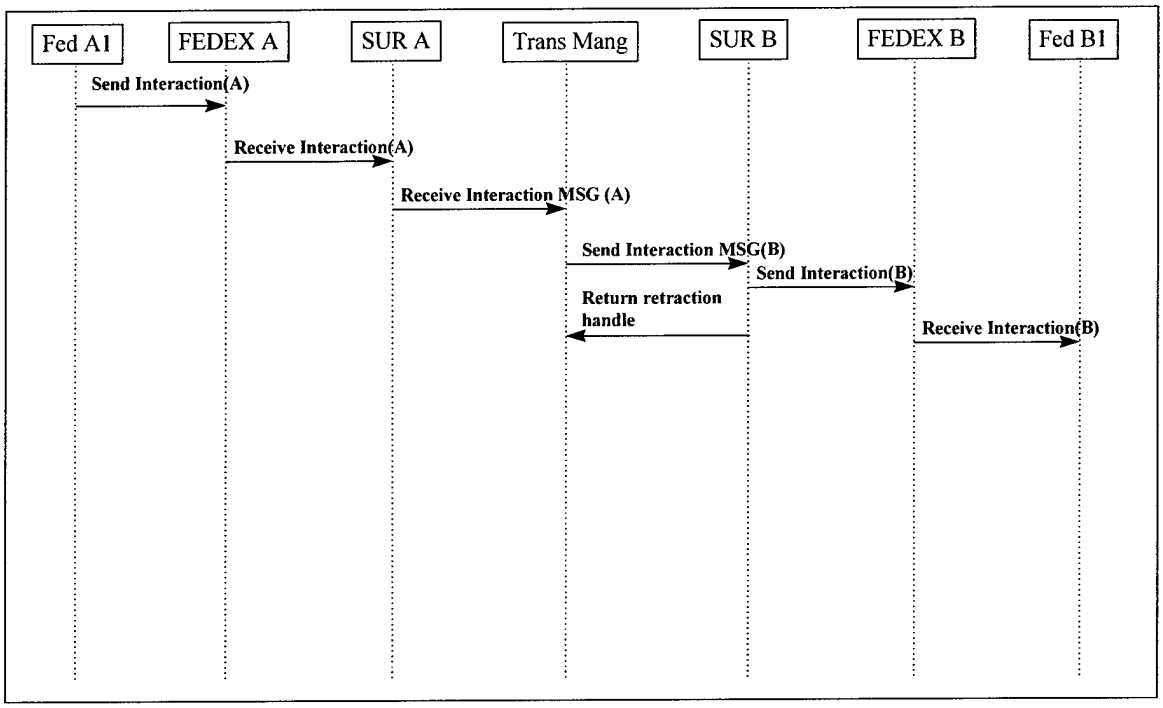
Federate A1 Updates the attribute values of an existing object that corresponds to an object instance found in federation FEDEX B as defined by the FOMAT.



1. Federate A1 Updates the attributes of the object to the FEDEX A.
2. SUR A passes a reflect attribute message to the Transformation Manager including the object ID, attribute value pairs, and retraction handle for federate A's object.
3. The Transformation Manager passes an update attribute value message to SUR B for the corresponding FEDEX B object instance, the translated attribute list (from FEDEX A attributes to FEDEX B attributes), and the transformed attribute values as specified by the FOMAT.
4. SUR B issues an update attribute value to FEDEX B.
5. SUR B returns the retraction handle corresponding to the Update Attribute value event to be associated with corresponding update attribute value retraction handle for FEDEX A.

Sending an interaction

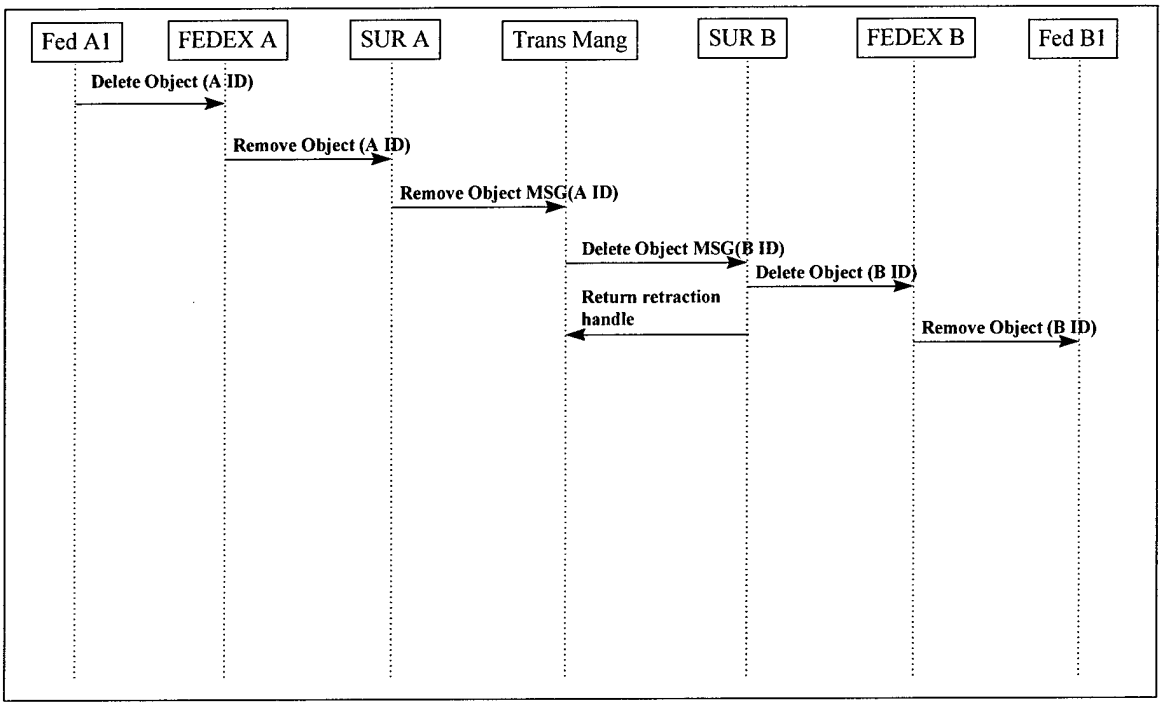
Federate A1 sends an interaction for an interaction class that corresponds to an interaction in federation FEDEX B as defined by the FOMAT.



1. Federate A1 invokes a send interaction to the FEDEX A.
2. SUR A receives a receive interaction from Federate A1.
3. SUR A passes a received interaction message to the Transformation Manager including the interaction class , the interaction parameter and value pairs, and the retraction handle.
4. The Transformation Manager passes a send interaction to SUR B for the corresponding FEDEX B interaction class, the translated parameter list (from FEDEX A parameters to FEDEX B parameters), and the transformed parameter values as specified by the FOMAT.
5. SUR B issues an send interaction to FEDEX B.
6. SUR B returns the retraction handle corresponding to the Send Interaction event to be associated with corresponding send interaction retraction handle for FEDEX A.
7. Federate B1 receives an receive interaction.

Delete Object

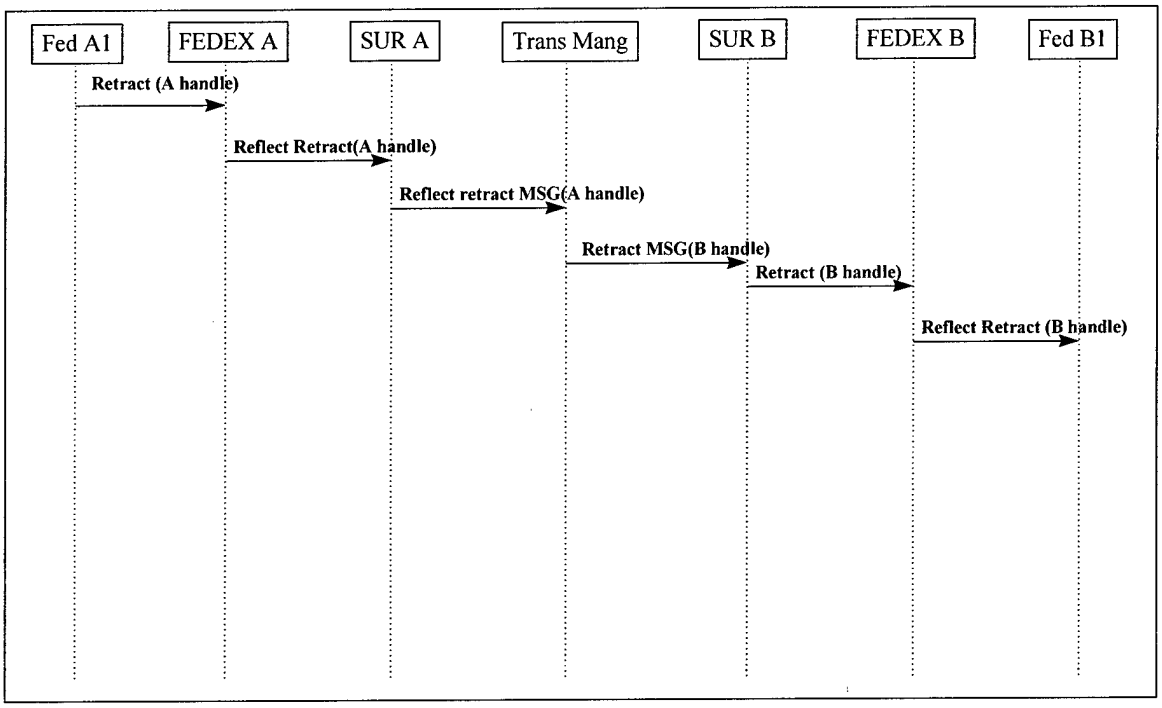
Federate A1 issues a delete object for an object class that corresponds to an object class in federation FEDEX B as defined by the FOMAT.



1. Federate A1 invokes a Delete Object to the FEDEX A.
2. SUR A receives a remove object from Federate A1.
3. SUR A passes a remove object message to the Transformation Manager including the object ID to be removed.
4. The Transformation Manager passes an delete object to SUR B for the corresponding FEDEX B object instance.
5. SUR B issues a delete object to FEDEX B.
6. SUR B returns the retraction handle corresponding to the delete object event to be associated with corresponding delete object retraction handle for FEDEX A.
7. Federate B1 receives an remove object.

Retract

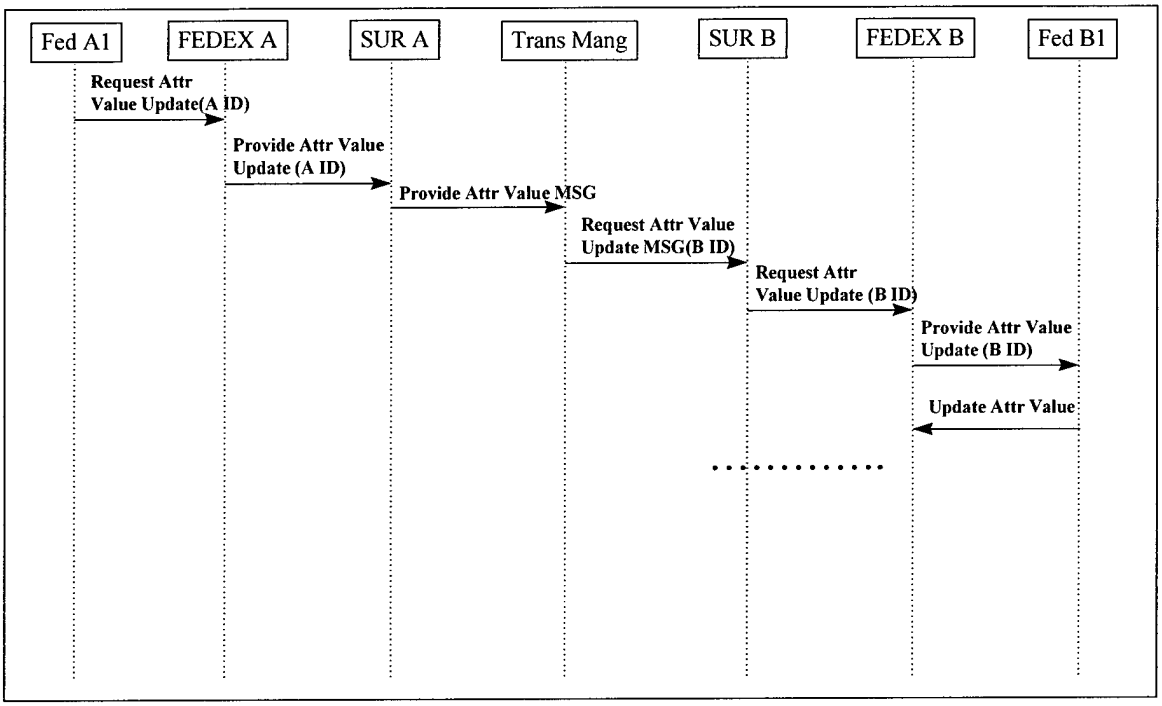
Federate A1 issues a retraction for an RTI event that corresponds to an event that implemented by the Bridge Federate between federations FEDEX A and B.



1. Federate A1 invokes a Retract to the FEDEX A.
2. SUR A receives a Reflect Retraction.
3. SUR A passes a reflect retraction to the Transformation Manager including the retract handle for FEDEX A.
4. The Transformation Manager passes an retract message to SUR B for the FEDEX B retraction handle corresponding to the given FEDEX A retraction handle.
5. SUR B issues an retract to FEDEX B.
6. Federate B1 receives an reflect retraction.

Request Attribute Value Update

Federate A1 issues a request attribute value update for an object instance that corresponds to a federation FEDEX B object instance as defined by the FOMAT.



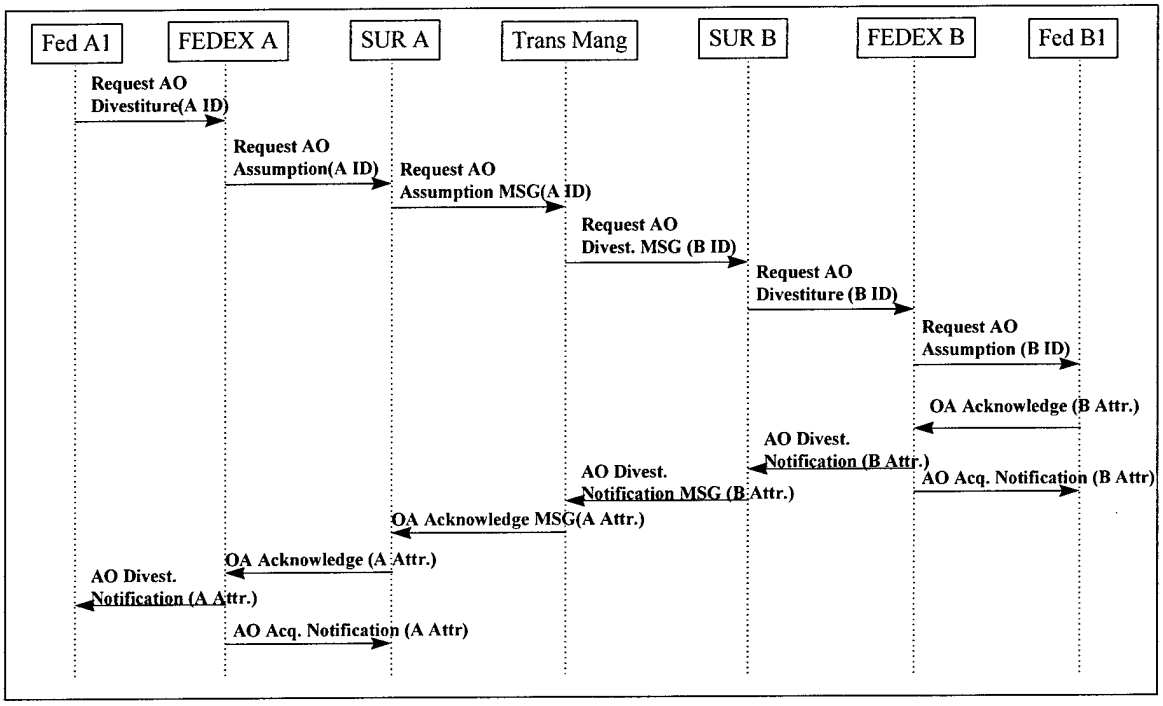
1. Federate A1 invokes a Request Attribute Value Update to the FEDEX A.
2. SUR A receives a Provide Attribute Value Update from FEDEX A.
3. SUR A passes a provide attribute value update message to the Transformation Manager including the object ID, and set of attribute to be queried.
4. The Transformation Manager passes an request attribute value update message to SUR B for the corresponding FEDEX B object instance and the translated attribute list (from FEDEX A attributes to FEDEX B attributes).
5. SUR B issues a request attribute value update to FEDEX B.
6. Federate B1 receives an Provide attribute value update and issues the requested Update Attribute value call.
7. The operations continues as shown in the update existing object attribute value operation.

1.2.4 Ownership Management

The ownership management group of services allow federates to transfer ownership of object attributes and can be implemented across a Bridge Federate. However, the Query Attribute Ownership services does not involve the Bridge Federate. If the object instance requested belongs to another federation, the FEDEX of the requesting federate will return the federation's surrogate as the owner.

Divesting Attribute Ownership

Federate A1 issues a request attribute ownership divestiture either specifying SUR A as the new owner or requests negotiation for the ownership. Note that the Interface Specification Version 1.1 does not support this activity for the Bridge Federate as noted below.



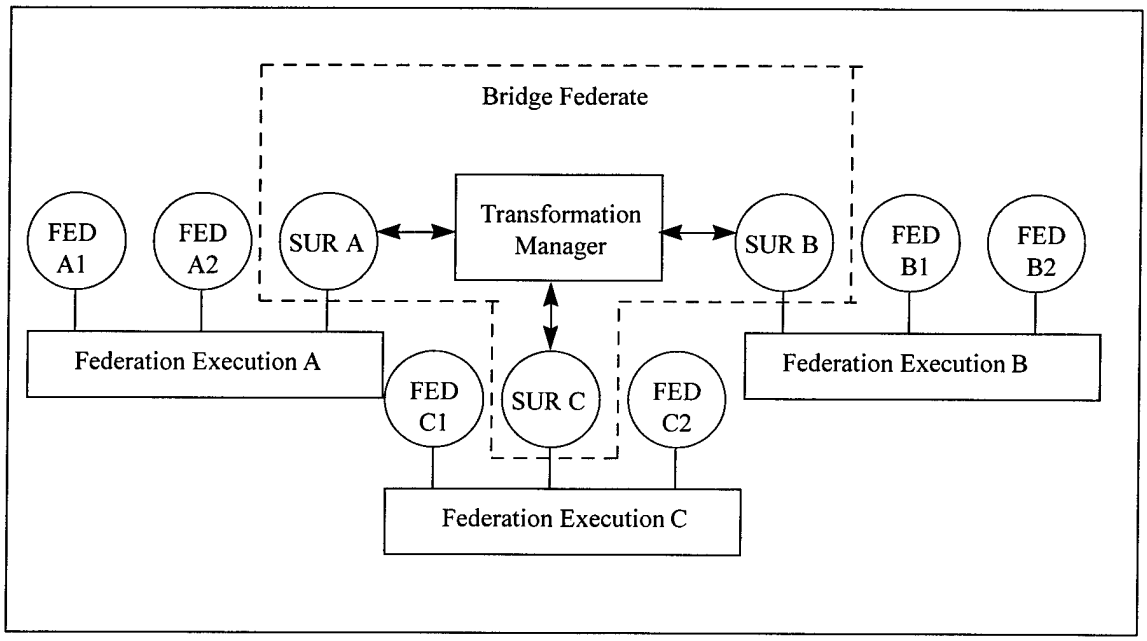
1. Federate A1 issues a Request Attribute Ownership Divestiture of a specific FEDEX A object instance and attributes. Federate may either explicitly request SUR A as the Federate to take ownership or request a negotiation to take place.
2. SUR A receives a request for Attribute Ownership Assumption for the object instances' attributes. *Note that in the Interface Specification, this RTI initiated services is a blocking two-way call. Given the amount of time this request to FEDEX B will take, it is recommended that this service be changed to a one way call and that a new Attribute Ownership Acknowledge service be added.*
3. SUR A sends a Request Attribute Ownership Assumption message to the Transformation Manager.
4. The transformation manger sends a Request Attribute Ownership Divestiture message to SUR B indicating the specific FEDEX B object instance and attributes corresponding to those provided by FEDEX A.
5. SUR B issues a Request Attribute Ownership Divestiture for the given FEDEX B object instances and attributes.
6. Federate B1 receives a request for Attribute Ownership Assumption for the object instances' attributes.
7. Federate B1 issues an Attribute Ownership Acknowledge indicating the list of attributes that it wishes to take ownership. Note that Federate B1 does not actually

own these attributes as this point. Also federate B1 will issue an Attribute Ownership Acknowledge with a NULL attributes list if it chooses not to accept ownership for any of the requested attributes.

8. SUR B receives an Attribute Ownership Divestiture Notification indicating the list of attributes that have been accepted by some federate in federation FEDEX B. Note that there may be multiple such calls received.
9. Federate B1 receives an Attribute Ownership Acquisition Notification indicating which attributes that it has acknowledged will it actually own. *Issues: assuming the RTI conducted the negotiation between multiple federates wishing the same attributes, how can this be extended to negotiating between multiple federations wanting ownership of the same set of attributes.*
10. SUR B issues an Attribute Ownership Divestiture Notification message to the transformation manager indicating the list of attributes now owned by federation FEDEX B.
11. The transformation manager issues an Attribute Ownership Acknowledge MSG to SUR A indicating the list of federation A attributes that FEDEX B wishes to own.
12. SUB A issues an Attribute Ownership Acknowledge to FEDEX A indicating the list of federation A attributes that it wishes to own on behalf of FEDEX B.
13. Federate A1 receives an Attribute Ownership Divestiture Notification indicating the list of attributes that have been accepted by some federate in federation FEDEX A.
14. SUR A receives an Attribute Ownership Acquisition Notification indicating which attributes for which it will actually receive ownership. *Again, there exists a potential negotiation conflict between FEDEX A and FEDEX B.*

Divesting Attribute Ownership Problem with Bridge Federate

The current HLA Interface Specification is not sufficient to support the ownership divestiture requirements for the Bridge Federate. The problem occurs when multiple fedexes are involved in the ownership neogotiations. The following funtional diagram and paragraph describe the problem in more detail.



Take for instance, federate A1 (Fed A1) issues a Request Attribute Ownership Divestiture of a specific FEDEX A object instance and attributes. Through the Bridge Federate, this Request Attribute Ownership Divestiture message is sent to Surrogate Federate B (SUR B) and SUR C. FEDEX A, B and C send a Request Attribute Ownership Assumption to their own federates, i.e. FEDEX A to FED A2 and SUR A, FEDEX B to B1 and B2, FEDEX C to C1 and C2. The problem occurs when there are federates in different FEDEXs that wants to take over the ownership of the attributes. We can examine a couple of cases :

Case 1 : If Fed C1 and Fed B1 both wants to take over the ownership, Fed C1 will send a Attribute Ownership Acknowledge to FEDEX C, FEDEX C will grant that ownership to Fed C1 with an Attribute Ownership Acquisition Notification. In the same time, Fed B1 will also send a Attribute Ownership Acknowledge to FEDEX B, and the ownership will be granted to Fed B1 by FEDEX B. Now both Fed C1 and Fed B1 thinks they have the ownership of the attributes.

Case 2 : If Fed A2 and Fed B2 both wants to take over the ownership, Fed B2 will send an Attribute Ownership Acknowledge to FEDEX B, and FEDEX B will grant the ownership to Fed B2. Surrogate B will get the Attribute Ownership Divestiture Notification from FEDEX B and passes the message to the Transformation Manager and to Surrogate A. Surrogate A then sends an Attribute Ownership Acknowledge to FEDEX A to ask for the owneship. However, Fed A2 also has informed FEDEX A that it wants to take over the ownership. FEDEX A now conducts negotiation and suppose FEDEX A granted the ownership to Fed A2. Now both Fed A2 and B2 thinks they have the ownership of the attributes.

In order to solve this problem, additional interfaces are proposed to allow nomination by RTI in addition to the current arbitration method. These additional interfaces are not known to be in any currently proposed HLA interface specification. The originator of the Request Attribute Ownership Divestiture will decide whether to accept the RTI's nominee by the as the new owner of the attributes it wants to give away. The additional interfaces that are proposed are as follows :

a. *Request Attibute Ownership Divestiture with Confirmation -*

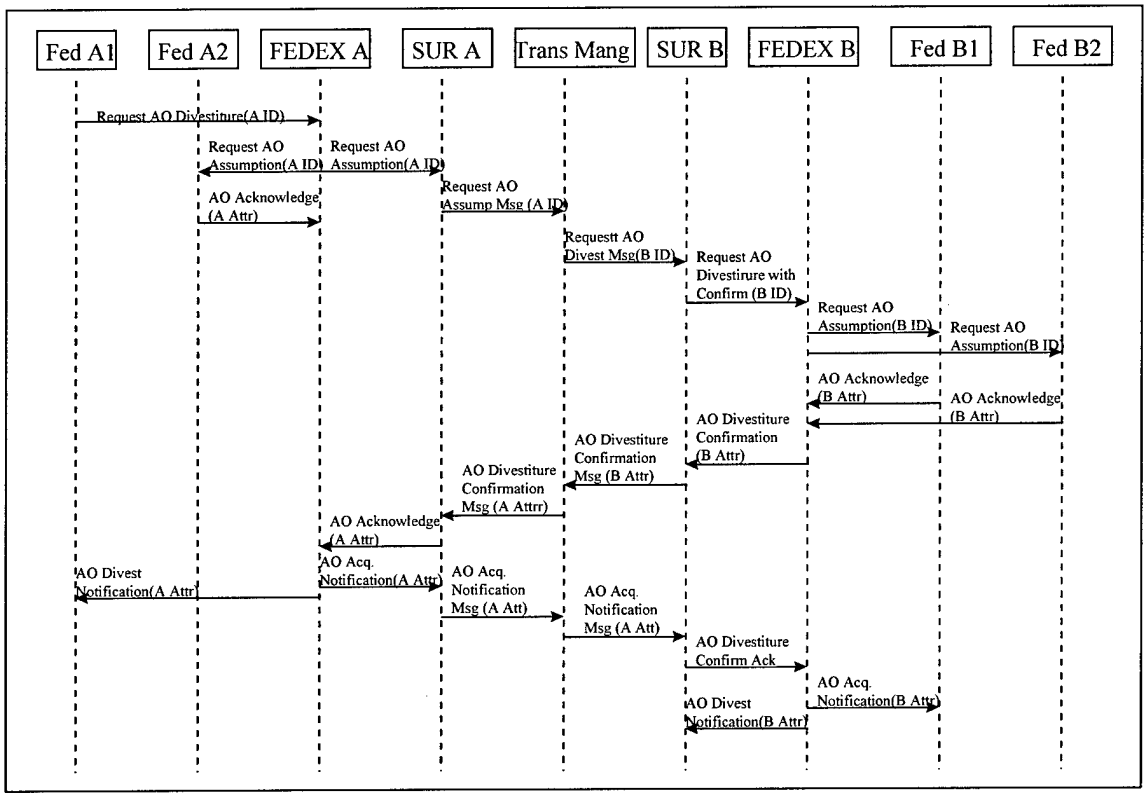
This interface will require the RTI to query the originator whether it should proceed with the ownership transfer after all the federates had acknowledge the Request Attribute Ownership Assumption.

b. *Attribute Ownership Divestiture Confirmation* -

This interface is initiated by the RTI to request confirmation from the originator that issued the Request Attribute Ownership Divestiture with Confirmation.

c. *Attribute Ownership Divestiture Confirmation Acknowledge* -
This interface is the acknowlegement of the Attribute Ownership Divest Confirmation

The following use case shows an example of how the new interface can accomplish the divesting attribute ownership across multiple federation executions. Federate A1 issues a request attribute ownership divestiture. Fed A2, Fed B1 and Fed B2 want to be the new owner of the attributes. Eventually, Fed B1 obtains the new ownership of the attributes.

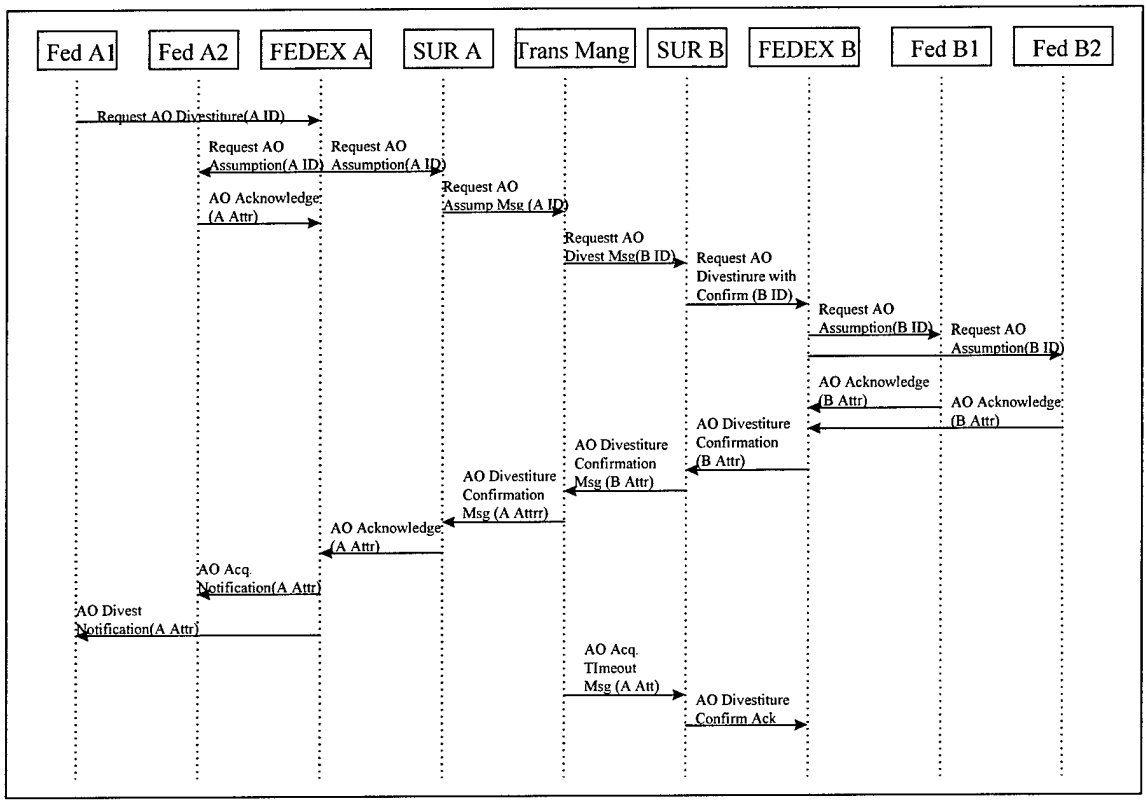


Description of the Use Case

1. Federate A1 issues a Request Attribute Ownership Divestiture of a specific FEDEX A object instance and attributes. In this case, Federate A1 request a negotiation to take place.
2. SUR A and Fed A2 receive a request for Attribute Ownership Assumption for the object instances' attributes.
3. Fed A2 returns the Attribute Ownership Assumption request with a list of attributes, A Attr.
4. SUR A sends a request Attribute Ownership Assumption message to the Transformation Manager.

5. The Transformation Manager sends a request Attribute Ownership Divestiture message to SUR B indicating the specific FEDEX B object instance and attributes corresponding to those provided by FEDEX A.
6. SUR B issues a Request *Attribute Ownership Divestiture with Confirmation* for the given FEDEX B object instances and attributes.
7. Federate B1 and B2 receive a request for Attribute Ownership Assumption for the object instances' attributes.
8. Both Federate B1 and B2 respond with an Attribute Ownership Acknowledge indicating the list of attributes that it wishes to take ownership.
9. After RTI conducted the negotiations, in this case, Fed B1 is negotiated as the potential new owner of the attribute, FEDEX B sends an *Attribute Ownership Divestiture Confirmation* with B attributes to SUR B.
10. SUR B sends a Attribute Ownership Divestiture Confirmation message to Transformation Manager.
11. The Transformation Manager sends a Attribute Ownership Divestiture Confirmation message to SUR A with A attributes.
12. SUR A returns the Attribute Ownership Assumption request (step 2) with a list of attriibutes, A Attr.
13. RTI conducts the negotiations, and in this case, SUR A is chosen as the new owner of the attributes. The FEDEX A then sends an Attribute Ownership Acquisition Notification to SUR A and an Attribute Ownership Divest Notification to Fed A1.
14. SUR A sends an Attribute Ownership Acquistion Notifiaction message to Translation Manager with A attributes.
15. Translation Manager sends an Attribute Ownership Acquistion Notification message to SUR B.
16. SUR B sends an *Attribute Ownership Divestiture Confirmation Acknowledge* (step 9) back to the FEDEX B.
17. FEDEX B then issues an Attribute Ownership Acquisition Notification to Fed B1 and an Attribute Ownership Divest Notification to SUR B.

The following use case is similar to the first use case, Federate A1 issues a request attribute ownership divestiture. Fed A2, Fed B1 and Fed B2 want to be the new owner of the attributes. This example, Fed A2 obtains the new ownership of the attributes.



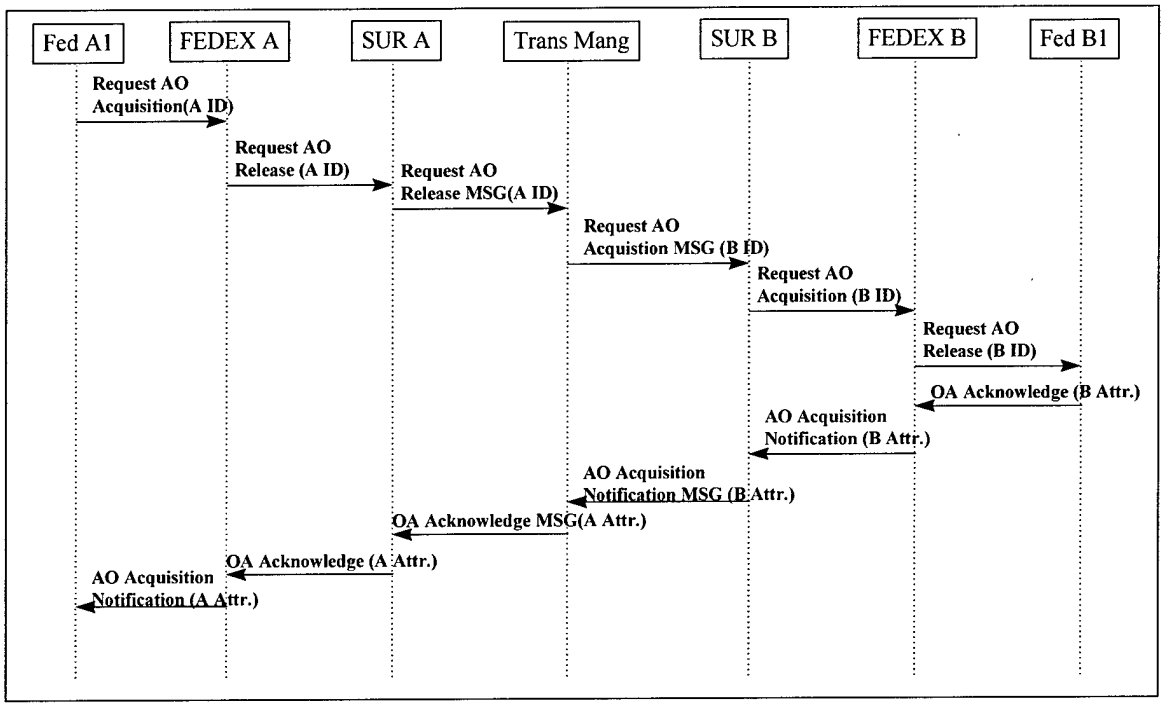
Description of the Use Case

1. Federate A1 issues a Request Attribute Ownership Divestiture of a specific FEDEX A object instance and attributes. In this case, Federate A1 request a negotiation to take place.
2. SUR A and Fed A2 receive a request for Attribute Ownership Assumption for the object instances' attributes.
3. Fed A2 returns the Attribute Ownership Assumption request with a list of attributes, A Attr.
4. SUR A sends a request Attribute Ownership Assumption message to the Transformation Manager.
5. The Transformation Manager sends a request Attribute Ownership Divestiture message to SUR B indicating the specific FEDEX B object instance and attributes corresponding to those provided by FEDEX A.
6. SUR B issues a Request *Attribute Ownership Divestiture with Confirmation* for the given FEDEX B object instances and attributes.
7. Federate B1 and B2 receive a request for Attribute Ownership Assumption for the object instances' attributes.
8. Both Federate B1 and B2 respond with an Attribute Ownership Acknowledge indicating the list of attributes that it wishes to take ownership.
9. After RTI conducted the negotiations, in this case, Fed B1 is negotiated as the potential new owner of the attribute, FEDEX B sends an *Attribute Ownership Divestiture Confirmation* with B attributes to SUR B.
10. SUR B sends a Attribute Ownership Divestiture Confirmation message to Transformation Manager.
11. The Transformation Manager sends a Attribute Ownership Divestiture Confirmation message to SUR A with A attributes.

12. SUR A returns the Attribute Ownership Assumption request (step 2) with a list of attriibutes, A Attr.
13. RTI conducts the negotiationsm and in this case, Fed A2 is chosen as the new owner of the attributes.
14. After a timeout period, the Trans Mang sends a Attribute Ownership Acquistion Timeout message to SUR B.
15. SUR B sends an *Attribute Ownership Divestiture Confirmation Acknowlege* to deny the divestiture nomination (step 9) back to FEDEX B.

Acquiring Attribute Ownership

Federate A1 issues a request attribute ownership acquisition for a set of object attributes that are owned by federate B1 in FEDEX B. Note that the Interface Specification Version 1.1 does not support this activity for the Bridge Federate as noted below.



1. Federate A1 issues a Request Attribute Ownership Acquisition of a specific FEDEX A object instance and attributes.
2. SUR A receives a request for Attribute Ownership Release for the object instances' attributes. *Note that in the Interface Specification, this RTI initiated services is a blocking two-way call. Given the amount of time this request to FEDEX B will take, it is recommended that this service be changed to a one way call and that a new Attribute Ownership Acknowledge service be added.*
3. SUR A sends a Request Attribute Ownership Release message to the Transformation Manager.
4. The transformation manger sends a Request Attribute Ownership Acquisition message to SUR B indicating the specific FEDEX B object instance and attributes corresponding to those provided by FEDEX A.
5. SUR B issues a Request Attribute Ownership Acquisition for the given FEDEX B object instances and attributes.
6. Federate B1 receives a request for Attribute Ownership Release for the object instances' attributes.
7. Federate B1 issues an Attribute Ownership Acknowledge indicating the list of attributes that it agrees to relinquish ownership. Federate B1 will issue an Attribute Ownership Acknowledge with a NULL attribute list if it chooses not release any of the requested attributes.

8. SUR B receives an Attribute Ownership Acquisition Notification indicating the list of attributes that has been awarded ownership.
9. SUR B issues an Attribute Ownership Acquisition Notification message to the transformation manager indicating the list of attributes now owned by federation FEDEX A.
10. The transformation manager issues an Attribute Ownership Acknowledge MSG to SUR A indicating the list of federation A attributes that FEDEX A now owns.
11. SUB A issues an Attribute Ownership Acknowledge to FEDEX A indicating the list of federation A attributes that it wishes to release to federate A1 on behalf of FEDEX B.
12. Federate A1 receives an Attribute Ownership Acquisition Notification indicating the list of attributes that it now owns.

1.2.5 Time Management

Time management is concerned with the mechanisms for controlling the advancement of federates along the federation time axis during the execution. The Use-Cases for the time management services of the bridge federate have not yet been completed.

1.2.6 Data Distribution Management

The Use-Cases for the Data Distribution Management services of the bridge federate have not yet been completed.

1.3 Requirements

The requirements specified in this section depend upon the context and content described in the Concept of Operation. Therefore, these requirements do not stand alone and must be used and tested in conjunction with the concept of operation.

1.3.1 FOM Mappings/Transformations Specification (FOMAT)

The FOMAT shall specify the number of federations to be bridged and the specifics about that bridging function. Specifically, the FOMAT shall define the federation name, object class name, attribute name and value transformation for each class attribute to be communicated across the bridge for each FEDEX that requires that object attribute value. The FOMAT shall define the federation name, interaction class name, parameter name and parameter transformation for each interaction to be communicated across the bridge for each FEDEX that requires that interaction. The FOMAT shall specify the data type for each object attribute and each interaction parameter within each FEDEX that requires this object attribute or interaction. The FOMAT shall also specify the class attributes that are required for the one-to-many object class mappings.

1.3.2 Surrogate Federate

Each surrogate federate shall behave as a normal federate in that it will provide an Federate Ambassador providing methods for the RTI initiated services defined in the

HLA Interface Specification Version 1.1. The behavior of these services are as defined in the Concept of Operations.

Each surrogate federate shall provide an RTI Ambassador that uses the Federation Management services Join Federation Execution and Resign Federation execution. These services shall be invoked in response to the join or resign message from the Transformation Manager as described in the Concept of Operations.

Each surrogate federate's RTI Ambassador shall also uses the Declaration Management services Publish Object, Publish Interaction, Subscribe Object Class Attribute, and Subscribe Interaction Class to be invoked in response to the publish and subscribe messages from the Transformation Manager as described in the Concept of Operations. Each surrogate federate shall send control updates and control interactions messages to the transformation manager as specified in the concept of operations.

Each surrogate federate's RTI Ambassador shall also use the Object Management services Request ID, Register Object, Update Attribute Values, Send Interaction, Delete Object, Request Attribute Value Update, and Retract to be invoked in response to the corresponding messages from the Transformation Manager as described in the Concept of Operations. Each surrogate federate shall send Discover Object, Reflect Attribute, Receive Interaction, Delete Object, Reflect Retraction, and Provide Attribute Value messages to the Transformation Manager as specified in the Concept of Operations. The Object Management services of Change Attribute Transportation Type, Change Attribute Order Type, Change Interaction Transportation Type and Change Interaction Order Type will not be addressed by these requirements.

Each surrogate federate's RTI Ambassador shall also use the Ownership Management services Request Attribute Ownership Divestiture and Request Attribute Ownership Acquisition to be invoked in response to the corresponding messages from the Transformation Manager as described in the Concept of Operations. Each surrogate federate shall send Request Attribute Ownership Assumption, Attribute Ownership Divestiture Notification, Attribute Ownership Acquisition Notification, Request Attribute Ownership Release, and Attribute Ownership Acknowledgment messages to the Transformation Manager as specified in the Concept of Operations.

1.3.3 Transformation Manager

The Transformation Manager shall accept instructions about the number of surrogate federates, the name of each FEDEX associated with each surrogate federate, and the FOMAT information described previously. The Transformation Manager shall, when instructed, send a join message to each surrogate federate indicating which FEDEX the surrogate will join. The Transformation Manager shall, when instructed, send a resign message to each surrogate federate indicating that the surrogate will resign from its FEDEX. Subsequent to sending the join messages, the Transformation Manager shall send to each surrogate, the set of object, attributes, and interactions relevant to the surrogates FEDEX as defined by FOMAT.

The Transformation Manager shall maintain the correspondence between the RTI object identifier number for the same object in multiple FEDEXs. The Transformation Manager shall maintain the correspondence between the RTI retraction handles for the same scheduled event in multiple FEDEXs.

The Transformation Manager shall accept the Control Updates, Control Interactions, Discover Object, Reflect Attribute, Receive Interaction, Remove Object, Reflect Retract, Provide Attribute Value Update, Request Attribute Ownership Assumption, Attribute Ownership Divestiture Notification, Attribute Ownership Acquisition Notification, Request Attribute Ownership Release, and Attribute Ownership Acknowledgment messages as specified by the Concept of Operations. In response to the appropriate surrogate message (as defined in the concept of operations), the Transformation Manager shall provide the Subscribe Object Class Attribute, Subscribe Interaction Class, Register Object, Update Attribute, Send Interaction, Delete Object, Retract, and Request Attribute Value, Request Attribute Ownership Divestiture, and Request Attribute Ownership Acquisition messages to the corresponding surrogate. The corresponding surrogates are determined using the correspondence specified by the FOMAT.

Appendix C – DESIGN REVIEW PRESENTATION

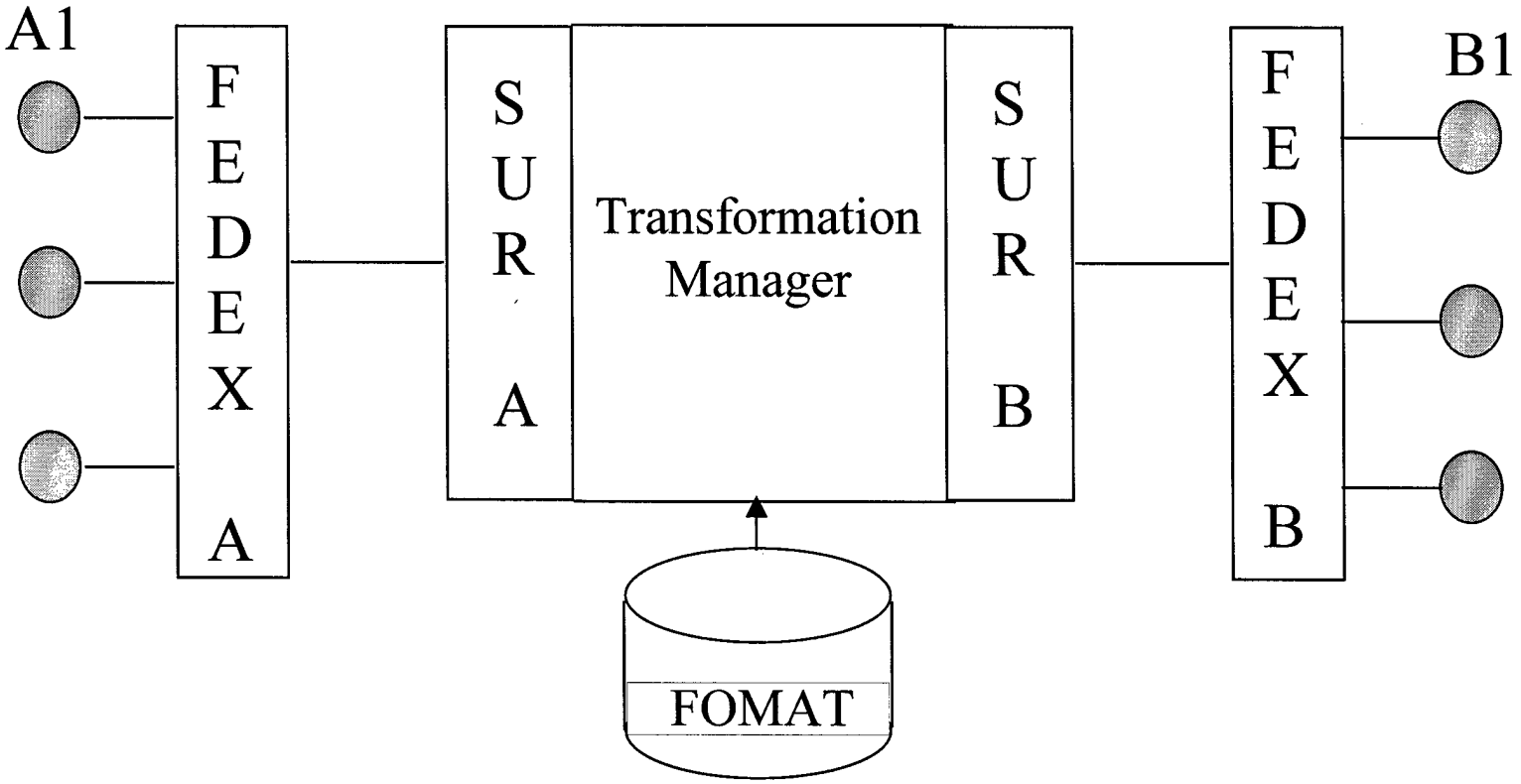
Bridge Federate Design Review

June 2, 1997

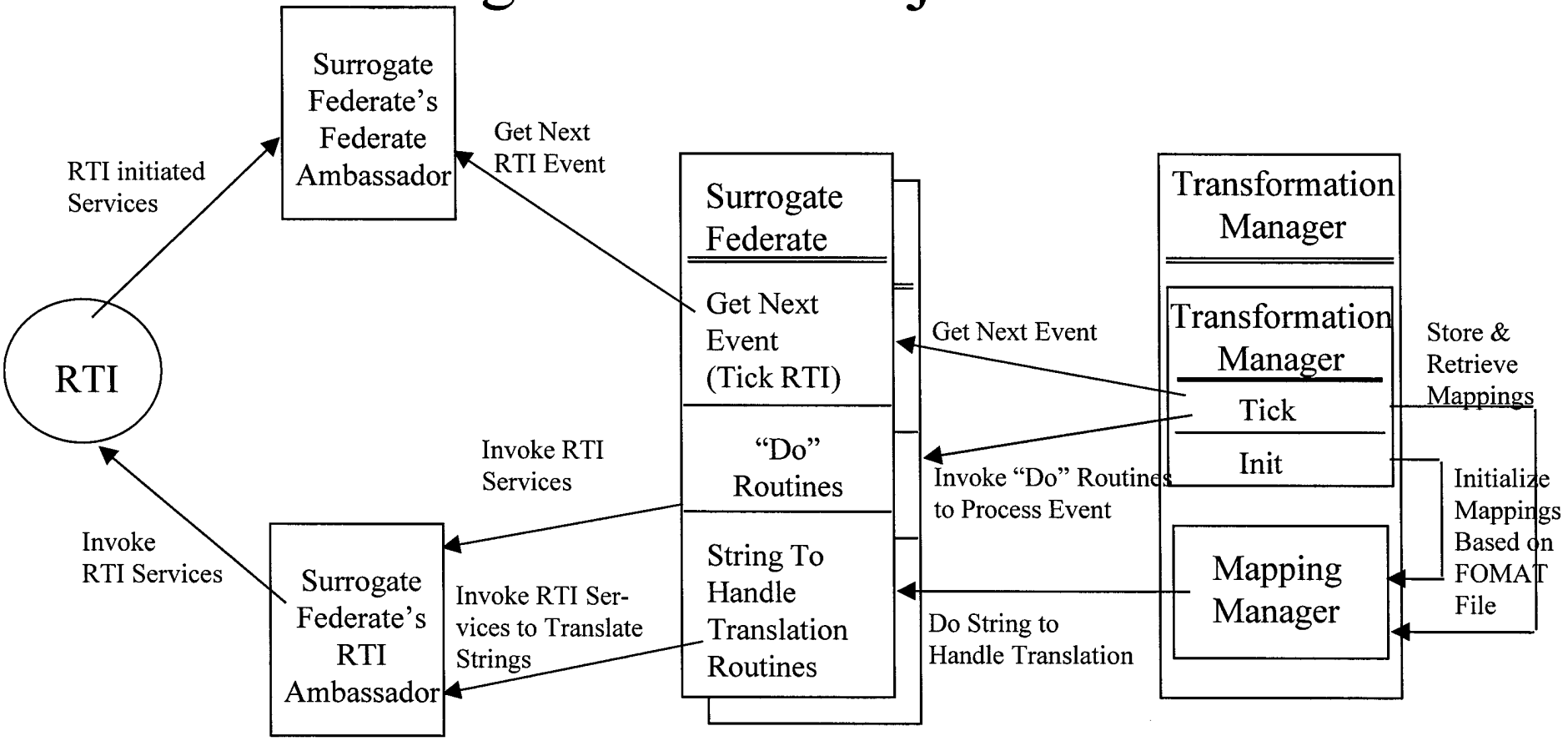
Bridge Federate Description

- The Bridge Federate combines multiple federation executions (FEDEXs)
- Internal components:
  - **surrogates (one for each federation it participates in), and**
  - **a transformer that coordinates among the surrogates.**
- Each FEDEX communicates with other FEDEXs using its own FOM and its Bridge Federate Surrogate.
- Each federate has no knowledge of the multiple FEDEXs or FOMs.
- The objects, attributes, and interaction mappings between FEDEXs is specified by the FOM Mappings/Transformations specification (FOMAT).
- Assumptions:
  - **Assumes that RTI Tick returns one event.**
  - **Assumes Save/Restore/Pause/Resume applied to all bridged FEDEXs.**

Bridge Federate Components



Bridge Federate Object Model




```
Main Processing Loop
   Transformation Manager.Tick
      for each surrogate federate loop
         next_event := surrogate federate.get_next_event;
         if next_event exists then process next_event;
      end loop;
```

Surrogate Federate Object

- Provides methods for the Transformation Manager to direct its activity within its FEDEX.
- Methods correspond to messages from TM/MM to Surrogate.
- Surrogate maintains state for Federation's Save/Restore, Pause/Resume status.
- Also saves/restores its own state.
- Get Next Event determines state change event; otherwise requests RTI Event Tick.

Surrogate Federate	
Name Federation_Handle Federate_Ambassador RTI_Ambassador Pause_Resume_State Set of Save_Restore_States	
Get_Next_Event Do_Join_Federation_Execution Do_Resign_Federation_Execution Do_Request_Pause Do_Pause_Achieved Do_Request_Resume Do_Resume_Achieved Do_Request_Federation_Save Do_Federation_Save_Achieved Do_Request_Restore Do_Restore_Achieved Do_Publish_Object_Class	Do_Publish_Interaction_Class Do_Subscribe_Object_Class_Attribute Do_Subscribe_Interaction_Class Do_Register_Object Do_Update_Attribute_Values Do_Send_Interaction Do_Delete_Object Do_Request_Attribute_Ownership_Divestiture Do_Request_Attribute_Ownership_Acquisition Get_Object_Class_Handle Get_Attribute_Handle Get_Interaction_Class_Handle Get_Parameter_Handle

Surrogate Federate Ambassador Object

- Implements the Federate Ambassador for each instance of the Surrogate Federate.
- Provides Get Next RTI Event that clears and returns data indicating the specific RTI initiated call from the last tick.
- Provides Methods for each RTI Initiated Service that stores the RTI initiated call data with this class.

Federate Ambassador For Surrogate Class
Identifier
Get_Next_RTI_Event Initiate_Pause Initiate_Resume Initiate_Federate_Save Initiate_Restore Start_Updates Stop_Updates Start_Interaction_Generation Stop_Interaction_Generation Discover_Object Reflect_Attribute_Values Receive_Interaction Remove_Object Provide_Attribute_Value_Update Reflect_Retraction Request_Attribute_Ownership_Assumption Attribute_Ownership_Divestiture_Notification Attribute_Ownership_Acquisition_Notification Request_Attribute_Ownership_Release Inform_Attribute_Ownership Time_Advance_Grant Change_Thresholds

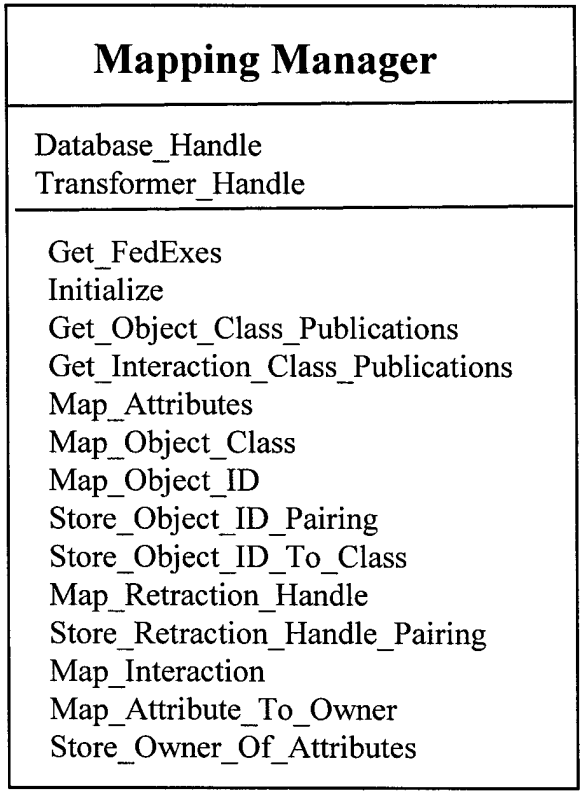
Transformation Manager Object

- TM is responsible for determining which FEDEX to bridge.
- TM is responsible for determining how to map a transaction in one FEDEX to 1 or more other FEDEXs.
- Join, Publish, Tick, and Resign are methods provided to the Main loop of the Bridge Federate.
- *Italics* methods support the Tick Method; these implement events in response to received surrogate messages.

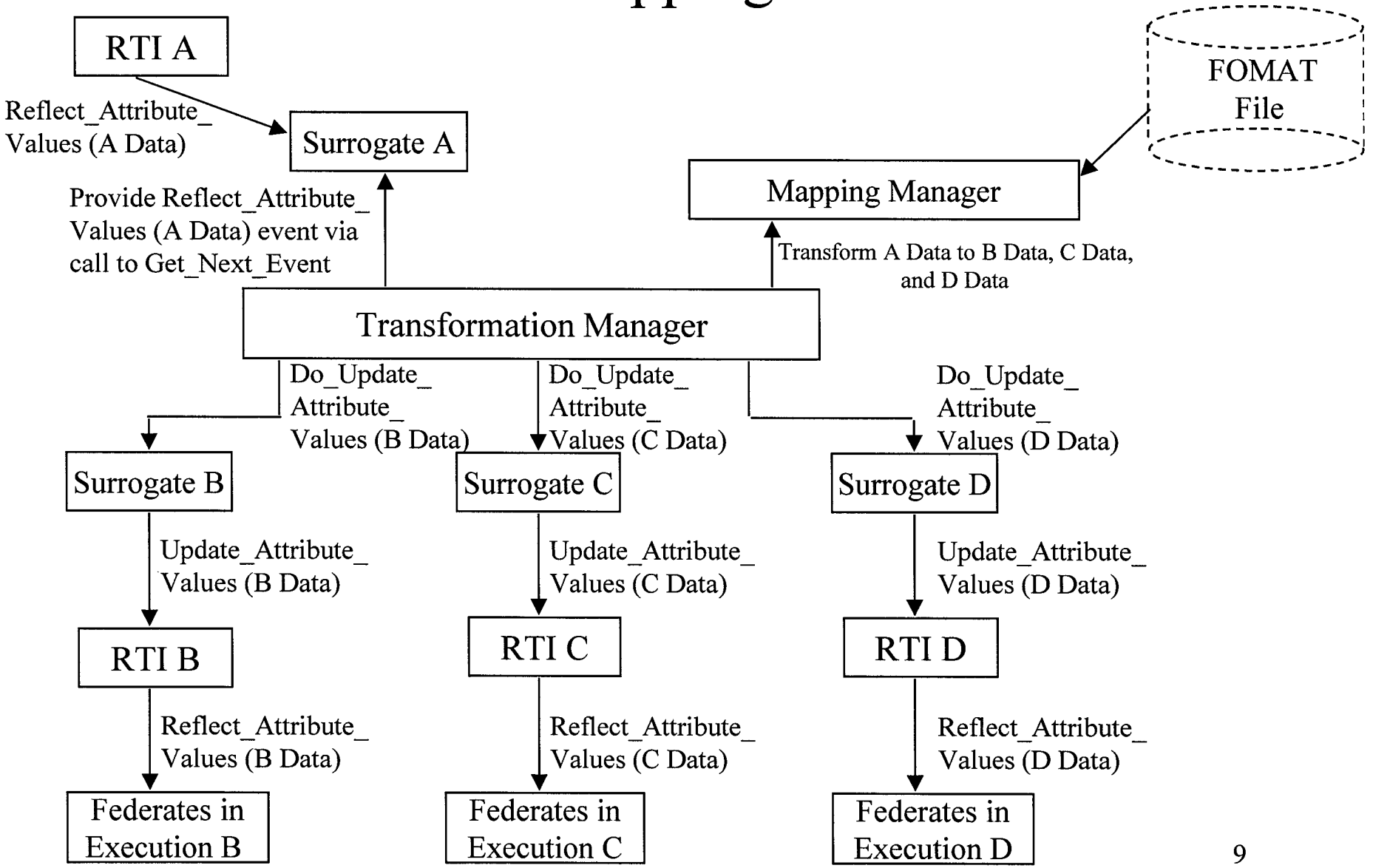
Transformation Manager
Surrogates Mapping_Manager
Join_Federation_Executions Publish Tick Resign_Federation_Executions *Received_Start_Updates* *Received_Stop_Updates* *Received_Start_Interaction_Generatioin* *Received_Stop_Interaction_Generation* *Propagate_Discover_Object* *Propagate_Reflect_Attribute_Values* *Propagate_Receive_Interaction* *Propagate_Remove_Object* *Propagate_Provide_Attribute_Value_Update* *Propagate_Reflect_Retraction* *Propagate_Initiate_Pause* *Pause_Achieved* *Propagate_Initiate_Resume* *Resume_Achieved* *Propagate_Initiate_Federate_Save* *Federate_Save_Achieved* *Propagate_Initiate_Restore* *Restore_Achieved*

Mapping Manager Object

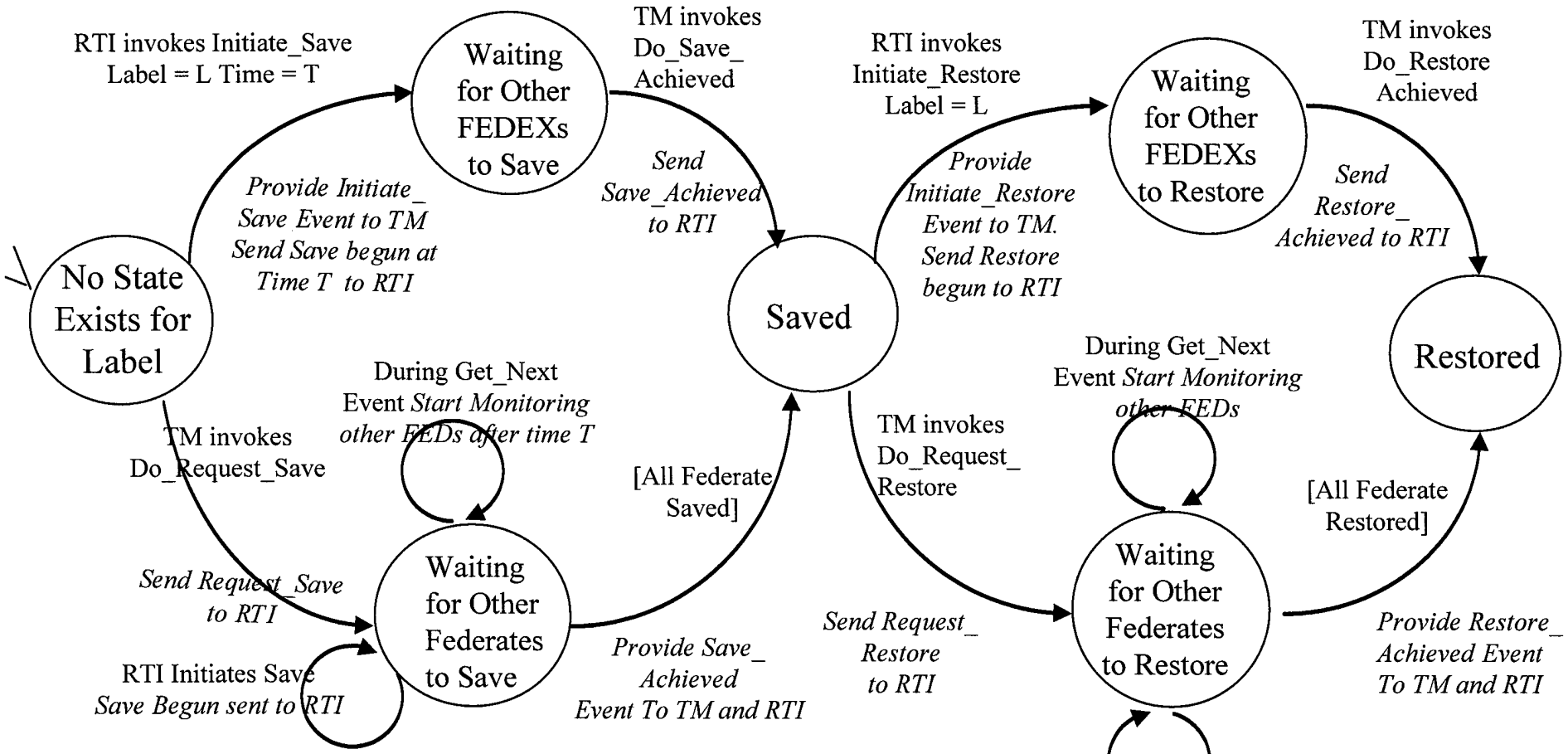
- Maintains all translations and transformations required to combine FEDEXs.
- Maintains the FOMAT Mapping/Transformations specification with name strings replaced by handles.
- Maintains the Object/Attribute ID and ownership Association between multiple FEDEXs.
- Maintains the Retraction Handle association between multiple FEDEXs.
- Will implement the value transformation process.
- Future: Time Management mapping implemented by MM object.



CONOPS: Mapping Attributes

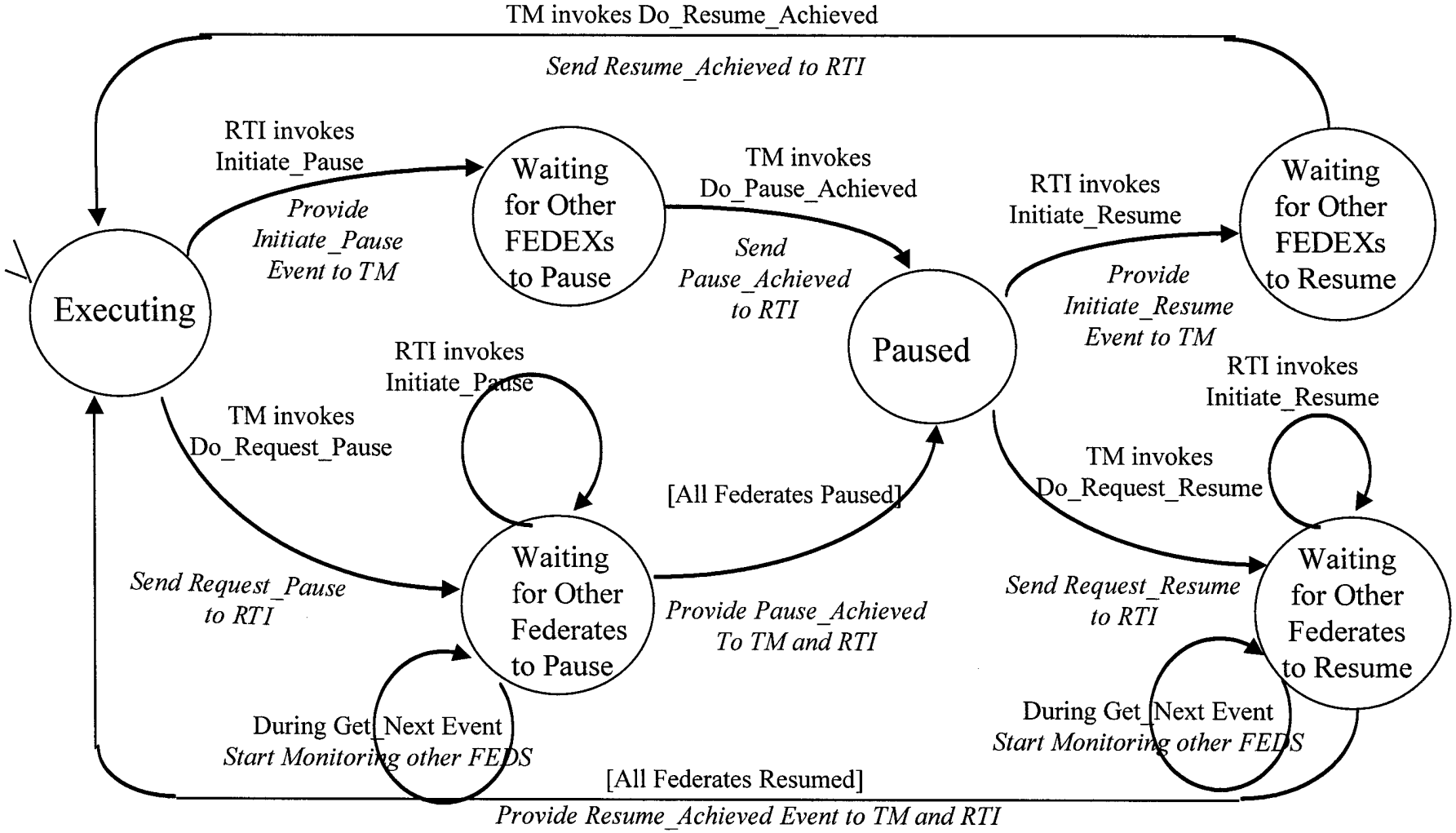


Surrogate Save/Restore State Diagram



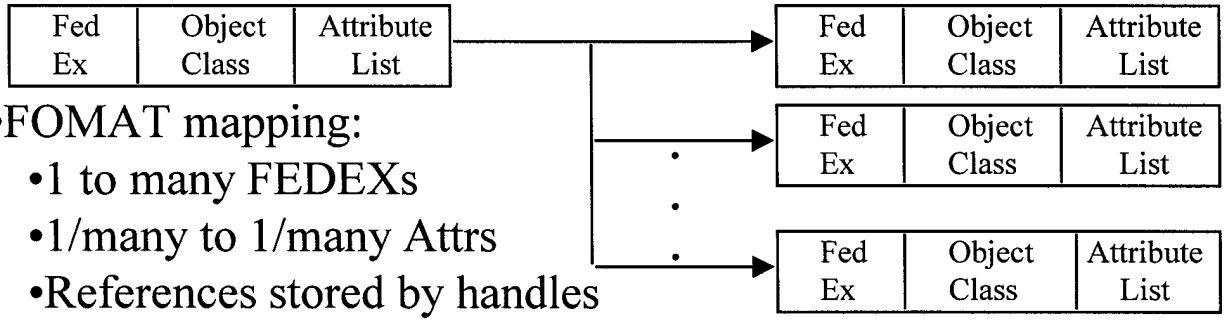
Note -- One State per Save Label
Legend --
Normal : Event invoked on Surrogate Federate
Italic : Actions taken by Surrogate Federate in response to Event
[ ] : Condition

Surrogate Pause/Resume State Diagram

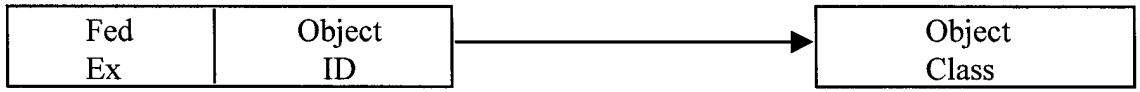


Legend
Normal : Event invoked on Surrogate Federate
Italic : Actions taken by Surrogate Federate in response to Event
[ ] : Condition

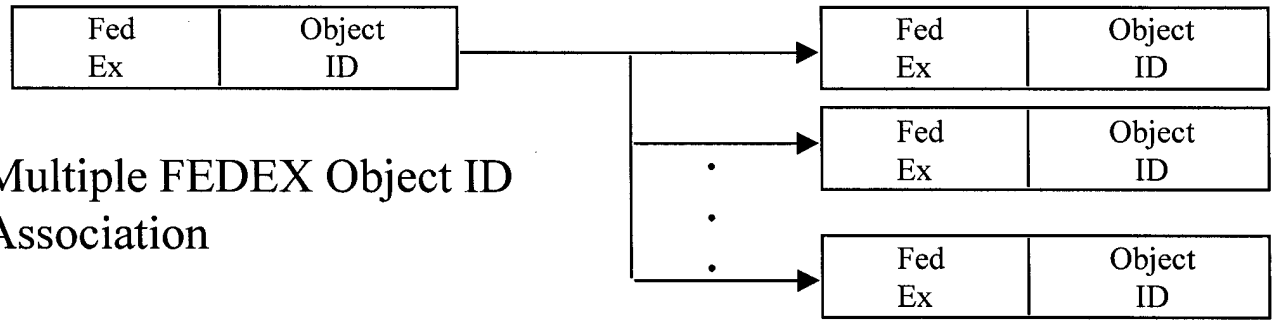
Transformation Mappings



- FOMAT mapping:
  - 1 to many FEDEXs
  - 1/many to 1/many Attrs
  - References stored by handles

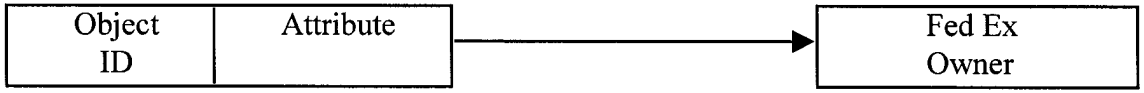


Object to Class handle for messages to Surrogates

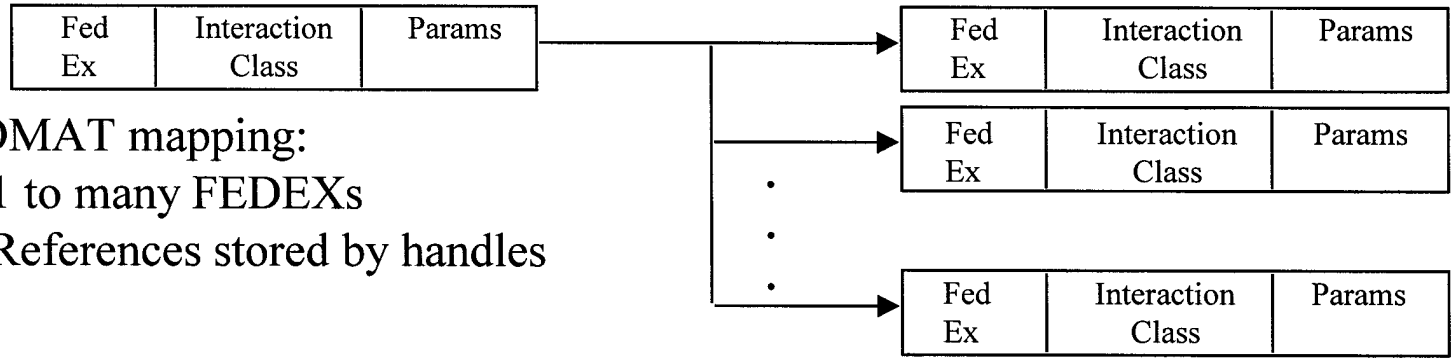


Multiple FEDEX Object ID Association

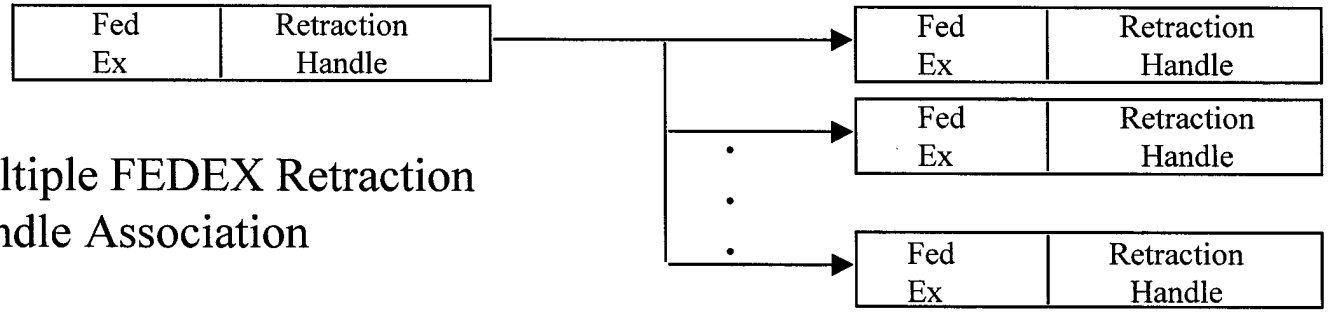
Transformation Mappings (cont'd)



Maintain all attribute owners for Provide Attr Value service

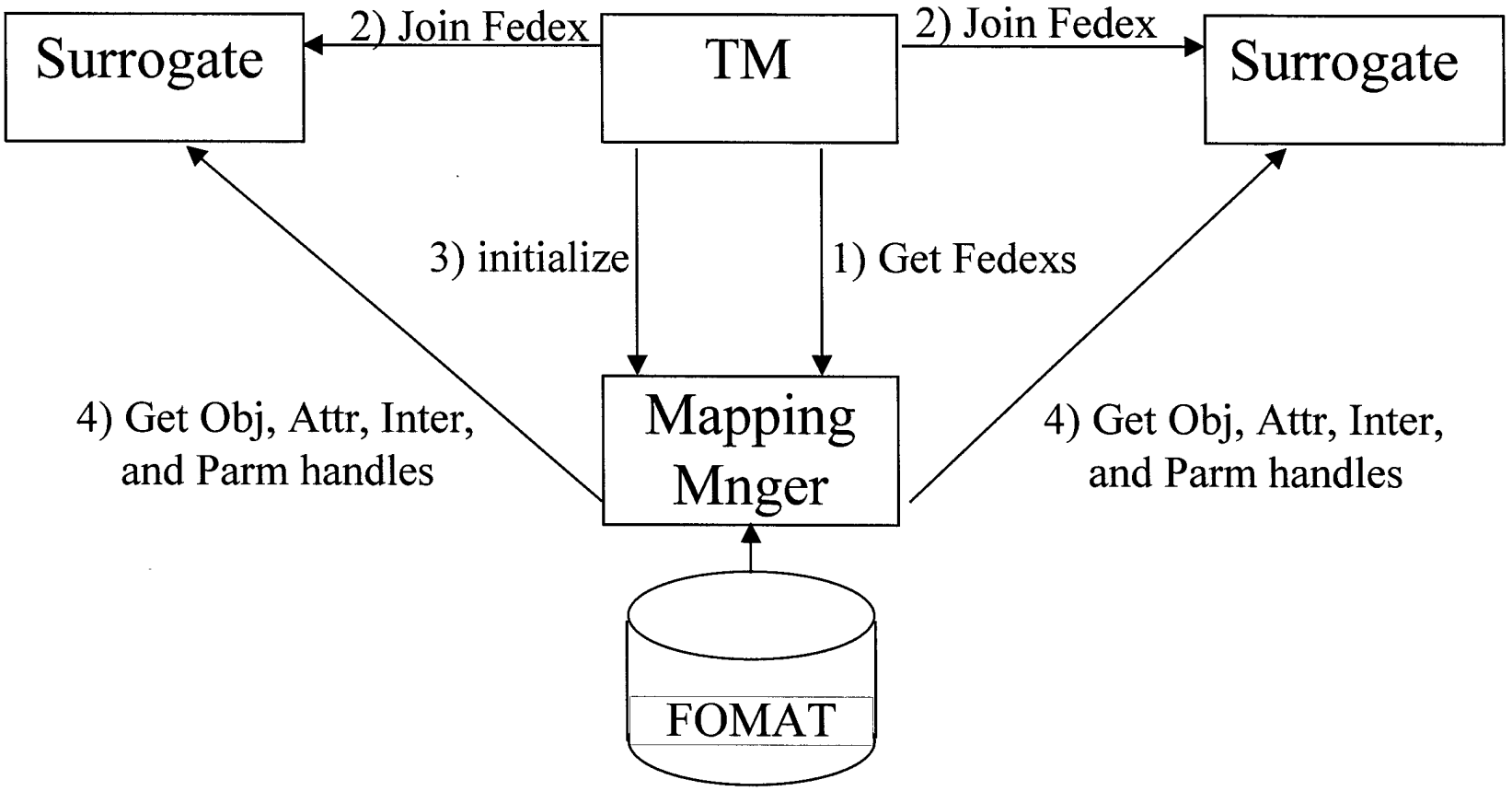


- FOMAT mapping:
  - 1 to many FEDEXs
  - References stored by handles



Multiple FEDEX Retraction Handle Association

Bridge Federate Initialization



FOM Mappings/Transformations Specification (FOMAT)

- Contains:
  - Federation data for each federation execution to be bridged. Only data to be mapped is specified in the federation data.
  - Attribute Mappings which specify the mapping from one or more attributes to one or more attributes
  - Interaction Mappings which specify the mapping from one interaction to another (there is no partial mapping of interactions)
  - Transformations - a transformation language will be used to specify the transformations between attribute values and between interaction parameter values.

Example of FOMAT file data

```
; Federation Data
(FEDEX-A
    (class X
        (attribute M, N complex1))
    ...
    (interaction D
        (parameter E Byte)
        (parameter F,G complex2))
    ...
)
(FEDEX-B
    (class Y
        (attribute P Boolean)
        (attribute Q Int16)
    )
    ...
    (interaction J
        (parameter K String(45))
        (parameter L Float32))
    ...
)
(FEDEX-C
    (class R
        (attribute Y Int16)
...
)
/* continued on next page */
```

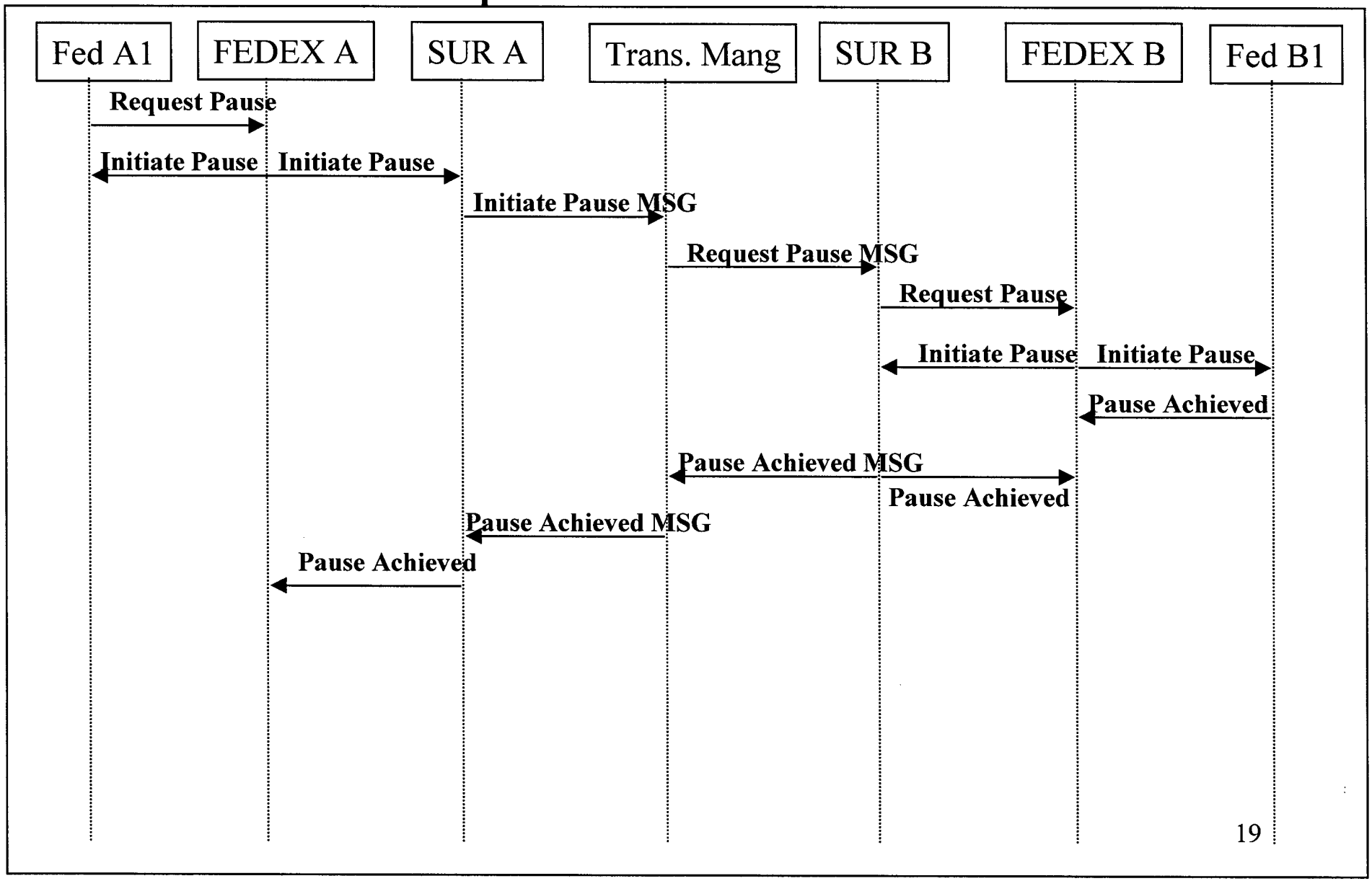
Example of FOMAT file data (cont'd)

```
;Complex type definitions
(complex1
                M Int32
                N Float64)
(complex2 .....)

; Transformation Information
(Data-Transform
    ((FEDEX-A X (M))       (FEDEX-C R (Y))     transformer-56)
    ((FEDEX-A X (M,N))     (FEDEX-B Y (P))     transformer-24)
    ((FEDEX-A D (E F G)    (FEDEX-B J  (K L))  transformer-33)

...
)
```

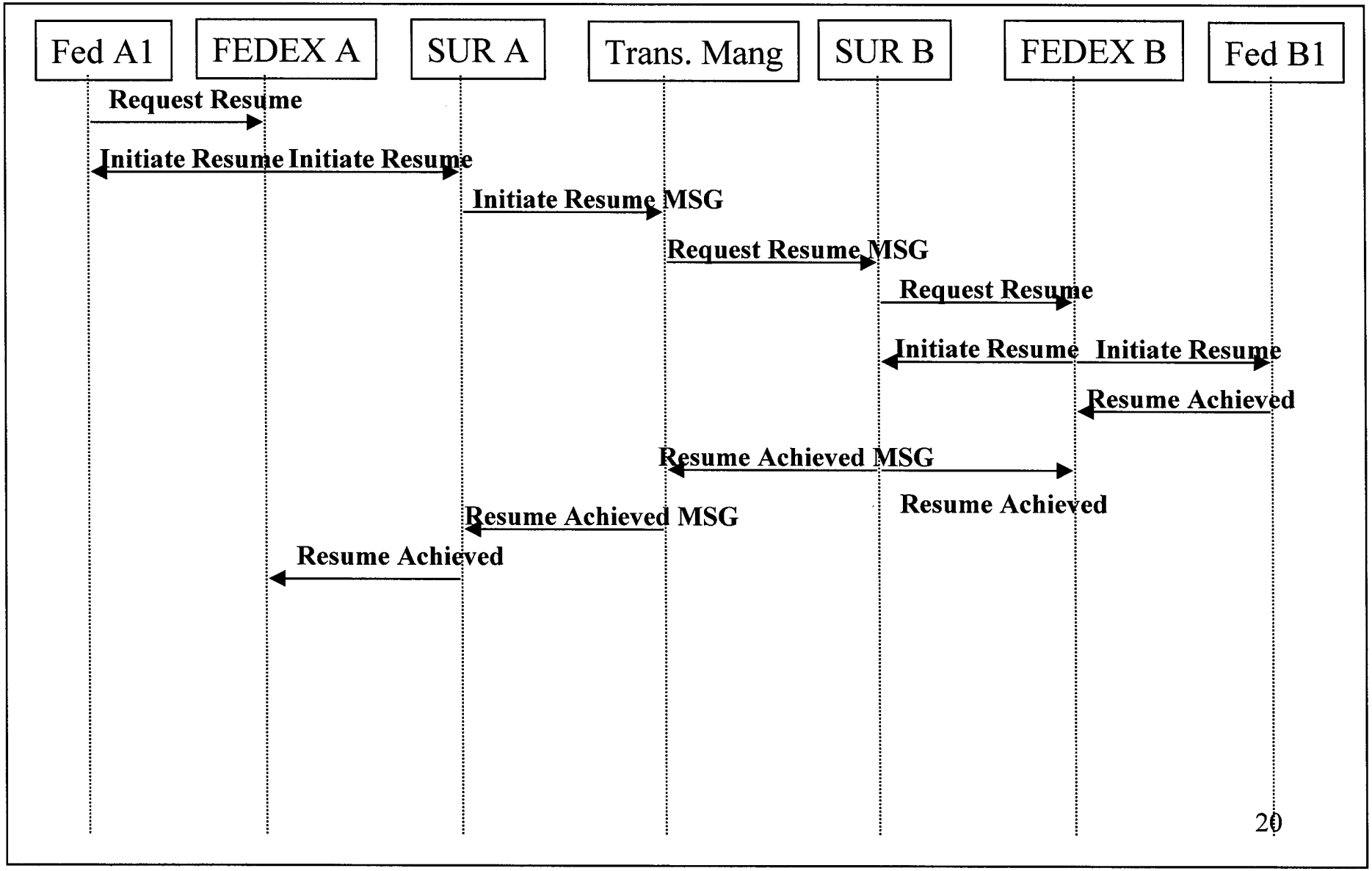
Design Issues

- HLA Ownership Management Services
  - **I/F Specification issues identified in Requirements Documents**
  - **Ownership Arbitration**
- Complex attribute transformation Issue
  - **Multiple attribute mapped to a single attribute in another FOM.**
  - **What should happen when not all required multiple attributes arrive in Update? Assume 1-to-1 mapping only for prototype.**
- Desired Experiment
  - **CCTT PPF FOM vs CCTT JPSD FOM?**
  - **CCTT PPF FOM vs ModSAF JPSD FOM?**
- Transformation Language
  - **Should an existing transformation language be used?**
  - **Are interpreters/parsers available?**
- Time Management
  - **Postpone to phase 2 of prototyping.**

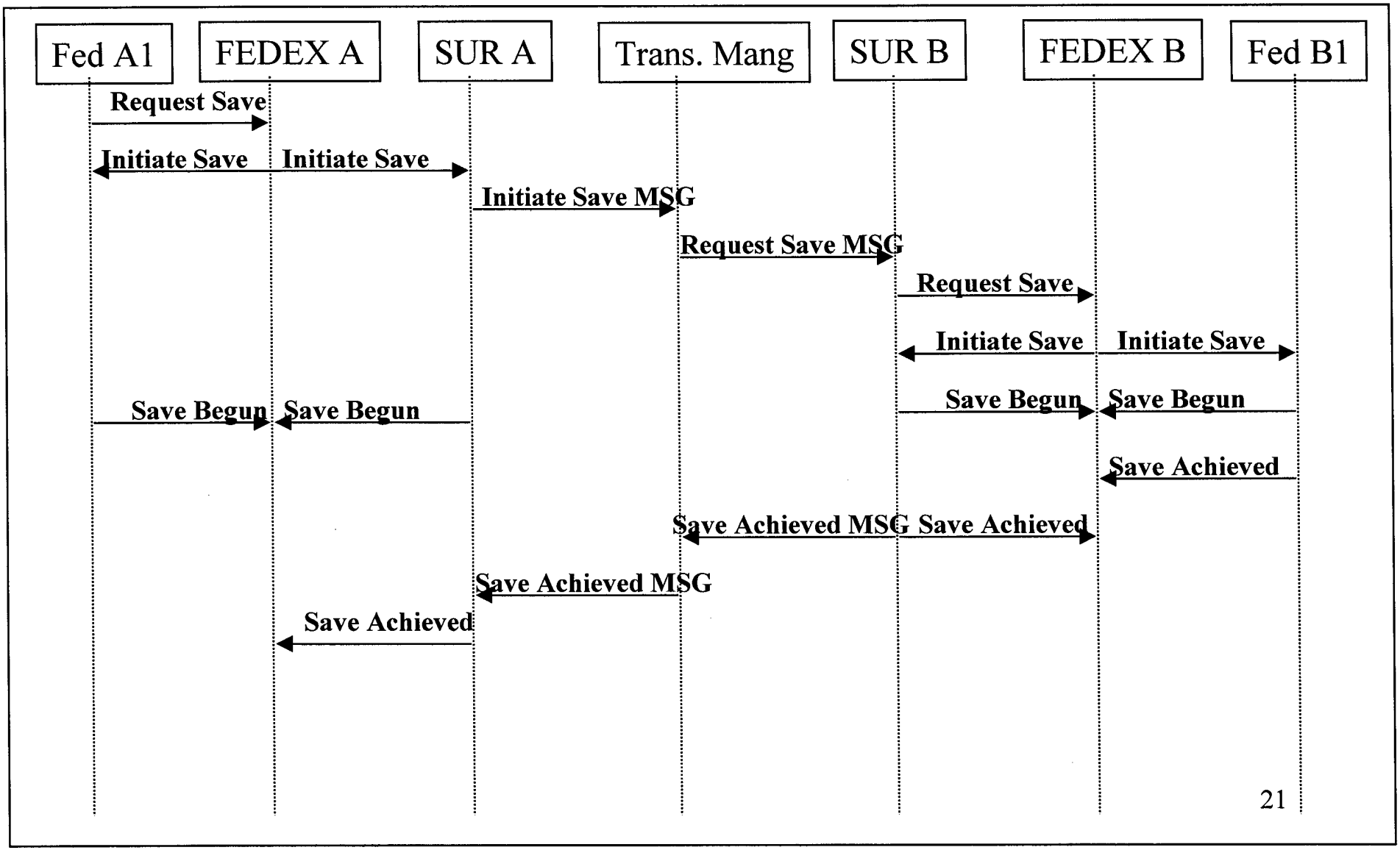
Backups: Event Flows: Pause



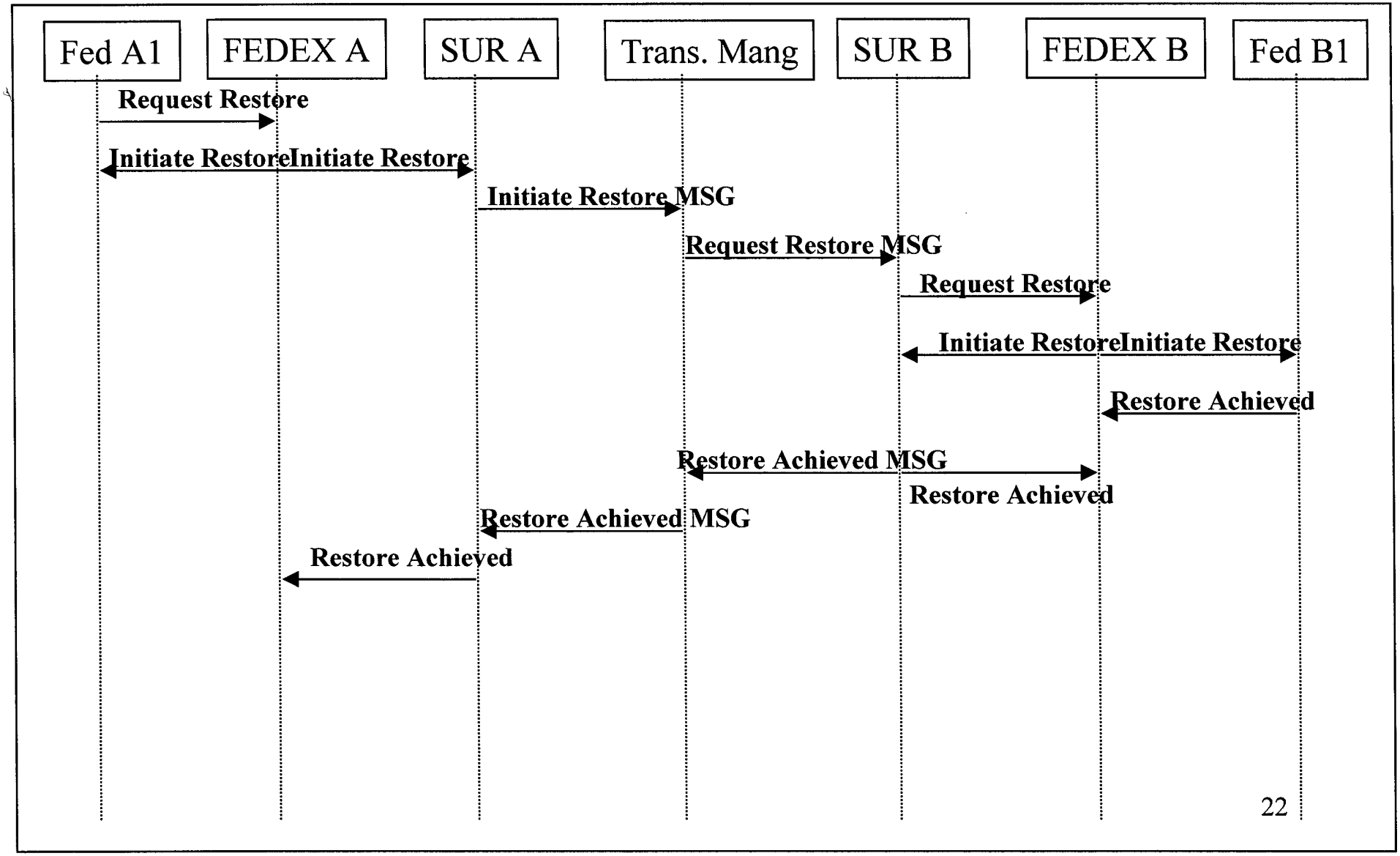
Resume


Fed A1
FEDEX A
SUR A
Trans. Mang
SUR B
FEDEX B
Fed B1
Request Resume
Initiate Resume Initiate Resume
Initiate Resume MSG
Request Resume MSG
Request Resume
Initiate Resume Initiate Resume
Resume Achieved
Resume Achieved MSG
Resume Achieved
Resume Achieved MSG
Resume Achieved

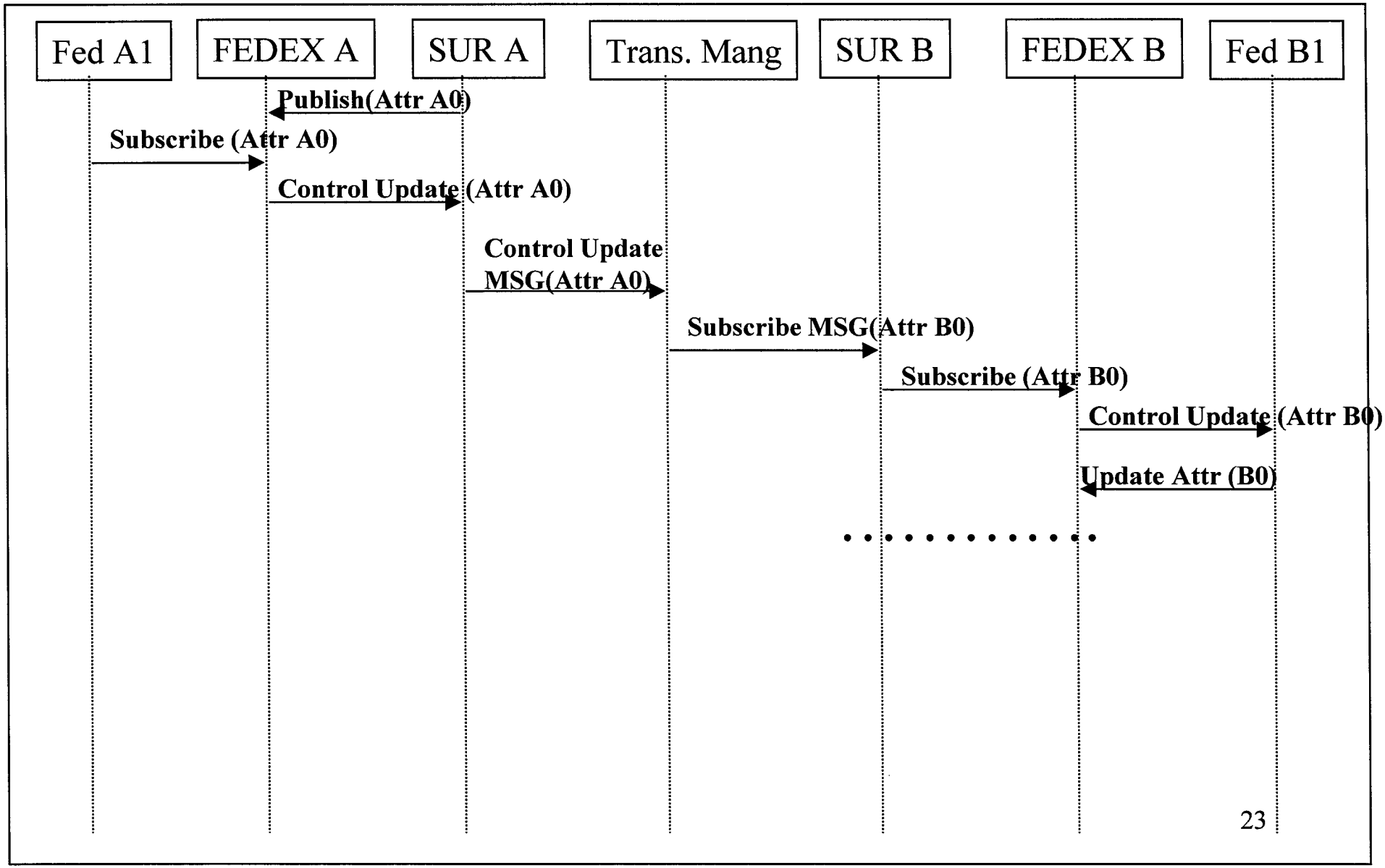
Save


Fed A1
FEDEX A
SUR A
Trans. Mang
SUR B
FEDEX B
Fed B1
Request Save
Initiate Save
Initiate Save
Initiate Save MSG
Request Save MSG
Request Save
Initiate Save
Initiate Save
Save Begun
Save Begun
Save Begun
Save Begun
Save Achieved
Save Achieved MSG
Save Achieved
Save Achieved MSG
Save Achieved

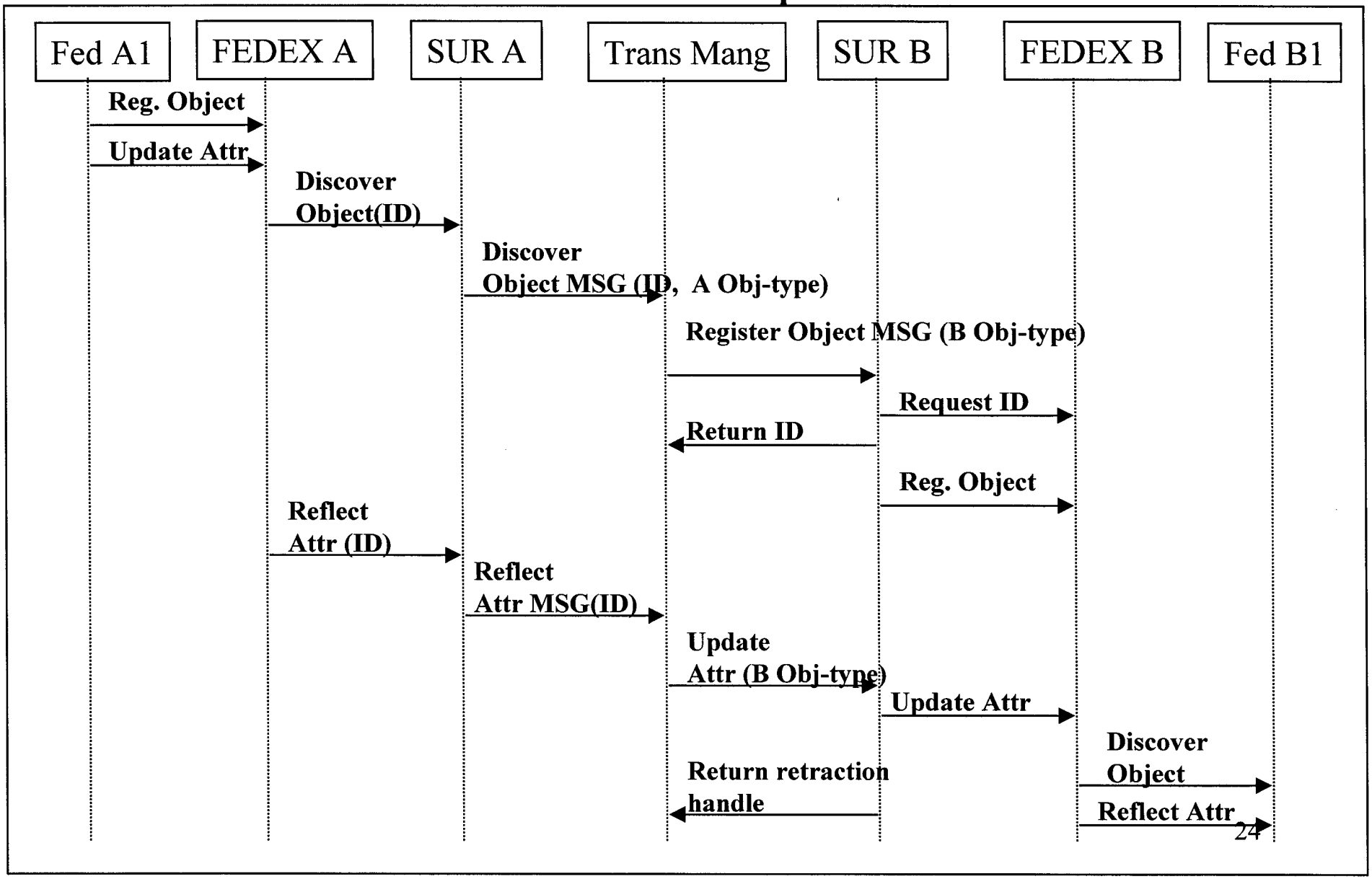
Restore


Fed A1
FEDEX A
SUR A
Trans. Mang
SUR B
FEDEX B
Fed B1
Request Restore
Initiate RestoreInitiate Restore
Initiate Restore MSG
Request Restore MSG
Request Restore
Initiate RestoreInitiate Restore
Restore Achieved
Restore Achieved MSG
Restore Achieved
Restore Achieved MSG
Restore Achieved

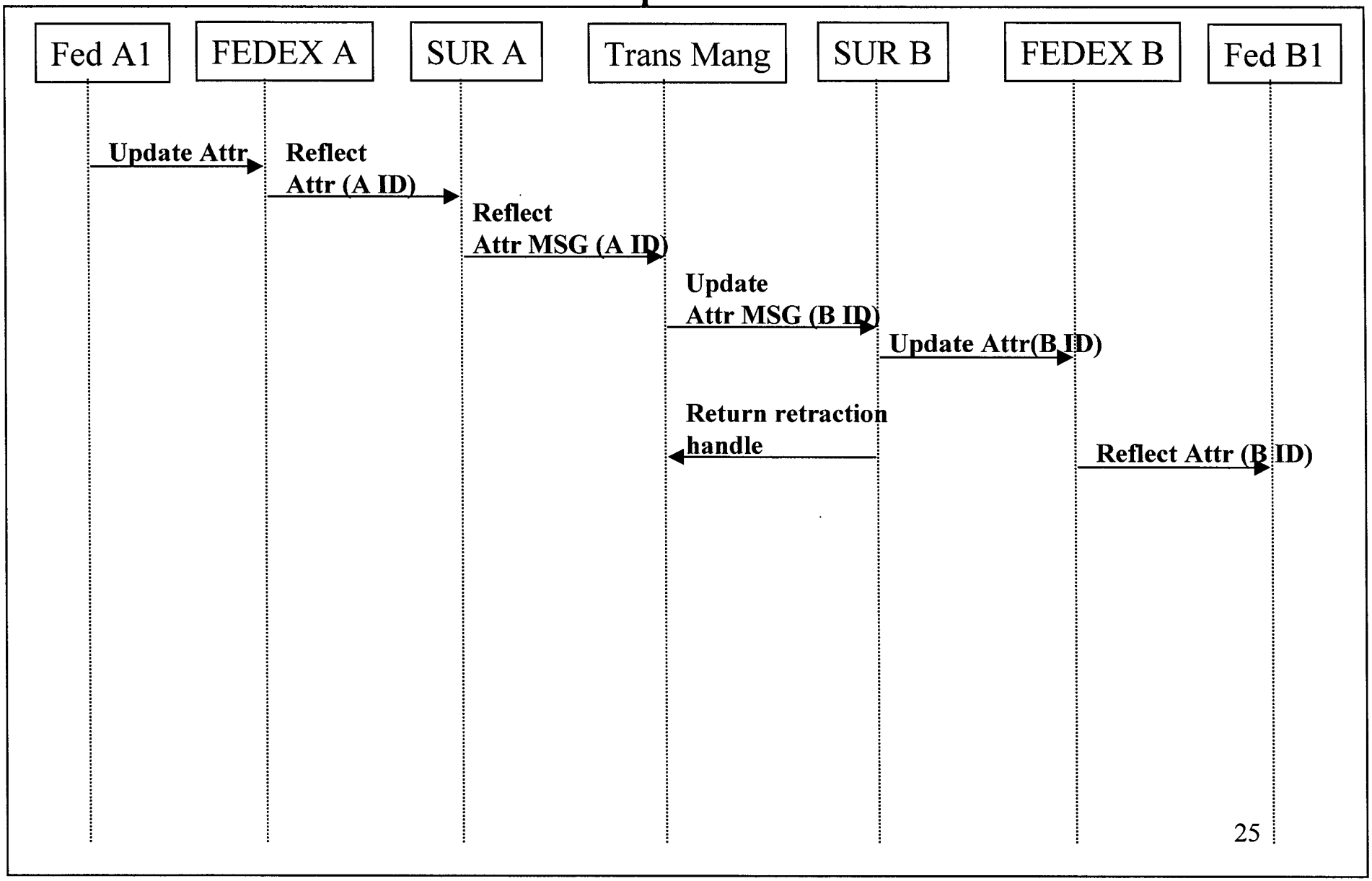
Subscribe


Fed A1
FEDEX A
SUR A
Trans. Mang
SUR B
FEDEX B
Fed B1
Publish(Attr A0)
Subscribe (Attr A0)
Control Update (Attr A0)
Control Update MSG(Attr A0)
Subscribe MSG(Attr B0)
Subscribe (Attr B0)
Control Update (Attr B0)
Update Attr (B0)

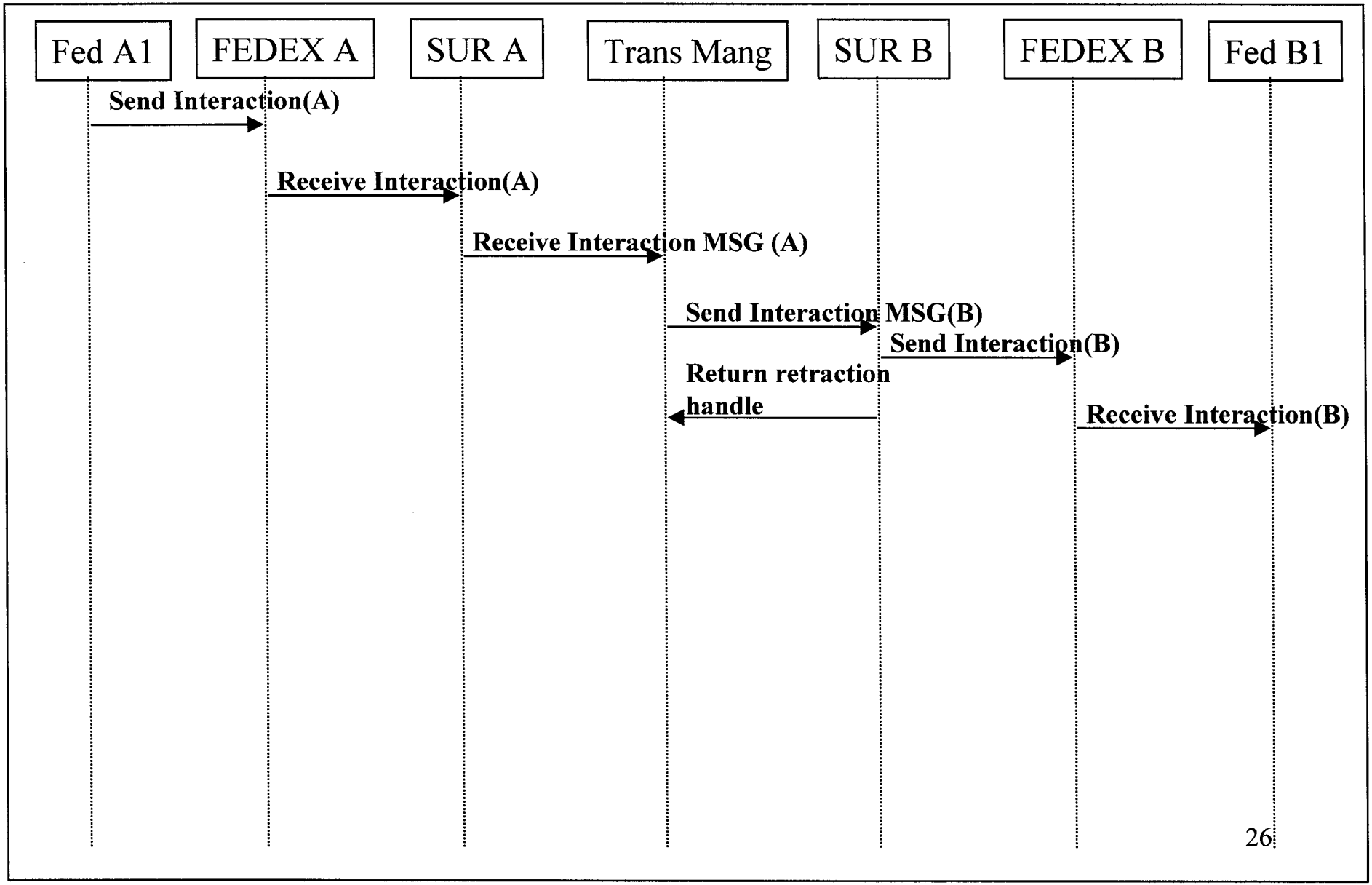
Create and Update



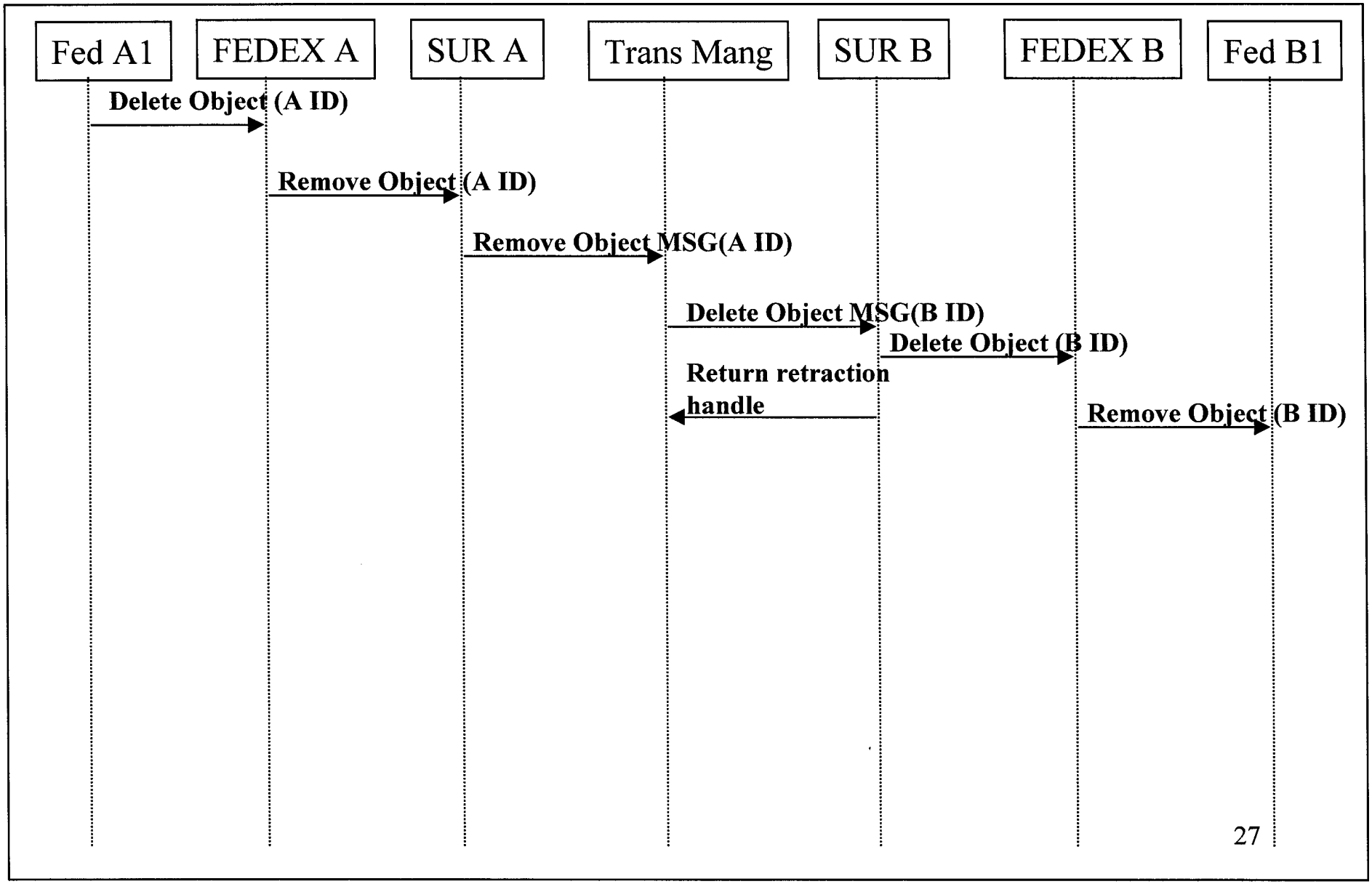
Update


Fed A1
FEDEX A
SUR A
Trans Mang
SUR B
FEDEX B
Fed B1
Update Attr
Reflect Attr (A ID)
Reflect Attr MSG (A ID)
Update Attr MSG (B ID)
Update Attr(B ID)
Return retraction handle
Reflect Attr (B ID)

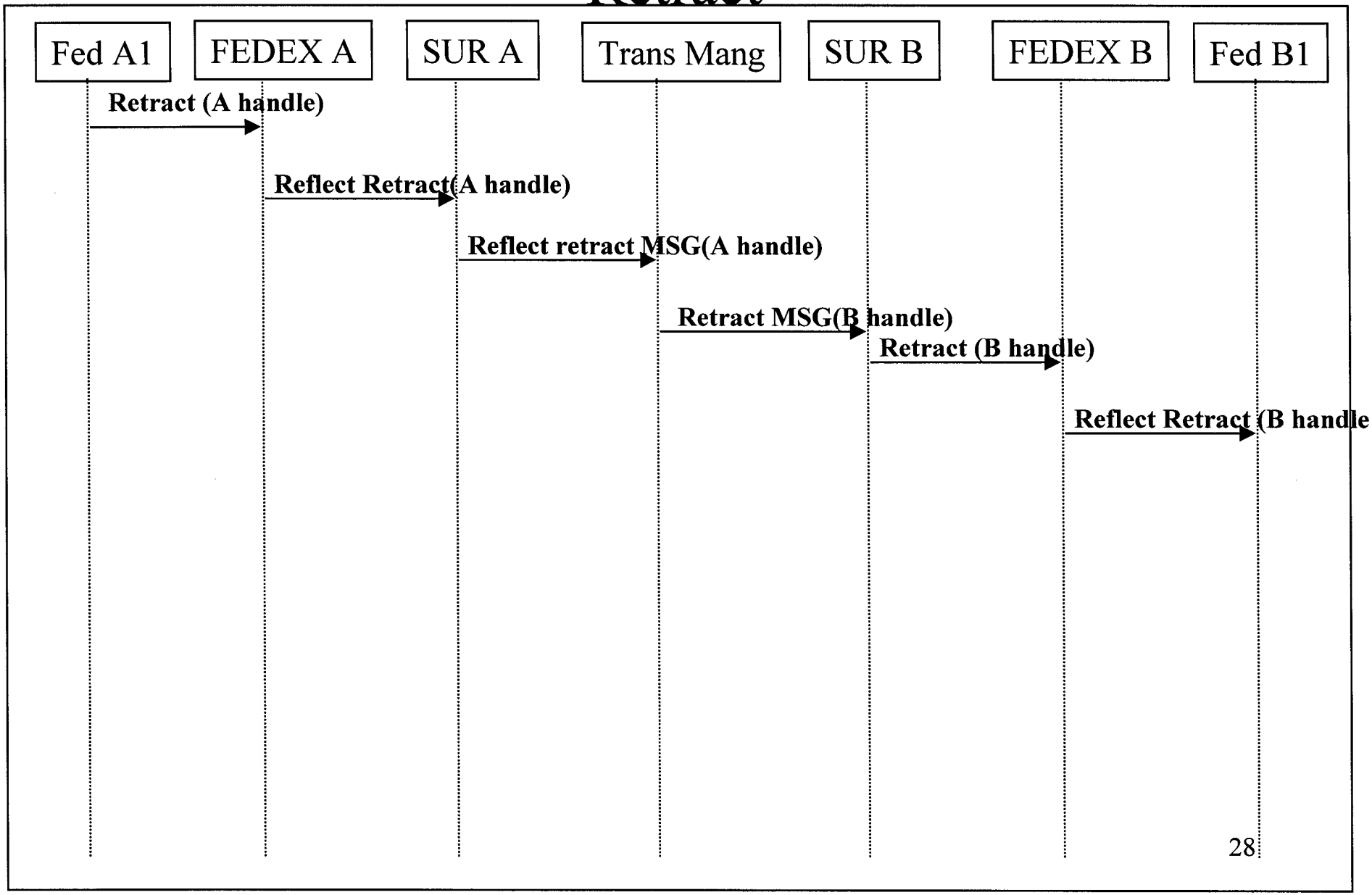
Send Interaction



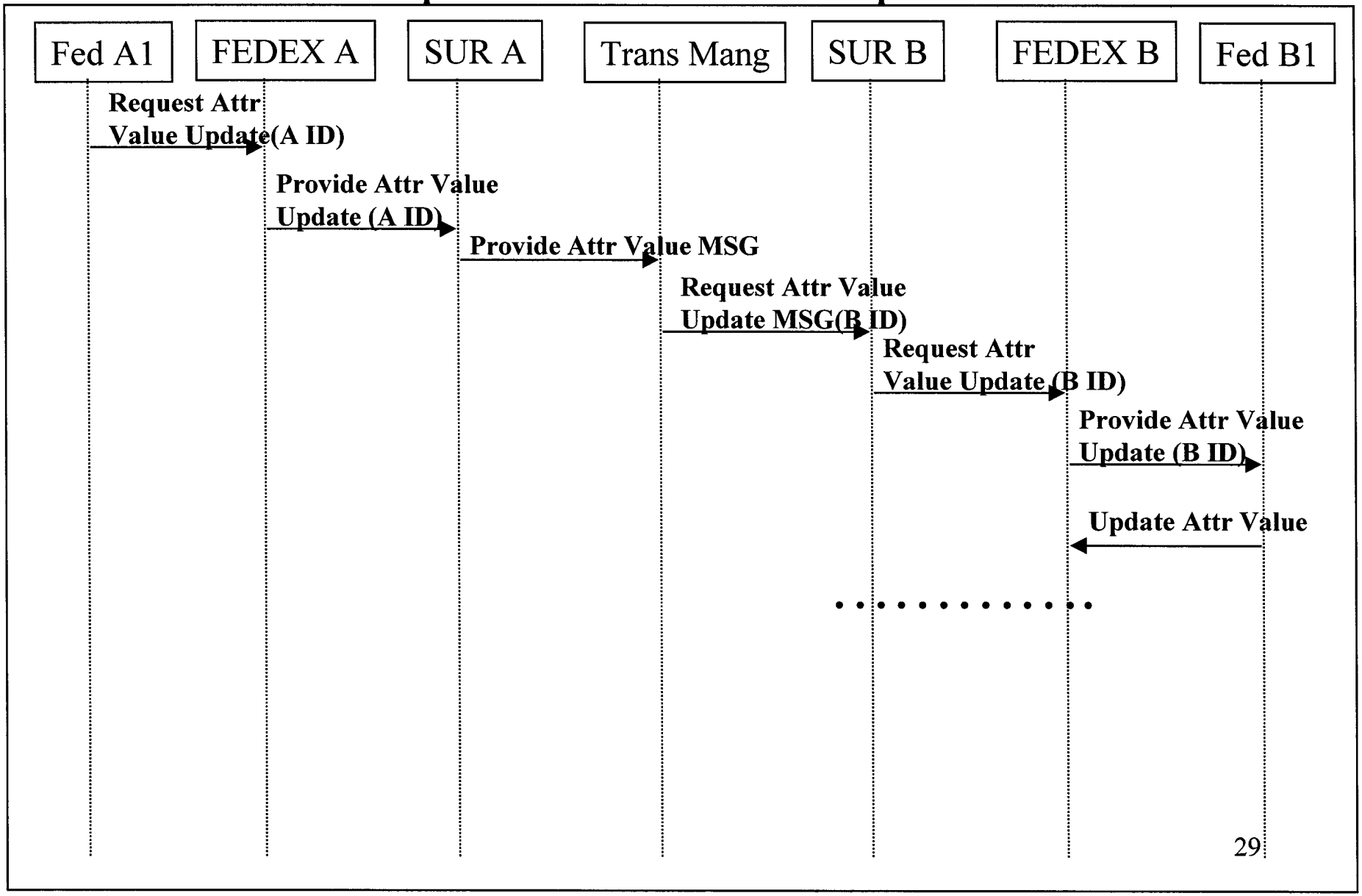
Delete


Fed A1
FEDEX A
SUR A
Trans Mang
SUR B
FEDEX B
Fed B1
Delete Object (A ID)
Remove Object (A ID)
Remove Object MSG(A ID)
Delete Object MSG(B ID)
Delete Object (B ID)
Return retraction handle
Remove Object (B ID)

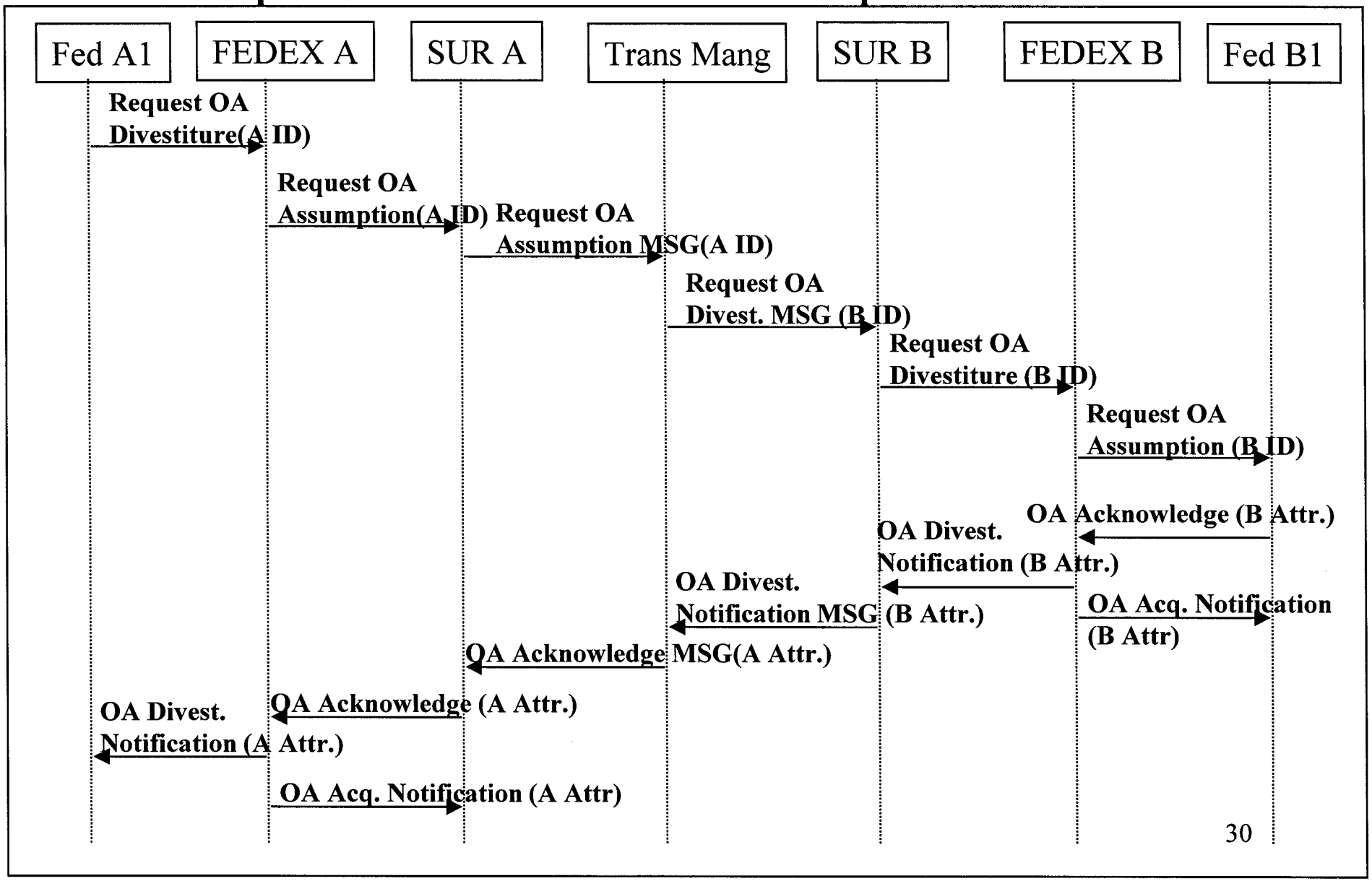
Retract


Fed A1
FEDEX A
SUR A
Trans Mang
SUR B
FEDEX B
Fed B1
Retract (A handle)
Reflect Retract(A handle)
Reflect retract MSG(A handle)
Retract MSG(B handle)
Retract (B handle)
Reflect Retract (B handle

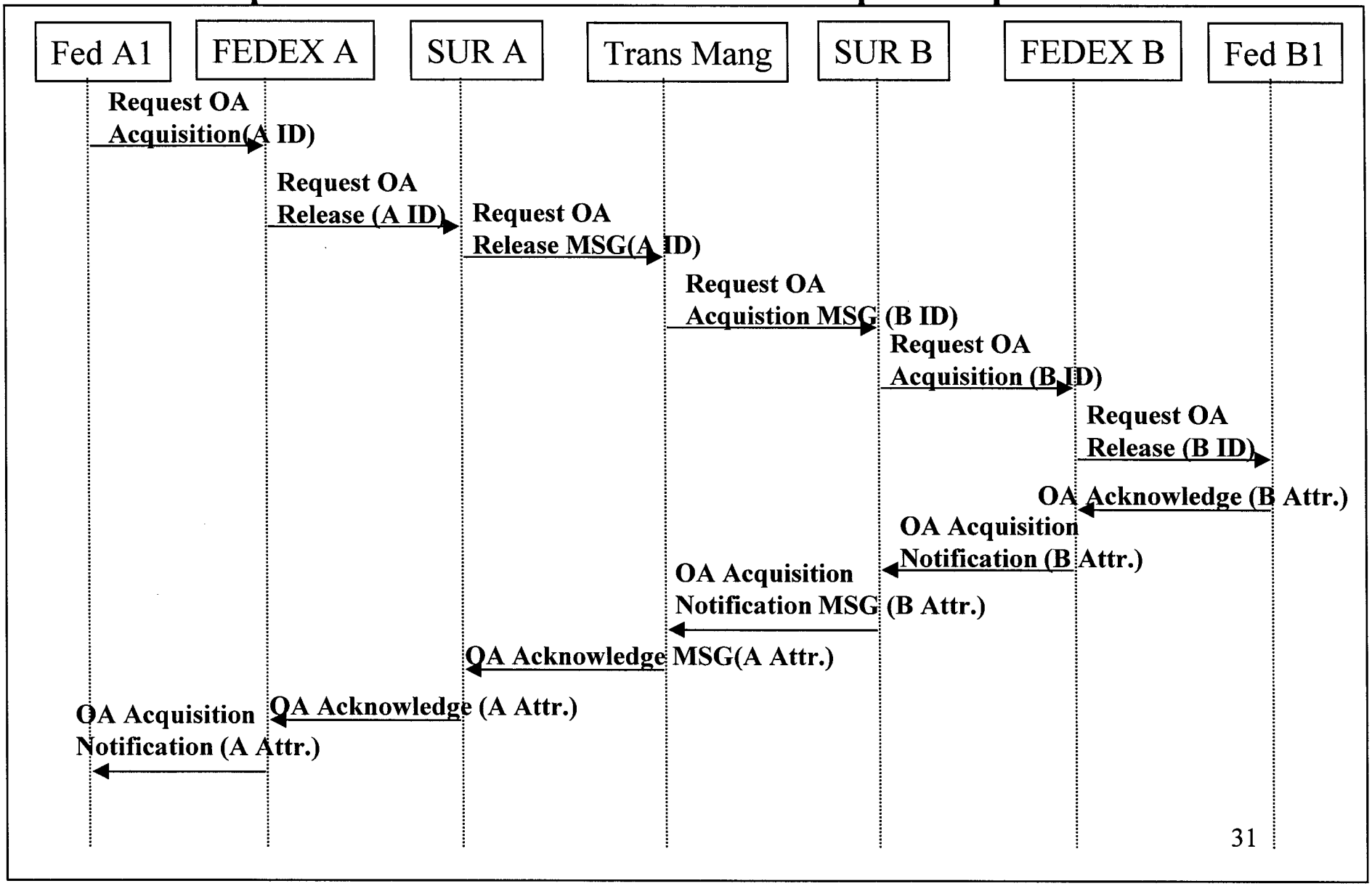
Request Attr Value Update



Request Attribute Ownership Divestiture

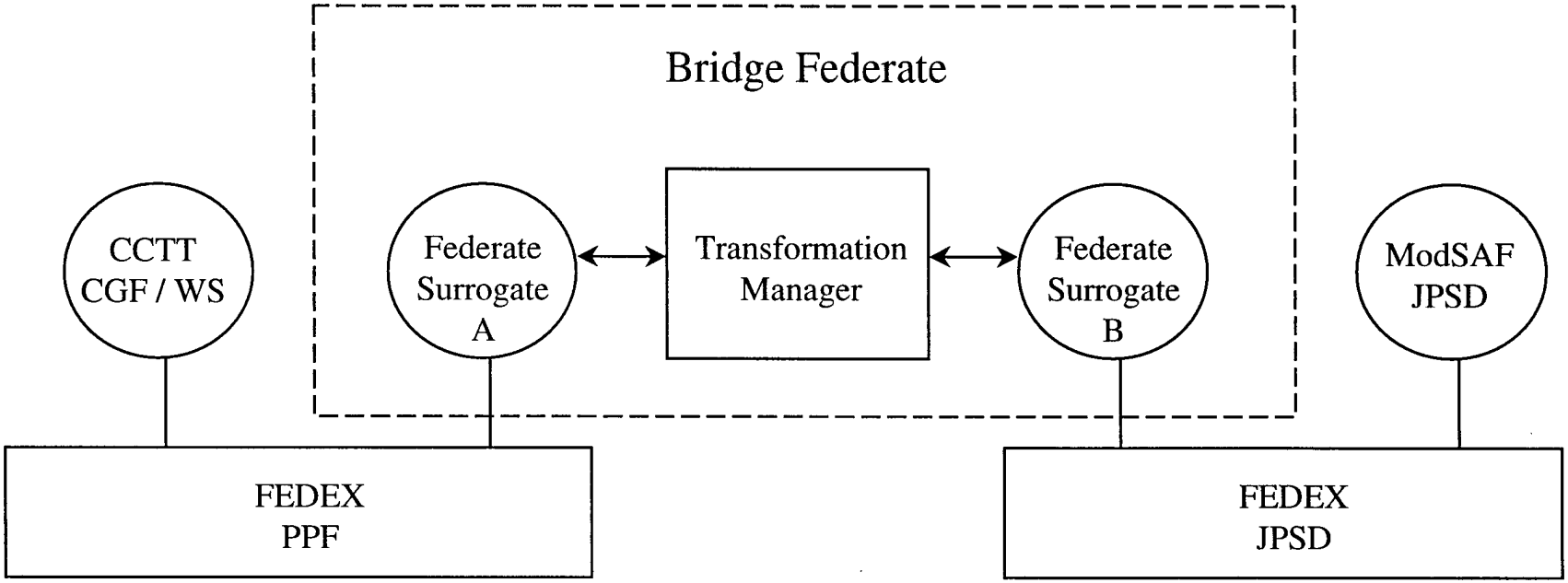


Request Attribute Ownership Acquisition

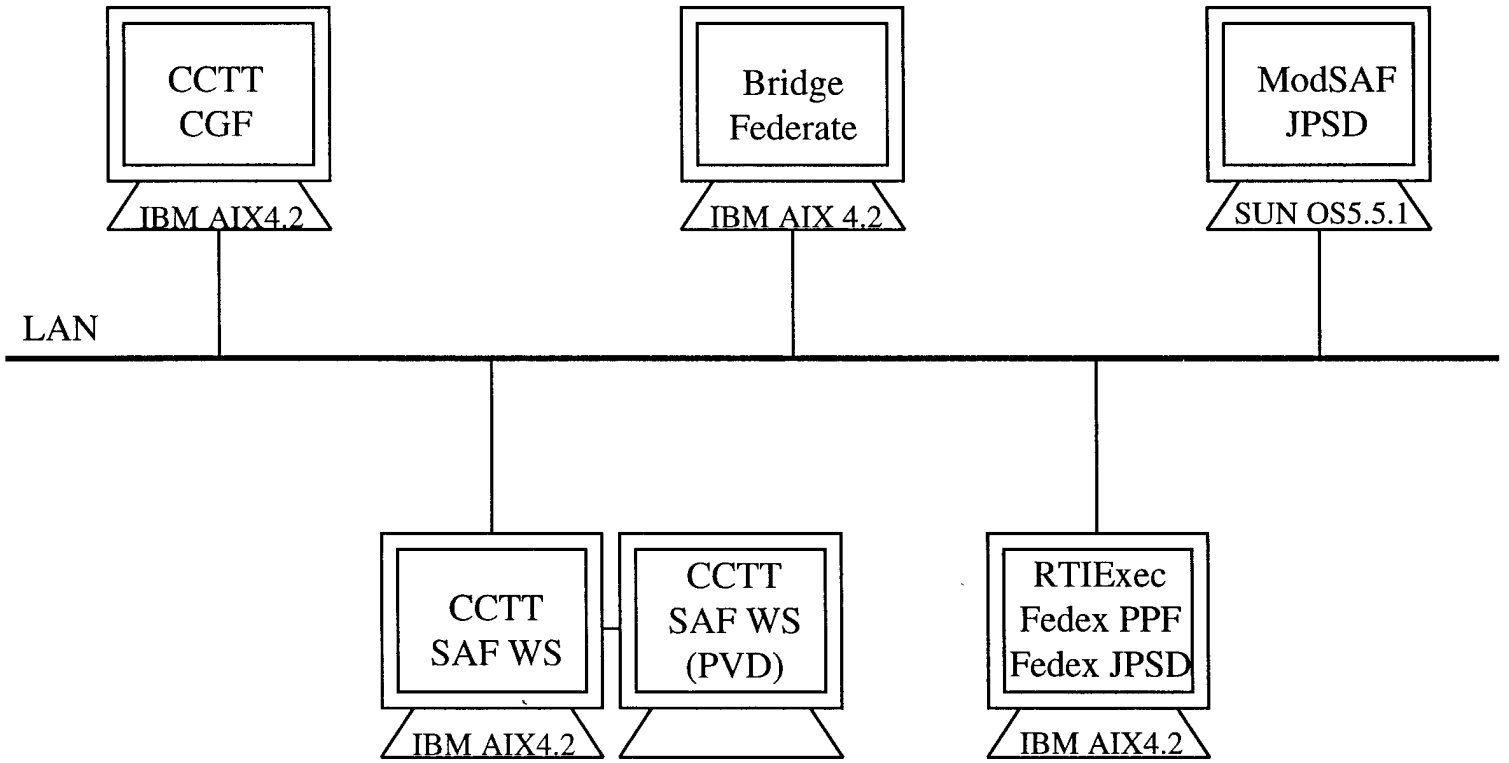


Appendix D – PHASE I FEDERATION WORKBOOK

Bridge Federate Application Demo
Functional Diagram



BRIDGE FEDERATE APPLICATION DEMO
HARDWARE CONFIGURATION



BRIDGE FEDERATE APPLICATION DEMO SCENARIO

- Consist of 1 M1 platoon and 1 T72 platoon.
- The M1 platoon executes a road march order on its designated route.
- The T72 Platoon executes an assembly order.
- The M1 platoon approaches the T72 platoon and the two platoons engage each other.

BRIDGE FEDERATE APPLICATION DEMO
INITIALIZATION AND RUN PROCEDURE

Initialization and Run Procedure		Expected Result
1. Start RTIexec	a. On machine "bullwinkle", type "cd /devel/rti" b. type "rtiexec 12345"	RTIexec is running. A log message "Acceptor open on #" is displayed.
2. Create Federation Executions	a. Open another window, make sure RTI environment variables are set correctly. b. type "cd /usr/devel/hla/ppf/driver/driver2" c. type "driver"	PPF Federation execution and JPSD federation execution JPSD are created in their respective windows
3. Start the Bridge Federate	a. On machine "natasha", type "cd /usr/devel/hla/ppf/driver/bridge1.0/driver" b. type "bridge_federate"	
4. Start CCTT CGF	a. On machine "boris", make sure the RTI envrionment varible is set correctly. b. type "cd /usr/devel/cctt/saf/build7/saf/environment" c. type ". menu_F0" d. type "1" to choose "Existing=:0.0" display. e. type "1" to choose "Net_And_EDB" to start the network process and entity database. f. type "2" to choose "Cgf_Simulator" g. Answer "n" to not turn on damage	

BRIDGE FEDERATE APPLICATION DEMO
INITIALIZATION AND RUN PROCEDURE

Initialization and Run Procedure		Expected Result
	assessment. h. type "2" to choose cgf_HLArev as the cgf executable.	When cgf finish starting up, a message "Ready for Execution" is shown on the window.
5. Start CCTT WS and DVI	a. On machine "rocky", make sure the RTI environment varible is set correctly. b. type "cd /usr/devel/cctt/saf/build7/saf/environment" c. type ". menu_F0" d. type "1" to choose "Existing=:0.0" display. e. type "1" to chosse "Net_And_EDB" to start the network process and entity database. f. type "3" to choose "DIS_Visual_Interface" g. type "4" to choose "SAF_WS_BLUFOR"	Network process and Entity Database windows are displayed. DVI window is displayed. The UCI and PVD shows up on the displays.
6. Start the FCS gateway and the JPSD ModSAF	a. On machine "centauri", type "cd /devel/rti/gateway". b. type " . setup" c. type "rungate". d. On machine "lumati", type "cd/devel/rti/gateway". e. type "runsaf".	The fcs gateway started. ModSAF is loaded.

BRIDGE FEDERATE APPLICATION DEMO
INITIALIZATION AND RUN PROCEDURE (Continued)

Initialization and Run Procedure		Expected Result
7. Initialize CCTT and load exercise file	a. From the DVI's "Send PDU pulldown menu, select "Create PDU". b. Change the Receiver Host to "broadcast" and click on "send" button. c. Select "Environment PDU" from the same pulldown menu. d. Change the Receiver Host to "broadcast" and click on "send" button. e. Select "CGF Specific PDU" from the menu. f. Change the Receiver Host to "boris" and click on the "send" button. g. Select "SAF Specific PDU" from the menu. Click on "SAF" selector, type in "b/bridge_ demo_blufor" as the file name, and click on the "send" button.	The M1 platoon and a route appear on the SAF WS PVD.
8. Load scenario for JPSD ModSAF	a. From the ModSAF's "File" pulldown menu, select "Load Scenario". b. Select the file "bridge_demo_opfor" and click "O.K."	The T72 platoon appears on the ModSAF PVD.

BRIDGE FEDERATE APPLICATION DEMO
INITIALIZATION AND RUN PROCEDURE (Continued)

Initialization and Run Procedure		Expected Result
9. Start CCTT exercise	a. From the DVI "Send PDU" pulldown menu select "Start PDU". b. Change the Receiver Host to "broadcast" and click on "send"	The M1 platoon starts executing the road march.
10. Start JPSD ModSAF exercise	a. Click on Resume Exercise button. b. Select the order from the On Order pulldown menu.	The T72 platoon stars executing the road march.

```
;; CCTTSAF FED
(fed
 (objects
 (class Entity
    (attribute FORCE_ID FED_RELIABLE FED_RECEIVE)
    (attribute ENTITYID_SITE FED_RELIABLE FED_RECEIVE)
    (attribute ENTITYID_APPLICATION FED_RELIABLE FED_RECEIVE)
    (attribute ENTITYID_ENTITY FED_RELIABLE FED_RECEIVE)
    (attribute LOCATION_X FED_RELIABLE FED_RECEIVE)
    (attribute LOCATION_Y FED_RELIABLE FED_RECEIVE)
    (attribute LOCATION_Z FED_RELIABLE FED_RECEIVE)
    (attribute VELOCITY_X FED_RELIABLE FED_RECEIVE)
    (attribute VELOCITY_Y FED_RELIABLE FED_RECEIVE)
    (attribute VELOCITY_Z FED_RELIABLE FED_RECEIVE)
    (attribute ACCELERATION_X FED_RELIABLE FED_RECEIVE)
    (attribute ACCELERATION_Y FED_RELIABLE FED_RECEIVE)
    (attribute ACCELERATION_Z FED_RELIABLE FED_RECEIVE)
    (attribute ORIENTATION_PSI FED_RELIABLE FED_RECEIVE)
    (attribute ORIENTATION_THETA FED_RELIABLE FED_RECEIVE)
    (attribute ORIENTATION_PHI FED_RELIABLE FED_RECEIVE)
    (attribute APPEARANCE_PAINT_SCHEME FED_RELIABLE FED_RECEIVE)
 (class Platform
    (attribute DAMAGE_STATE_APPEARANCE FED_RELIABLE FED_RECEIVE)
 (class Ground_Vehicle
    (attribute APPEARANCE_SMOKE FED_RELIABLE FED_RECEIVE)
    (attribute APPEARANCE_TRAILING FED_RELIABLE FED_RECEIVE)
    (attribute APPEARANCE_HATCH FED_RELIABLE FED_RECEIVE)
    (attribute APPEARANCE_LIGHTS FED_RELIABLE FED_RECEIVE)
    (attribute APPEARANCE_FLAMING FED_RELIABLE FED_RECEIVE)
    (attribute DAMAGE_STATE_MOBILITY FED_RELIABLE FED_RECEIVE)
    (attribute DAMAGE_STATE_FIRE_POWER FED_RELIABLE FED_RECEIVE)
 (class Tracked
    (attribute TURRET_HEADING FED_RELIABLE FED_RECEIVE)
    (attribute MAIN_GUN_PITCH FED_RELIABLE FED_RECEIVE)
    (attribute COMMANDERS_GUN_PITCH FED_RELIABLE FED_RECEIVE)
    (attribute MACHINE_GUN_PITCH FED_RELIABLE FED_RECEIVE)
 (class Tank
 (class M1
 )
 (class T72
 )
 )
 (class Armored_Fighting_Vehicle
    (attribute APPEARANCE_LAUNCHER FED_RELIABLE FED_RECEIVE)
 (class M2
 )
 (class BMP
 )
 )
 )
 )
 (class Air_Vehicle
    (attribute ANGULAR_VELOCITY_PSI FED_RELIABLE FED_RECEIVE)
    (attribute ANGULAR_VELOCITY_THETA FED_RELIABLE FED_RECEIVE)
    (attribute ANGULAR_VELOCITY_PHI FED_RELIABLE FED_RECEIVE)
```

```
(class Attack
(class FA18
)
)
)
(class Human
(class INFANTRY
)
)
)
(class Munition
   (attribute ANGULAR_VELOCITY_PSI FED_RELIABLE FED_RECEIVE)
   (attribute ANGULAR_VELOCITY_THETA FED_RELIABLE FED_RECEIVE)
   (attribute ANGULAR_VELOCITY_PHI FED_RELIABLE FED_RECEIVE)
   (attribute FIRING_ID FED_RELIABLE FED_RECEIVE)
(class Guided
(class Anti_Ground
(class TOW
)
(class AT5
)
(class LGB
)
)
(class Anti_Air
(class SA16
)
)
)
)
)
 (class Manager
      (class Federate
           (attribute FederateHost FED_RELIABLE FED_RECEIVE)
           (attribute FederateHandle FED_RELIABLE FED_RECEIVE)
           (attribute FederateState FED_RELIABLE FED_RECEIVE)
           (attribute FederateName FED_RELIABLE FED_RECEIVE)
           (attribute RTIversion FED_RELIABLE FED_RECEIVE)
           (attribute TimeManagerState FED_RELIABLE FED_RECEIVE)
           (attribute FederateLookahead FED_RELIABLE FED_RECEIVE)
           (attribute FederateTime FED_RELIABLE FED_RECEIVE)
           (attribute TimeConstrained FED_RELIABLE FED_RECEIVE)
           (attribute TimeRegulating FED_RELIABLE FED_RECEIVE)
           (attribute FIFOlength FED_RELIABLE FED_RECEIVE)
           (attribute TSOlength FED_RELIABLE FED_RECEIVE)
           (attribute DequeueFIFOasync FED_RELIABLE FED_RECEIVE)
           (attribute TotalObjectCount FED_RELIABLE FED_RECEIVE)
           (attribute HoldingTokensObjectCount FED_RELIABLE FED_RECEIVE)
           (attribute DeletedObjectCount FED_RELIABLE FED_RECEIVE)
           (attribute NumAttributes FED_RELIABLE FED_RECEIVE)
           (attribute NumParameters FED_RELIABLE FED_RECEIVE)
           )
      (class Federation
           (attribute FederationName FED_RELIABLE FED_RECEIVE)
           (attribute FederationState FED_RELIABLE FED_RECEIVE)
```

```
            (attribute FederatesInFederation FED_RELIABLE FED_RECEIVE)
            (attribute SaveIsScheduled FED_RELIABLE FED_RECEIVE)
            (attribute ScheduledSaveTime FED_RELIABLE FED_RECEIVE)
            (attribute RTIversion FED_RELIABLE FED_RECEIVE)
            )
        )
)
(interactions
 (class DETONATION FED_RELIABLE FED_RECEIVE
   (parameter Munition_ID)
   (parameter Target_ID)
   (parameter Firing_ID)
   (parameter Location_X)
   (parameter Location_Y)
   (parameter Location_Z)
   (parameter Relative_Location_X)
   (parameter Relative_Location_Y)
   (parameter Relative_Location_Z)
   (parameter Velocity_X)
   (parameter Velocity_Y)
   (parameter Velocity_Z)
   (parameter Burst_Descriptor_Warhead)
   (parameter Burst_Descriptor_Fuze)
   (parameter Burst_Descriptor_Detonation_Result)
   (parameter Quantity)
   (parameter Rate)
 )

 (class WEAPON_FIRE FED_RELIABLE FED_RECEIVE
   (parameter Firing_ID)
   (parameter Target_ID)
   (parameter Location_X)
   (parameter Location_Y)
   (parameter Location_Z)
   (parameter Velocity_X)
   (parameter Velocity_Y)
   (parameter Velocity_Z)
   (parameter Munition_Type)
   (parameter Quantity)
   (parameter Rate)
 )

 (class WEAPON_LAUNCH FED_RELIABLE FED_RECEIVE
   (parameter Launch_Platform_ID)
   (parameter Munitiion_ID)
   (parameter Target_ID)
   (parameter Location_X)
   (parameter Location_Y)
   (parameter Location_Z)
   (parameter Velocity_X)
   (parameter Velocity_Y)
   (parameter Velocity_Z)
 )

 (class COLLISION FED_RELIABLE FED_RECEIVE
```

```
  (parameter Issuing_ID)
  (parameter Colliding_ID)
  (parameter Mass)
  (parameter Relative_Location_X)
  (parameter Relative_Location_Y)
  (parameter Relative_Location_Z)
  (parameter Velocity_X)
  (parameter Velocity_Y)
  (parameter Velocity_Z)
)

(class AIR_VEHICLE_LAUNCH FED_RELIABLE FED_RECEIVE
  (parameter Launch_Platform_ID)
  (parameter Air_Vehicle_ID)
  (parameter Location_X)
  (parameter Location_Y)
  (parameter Location_Z)
  (parameter Velocity_X)
  (parameter Velocity_Y)
  (parameter Velocity_Z)
)

(class Manager FED_RELIABLE FED_RECEIVE
    (class Federate FED_RELIABLE FED_RECEIVE
        (parameter FromFederate)
        (class Alert FED_RELIABLE FED_RECEIVE
            (parameter AlertSeverity)
            (parameter AlertText)
            (parameter AlertID)
            )
        (class ServiceLog FED_RELIABLE FED_RECEIVE
            (parameter ServiceName)
            (parameter ServiceInitiator)
            (class ServiceLogArguments FED_RELIABLE FED_RECEIVE
                (parameter Handle1)
                (parameter Handle2)
                (parameter HandleSet)
                (parameter ObjectIDorCount)
                (parameter TagOrLabelOrName)
                (parameter Time)
                (parameter Enumeration)
                (parameter Boolean)
                )
            )
        (class ObjectInformation FED_RELIABLE FED_RECEIVE
            (parameter ObjectID)
            (parameter LockedAttributes)
            (parameter RegisteredClass)
            (parameter RepresentedClass)
            )
        (class PublishingClass FED_RELIABLE FED_RECEIVE
            (parameter ObjectClass)
            ;; format = ClassHandle:attrHandle,attrHandle,...,attrHandle'
            (parameter InteractionClass)
            ;; format = ClassHandle
```

```
        )
    (class SubscribingClass FED_RELIABLE FED_RECEIVE
        (parameter ObjectClass)
        ;; format = ClassHandle:attrHandle,attrHandle,...,attrHandle
        (parameter InteractionClass)
        ;; format = ClassHandle
        )
    (class Action FED_RELIABLE FED_RECEIVE
        (parameter ToFederate)
        (class RequestPublicationTree FED_RELIABLE FED_RECEIVE
            )
        (class RequestSubscriptionTree FED_RELIABLE FED_RECEIVE
            )
        (class SetTiming FED_RELIABLE FED_RECEIVE
            (parameter FedReportPeriod)
            (parameter TimeReportPeriod)
            (parameter ObjectReportPeriod)
            )
        (class RequestObjectInformation FED_RELIABLE FED_RECEIVE
            (parameter ObjectID)
            )
        (class ModifyAttributeState FED_RELIABLE FED_RECEIVE
            (parameter ObjectID)
            (parameter AttributeID)
            (parameter TokenState)
            )
        (class RemoteServiceInvocation FED_RELIABLE FED_RECEIVE
            (class DoResignFederationExecution FED_RELIABLE FED_RECEIVE
                (parameter ResignAction)
                )
            (class DoDeleteObject FED_RELIABLE FED_RECEIVE
                (parameter ObjectID)
                (parameter Time)
                (parameter Tag)
                )
            (class DoSetLookahead FED_RELIABLE FED_RECEIVE
                (parameter Lookahead)
                )
            (class DoSetTimeConstrained FED_RELIABLE FED_RECEIVE
                (parameter State)
                )
            (class DoTurnRegulationOn FED_RELIABLE FED_RECEIVE
                )
            (class DoTurnRegulationOff FED_RELIABLE FED_RECEIVE
                )
            )
        (class Control FED_RELIABLE FED_RECEIVE
            (parameter SetServiceLogging)
            (parameter SetLogFile)
            (parameter DeleteObject)
            (parameter DequeueFIFO)
            )
        )
    )
)
```

```
;; MODSAF FED
(fed
 ;; objects
 (objects
   (class Entity
        (attribute Entity_ID_site FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_ID_application FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_ID_entity FED_BEST_EFFORT FED_RECEIVE )
        (attribute Force_ID FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Type_Kind FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Type_Domain FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Type_Country FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Type_Category FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Type_Subcategory FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Type_Specific FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Type_Extra FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Guise_Kind FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Guise_Domain FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Guise_Country FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Guise_Category FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Guise_Scategory FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Guise_Specific FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Guise_Extra FED_BEST_EFFORT FED_RECEIVE )
        (attribute Orientation_Psi FED_BEST_EFFORT FED_RECEIVE )
        (attribute Orientation_Theta FED_BEST_EFFORT FED_RECEIVE )
        (attribute Orientation_Phi FED_BEST_EFFORT FED_RECEIVE )
        (attribute Location_X FED_BEST_EFFORT FED_RECEIVE )
        (attribute Location_Y FED_BEST_EFFORT FED_RECEIVE )
        (attribute Location_Z FED_BEST_EFFORT FED_RECEIVE )
        (attribute Velocity_X FED_BEST_EFFORT FED_RECEIVE )
        (attribute Velocity_Y FED_BEST_EFFORT FED_RECEIVE )
        (attribute Velocity_Z FED_BEST_EFFORT FED_RECEIVE )
        (attribute marking_charset FED_BEST_EFFORT FED_RECEIVE )
        (attribute marking_text FED_BEST_EFFORT FED_RECEIVE )
        (attribute Appearance FED_BEST_EFFORT FED_RECEIVE )
        (attribute Dead_Reckoning_Algorithm FED_BEST_EFFORT FED_RECEIVE )
        (attribute Acceleration_X FED_BEST_EFFORT FED_RECEIVE )
        (attribute Acceleration_Y FED_BEST_EFFORT FED_RECEIVE )
        (attribute Acceleration_Z FED_BEST_EFFORT FED_RECEIVE )
        (attribute Angular_Velocity_Psi FED_BEST_EFFORT FED_RECEIVE )
        (attribute Angular_Velocity_Theta FED_BEST_EFFORT FED_RECEIVE )
        (attribute Angular_Velocity_Phi FED_BEST_EFFORT FED_RECEIVE )
        (attribute TimeStamp FED_BEST_EFFORT FED_RECEIVE )
        (attribute MarkingCharSet FED_BEST_EFFORT FED_RECEIVE )
        (attribute Capabilities FED_BEST_EFFORT FED_RECEIVE )
        (attribute ArticulatedStruct FED_BEST_EFFORT FED_RECEIVE )
  )
  (class Manager

   (class Federate
    (attribute FederateHost FED_RELIABLE FED_RECEIVE)
    (attribute FederateHandle FED_RELIABLE FED_RECEIVE)
    (attribute FederateState FED_RELIABLE FED_RECEIVE)
    (attribute FederateName FED_RELIABLE FED_RECEIVE)
    (attribute RTIversion FED_RELIABLE FED_RECEIVE)
```

```
    (attribute TimeManagerState FED_RELIABLE FED_RECEIVE)
    (attribute FederateLookahead FED_RELIABLE FED_RECEIVE)
    (attribute FederateTime FED_RELIABLE FED_RECEIVE)
    (attribute TimeConstrained FED_RELIABLE FED_RECEIVE)
    (attribute TimeRegulating FED_RELIABLE FED_RECEIVE)
    (attribute FIFOlength FED_RELIABLE FED_RECEIVE)
    (attribute TSOlength FED_RELIABLE FED_RECEIVE)
    (attribute DequeueFIFOasync FED_RELIABLE FED_RECEIVE)
    (attribute TotalObjectCount FED_RELIABLE FED_RECEIVE)
    (attribute HoldingTokensObjectCount FED_RELIABLE FED_RECEIVE)
    (attribute DeletedObjectCount FED_RELIABLE FED_RECEIVE)
    (attribute NumAttributes FED_RELIABLE FED_RECEIVE)
    (attribute NumParameters FED_RELIABLE FED_RECEIVE)
   )
   (class Federation
    (attribute FederationName FED_RELIABLE FED_RECEIVE)
    (attribute FederationState FED_RELIABLE FED_RECEIVE)
    (attribute FederatesInFederation FED_RELIABLE FED_RECEIVE)
    (attribute SaveIsScheduled FED_RELIABLE FED_RECEIVE)
    (attribute ScheduledSaveTime FED_RELIABLE FED_RECEIVE)
    (attribute RTIversion FED_RELIABLE FED_RECEIVE)
   )


  )
)
;; interactions
(interactions
 (class Detonation FED_BEST_EFFORT FED_RECEIVE
 ;;(class Detonation FED_BEST_EFFORT FED_TIMESTAMP
  (parameter Munition_ID_site)
  (parameter Munition_ID_application)
  (parameter Munition_ID_entity)
  (parameter Attacker_ID_site)
  (parameter Attacker_ID_application)
  (parameter Attacker_ID_entity)
  (parameter Target_ID_site)
  (parameter Target_ID_application)
  (parameter Target_ID_entity)
  (parameter Event_ID_site)
  (parameter Event_ID_application)
  (parameter Event_ID_entity)
  (parameter WorldLocationX)
  (parameter WorldLocationY)
  (parameter WorldLocationZ)
  (parameter EntityLocationX)
  (parameter EntityLocationY)
  (parameter EntityLocationZ)
  (parameter VelocityX)
  (parameter VelocityY)
  (parameter VelocityZ)
  (parameter BurstKind)
  (parameter BurstDomain)
  (parameter BurstCountry)
  (parameter BurstCategory)
```

```
  (parameter BurstSubcategory)
  (parameter BurstSpecific)
  (parameter BurstExtra)
  (parameter BurstWarhead)
  (parameter BurstFuze)
  (parameter BurstQuantity)
  (parameter BurstRate)
  (parameter DetonationResult)
  (parameter TimeStamp)
  (parameter ArticulatedStruct)
 )
 ;;(class Fire FED_BEST_EFFORT FED_TIMESTAMP
 (class Fire FED_BEST_EFFORT FED_RECEIVE
  (parameter Launch_Platform_ID_site )
  (parameter Launch_Platform_ID_application )
  (parameter Launch_Platform_ID_entity )
  (parameter Target_ID_site )
  (parameter Target_ID_application )
  (parameter Target_ID_entity )
  (parameter Weapon_ID_site )
  (parameter Weapon_ID_application )
  (parameter Weapon_ID_entity )
  (parameter Event_ID_site )
  (parameter Event_ID_application )
  (parameter Event_ID_entity )
  (parameter LocationX )
  (parameter LocationY )
  (parameter LocationZ )
  (parameter VelocityX )
  (parameter VelocityY )
  (parameter VelocityZ )
  (parameter Range)
  (parameter BurstKind)
  (parameter BurstDomain)
  (parameter BurstCountry)
  (parameter BurstCategory)
  (parameter BurstSubcategory)
  (parameter BurstSpecific)
  (parameter BurstExtra)
  (parameter BurstWarhead)
  (parameter BurstFuze)
  (parameter BurstQuantity)
  (parameter BurstRate)
  (parameter TimeStamp)
 )
 (class Signal FED_BEST_EFFORT FED_RECEIVE
  (parameter TimeStamp)
  (parameter EntitySite)
  (parameter EntityHost)
  (parameter EntityId)
  (parameter RadioId)
  (parameter EncodingScheme)
  (parameter TdlType)
  (parameter SampleRate)
  (parameter Samples)
```

```
  (parameter DataStruct)
 )
 (class Collision FED_BEST_EFFORT FED_RECEIVE
  (parameter TimeStamp)
  (parameter IssuingEntitySite)
  (parameter IssuingEntityHost)
  (parameter IssuingEntityId)
  (parameter CollidingEntitySite)
  (parameter CollidingEntityHost)
  (parameter CollidingEntityId)
  (parameter EventIdSite)
  (parameter EventIdHost)
  (parameter EventIdEvent)
  (parameter VelocityX)
  (parameter VelocityY)
  (parameter VelocityZ)
  (parameter Mass)
  (parameter RelLocationX)
  (parameter RelLocationY)
  (parameter RelLocationZ)
 )

 (class Manager FED_RELIABLE FED_RECEIVE
  (class Federate FED_RELIABLE FED_RECEIVE
   (parameter FromFederate)
   (class Alert FED_RELIABLE FED_RECEIVE
    (parameter AlertSeverity)
    (parameter AlertText)
    (parameter AlertID)
   )
   (class ServiceLog FED_RELIABLE FED_RECEIVE
    (parameter ServiceName)
    (parameter ServiceInitiator)
    (class ServiceLogArguments FED_RELIABLE FED_RECEIVE
     (parameter Handle1)
     (parameter Handle2)
     (parameter HandleSet)
     (parameter ObjectIDorCount)
     (parameter TagOrLabelOrName)
     (parameter Time)
     (parameter Enumeration)
     (parameter Boolean)
    )
   )
   (class ObjectInformation FED_RELIABLE FED_RECEIVE
    (parameter ObjectID)
    (parameter LockedAttributes)
    (parameter RegisteredClass)
    (parameter RepresentedClass)
   )
   (class PublishingClass FED_RELIABLE FED_RECEIVE
    (parameter ObjectClass)
    ;; format = ClassHandle:attrHandle,attrHandle,...,attrHandle
    (parameter InteractionClass)
    ;; format = ClassHandle
```

```
  )
  (class SubscribingClass FED_RELIABLE FED_RECEIVE
   (parameter ObjectClass)
   ;; format = ClassHandle:attrHandle,attrHandle,...,attrHandle
   (parameter InteractionClass)
   ;; format = ClassHandle
  )
  (class Action FED_RELIABLE FED_RECEIVE
   (parameter ToFederate)
   (class RequestPublicationTree FED_RELIABLE FED_RECEIVE
   )
   (class RequestSubscriptionTree FED_RELIABLE FED_RECEIVE
   )
   (class SetTiming FED_RELIABLE FED_RECEIVE
    (parameter FedReportPeriod)
    (parameter TimeReportPeriod)
    (parameter ObjectReportPeriod)
   )
   (class RequestObjectInformation FED_RELIABLE FED_RECEIVE
    (parameter ObjectID)
   )
   (class ModifyAttributeState FED_RELIABLE FED_RECEIVE
    (parameter ObjectID)
    (parameter AttributeID)
    (parameter TokenState)
   )
   (class RemoteServiceInvocation FED_RELIABLE FED_RECEIVE
    (class DoResignFederationExecution FED_RELIABLE FED_RECEIVE
      (parameter ResignAction)
    )
    (class DoDeleteObject FED_RELIABLE FED_RECEIVE
      (parameter ObjectID)
      (parameter Time)
      (parameter Tag)
    )
    (class DoSetLookahead FED_RELIABLE FED_RECEIVE
      (parameter Lookahead)
    )
    (class DoSetTimeConstrained FED_RELIABLE FED_RECEIVE
      (parameter State)
    )
    (class DoTurnRegulationOn FED_RELIABLE FED_RECEIVE
    )
    (class DoTurnRegulationOff FED_RELIABLE FED_RECEIVE
    )
   )
   (class Control FED_RELIABLE FED_RECEIVE
    (parameter SetServiceLogging)
    (parameter SetLogFile)
    (parameter DeleteObject)
    (parameter DequeueFIFO)
   )
  )
 )
)
```

```
;; FOR CCTTSAF AND MODSAF
(cctt_federation
 (class M1
   (attribute FORCE_ID integer32)
   (attribute ENTITYID_SITE integer32 )
   (attribute ENTITYID_APPLICATION integer32 )
   (attribute ENTITYID_ENTITY integer32 )
   (attribute LOCATION_X integer32 )
   (attribute LOCATION_Y integer32 )
   (attribute LOCATION_Z integer32 )
   (attribute VELOCITY_X integer32 )
   (attribute VELOCITY_Y integer32 )
   (attribute VELOCITY_Z integer32 )
   (attribute ACCELERATION_X integer32 )
   (attribute ACCELERATION_Y integer32 )
   (attribute ACCELERATION_Z integer32 )
   (attribute ORIENTATION_PSI integer32 )
   (attribute ORIENTATION_THETA integer32 )
   (attribute ORIENTATION_PHI integer32 )
   (attribute APPEARANCE_PAINT_SCHEME integer32 )
   (attribute DAMAGE_STATE_APPEARANCE integer32 )
   (attribute APPEARANCE_SMOKE integer32 )
   (attribute APPEARANCE_TRAILING integer32 )
   (attribute APPEARANCE_HATCH integer32 )
   (attribute APPEARANCE_LIGHTS integer32 )
   (attribute APPEARANCE_FLAMING integer32 )
   (attribute DAMAGE_STATE_MOBILITY integer32 )
   (attribute DAMAGE_STATE_FIRE_POWER integer32 )
 )
 (class T72
   (attribute FORCE_ID integer32)
   (attribute ENTITYID_SITE integer32 )
   (attribute ENTITYID_APPLICATION integer32 )
   (attribute ENTITYID_ENTITY integer32 )
   (attribute LOCATION_X integer32 )
   (attribute LOCATION_Y integer32 )
   (attribute LOCATION_Z integer32 )
   (attribute VELOCITY_X integer32 )
   (attribute VELOCITY_Y integer32 )
   (attribute VELOCITY_Z integer32 )
   (attribute ACCELERATION_X integer32 )
   (attribute ACCELERATION_Y integer32 )
   (attribute ACCELERATION_Z integer32 )
   (attribute ORIENTATION_PSI integer32 )
   (attribute ORIENTATION_THETA integer32 )
   (attribute ORIENTATION_PHI integer32 )
   (attribute APPEARANCE_PAINT_SCHEME integer32 )
   (attribute DAMAGE_STATE_APPEARANCE integer32 )
   (attribute APPEARANCE_SMOKE integer32 )
   (attribute APPEARANCE_TRAILING integer32 )
   (attribute APPEARANCE_HATCH integer32 )
   (attribute APPEARANCE_LIGHTS integer32 )
   (attribute APPEARANCE_FLAMING integer32 )
   (attribute DAMAGE_STATE_MOBILITY integer32 )
   (attribute DAMAGE_STATE_FIRE_POWER integer32 )
```

```
 )
 (interaction DETONATION
    (parameter Munition_ID integer32)
    (parameter Target_ID integer32)
    (parameter Firing_ID integer32)
    (parameter Location_X integer32)
    (parameter Location_Y integer32)
    (parameter Location_Z integer32)
    (parameter Relative_Location_X integer32)
    (parameter Relative_Location_Y integer32)
    (parameter Relative_Location_Z integer32)
    (parameter Velocity_X integer32)
    (parameter Velocity_Y integer32)
    (parameter Velocity_Z integer32)
    (parameter Burst_Descriptor_Warhead integer32)
    (parameter Burst_Descriptor_Fuze integer32)
    (parameter Burst_Descriptor_Detonation_Result integer32)
    (parameter Quantity integer32)
    (parameter Rate integer32)
 )
 (interaction WEAPON_FIRE
    (parameter Firing_ID integer32)
    (parameter Target_ID integer32)
    (parameter Location_X integer32)
    (parameter Location_Y integer32)
    (parameter Location_Z integer32)
    (parameter Velocity_X integer32)
    (parameter Velocity_Y integer32)
    (parameter Velocity_Z integer32)
    (parameter Munition_Type integer32)
    (parameter Quantity integer32)
    (parameter Rate integer32)
 )
)

(fcs_federation
 (class Entity
        (attribute Entity_ID_site integer32)
        (attribute Entity_ID_application integer32 )
        (attribute Entity_ID_entity integer32 )
        (attribute Force_ID integer32 )
        (attribute Entity_Type_Kind integer32 )
        (attribute Entity_Type_Domain integer32 )
        (attribute Entity_Type_Country integer32 )
        (attribute Entity_Type_Category integer32 )
        (attribute Entity_Type_Subcategory integer32 )
        (attribute Entity_Type_Specific integer32 )
        (attribute Entity_Type_Extra integer32 )
        (attribute Entity_Guise_Kind integer32 )
        (attribute Entity_Guise_Domain integer32 )
        (attribute Entity_Guise_Country integer32 )
        (attribute Entity_Guise_Category integer32 )
        (attribute Entity_Guise_Scategory integer32 )
        (attribute Entity_Guise_Specific integer32 )
        (attribute Entity_Guise_Extra integer32 )
```

```
        (attribute Orientation_Psi integer32 )
        (attribute Orientation_Theta integer32 )
        (attribute Orientation_Phi integer32 )
        (attribute Location_X integer32 )
        (attribute Location_Y integer32 )
        (attribute Location_Z integer32 )
        (attribute Velocity_X integer32 )
        (attribute Velocity_Y integer32 )
        (attribute Velocity_Z integer32 )
        (attribute Appearance integer32 )
        (attribute Dead_Reckoning_Algorithm integer32 )
        (attribute Acceleration_X integer32 )
        (attribute Acceleration_Y integer32 )
        (attribute Acceleration_Z integer32 )
)
(interaction Detonation
  (parameter Munition_ID_site integer32)
  (parameter Munition_ID_application integer32)
  (parameter Munition_ID_entity integer32)
  (parameter Attacker_ID_site integer32)
  (parameter Attacker_ID_application integer32)
  (parameter Attacker_ID_entity integer32)
  (parameter Target_ID_site integer32)
  (parameter Target_ID_application integer32)
  (parameter Target_ID_entity integer32)
  (parameter Event_ID_site integer32)
  (parameter Event_ID_application integer32)
  (parameter Event_ID_entity integer32)
  (parameter WorldLocationX integer32)
  (parameter WorldLocationY integer32)
  (parameter WorldLocationZ integer32)
  (parameter EntityLocationX integer32)
  (parameter EntityLocationY integer32)
  (parameter EntityLocationZ integer32)
  (parameter VelocityX integer32)
  (parameter VelocityY integer32)
  (parameter VelocityZ integer32)
  (parameter BurstKind integer32)
  (parameter BurstDomain integer32)
  (parameter BurstCountry integer32)
  (parameter BurstCategory integer32)
  (parameter BurstSubcategory integer32)
  (parameter BurstSpecific integer32)
  (parameter BurstExtra integer32)
  (parameter BurstWarhead integer32)
  (parameter BurstFuze integer32)
  (parameter BurstQuantity integer32)
  (parameter BurstRate integer32)
  (parameter DetonationResult integer32)
  (parameter TimeStamp integer32)
  (parameter ArticulatedStruct integer32)
)
(interaction Fire
  (parameter Launch_Platform_ID_site integer32 )
  (parameter Launch_Platform_ID_application integer32 )
```

```
    (parameter Launch_Platform_ID_entity integer32 )
    (parameter Target_ID_site integer32 )
    (parameter Target_ID_application integer32 )
    (parameter Target_ID_entity integer32 )
    (parameter Weapon_ID_site integer32 )
    (parameter Weapon_ID_application integer32)
    (parameter Weapon_ID_entity integer32)
    (parameter Event_ID_site integer32)
    (parameter Event_ID_application integer32)
    (parameter Event_ID_entity integer32)
    (parameter LocationX integer32 )
    (parameter LocationY integer32 )
    (parameter LocationZ integer32 )
    (parameter VelocityX integer32 )
    (parameter VelocityY integer32 )
    (parameter VelocityZ integer32 )
    (parameter Range integer32)
    (parameter BurstKind integer32)
    (parameter BurstDomain integer32)
    (parameter BurstCountry integer32)
    (parameter BurstCategory integer32)
    (parameter BurstSubcategory integer32)
    (parameter BurstSpecific integer32)
    (parameter BurstExtra integer32)
    (parameter BurstWarhead integer32)
    (parameter BurstFuze integer32)
    (parameter BurstQuantity integer32)
    (parameter BurstRate integer32)
    (parameter TimeStamp integer32)
  )
)

object_mappings
(
  ((cctt_federation M1 (FORCE_ID)) (fcs_federation Entity (Force_ID Entity_Type_Kind
Entity_Type_Domain Entity_Type_Country Entity_Type_Category Entity_Type_Subcategory
Entity_Type_Specific Entity_Type_Extra Entity_Guise_Kind Entity_Guise_Domain Entity_Guise_Country
Entity_Guise_Category Entity_Guise_Scategory Entity_Guise_Specific Entity_Guise_Extra
Dead_Reckoning_Algorithm )) force_id_pj N)
  ((cctt_federation M1 (ENTITYID_SITE ENTITYID_APPLICATION ENTITYID_ENTITY))
(fcs_federation Entity (Entity_ID_site Entity_ID_application Entity_ID_entity)) site_appl_entity_pj Y)
  ((cctt_federation M1 (LOCATION_X)) (fcs_federation Entity (Location_X)) identity N)
  ((cctt_federation M1 (LOCATION_Y)) (fcs_federation Entity (Location_Y)) identity N)
  ((cctt_federation M1 (LOCATION_Z)) (fcs_federation Entity (Location_Z)) identity N)
  ((cctt_federation M1 (VELOCITY_X)) (fcs_federation Entity (Velocity_X)) identity N)
  ((cctt_federation M1 (VELOCITY_Y)) (fcs_federation Entity (Velocity_Y)) identity N)
  ((cctt_federation M1 (VELOCITY_Z)) (fcs_federation Entity (Velocity_Z)) identity N)
  ((cctt_federation M1 (ACCELERATION_X)) (fcs_federation Entity (Acceleration_X)) identity N)
  ((cctt_federation M1 (ACCELERATION_Y)) (fcs_federation Entity (Acceleration_Y)) identity N)
  ((cctt_federation M1 (ACCELERATION_Z)) (fcs_federation Entity (Acceleration_Z)) identity N)
  ((cctt_federation M1 (ORIENTATION_PSI)) (fcs_federation Entity (Orientation_Psi)) identity N)
  ((cctt_federation M1 (ORIENTATION_THETA)) (fcs_federation Entity (Orientation_Theta)) identity N)
  ((cctt_federation M1 (ORIENTATION_PHI)) (fcs_federation Entity (Orientation_Phi)) identity N)
  ((cctt_federation M1 (APPEARANCE_PAINT_SCHEME DAMAGE_STATE_APPEARANCE
APPEARANCE_SMOKE APPEARANCE_TRAILING APPEARANCE_HATCH
```

```
APPEARANCE_LIGHTS APPEARANCE_FLAMING DAMAGE_STATE_MOBILITY
DAMAGE_STATE_FIRE_POWER)) (fcs_federation Entity (Appearance)) appearance_pj N)

 ((fcs_federation Entity (Force_ID )) (cctt_federation T72 (FORCE_ID)) force_id_jp N)
 ((fcs_federation Entity (Entity_ID_site Entity_ID_application Entity_ID_entity)) (cctt_federation T72
(ENTITYID_SITE ENTITYID_APPLICATION ENTITYID_ENTITY)) site_appl_entity_jp Y)
 ((fcs_federation Entity (Location_X))  (cctt_federation T72 (LOCATION_X)) identity N)
 ((fcs_federation Entity (Location_Y))  (cctt_federation T72 (LOCATION_Y)) identity N)
 ((fcs_federation Entity (Location_Z))  (cctt_federation T72 (LOCATION_Z)) identity N)
 ((fcs_federation Entity (Velocity_X))  (cctt_federation T72 (VELOCITY_X)) identity N)
 ((fcs_federation Entity (Velocity_Y))  (cctt_federation T72 (VELOCITY_Y)) identity N)
 ((fcs_federation Entity (Velocity_Z))  (cctt_federation T72 (VELOCITY_Z)) identity N)
 ((fcs_federation Entity (Acceleration_X)) (cctt_federation T72 (ACCELERATION_X)) identity N)
 ((fcs_federation Entity (Acceleration_Y)) (cctt_federation T72 (ACCELERATION_Y)) identity N)
 ((fcs_federation Entity (Acceleration_Z)) (cctt_federation T72 (ACCELERATION_Z)) identity N)
 ((fcs_federation Entity (Orientation_Psi))  (cctt_federation T72 (ORIENTATION_PSI))  identity N)
 ((fcs_federation Entity (Orientation_Theta)) (cctt_federation T72 (ORIENTATION_THETA)) identity N)
 ((fcs_federation Entity (Orientation_Phi))  (cctt_federation T72 (ORIENTATION_PHI))  identity N)
 ((fcs_federation Entity (Appearance))      (cctt_federation T72 (APPEARANCE_PAINT_SCHEME
DAMAGE_STATE_APPEARANCE APPEARANCE_SMOKE APPEARANCE_TRAILING
APPEARANCE_HATCH APPEARANCE_LIGHTS APPEARANCE_FLAMING
DAMAGE_STATE_MOBILITY DAMAGE_STATE_FIRE_POWER)) appearance_jp N)
)

inter_mappings
(
 ((cctt_federation DETONATION (Munition_ID))  (fcs_federation Detonation (Munition_ID_site
Munition_ID_application Munition_ID_entity)) null_id_pj)
 ((cctt_federation DETONATION (Target_ID))    (fcs_federation Detonation (Target_ID_site
Target_ID_application Target_ID_entity)) real_id_pj)
 ((cctt_federation DETONATION (Firing_ID))    (fcs_federation Detonation (Attacker_ID_site
Attacker_ID_application Attacker_ID_entity)) real_id_pj)
 ((cctt_federation DETONATION (Location_X))   (fcs_federation Detonation (WorldLocationX))
identity)
 ((cctt_federation DETONATION (Location_Y))   (fcs_federation Detonation (WorldLocationY))
identity)
 ((cctt_federation DETONATION (Location_Z))   (fcs_federation Detonation (WorldLocationZ))
identity)
 ((cctt_federation DETONATION (Velocity_X))   (fcs_federation Detonation (VelocityX))      identity )
 ((cctt_federation DETONATION (Velocity_Y))   (fcs_federation Detonation (VelocityY))      identity )
 ((cctt_federation DETONATION (Velocity_Z))   (fcs_federation Detonation (VelocityZ))      identity )
 ((cctt_federation DETONATION (Relative_Location_X)) (fcs_federation Detonation (EntityLocationX))
identity )
 ((cctt_federation DETONATION (Relative_Location_Y)) (fcs_federation Detonation (EntityLocationY))
identity )
 ((cctt_federation DETONATION (Relative_Location_Z)) (fcs_federation Detonation (EntityLocationZ))
identity )
 ((cctt_federation DETONATION (Burst_Descriptor_Warhead)) (fcs_federation Detonation
(BurstWarhead))     make_it5000 )
 ((cctt_federation DETONATION (Burst_Descriptor_Fuze))   (fcs_federation Detonation (BurstFuze))
make_it0 )
 ((cctt_federation DETONATION (Burst_Descriptor_Detonation_Result))   (fcs_federation Detonation
(DetonationResult))      make_it_one )
 ((cctt_federation DETONATION (Quantity))             (fcs_federation Detonation (BurstQuantity))
identity )
```

((cctt_federation DETONATION (Rate)) (fcs_federation Detonation (BurstRate BurstKind BurstDomain BurstCountry BurstCategory BurstSubcategory BurstSpecific BurstExtra Event_ID_site Event_ID_application Event_ID_entity TimeStamp ArticulatedStruct)) rate_plus_extra_pj)

((fcs_federation Detonation (Munition_ID_site Munition_ID_application Munition_ID_entity)) (cctt_federation DETONATION (Munition_ID)) null_id_jp)
((fcs_federation Detonation (Attacker_ID_site Attacker_ID_application Attacker_ID_entity)) (cctt_federation DETONATION (Firing_ID)) real_id_jp)
((fcs_federation Detonation (Target_ID_site Target_ID_application Target_ID_entity)) (cctt_federation DETONATION (Target_ID)) real_id_jp)
((fcs_federation Detonation (WorldLocationX)) (cctt_federation DETONATION (Location_X)) identity)
((fcs_federation Detonation (WorldLocationY)) (cctt_federation DETONATION (Location_Y)) identity)
((fcs_federation Detonation (WorldLocationZ)) (cctt_federation DETONATION (Location_Z)) identity)
((fcs_federation Detonation (VelocityX)) (cctt_federation DETONATION (Velocity_X)) identity)
((fcs_federation Detonation (VelocityY)) (cctt_federation DETONATION (Velocity_Y)) identity)
((fcs_federation Detonation (VelocityZ)) (cctt_federation DETONATION (Velocity_Z)) identity)
((fcs_federation Detonation (EntityLocationX)) (cctt_federation DETONATION (Relative_Location_X)) identity)
((fcs_federation Detonation (EntityLocationY)) (cctt_federation DETONATION (Relative_Location_Y)) identity)
((fcs_federation Detonation (EntityLocationZ)) (cctt_federation DETONATION (Relative_Location_Z)) identity)
((fcs_federation Detonation (BurstWarhead)) (cctt_federation DETONATION (Burst_Descriptor_Warhead)) make_it1400)
((fcs_federation Detonation (BurstFuze)) (cctt_federation DETONATION (Burst_Descriptor_Fuze)) make_it1000)
((fcs_federation Detonation (BurstQuantity)) (cctt_federation DETONATION (Quantity)) identity)
((fcs_federation Detonation (BurstRate)) (cctt_federation DETONATION (Rate)) identity)
((fcs_federation Detonation (DetonationResult)) (cctt_federation DETONATION (Burst_Descriptor_Detonation_Result)) identity)

((cctt_federation WEAPON_FIRE (Firing_ID)) (fcs_federation Fire (Launch_Platform_ID_site Launch_Platform_ID_application Launch_Platform_ID_entity)) real_id_pj)
((cctt_federation WEAPON_FIRE (Target_ID)) (fcs_federation Fire (Target_ID_site Target_ID_application Target_ID_entity)) real_id_pj)
((cctt_federation WEAPON_FIRE (Location_X)) (fcs_federation Fire (LocationX)) identity)
((cctt_federation WEAPON_FIRE (Location_Y)) (fcs_federation Fire (LocationY)) identity)
((cctt_federation WEAPON_FIRE (Location_Z)) (fcs_federation Fire (LocationZ)) identity)
((cctt_federation WEAPON_FIRE (Velocity_X)) (fcs_federation Fire (VelocityX)) identity)
((cctt_federation WEAPON_FIRE (Velocity_Y)) (fcs_federation Fire (VelocityY)) identity)
((cctt_federation WEAPON_FIRE (Velocity_Z)) (fcs_federation Fire (VelocityZ)) identity)
((cctt_federation WEAPON_FIRE (Munition_Type)) (fcs_federation Fire (BurstKind BurstDomain BurstCountry BurstCategory BurstSubcategory BurstSpecific BurstExtra BurstWarhead BurstFuze)) munition_type_pj)
((cctt_federation WEAPON_FIRE (Quantity)) (fcs_federation Fire (BurstQuantity)) identity)
((cctt_federation WEAPON_FIRE (Rate)) (fcs_federation Fire (BurstRate Weapon_ID_site Weapon_ID_application Weapon_ID_entity Event_ID_site Event_ID_application Event_ID_entity Range TimeStamp)) fire_rate_plus_extra_pj)

((fcs_federation Fire (Launch_Platform_ID_site Launch_Platform_ID_application Launch_Platform_ID_entity)) (cctt_federation WEAPON_FIRE (Firing_ID)) real_id_jp)
((fcs_federation Fire (Target_ID_site Target_ID_application Target_ID_entity)) (cctt_federation WEAPON_FIRE (Target_ID)) real_id_jp)

```
  ((fcs_federation Fire (LocationX)) (cctt_federation WEAPON_FIRE (Location_X))       identity)
  ((fcs_federation Fire (LocationY)) (cctt_federation WEAPON_FIRE (Location_Y))       identity)
  ((fcs_federation Fire (LocationZ)) (cctt_federation WEAPON_FIRE (Location_Z))       identity)
  ((fcs_federation Fire (VelocityX)) (cctt_federation WEAPON_FIRE (Velocity_X))       identity)
  ((fcs_federation Fire (VelocityY)) (cctt_federation WEAPON_FIRE (Velocity_Y))       identity)
  ((fcs_federation Fire (VelocityZ)) (cctt_federation WEAPON_FIRE (Velocity_Z))       identity)
  ((fcs_federation Fire (BurstKind BurstDomain BurstCountry BurstCategory BurstSubcategory
BurstSpecific BurstExtra BurstWarhead BurstFuze)) (cctt_federation WEAPON_FIRE (Munition_Type))
munition_type_jp )
  ((fcs_federation Fire (BurstQuantity )) (cctt_federation WEAPON_FIRE (Quantity))    identity )
  ((fcs_federation Fire (BurstRate ))     (cctt_federation WEAPON_FIRE (Rate))        identity )
)
```

Federation Execution Summary Table

Federation Execution Name:	FCS Federation & CCTT Federation
Number of Concurrent Federation Executions (total including this Federation Execution):	2
RTI Software Used (Version):	1.0

Federate Summary Information

	Name	API (C++,Ada, IDL, Java)	Time Management Switches		Host (assign # to each host) (List data on Host Table)	LAN (assign # to each LAN) (List data on LAN Tables)
			Regulating (y or n)	Constraining (y or n)		
Fed 1	Surrogate For FCS	Ada	N	N	1	1
Fed 2	FCS	C	N	N	2	1
Fed 3	Surrogate For cctt_federation	Ada	N	N	3	1
Fed 4	dis_federate	Ada	N	N	4	1
Fed 5	edb_federate	Ada	N	N	5	1
Fed 6						
Fed 7						
Fed 8						
Fed 9						
Fed 10						
Fed 11						
Fed N						

Host Table

	Hardware	Operating System	Memory available to RTI (MB)	% CPU Available to RTI
Host $_1$	IBM 43P	AIX 4.1		
Host $_2$	Sun	SunOS5.5.2		
Host $_3$	IBM 43P	AIX 4.1		
Host $_4$	IBM 43P	AIX 4.1		
Host $_5$	IBM 43P	AIX 4.1		
Host $_6$				
Host $_7$				
Host $_n$				

LAN Tables

LAN Table 1: LAN Descriptions

	Physical Type (Ethernet, ATM,etc.)	Throughtput Available to FEDEX
LAN_1	Ethernet 10 Base-T	
LAN_2		
LAN_3		
LAN_4		
LAN_5		
LAN_6		
LAN_7		
LAN_n		

LAN Table 2: LAN to LAN Connectivity

	LAN_1	LAN_2	LAN_3	LAN_4	LAN_5	LAN_6	.	.	.	LAN_N
LAN_1										
LAN_2	1. ____ • 2. ____ • 3. ____ •									
LAN_3	1. ____ • 2. ____ • 3. ____ •	1. ____ • 2. ____ • 3. ____ •								
LAN_4	1. ____ • 2. ____ • 3. ____ •	1. ____ • 2. ____ • 3. ____ •	1. ____ • 2. ____ • 3. ____ •							
LAN_5	1. ____ • 2. ____ • 3. ____ •	1. ____ • 2. ____ • 3. ____ •	1. ____ • 2. ____ • 3. ____ •	1. ____ • 2. ____ • 3. ____ •						
LAN_6	1. ____ • 2. ____ • 3. ____ •	1. ____ • 2. ____ • 3. ____ •	1. ____ • 2. ____ • 3. ____ •	1. ____ • 2. ____ • 3. ____ •	1. ____ • 2. ____ • 3. ____ •					
...										
LAN_N	1. ____ • 2. ____ • 3. ____ •	1. ____ • 2. ____ • 3. ____ •	1. ____ • 2. ____ • 3. ____ •	1. ____ • 2. ____ • 3. ____ •	1. ____ • 2. ____ • 3. ____ •	1. ____ • 2. ____ • 3. ____ •				

•

1. Device type means type of switch employed to connect the LANs
2. Throughput means the effective throughput available through the LAN connection for Federation execution, expressed in Mb
3. Latency contribution from devices connecting the LANs, expressed in milliseconds.

RTI Services Table (Check if service to be used at least once during this Federation execution)		
Service	**IF Ref**	**Service Used?**
Create Federation Execution	2.1	✓
Destroy Federation Execution	2.2	✓
Join Federation Execution	2.3	✓
Resign Federation Execution	2.4	✓
Request **Pause**	2.5	
Initiate **Pause**	2.6	
Pause Achieved	2.7	
Request **Resume**	2.8	
Initiate **Resume**	2.9	
Resume Achieved	2.10	
Request **Federation Save**	2.11	
Initiate **Federation Save**	2.12	
Federation Save Begun	2.13	
Federation Save Achieved	2.14	
Request **Restore**	2.15	
Initiate **Restore**	2.16	
Restore Achieved	2.17	
Publish Object Class	3.1	✓
Subscribe Object Class **Attributes**	3.2	✓
Publish Interaction	3.3	✓
Subscribe Interaction	3.4	✓
Control Updates	3.5	✓
Control Interactions	3.6	✓
Request ID	4.1	✓
Register Object	4.2	✓
Update Attribute Values	4.3	✓
Delete Object	4.8	
Remove Object	4.9	
Change Attribute Transportation Type	4.10	
Change Attribute Order Type	4.11	
Change Interaction Transportation Type	4.12	
Change Interaction Order Type	4.13	
Request Attribute Value Update	4.14	✓
Provide Attribute Value Update	4.15	✓
Retract	4.16	
Reflect Retract	4.17	
Request Attribute **Ownership Divestiture**	5.1	
Request Attribute **Ownership Assumptio**	5.2	
Attribute Ownership **Divestiture Notifica**	5.3	
Attribute Ownership **Acquisition Notifica**	5.4	
Request Attribute **Ownership Acquisitio**	5.5	
Request Attribute **Ownership Release**	5.6	
Query Attribute Ownership	5.7	
Inform Attribute Ownership	5.8	
Is **Attribute Owned by Federate?**	5.9	
Request **Federation Time**	6.1	
Request **LBTS**	6.2	
Request **Federate Time**	6.3	
Request **Min Next Event Time**	6.4	
Set **Lookahead**	6.5	
Request **Lookahead**	6.6	
Time Advance Request	6.7	

Host Table

	Hardware	Operating System	Memory available to RTI (MB)	% CPU Available to RTI
Host $_1$	IBM 43P	AIX 4.1		
Host $_2$	IBM 43P	AIX 4.1		
Host $_3$	IBM 43P	AIX 4.1		
Host $_4$				
Host $_5$				
Host $_6$				
Host $_7$				
Host $_n$				

NOTE:
Complete one of these tables for each Federation execution

LAN Tables

LAN Table 1: LAN Descriptions

	Physical Type (Ethernet, ATM, etc.)	Throughtput Available to FEDEX
LAN_1	Ethernet 10 Base-T	
LAN_2		
LAN_3		
LAN_4		
LAN_5		
LAN_6		
LAN_7		
LAN_n		

NOTE:
Complete one of these tables for each Federation execution

LAN Table 2: LAN to LAN Connectivity

	LAN_1	LAN_2	LAN_3	LAN_4	LAN_5	LAN_6	.	.	.	LAN_N
LAN_1										
LAN_2	1. ____ * 2. ____ * 3. ____ *									
LAN_3	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *								
LAN_4	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *							
LAN_5	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *						
LAN_6	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *					
...										
LAN_N	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *				

*

1. Device type means type of switch employed to connect the LANs
2. Throughput means the effective throughput available through the LAN connection for Federation execution, expressed in Mb
3. Latency contribution from devices connecting the LANs, expressed in milliseconds.

RTI Services Table (Check if service to be used at least once during this Federation execution)		
Service	**IF Ref**	**Service Used?**
Create Federation Execution	2.1	✓
Destroy Federation Execution	2.2	✓
Join Federation Execution	2.3	✓
Resign Federation Execution	2.4	✓
Request **Pause**	2.5	
Initiate **Pause**	2.6	
Pause Achieved	2.7	
Request **Resume**	2.8	
Initiate **Resume**	2.9	
Resume Achieved	2.10	
Request **Federation Save**	2.11	
Initiate **Federation Save**	2.12	
Federation Save Begun	2.13	
Federation Save Achieved	2.14	
Request **Restore**	2.15	
Initiate **Restore**	2.16	
Restore Achieved	2.17	
Publish Object Class	3.1	✓
Subscribe Object Class **Attributes**	3.2	✓
Publish Interaction	3.3	✓
Subscribe Interaction	3.4	✓
Control Updates	3.5	✓
Control Interactions	3.6	✓
Request ID	4.1	✓
Register Object	4.2	✓
Update Attribute Values	4.3	✓
Delete Object	4.8	
Remove Object	4.9	
Change Attribute Transportation Type	4.10	
Change Attribute Order Type	4.11	
Change Interaction Transportation Type	4.12	
Change Interaction Order Type	4.13	
Request Attribute Value Update	4.14	✓
Provide Attribute Value Update	4.15	✓
Retract	4.16	
Reflect Retract	4.17	
Request Attribute **Ownership Divestiture**	5.1	
Request Attribute **Ownership Assumptio**	5.2	
Attribute Ownership **Divestiture Notifica**	5.3	
Attribute Ownership **Acquisition Notifica**	5.4	
Request Attribute **Ownership Acquisitio**	5.5	
Request Attribute **Ownership Release**	5.6	
Query Attribute Ownership	5.7	
Inform Attribute Ownership	5.8	
Is **Attribute Owned by Federate?**	5.9	
Request **Federation Time**	6.1	
Request **LBTS**	6.2	
Request **Federate Time**	6.3	
Request **Min Next Event Time**	6.4	
Set **Lookahead**	6.5	
Request **Lookahead**	6.6	
Time Advance Request	6.7	

NOTE: Complete one of these tables for each Federation execution

Object/Interaction Table

NOTE: Complete one of these tables for each Federate

Federate ,
Surrogate_for : cctt_federation

					If Update =" y"					If Subscribe = y?	Ownership	
Object/ Interaction Class	**Attribute/ Parameter**	**Count**	**Size**	**Update? (y or n)**	**Update Rate # updates/unit time**	**Update Grouping (Assign same letter to attributes which will all be updated at the same time)**	**Transport R= Reliable B= Best Effort**	**Ordering TSO or FIFO**	**Subscribe? (y or n)**	**Maximum tolerable latency from any source (milliseconds)**	**Attribute Ownership Transfer Rate # times/unit time**	**Ownership Transfer Grouping (Assign same letter to attributes which will all be transferred together)**
M1												
	FORCE_ID		2 bytes	n			R	FIFO	y			
	ENTITYID_SITE		2 bytes	n			R	FIFO	y			
	ENTITYID_APPLICATION		2 bytes	n			R	FIFO	y			
	ENTITYID_ENTITY		2 bytes	n			R	FIFO	y			
	LOCATION_X		8 bytes	y	15	A	R	FIFO	y			
	LOCATION_Y		8 bytes	y	15	A	R	FIFO	y			
	LOCATION_Z		8 bytes	y	15	A	R	FIFO	y			
	VELOCITY_X		4 bytes	y	15	A	R	FIFO	y			
	VELOCITY_Y		4 bytes	y	15	A	R	FIFO	y			
	VELOCITY_Z		4 bytes	y	15	A	R	FIFO	y			
	ACCELERATION_X		4 bytes	y	15	A	R	FIFO	y			
	ACCELERATION_Y		4 bytes	y	15	A	R	FIFO	y			
	ACCELERATION_Z		4 bytes	y	15	A	R	FIFO	y			
	ORIENTATION_PSI		4 bytes	y	15	A	R	FIFO	y			
	ORIENTATION_THETA		4 bytes	y	15	A	R	FIFO	y			
	ORIENTATION_PHI		4 bytes	y	15	A	R	FIFO	y			
	APPEARANCE_PAINT_SCHEME		2 bytes	y	15	A	R	FIFO	y			
	DAMAGE_STATE_APPEARANCE		2 bytes	y	15	A	R	FIFO	y			
	APPEARANCE_SMOKE		2 bytes	y	15	A	R	FIFO	y			
	APPEARANCE_TRAILING		2 bytes	y	15	A	R	FIFO	y			
	APPEARANCE_HATCH		2 bytes	y	15	A	R	FIFO	y			
	APPEARANCE_LIGHTS		2 bytes	y	15	A	R	FIFO	y			
	APPEARANCE_FLAMING		2 bytes	y	15	A	R	FIFO	y			
	DAMAGE_STATE_MOBILITY		2 bytes	y	15	A	R	FIFO	y			
	DAMAGE_STATE_FIRE_POWER		2 bytes	y	15	A	R	FIFO	y			
T72												
	FORCE_ID		2 bytes	n			R	FIFO	y			
	ENTITYID_SITE		2 bytes	n			R	FIFO	y			
	ENTITYID_APPLICATION		2 bytes	n			R	FIFO	y			
	ENTITYID_ENTITY		2 bytes	n			R	FIFO	y			
	LOCATION_X		8 bytes	y	15	B	R	FIFO	y			
	LOCATION_Y		8 bytes	y	15	B	R	FIFO	y			
	LOCATION_Z		8 bytes	y	15	B	R	FIFO	y			
	VELOCITY_X		4 bytes	y	15	B	R	FIFO	y			

	VELOCITY_Y		4 bytes	y	15	B	R	FIFO	y			
	VELOCITY_Z		4 bytes	y	15	B	R	FIFO	y			
	ACCELERATION_X		4 bytes	y	15	B	R	FIFO	y			
	ACCELERATION_Y		4 bytes	y	15	B	R	FIFO	y			
	ACCELERATION_Z		4 bytes	y	15	B	R	FIFO	y			
	ORIENTATION_PSI		4 bytes	y	15	B	R	FIFO	y			
	ORIENTATION_THETA		4 bytes	y	15	B	R	FIFO	y			
	ORIENTATION_PHI		4 bytes	y	15	B	R	FIFO	y			
	APPEARANCE_PAINT_SCHEME		2 bytes	y	15	B	R	FIFO	y			
	DAMAGE_STATE_APPEARANCE		2 bytes	y	15	B	R	FIFO	y			
	APPEARANCE_SMOKE		2 bytes	y	15	B	R	FIFO	y			
	APPEARANCE_TRAILING		2 bytes	y	15	B	R	FIFO	y			
	APPEARANCE_HATCH		2 bytes	y	15	B	R	FIFO	y			
	APPEARANCE_LIGHTS		2 bytes	y	15	B	R	FIFO	y			
	APPEARANCE_FLAMING		2 bytes	y	15	B	R	FIFO	y			
	DAMAGE_STATE_MOBILITY		2 bytes	y	15	B	R	FIFO	y			
	DAMAGE_STATE_FIRE_POWER		2 bytes	y	15	B	R	FIFO	y			
DETONATION	Munition_ID		2 bytes			C	R	FIFO	y			
	Target_ID		2 bytes			C	R	FIFO	y			
	Firing_ID		2 bytes			C	R	FIFO	y			
	Location_X		8 bytes			C	R	FIFO	y			
	Location_Y		8 bytes			C	R	FIFO	y			
	Location_Z		8 bytes			C	R	FIFO	y			
	Relative_Location_X		4 bytes			C	R	FIFO	y			
	Relative_Location_Y		4 bytes			C	R	FIFO	y			
	Relative_Location_Z		4 bytes			C	R	FIFO	y			
	Velocity_X		4 bytes			C	R	FIFO	y			
	Velocity_Y		4 bytes			C	R	FIFO	y			
	Velocity_Z		4 bytes			C	R	FIFO	y			
	Burst_Descriptor_Warhead		2 bytes			C	R	FIFO	y			
	Burst_Descriptor_Fuze		2 bytes			C	R	FIFO	y			
	Burst_Descriptor_Detonation_Result		2 bytes			C	R	FIFO	y			
	Quantity		2 bytes			C	R	FIFO	y			
	Rate		2 bytes			C	R	FIFO	y			
WEAPON_FIRE	Firing_ID		2 bytes			D	R	FIFO	y			
	Target_ID		2 bytes			D	R	FIFO	y			
	Location_X		8 bytes			D	R	FIFO	y			
	Location_Y		8 bytes			D	R	FIFO	y			
	Location_Z		8 bytes			D	R	FIFO	y			
	Velocity_X		8 bytes			D	R	FIFO	y			
	Velocity_Y		8 bytes			D	R	FIFO	y			
	Velocity_Z		8 bytes			D	R	FIFO	y			
	Munition_Type		2 bytes			D	R	FIFO	y			
	Quantity		2 bytes			D	R	FIFO	y			
	Rate		2 bytes			D	R	FIFO	y			
COLLISION	Issuing_ID		2 bytes			E	R	FIFO	y			
	Colliding_ID		2 bytes			E	R	FIFO	y			
	Mass		2 bytes			E	R	FIFO	y			
	Relative_Location_X		4 bytes			E	R	FIFO	y			

	Relative_Location_Y		4 bytes			E	R	FIFO	y			
	Relative_Location_Z		4 bytes			E	R	FIFO	y			
	Velocity_X		4 bytes			E	R	FIFO	y			
	Velocity_Y		4 bytes			E	R	FIFO	y			
	Velocity_Z		4 bytes			E	R	FIFO	y			

Object/Interaction Table

NOTE: Complete one of these tables for each Federate

Federate,
edb_federate

					If Update = "y"					If Subscribe = y?	Ownership	
Object/ Interaction Class	**Attribute/ Parameter**	**Count**	**Size**	**Update? (y or n)**	**Update Rate # updates/unit time**	**Update Grouping (Assign same letter to attributes which will all be updated at the same time)**	**Transport R= Reliable B= Best Effort**	**Ordering TSO or FIFO**	**Subscribe? (y or n)**	**Maximum tolerable latency from any source (milliseconds)**	**Attribute Ownership Transfer Rate # times/unit time**	**Ownership Transfer Grouping (Assign same letter to attributes which will all be transferred together)**
M1												
	FORCE_ID		2 bytes	n			R	FIFO	y			
	ENTITYID_SITE		2 bytes	n			R	FIFO	y			
	ENTITYID_APPLICATION		2 bytes	n			R	FIFO	y			
	ENTITYID_ENTITY		2 bytes	n			R	FIFO	y			
	LOCATION_X		8 bytes	y	15	A	R	FIFO	y			
	LOCATION_Y		8 bytes	y	15	A	R	FIFO	y			
	LOCATION_Z		8 bytes	y	15	A	R	FIFO	y			
	VELOCITY_X		4 bytes	y	15	A	R	FIFO	y			
	VELOCITY_Y		4 bytes	y	15	A	R	FIFO	y			
	VELOCITY_Z		4 bytes	y	15	A	R	FIFO	y			
	ACCELERATION_X		4 bytes	y	15	A	R	FIFO	y			
	ACCELERATION_Y		4 bytes	y	15	A	R	FIFO	y			
	ACCELERATION_Z		4 bytes	y	15	A	R	FIFO	y			
	ORIENTATION_PSI		4 bytes	y	15	A	R	FIFO	y			
	ORIENTATION_THETA		4 bytes	y	15	A	R	FIFO	y			
	ORIENTATION_PHI		4 bytes	y	15	A	R	FIFO	y			
	APPEARANCE_PAINT_SCHEME		2 bytes	y	15	A	R	FIFO	y			
	DAMAGE_STATE_APPEARANCE		2 bytes	y	15	A	R	FIFO	y			
	APPEARANCE_SMOKE		2 bytes	y	15	A	R	FIFO	y			
	APPEARANCE_TRAILING		2 bytes	y	15	A	R	FIFO	y			
	APPEARANCE_HATCH		2 bytes	y	15	A	R	FIFO	y			
	APPEARANCE_LIGHTS		2 bytes	y	15	A	R	FIFO	y			
	APPEARANCE_FLAMING		2 bytes	y	15	A	R	FIFO	y			
	DAMAGE_STATE_MOBILITY		2 bytes	y	15	A	R	FIFO	y			
	DAMAGE_STATE_FIRE_POWER		2 bytes	y	15	A	R	FIFO	y			
T72												
	FORCE_ID		2 bytes	n			R	FIFO	y			
	ENTITYID_SITE		2 bytes	n			R	FIFO	y			
	ENTITYID_APPLICATION		2 bytes	n			R	FIFO	y			
	ENTITYID_ENTITY		2 bytes	n			R	FIFO	y			
	LOCATION_X		8 bytes	y	15	B	R	FIFO	y			
	LOCATION_Y		8 bytes	y	15	B	R	FIFO	y			
	LOCATION_Z		8 bytes	y	15	B	R	FIFO	y			
	VELOCITY_X		4 bytes	y	15	B	R	FIFO	y			

	VELOCITY_Y		4 bytes	y	15	B	R	FIFO	y				
	VELOCITY_Z		4 bytes	y	15	B	R	FIFO	y				
	ACCELERATION_X		4 bytes	y	15	B	R	FIFO	y				
	ACCELERATION_Y		4 bytes	y	15	B	R	FIFO	y				
	ACCELERATION_Z		4 bytes	y	15	B	R	FIFO	y				
	ORIENTATION_PSI		4 bytes	y	15	B	R	FIFO	y				
	ORIENTATION_THETA		4 bytes	y	15	B	R	FIFO	y				
	ORIENTATION_PHI		4 bytes	y	15	B	R	FIFO	y				
	APPEARANCE_PAINT_SCHEME		2 bytes	y	15	B	R	FIFO	y				
	DAMAGE_STATE_APPEARANCE		2 bytes	y	15	B	R	FIFO	y				
	APPEARANCE_SMOKE		2 bytes	y	15	B	R	FIFO	y				
	APPEARANCE_TRAILING		2 bytes	y	15	B	R	FIFO	y				
	APPEARANCE_HATCH		2 bytes	y	15	B	R	FIFO	y				
	APPEARANCE_LIGHTS		2 bytes	y	15	B	R	FIFO	y				
	APPEARANCE_FLAMING		2 bytes	y	15	B	R	FIFO	y				
	DAMAGE_STATE_MOBILITY		2 bytes	y	15	B	R	FIFO	y				
	DAMAGE_STATE_FIRE_POWER		2 bytes	y	15	B	R	FIFO	y				

Object/Interaction Table

NOTE: Complete one of these tables for each Federate

Federate #
dis_federate

					If Update =" y"					If Subscribe = y?	Ownership	
Object/ Interaction Class	**Attribute/ Parameter**	**Count**	**Size**	**Update? (y or n)**	**Update Rate # updates/unit time**	**Update Grouping (Assign same letter to attributes which will all be updated at the same time)**	**Transport R= Reliable B= Best Effort**	**Ordering TSO or FIFO**	**Subscribe? (y or n)**	**Maximum tolerable latency from any source (milliseconds)**	**Attribute Ownership Transfer Rate # times/unit time**	**Ownership Transfer Grouping (Assign same letter to attributes which will all be transferred together)**
DETONATION	Munition_ID		2 bytes			A	R	FIFO	y			
	Target_ID		2 bytes			A	R	FIFO	y			
	Firing_ID		2 bytes			A	R	FIFO	y			
	Location_X		8 bytes			A	R	FIFO	y			
	Location_Y		8 bytes			A	R	FIFO	y			
	Location_Z		8 bytes			A	R	FIFO	y			
	Relative_Location_X		4 bytes			A	R	FIFO	y			
	Relative_Location_Y		4 bytes			A	R	FIFO	y			
	Relative_Location_Z		4 bytes			A	R	FIFO	y			
	Velocity_X		4 bytes			A	R	FIFO	y			
	Velocity_Y		4 bytes			A	R	FIFO	y			
	Velocity_Z		4 bytes			A	R	FIFO	y			
	Burst_Descriptor_Warhead		2 bytes			A	R	FIFO	y			
	Burst_Descriptor_Fuze		2 bytes			A	R	FIFO	y			
	Burst_Descriptor_Detonation_Result		2 bytes			A	R	FIFO	y			
	Quantity		2 bytes			A	R	FIFO	y			
	Rate		2 bytes			A	R	FIFO	y			
WEAPON_FIRE	Firing_ID		2 bytes			B	R	FIFO	y			
	Target_ID		2 bytes			B	R	FIFO	y			
	Location_X		8 bytes			B	R	FIFO	y			
	Location_Y		8 bytes			B	R	FIFO	y			
	Location_Z		8 bytes			B	R	FIFO	y			
	Velocity_X		8 bytes			B	R	FIFO	y			
	Velocity_Y		8 bytes			B	R	FIFO	y			
	Velocity_Z		8 bytes			B	R	FIFO	y			
	Munition_Type		2 bytes			B	R	FIFO	y			
	Quantity		2 bytes			B	R	FIFO	y			
	Rate		2 bytes			B	R	FIFO	y			
COLLISION	Issuing_ID		2 bytes			C	R	FIFO	y			
	Colliding_ID		2 bytes			C	R	FIFO	y			
	Mass		2 bytes			C	R	FIFO	y			
	Relative_Location_X		4 bytes			C	R	FIFO	y			
	Relative_Location_Y		4 bytes			C	R	FIFO	y			
	Relative_Location_Z		4 bytes			C	R	FIFO	y			

	Velocity_X		4 bytes			C	R	FIFO	y				
	Velocity_Y		4 bytes			C	R	FIFO	y				
	Velocity_Z		4 bytes			C	R	FIFO	y				

Federation Execution Summary Table

Federation Execution Name: FCS Federation

Number of Concurrent Federation Executions (total including this Federation Execution): 2

RTI Software Used (Version): 1.0

NOTE:
Complete one of these tables for each Federation execution

Federate Summary Information

	Name	API (C++,Ada, IDL, Java)	Time Management Switches		Host (assign # to each host) (List data on Host Table)	LAN (assign # to each LAN) (List data on LAN Tables)
			Regulating (y or n)	Constraining (y or n)		
Fed 1	Surrogate For FCS	Ada	N	N	1	1
Fed 2	FCS	C	N	N	2	1
Fed 3						
Fed 4						
Fed 5						
Fed 6						
Fed 7						
Fed 8						
Fed 9						
Fed 10						
Fed 11						
Fed N						

Host Table

	Hardware	Operating System	Memory available to RTI (MB)	% CPU Available to RTI
Host $_1$	IBM 43P	AIX 4.1		
Host $_2$	Sun	SunOS5.5.2		
Host $_3$				
Host $_4$				
Host $_5$				
Host $_6$				
Host $_7$				
Host $_n$				

NOTE:
Complete one of these tables for each Federation execution

LAN Tables

LAN Table 1: LAN Descriptions

	Physical Type (Ethernet, ATM,etc.)	Throughtput Available to FEDEX
LAN_1	Ethernet 10 Base-T	
LAN_2		
LAN_3		
LAN_4		
LAN_5		
LAN_6		
LAN_7		
LAN_n		

NOTE:
Complete one of these tables for each Federation execution

LAN Table 2: LAN to LAN Connectivity

	LAN_1	LAN_2	LAN_3	LAN_4	LAN_5	LAN_6	.	.	.	LAN_N
LAN_1										
LAN_2	1. ____ 2. ____ 3. ____									
LAN_3	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____								
LAN_4	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____							
LAN_5	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____						
LAN_6	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____					
...										
LAN_N	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____				

1. Device type means type of switch employed to connect the LANs
2. Throughput means the effective throughput available through the LAN connection for Federation execution, expressed in Mb
3. Latency contribution from devices connecting the LANs, expressed in milliseconds.

RTI Services Table (Check if service to be used at least once during this Federation execution)		
Service	**IF Ref**	**Service Used?**
Create Federation Execution	2.1	✓
Destroy Federation Execution	2.2	✓
Join Federation Execution	2.3	✓
Resign Federation Execution	2.4	✓
Request **Pause**	2.5	
Initiate **Pause**	2.6	
Pause Achieved	2.7	
Request **Resume**	2.8	
Initiate **Resume**	2.9	
Resume Achieved	2.10	
Request **Federation Save**	2.11	
Initiate **Federation Save**	2.12	
Federation Save Begun	2.13	
Federation Save Achieved	2.14	
Request **Restore**	2.15	
Initiate **Restore**	2.16	
Restore Achieved	2.17	
Publish Object Class	3.1	✓
Subscribe Object Class **Attributes**	3.2	✓
Publish Interaction	3.3	✓
Subscribe Interaction	3.4	✓
Control Updates	3.5	✓
Control Interactions	3.6	✓
Request ID	4.1	✓
Register Object	4.2	✓
Update Attribute Values	4.3	✓
Delete Object	4.8	
Remove Object	4.9	
Change Attribute Transportation Type	4.10	
Change Attribute Order Type	4.11	
Change Interaction Transportation Type	4.12	
Change Interaction Order Type	4.13	
Request Attribute Value Update	4.14	✓
Provide Attribute Value Update	4.15	✓
Retract	4.16	
Reflect Retract	4.17	
Request Attribute **Ownership Divestiture**	5.1	
Request Attribute **Ownership Assumptio**	5.2	
Attribute Ownership **Divestiture Notifica**	5.3	
Attribute Ownership **Acquisition Notifica**	5.4	
Request Attribute **Ownership Acquisitio**	5.5	
Request Attribute **Ownership Release**	5.6	
Query Attribute Ownership	5.7	
Inform Attribute Ownership	5.8	
Is **Attribute Owned by Federate?**	5.9	
Request **Federation Time**	6.1	
Request **LBTS**	6.2	
Request **Federate Time**	6.3	
Request **Min Next Event Time**	6.4	
Set **Lookahead**	6.5	
Request **Lookahead**	6.6	
Time Advance Request	6.7	

NOTE: Complete one of these tables for each Federation execution

Object/Interaction Table

NOTE: Complete one of these tables for each Federate

Federate #
JPSD Modsaf

					If Update =" y"					If Subscribe = y?	Ownership	
Object/ Interaction Class	**Attribute/ Parameter**	**Count**	**Size**	**Update?** (y or n)	**Update Rate** # updates/unit time	**Update Grouping** (Assign same letter to attributes which will all be updated at the same time)	**Transport** R= Reliable B= Best Effort	**Ordering** TSO or FIFO	**Subscribe?** (y or n)	**Maximum tolerable latency from any source** (milliseconds)	**Attribute Ownership Transfer Rate** # times/unit time	**Ownership Transfer Grouping** (Assign same letter to attributes which will all be transferred together)
Entity												
	Entity_ID_site		2 bytes	n			R	FIFO	y			
	Entity_ID_application		2 bytes	n			R	FIFO	y			
	Entity_ID_entity		2 bytes	n			R	FIFO	y			
	Force_ID		2 bytes	n			R	FIFO	y			
	Entity_Type_Kind		2 bytes	n			R	FIFO	y			
	Entity_Type_Domain		2 bytes	n			R	FIFO	y			
	Entity_Type_Country		2 bytes	n			R	FIFO	y			
	Entity_Type_Category		2 bytes	n			R	FIFO	y			
	Entity_Type_Subcategory		2 bytes	n			R	FIFO	y			
	Entity_Type_Specific		2 bytes	n			R	FIFO	y			
	Entity_Type_Extra		2 bytes	n			R	FIFO	y			
	Orientation_Psi		8 bytes	y		A	R	FIFO	y			
	Orientation_Theta		8 bytes	y		A	R	FIFO	y			
	Orientation_Phi		8 bytes	y		A	R	FIFO	y			
	Location_X		8 bytes	y		A	R	FIFO	y			
	Location_Y		8 bytes	y		A	R	FIFO	y			
	Location_Z		8 bytes	y		A	R	FIFO	y			
	Velocity_X		8 bytes	y		A	R	FIFO	y			
	Velocity_Y		8 bytes	y		A	R	FIFO	y			
	Velocity_Z		8 bytes	y		A	R	FIFO	y			
	marking_text			y		A	R	FIFO	y			
	Appearance		4 bytes	y		A	R	FIFO	y			
	Acceleration_X		8 bytes	y		A	R	FIFO	y			
	Acceleration_Y		8 bytes	y		A	R	FIFO	y			
	Acceleration_Z		8 bytes	y		A	R	FIFO	y			
	Angular_Velocity_Psi		8 bytes	y		A	R	FIFO	y			
	Angular_Velocity_Theta		8 bytes	y		A	R	FIFO	y			
	Angular_Velocity_Phi		8 bytes	y		A	R	FIFO	y			
Fire												
	Launch_Platform_ID_site		2 bytes			B	R	FIFO	y			
	Launch_Platform_ID_application		2 bytes			B	R	FIFO	y			
	Launch_Platform_ID_entity		2 bytes			B	R	FIFO	y			
	Target_ID_site		2 bytes			B	R	FIFO	y			
	Target_ID_application		2 bytes			B	R	FIFO	y			
	Target_ID_entity		2 bytes			B	R	FIFO	y			

	Weapon_ID_site		2 bytes			B	R	FIFO	y				
	Weapon_ID_application		2 bytes			B	R	FIFO	y				
	Weapon_ID_entity		2 bytes			B	R	FIFO	y				
	Event_ID_site		2 bytes			B	R	FIFO	y				
	Event_ID_application		2 bytes			B	R	FIFO	y				
	Event_ID_entity		2 bytes			B	R	FIFO	y				
	LocationX		8 bytes			B	R	FIFO	y				
	LocationY		8 bytes			B	R	FIFO	y				
	LocationZ		8 bytes			B	R	FIFO	y				
	VelocityX		8 bytes			B	R	FIFO	y				
	VelocityY		8 bytes			B	R	FIFO	y				
	VelocityZ		8 bytes			B	R	FIFO	y				
	Range		8 bytes			B	R	FIFO	y				
	BurstKind		2 bytes			B	R	FIFO	y				
	BurstDomain		2 bytes			B	R	FIFO	y				
	BurstCountry		2 bytes			B	R	FIFO	y				
	BurstCategory		2 bytes			B	R	FIFO	y				
	BurstSubcategory		2 bytes			B	R	FIFO	y				
	BurstSpecific		2 bytes			B	R	FIFO	y				
	BurstExtra		2 bytes			B	R	FIFO	y				
	BurstWarhead		2 bytes			B	R	FIFO	y				
	BurstFuze		2 bytes			B	R	FIFO	y				
	BurstQuantity		2 bytes			B	R	FIFO	y				
	BurstRate		2 bytes			B	R	FIFO	y				
Detonation													
	Munition_ID_site		2 bytes			C	R	FIFO	y				
	Munition_ID_application		2 bytes			C	R	FIFO	y				
	Munition_ID_entity		2 bytes			C	R	FIFO	y				
	Target_ID_site		2 bytes			C	R	FIFO	y				
	Target_ID_application		2 bytes			C	R	FIFO	y				
	Target_ID_entity		2 bytes			C	R	FIFO	y				
	Event_ID_site		2 bytes			C	R	FIFO	y				
	Event_ID_application		2 bytes			C	R	FIFO	y				
	Event_ID_entity		2 bytes			C	R	FIFO	y				
	WorldLocationX		8 bytes			C	R	FIFO	y				
	WorldLocationY		8 bytes			C	R	FIFO	y				
	WorldLocationZ		8 bytes			C	R	FIFO	y				
	EntityLocationX		8 bytes			C	R	FIFO	y				
	EntityLocationY		8 bytes			C	R	FIFO	y				
	EntityLocationZ		8 bytes			C	R	FIFO	y				
	VelocityX		8 bytes			C	R	FIFO	y				
	VelocityY		8 bytes			C	R	FIFO	y				
	VelocityZ		8 bytes			C	R	FIFO	y				
	BurstKind		2 bytes			C	R	FIFO	y				
	BurstDomain		2 bytes			C	R	FIFO	y				
	BurstCountry		2 bytes			C	R	FIFO	y				
	BurstCategory		2 bytes			C	R	FIFO	y				
	BurstSubcategory		2 bytes			C	R	FIFO	y				
	BurstSpecific		2 bytes			C	R	FIFO	y				
	BurstExtra		2 bytes			C	R	FIFO	y				
	BurstWarhead		2 bytes			C	R	FIFO	y				
	BurstFuze		2 bytes			C	R	FIFO	y				
	BurstQuantity		2 bytes			C	R	FIFO	y				

	BurstRate		2 bytes			C	R	FIFO	y			
	DetonationResult		2 bytes			C	R	FIFO	y			

Object/Interaction Table

NOTE: Complete one of these tables for each Federate

Federate #
JPSD Modsaf

					If Update =" y"					If Subscribe = y?	Ownership	
Object/ Interaction Class	**Attribute/ Parameter**	**Count**	**Size**	**Update?**	**Update Rate**	**Update Grouping**	**Transport**	**Ordering**	**Subscribe?**	**Maximum tolerable latency from any source**	**Attribute Ownership Transfer Rate**	**Ownership Transfer Grouping**
				(y or n)	**# updates/unit time**	(Assign same letter to attributes which will all be updated at the same time)	**R= Reliable B= Best Effort**	**TSO or FIFO**	**(y or n)**	**(milliseconds)**	**# times/unit time**	(Assign same letter to attributes which will all be transferred together)
Entity												
	Entity_ID_site		2 bytes	n			R	FIFO	y			
	Entity_ID_application		2 bytes	n			R	FIFO	y			
	Entity_ID_entity		2 bytes	n			R	FIFO	y			
	Force_ID		2 bytes	n			R	FIFO	y			
	Entity_Type_Kind		2 bytes	n			R	FIFO	y			
	Entity_Type_Domain		2 bytes	n			R	FIFO	y			
	Entity_Type_Country		2 bytes	n			R	FIFO	y			
	Entity_Type_Category		2 bytes	n			R	FIFO	y			
	Entity_Type_Subcategory		2 bytes	n			R	FIFO	y			
	Entity_Type_Specific		2 bytes	n			R	FIFO	y			
	Entity_Type_Extra		2 bytes	n			R	FIFO	y			
	Orientation_Psi		8 bytes	y		A	R	FIFO	y			
	Orientation_Theta		8 bytes	y		A	R	FIFO	y			
	Orientation_Phi		8 bytes	y		A	R	FIFO	y			
	Location_X		8 bytes	y		A	R	FIFO	y			
	Location_Y		8 bytes	y		A	R	FIFO	y			
	Location_Z		8 bytes	y		A	R	FIFO	y			
	Velocity_X		8 bytes	y		A	R	FIFO	y			
	Velocity_Y		8 bytes	y		A	R	FIFO	y			
	Velocity_Z		8 bytes	y		A	R	FIFO	y			
	marking_text			y		A	R	FIFO	y			
	Appearance		4 bytes	y		A	R	FIFO	y			
	Acceleration_X		8 bytes	y		A	R	FIFO	y			
	Acceleration_Y		8 bytes	y		A	R	FIFO	y			
	Acceleration_Z		8 bytes	y		A	R	FIFO	y			
	Angular_Velocity_Psi		8 bytes	y		A	R	FIFO	y			
	Angular_Velocity_Theta		8 bytes	y		A	R	FIFO	y			
	Angular_Velocity_Phi		8 bytes	y		A	R	FIFO	y			
Fire												
	Launch_Platform_ID_site		2 bytes			B	R	FIFO	y			
	Launch_Platform_ID_application		2 bytes			B	R	FIFO	y			
	Launch_Platform_ID_entity		2 bytes			B	R	FIFO	y			
	Target_ID_site		2 bytes			B	R	FIFO	y			
	Target_ID_application		2 bytes			B	R	FIFO	y			
	Target_ID_entity		2 bytes			B	R	FIFO	y			

	Weapon_ID_site		2 bytes			B	R	FIFO	y				
	Weapon_ID_application		2 bytes			B	R	FIFO	y				
	Weapon_ID_entity		2 bytes			B	R	FIFO	y				
	Event_ID_site		2 bytes			B	R	FIFO	y				
	Event_ID_application		2 bytes			B	R	FIFO	y				
	Event_ID_entity		2 bytes			B	R	FIFO	y				
	LocationX		8 bytes			B	R	FIFO	y				
	LocationY		8 bytes			B	R	FIFO	y				
	LocationZ		8 bytes			B	R	FIFO	y				
	VelocityX		8 bytes			B	R	FIFO	y				
	VelocityY		8 bytes			B	R	FIFO	y				
	VelocityZ		8 bytes			B	R	FIFO	y				
	Range		8 bytes			B	R	FIFO	y				
	BurstKind		2 bytes			B	R	FIFO	y				
	BurstDomain		2 bytes			B	R	FIFO	y				
	BurstCountry		2 bytes			B	R	FIFO	y				
	BurstCategory		2 bytes			B	R	FIFO	y				
	BurstSubcategory		2 bytes			B	R	FIFO	y				
	BurstSpecific		2 bytes			B	R	FIFO	y				
	BurstExtra		2 bytes			B	R	FIFO	y				
	BurstWarhead		2 bytes			B	R	FIFO	y				
	BurstFuze		2 bytes			B	R	FIFO	y				
	BurstQuantity		2 bytes			B	R	FIFO	y				
	BurstRate		2 bytes			B	R	FIFO	y				
Detonation													
	Munition_ID_site		2 bytes			C	R	FIFO	y				
	Munition_ID_application		2 bytes			C	R	FIFO	y				
	Munition_ID_entity		2 bytes			C	R	FIFO	y				
	Target_ID_site		2 bytes			C	R	FIFO	y				
	Target_ID_application		2 bytes			C	R	FIFO	y				
	Target_ID_entity		2 bytes			C	R	FIFO	y				
	Event_ID_site		2 bytes			C	R	FIFO	y				
	Event_ID_application		2 bytes			C	R	FIFO	y				
	Event_ID_entity		2 bytes			C	R	FIFO	y				
	WorldLocationX		8 bytes			C	R	FIFO	y				
	WorldLocationY		8 bytes			C	R	FIFO	y				
	WorldLocationZ		8 bytes			C	R	FIFO	y				
	EntityLocationX		8 bytes			C	R	FIFO	y				
	EntityLocationY		8 bytes			C	R	FIFO	y				
	EntityLocationZ		8 bytes			C	R	FIFO	y				
	VelocityX		8 bytes			C	R	FIFO	y				
	VelocityY		8 bytes			C	R	FIFO	y				
	VelocityZ		8 bytes			C	R	FIFO	y				
	BurstKind		2 bytes			C	R	FIFO	y				
	BurstDomain		2 bytes			C	R	FIFO	y				
	BurstCountry		2 bytes			C	R	FIFO	y				
	BurstCategory		2 bytes			C	R	FIFO	y				
	BurstSubcategory		2 bytes			C	R	FIFO	y				
	BurstSpecific		2 bytes			C	R	FIFO	y				
	BurstExtra		2 bytes			C	R	FIFO	y				
	BurstWarhead		2 bytes			C	R	FIFO	y				
	BurstFuze		2 bytes			C	R	FIFO	y				
	BurstQuantity		2 bytes			C	R	FIFO	y				

	BurstRate		2 bytes			C	R	FIFO	y			
	DetonationResult		2 bytes			C	R	FIFO	y			

Class1	Class2	Class3	Class4	Class5	Class6
Entity (PS)	Platform (PS)	Ground_Vehicle (PS)	Tracked (PS)	Tank (PS)	M1 (PS)
					T72 (PS)
				Armored_Fighting_Vehicle (PS)	M2 (PS)
					BMP (PS)
		Air_Vehicle (PS)	Attack (PS)	FA18 (PS)	
		Human (PS)	INFANTRY (PS)		
	Munition (PS)	Guided (PS)	Anti_Ground (PS)	TOW (PS)	
				AT5 (PS)	
				LGB (PS)	
			Anti_Air (PS)	SA16 (PS)	

Term	Definition
Entity	The class representing the highest level. All the simulation entities are under this class.
Platform	Class defining platform
Munition	Class defining munition
INFANTRY	Class defining infantry warfighter
Ground_Vehicle	Class defining ground vehicle of a platform class.
Air_Vehicle	Class defining air vehicle of the platform class
Human	Class defining human warfighter.
Tracked	Class defining tracked ground vehicle
Attack	Class defining attack air vehicle
Tank	Class defining tank
Armored_Fighting_Vehicle	Class defining armored fighting vehicle
M1	Class defining M1 tank
T72	class defining T72 tank
M2	class defining M2 AFV
BMP	class defining BMP AFV
FA18	class defining FA18 attack air vehicle
Guided	class defining guided munition
Anti_Ground	class defining anti ground guided munition
Anti_Air	class defining anti-air guided munition
SA16	Class defining SA16 missile
TOW	class defining a TOW missile
AT5	class defining AT5 missile
LGB	class defining LG8 missile

Interaction Structure	Initiating Object		Receiving Object/Area		Interaction Parameters	Init/ Sense/ React
	Class	Affected Attributes	Class	Affected Attributes		
DETONATION	Entity	None	None	None	Munition_ID	IR
					Target_ID	
					Firing_ID	
					Location_X	
					Location_Y	
					Location_Z	
					Relative_Location_X	
					Relative_Location_Y	
					Relative_Location_Z	
					Velocity_X	
					Velocity_Y	
					Velocity_Z	
					Burst_Descriptor_Warhead	
					Burst_Descriptor_Fuze	
					Burst_Descriptor_Detonation_Result	
					Quantity	
					Rate	
WEAPON_FIRE	Entity	None	None	None	Firing_ID	IR
					Target_ID	
					Location_X	
					Location_Y	
					Location_Z	
					Velocity_X	
					Velocity_Y	
					Velocity_Z	
					Munition_Type	
					Quantity	
					Rate	
WEAPON_LAUNCH	Entity	None	None	None	Launch_Platform_ID	IR
					Munition_ID	
					Target_ID	
					Location_X	

Interaction Structure		Initiating Object		Receiving Object/Area		Interaction Parameters	Init/ Sense/ React
		Class	Affected Attributes	Class	Affected Attributes		
						Location_Y	
						Location_Z	
						Velocity_X	
						Velocity_Y	
						Velocity_Z	
COLLISION		Entity	None	None	None	Issuing_ID	IR
						Colliding_ID	
						Mass	
						Relative_Location_X	
						Relative_Location_Y	
						Relative_Location_Z	
						Velocity_X	
						Velocity_Y	
						Velocity_Z	

Term	Definition
DETONATION	Defines a detonation interaction
WEAPON_FIRE	Defines a weapon fire interaction.
WEAPON_LAUNCH	Defines a weapon launch interacion.
COLLISION	Defines a collition interaction.

Object/Interaction	Attribute/Parameter	Datatype	Cardinality	Units	Resolution	Accuracy	Accuracy Condition	Update Type
Entity	FORCE_ID	short	1	enumeration	n/a	perfect	n/a	Static
	ENTITYID_SITE	short	1		n/a	perfect	n/a	Static
	ENTITYID_APPLICATION	short	1		n/a	perfect	n/a	Static
	ENTITYID_ENTITY	short	1		n/a	perfect	n/a	Static
	LOCATION_X	double	1	meters		n/a	DR	Conditional
	LOCATION_Y	double	1	meters		n/a	DR	Conditional
	LOCATION_Z	double	1	meters		n/a	DR	Conditional
	VELOCITY_X	float	1	meters/sec		n/a	DR	Conditional
	VELOCITY_Y	float	1	meters/sec		n/a	DR	Conditional
	VELOCITY_Z	float	1	meters/sec		n/a	DR	Conditional
	ACCELERATION_X	float	1	meters/sec^2		n/a	DR	Conditional
	ACCELERATION_Y	float	1	meters/sec^2		n/a	DR	Conditional
	ACCELERATION_Z	float	1	meters/sec^2		n/a	DR	Conditional
	ORIENTATION_PSI	float	1	radians		n/a	DR	Conditional
	ORIENTATION_THETA	float	1	radians		n/a	DR	Conditional
	ORIENTATION_PHI	float	1	radians		n/a	DR	Conditional
	APPEARANCE_PAINT_SCHEME	short	1	enumeration	n/a	perfect	n/a	Static
Platform	DAMAGE_STATE_APPEARANCE	short	1	enumeration	n/a	perfect	n/a	Conditional
Munition	FIRING_ID	long	1		n/a	perfect	n/a	Static
Ground_Vehicle	APPEARANCE_SMOKE	short	1	enumeration	n/a	n/a	n/a	Conditional
	APPEARANCE_TRAILING	short	1	enumeration	n/a	n/a	n/a	Conditional
	APPEARANCE_HATCH	short	1	enumeration	n/a	n/a	n/a	Conditional
	APPEARANCE_LIGHTS	short	1	enumeration	n/a	n/a	n/a	Conditional
	APPEARACNE_FLAMING	short	1	enumeration	n/a	n/a	n/a	Conditional
	DAMAGE_STATE_MOBILITY	short	1	enumeration	n/a	n/a	n/a	Conditional
	DAMAGE_STATE_FIRE_POWER	short	1	enumeration	n/a	n/a	n/a	Conditional
Armored_Fighting_Vehi	APPEARANCE_LAUNCHER	short	1	enumeration	n/a	n/a	n/a	Conditional
DETONATION	Munition_ID	long	1			perfect	n/a	N/A
	Target_ID	long	1			perfect	n/a	N/A
	Firing_ID	long	1			perfect	n/a	N/A
	Location_X	double	1	meters		n/a	n/a	N/A
	Location_Y	double	1	meters		n/a	n/a	N/A
	Location_Z	double	1	meters		n/a	n/a	N/A

Update Condition	Transferable/ Acceptable	Updateable/ Reflectable
	N	UR
	N	UR
	N	UR
	N	UR
DR	N	UR
DR	N	UR
DR	N	UR
DR	N	UR
DR	N	UR
DR	N	UR
DR	N	UR
DR	N	UR
DR	N	UR
DR	N	UR
DR	N	UR
DR	N	UR
	N	UR
	N	UR
	N	UR
	N	UR
	N	UR
	N	UR
	N	UR
	N	UR
	N	UR
	N	UR
	N	UR
	N	UR
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A

Object/Interaction	Attribute/Parameter	Datatype	Cardinality	Units	Resolution	Accuracy	Accuracy Condition	Update Type
	Relative_Location_X	double	1	meters		n/a	n/a	N/A
	Relative_Location_Y	double	1	meters		n/a	n/a	N/A
	Relative_Location_Z	double	1	meters		n/a	n/a	N/A
	Velocity_X	float	1	meters/sec		n/a	n/a	N/A
	Velocity_Y	float	1	meters/sec		n/a	n/a	N/A
	Velocity_Z	float	1	meters/sec		n/a	n/a	N/A
	Burst_Descriptor_Warhead	short	1	enumeration		n/a	n/a	N/A
	Burst_Descriptor_Fuze	short	1	enumeration		n/a	n/a	N/A
	Burst_Descriptor_Detonation_Result	short	1	enumeration		n/a	n/a	N/A
	Quantity	short	1			n/a	n/a	N/A
	Rate	float	1			n/a	n/a	N/A
WEAPON_FIRE	Firing_ID	long	1			n/a	n/a	N/A
	Target_ID	long	1			n/a	n/a	N/A
	Location_X	double	1	meters		n/a	n/a	N/A
	Location_Y	double	1	meters		n/a	n/a	N/A
	Location_Z	double	1	meters		n/a	n/a	N/A
	Velocity_X	float	1	meters/sec		n/a	n/a	N/A
	Velocity_Y	float	1	meters/sec		n/a	n/a	N/A
	Velocity_Z	float	1	meters/sec		n/a	n/a	N/A
	Munition_Type	string	1			n/a	n/a	N/A
	Quantity	short	1			n/a	n/a	N/A
	Rate	float	1			n/a	n/a	N/A
WEAPON_LAUNCH	Launch_Platform_ID	long	1			perfect	n/a	N/A
	Munitiion_ID	long	1			perfect	n/a	N/A
	Target_ID	long	1			perfect	n/a	N/A
	Location_X	double	1	meters		n/a	n/a	N/A
	Location_Y	double	1	meters		n/a	n/a	N/A
	Location_Z	double	1	meters		n/a	n/a	N/A
	Velocity_X	float	1	meters/sec		n/a	n/a	N/A
	Velocity_Y	float	1	meters/sec		n/a	n/a	N/A
	Velocity_Z	float	1	meters/sec		n/a	n/a	N/A
COLLISION	Issuing_ID	long	1			perfect	n/a	N/A
	Colliding_ID	long	1			perfect	n/a	N/A

Update Condition	Transferable/ Acceptable	Updateable/ Reflectable
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A

Object/Interaction	Attribute/Parameter	Datatype	Cardinality	Units	Resolution	Accuracy	Accuracy Condition	Update Type
	Mass	float	1	kg		n/a	n/a	N/A
	Relative_Location_X	double	1	meters		n/a	n/a	N/A
	Relative_Location_Y	double	1	meters		n/a	n/a	N/A
	Relative_Location_Z	double	1	meters		n/a	n/a	N/A
	Velocity_X	float	1	meters/sec		n/a	n/a	N/A
	Velocity_Y	float	1	meters/sec		n/a	n/a	N/A
	Velocity_Z	float	1	meters/sec		n/a	n/a	N/A

Update Condition	Transferable/ Acceptable	Updateable/ Reflectable
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A
N/A	N/A	N/A

Class/Interaction	Term	Definition
Entity	FORCE_ID	The ID of the force of the entity class
	ENTITYID_SITE	The site id of the entity class
	ENTITYID_APPLICATION	The application id of the entity class
	ENTITYID_ENTITY	The entity part of the entity id of the entity class
	LOCATION_X	The x coordinate of the location of the entity.
	LOCATION_Y	This defines the y coordinate of the entity location
	LOCATION_Z	This attribute defines the z coordinate of the location of the entity class
	VELOCITY_X	defines the x coordinate of the velocity of the entity class.
	VELOCITY_Y	Defines the y coordinate of the velocity of the entity class
	VELOCITY_Z	Defines the z coordinate of the velocity of the entity class.
	ACCELERATION_X	Defines the x coordinate of the acceleration of the entity class.
	ACCELERATION_Y	Defines the y coordiante of the acceleration of the entity class.
	ACCELERATION_Z	Defines the z coordinate of the acceleration of the entity class,
	ORIENTATION_PSI	Defines the psi angle of the orientation of the entity class.
	ORIENTATION_THETA	Defines the theta angle of the orientation of the entity class.
	ORIENTATION_PHI	Defines the phi angle of the orientation of the entity class.
	APPEARANCE_PAINT_SCHEME	Defines the paint scheme of the entity class.
Platform	DAMAGE_STATE_APPEARANCE	Defines the damange state of the platform class.
Munition	FIRING_ID	Defines the entityid that fires this munition class.
Ground_Vehicle	APPEARANCE_SMOKE	Defines the smoke appearance of the ground vehicle class.
	APPEARANCE_TRAILING	Defines the trailing appearance of the ground vehicle class.
	APPEARANCE_HATCH	Defines the hatch appearance of the ground vehicle class.
	APPEARANCE_LIGHTS	Defines the lights appearance of the ground vehicle class.
	APPEARACNE_FLAMING	Define the flaming appearance of the ground vehicle class.
	DAMAGE_STATE_MOBILITY	Defines the mobility state of the ground vehicle class.
	DAMAGE_STATE_FIRE_POWER	Defines the fire power state of the ground vehicle class.
Armored_Fighting_Vehicle	APPEARANCE_LAUNCHER	Defines the launcher appearance of the Armored_Fighting_Vehicle clas
DETONATION	Munition_ID	Defines the id of the munition
	Target_ID	Defines the id of the target
	Firing_ID	defines the id of the firing entity.
	Location_X	defines the x coordiante of the detonation location
	Location_Y	defines the y coordiante of the detonation location.
	Location_Z	Defines the z coordiante of the detonation location.
	Relative_Location_X	Defines the x coordinate of the detonation location relative to the center

Class/Interaction	Term	Definition
	Relative_Location_Y	Defines the y coordinate of the detonation location relative to the center
	Relative_Location_Z	Defines the z coordinate of the detonation that is relative to the center o
	Velocity_X	Defines the impacting velocity x
	Velocity_Y	Defines the impacting velocity y
	Velocity_Z	defines impacting velocity z
	Burst_Descriptor_Warhead	Defines the warhead
	Burst_Descriptor_Fuze	Defines the fuze
	Burst_Descriptor_Detonation_Result	Defines the result of the detonation
	Quantity	Defines the quantity of the warhead.
	Rate	Defines the rate of the munition
WEAPON_FIRE	Firing_ID	Defines the id of the firing entity
	Target_ID	defines the id of the target
	Location_X	Defines the x coordinate of the location where it is firing at.
	Location_Y	Defines the y coordinate of the location where it is firing at.
	Location_Z	Defines the z coordinate of the location where it is firing at.
	Velocity_X	Defines the x velocity of the munition when it leaves the weapon.
	Velocity_Y	Defines the y velocity of the munition when it leaves the weapon.
	Velocity_Z	Defines the z velocity of the munition when it leaves the weapon.
	Munition_Type	Defines the type of munition
	Quantity	Defines the number of the munition.
	Rate	Defines the rate of the munition.
WEAPON_LAUNCH	Launch_Platform_ID	Defines the id of the launch platorm.
	Munitiion_ID	Defines the id of the munition
	Target_ID	Defines the id of the target.
	Location_X	Defines the x coordinate of the location where the weapon is targeted.
	Location_Y	Defines the y coordinate of the location the wepaon is targeting at.
	Location_Z	Defines the z coordinate of the location where the weapon is targeting a
	Velocity_X	Defines the missile x velocity when it leaves the launcher.
	Velocity_Y	Defines the missile y velocity when it leaves the launcher.
	Velocity_Z	Defines the missile z velocity when the missile leaves the launcher.
COLLISION	Issuing_ID	Defines the id of the entity that issues this interaction.
	Colliding_ID	Defines the id which the entity has colided.
	Mass	Defines the mass of the entity which issues this interaction.
	Relative_Location_X	Defines the x coordinate of the relative location of the center of the issui

Class/Interaction	Term	Definition
	Relative_Location_Y	Defines the y coordinate of the location relative to the center of the issui
	Relative_Location_Z	Defines the z coordiante of location relative to the center of the issuing
	Velocity_X	Defines the x velocity of the issuing entity.
	Velocity_Y	Defines the y location of the issuing entity.
	Velocity_Z	Defines the z velocity of the enitty.

Complex Datatype	Field Name	Datatype	Cardinality	Units	Resolution	Accuracy	Accuracy Condition
idlBurstDescriptor	idlWarhead	any	1	enumeration		perfect	always
	idlFuse	any	1	enumeration		perfect	always
	idlDetonationResult	any	1	enumeration		perfect	always

Class1
Entity (PS)

Term	Definition
Entity	

Interaction Structure	Initiating Object		Receiving Object/Area		Interaction Parameters	Init/ Sense/ React
	Class	Affected Attributes	Class	Affected Attributes		
Detonation	Entity	None	None	None	Munition_ID_site	IR
					Munition_ID_application	
					Munition_ID_entity	
					Attacker_ID_site	
					Attacker_ID_application	
					Attacker_ID_entity	
					Target_ID_site	
					Target_ID_application	
					Target_ID_entity	
					Event_ID_site	
					Event_ID_application	
					Event_ID_entity	
					WorldLocationX	
					WorldLocationY	
					WorldLocationZ	
					EntityLocationX	
					EntityLocationY	
					EntityLocationZ	
					VelocityX	
					VelocityY	
					VelocityZ	
					BurstKind	
					BurstDomain	
					BurstCountry	
					BurstCategory	
					BurstSubcategory	
					BurstSpecific	
					BurstExtra	
					BurstWarhead	
					BurstFuze	
					BurstQuantity	
					BurstRate	
					DetonationResult	
					TimeStamp	
					ArticulatedStruct	
Fire	Entity	None	None	None	Launch_Platform_ID_site	IR
					Launch_Platform_ID_application	
					Launch_Platform_ID_entity	
					Target_ID_site	
					Target_ID_application	
					Target_ID_entity	
					Weapon_ID_site	
					Weapon_ID_application	
					Weapon_ID_entity	
					Event_ID_site	
					Event_ID_application	
					Event_ID_entity	
					LocationX	
					LocationY	

Interaction Structure		Initiating Object		Receiving Object/Area		Interaction Parameters	Init/ Sense/ React
		Class	Affected Attributes	Class	Affected Attributes		
						LocationZ	
						VelocityX	
						VelocityY	
						VelocityZ	
						Range	
						TimeStamp	
						BurstKind	
						BurstDomain	
						BurstCountry	
						BurstCategory	
						BurstSubcategory	
						BurstSpecific	
						BurstExtra	
						BurstWarhead	
						BurstFuze	
						BurstQuantity	
						BurstRate	
Signal		Entity	None	None	None	TimeStamp	IR
						EntitySite	
						EntityHost	
						EntityId	
						RadioId	
						EncodingScheme	
						TdlType	
						SampleRate	
						Samples	
						DataStruct	
Collision		Entity	None	None	None	TimeStamp	IR
						IssuingEntitySite	
						IssuingEntityHost	
						IssuingEntityId	
						CollidingEntitySite	
						CollidingEntityHost	
						CollidingEntityId	
						EventIdSite	
						EventIdHost	
						EventIdEvent	
						VelocityX	
						VelocityY	
						VelocityZ	
						Mass	
						RelLocationX	
						RelLocationY	
						RelLocationZ	

Term	Definition
Detonation	
Fire	
Signal	
Collision	

Object/Interaction	Attribute/Parameter	Datatype	Cardinality	Units	Resolution	Accuracy	Accuracy Condition	Update Type	Update Condition	Transferable/ Acceptable
Entity	Entity_ID_site	short	1		n/a	perfect	always	Conditional		N
	Entity_ID_application	short	1		n/a	perfect	always	Conditional		N
	Entity_ID_entity	short	1		n/a	perfect	always	Conditional		N
	Force_ID	short	1	enumeration	n/a	perfect	always	Conditional		N
	Orientation_Psi	float	1	radians		perfect	always	Conditional		N
	Orientation_Theta	float	1	radians		perfect	always	Conditional		N
	Orientation_Phi	float	1	radians		perfect	always	Conditional		N
	Location_X	double	1			perfect	always	Conditional		N
	Location_Y	double	1			perfect	always	Conditional		N
	Location_Z	double	1			perfect	always	Conditional		N
	Velocity_X	float	1			perfect	always	Conditional		N
	Velocity_Y	float	1			perfect	always	Conditional		N
	Velocity_Z	float	1			perfect	always	Conditional		N
	Acceleration_X	float	1			perfect	always	Conditional		N
	Acceleration_Y	float	1			perfect	always	Conditional		N
	Acceleration_Z	float	1			perfect	always	Conditional		N
	Angular_Velocity_Psi	any	1			perfect	always	Conditional		N
	Angular_Velocity_Theta	any	1			perfect	always	Conditional		N
	Angular_Velocity_Phi	any	1			perfect	always	Conditional		N
	Entity_Type_Kind	any	1			perfect	always	Conditional		N
	Entity_Type_Domain	any	1			perfect	always	Conditional		N
	Entity_Type_Country	any	1			perfect	always	Conditional		N
	Entity_Type_Category	any	1			perfect	always	Conditional		N
	Entity_Type_Subcategory	any	1			perfect	always	Conditional		N
	Entity_Type_Specific	any	1			perfect	always	Conditional		N
	Entity_Type_Extra	any	1			perfect	always	Conditional		N
	Entity_Guise_Kind	any	1			perfect	always	Conditional		N
	Entity_Guise_Domain	any	1			perfect	always	Conditional		N
	Entity_Guise_Country	any	1			perfect	always	Conditional		N
	Entity_Guise_Category	any	1			perfect	always	Conditional		N
	Entity_Guise_Scategory	any	1			perfect	always	Conditional		N
	Entity_Guise_Specific	any	1			perfect	always	Conditional		N
	Entity_Guise_Extra	any	1			perfect	always	Conditional		N

Updateable/ Reflectable
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR
UR

Object/Interaction	Attribute/Parameter	Datatype	Cardinality	Units	Resolution	Accuracy	Accuracy Condition	Update Type	Update Condition	Transferable/ Acceptable
	marking_charset	any	1			perfect	always	Conditional		N
	marking_text	any	1			perfect	always	Conditional		N
	Appearance	any	1			perfect	always	Conditional		N
	Dead_Reckoning_Algorithm	any	1			perfect	always	Conditional		N
	TimeStamp	any	1			perfect	always	Conditional		N
	MarkingCharSet	any	1			perfect	always	Conditional		N
	Capabilities	any	1			perfect	always	Conditional		N
	ArticulatedStruct	any	1			perfect	always	Conditional		N
Detonation	Munition_ID_site	any	1			perfect	always	N/A	N/A	N/A
	Munition_ID_application	any	1			perfect	always	N/A	N/A	N/A
	Munition_ID_entity	any	1			perfect	always	N/A	N/A	N/A
	Attacker_ID_site	any	1			perfect	always	N/A	N/A	N/A
	Attacker_ID_application	any	1			perfect	always	N/A	N/A	N/A
	Attacker_ID_entity	any	1			perfect	always	N/A	N/A	N/A
	Target_ID_site	any	1			perfect	always	N/A	N/A	N/A
	Target_ID_application	any	1			perfect	always	N/A	N/A	N/A
	Target_ID_entity	any	1			perfect	always	N/A	N/A	N/A
	Event_ID_site	any	1			perfect	always	N/A	N/A	N/A
	Event_ID_application	any	1			perfect	always	N/A	N/A	N/A
	Event_ID_entity	any	1			perfect	always	N/A	N/A	N/A
	WorldLocationX	double	1			perfect	always	N/A	N/A	N/A
	WorldLocationY	double	1			perfect	always	N/A	N/A	N/A
	WorldLocationZ	double	1			perfect	always	N/A	N/A	N/A
	EntityLocationX	double	1			perfect	always	N/A	N/A	N/A
	EntityLocationY	double	1			perfect	always	N/A	N/A	N/A
	EntityLocationZ	double	1			perfect	always	N/A	N/A	N/A
	VelocityX	float	1			perfect	always	N/A	N/A	N/A
	VelocityY	float	1			perfect	always	N/A	N/A	N/A
	VelocityZ	float	1			perfect	always	N/A	N/A	N/A
	BurstKind	any	1			perfect	always	N/A	N/A	N/A
	BurstDomain	any	1			perfect	always	N/A	N/A	N/A
	BurstCountry	any	1			perfect	always	N/A	N/A	N/A
	BurstCategory	any	1			perfect	always	N/A	N/A	N/A

Updateable/ Reflectable
UR
UR
UR
UR
UR
UR
UR
UR
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A

Object/Interaction	Attribute/Parameter	Datatype	Cardinality	Units	Resolution	Accuracy	Accuracy Condition	Update Type	Update Condition	Transferable/ Acceptable
	BurstSubcategory	any	1			perfect	always	N/A	N/A	N/A
	BurstSpecific	any	1			perfect	always	N/A	N/A	N/A
	BurstExtra	any	1			perfect	always	N/A	N/A	N/A
	BurstWarhead	short	1			perfect	always	N/A	N/A	N/A
	BurstFuze	short	1			perfect	always	N/A	N/A	N/A
	BurstQuantity	short	1			perfect	always	N/A	N/A	N/A
	BurstRate	float	1			perfect	always	N/A	N/A	N/A
	DetonationResult	short	1			perfect	always	N/A	N/A	N/A
	TimeStamp	any	1			perfect	always	N/A	N/A	N/A
	ArticulatedStruct	any	1			perfect	always	N/A	N/A	N/A
Fire	Launch_Platform_ID_site	any	1			perfect	always	N/A	N/A	N/A
	Launch_Platform_ID_applic	any	1			perfect	always	N/A	N/A	N/A
	Launch_Platform_ID_entity	any	1			perfect	always	N/A	N/A	N/A
	Target_ID_site	any	1			perfect	always	N/A	N/A	N/A
	Target_ID_application	any	1			perfect	always	N/A	N/A	N/A
	Target_ID_entity	any	1			perfect	always	N/A	N/A	N/A
	Weapon_ID_site	any	1			perfect	always	N/A	N/A	N/A
	Weapon_ID_application	any	1			perfect	always	N/A	N/A	N/A
	Weapon_ID_entity	any	1			perfect	always	N/A	N/A	N/A
	Event_ID_site	any	1			perfect	always	N/A	N/A	N/A
	Event_ID_application	any	1			perfect	always	N/A	N/A	N/A
	Event_ID_entity	any	1			perfect	always	N/A	N/A	N/A
	LocationX	double	1			perfect	always	N/A	N/A	N/A
	LocationY	double	1			perfect	always	N/A	N/A	N/A
	LocationZ	double	1			perfect	always	N/A	N/A	N/A
	VelocityX	float	1			perfect	always	N/A	N/A	N/A
	VelocityY	float	1			perfect	always	N/A	N/A	N/A
	VelocityZ	float	1			perfect	always	N/A	N/A	N/A
	Range	any	1			perfect	always	N/A	N/A	N/A
	TimeStamp	any	1			perfect	always	N/A	N/A	N/A
	BurstKind	any	1			perfect	always	N/A	N/A	N/A
	BurstDomain	any	1			perfect	always	N/A	N/A	N/A
	BurstCountry	any	1			perfect	always	N/A	N/A	N/A

Updateable/ Reflectable
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A

Object/Interaction	Attribute/Parameter	Datatype	Cardinality	Units	Resolution	Accuracy	Accuracy Condition	Update Type	Update Condition	Transferable/ Acceptable
	BurstCategory	any	1			perfect	always	N/A	N/A	N/A
	BurstSubcategory	any	1			perfect	always	N/A	N/A	N/A
	BurstSpecific	any	1			perfect	always	N/A	N/A	N/A
	BurstExtra	any	1			perfect	always	N/A	N/A	N/A
	BurstWarhead	any	1			perfect	always	N/A	N/A	N/A
	BurstFuze	any	1			perfect	always	N/A	N/A	N/A
	BurstQuantity	any	1			perfect	always	N/A	N/A	N/A
	BurstRate	any	1			perfect	always	N/A	N/A	N/A
Signal	TimeStamp	any	1			perfect	always	N/A	N/A	N/A
	EntitySite	any	1			perfect	always	N/A	N/A	N/A
	EntityHost	any	1			perfect	always	N/A	N/A	N/A
	EntityId	any	1			perfect	always	N/A	N/A	N/A
	RadioId	any	1			perfect	always	N/A	N/A	N/A
	EncodingScheme	any	1			perfect	always	N/A	N/A	N/A
	TdlType	any	1			perfect	always	N/A	N/A	N/A
	SampleRate	any	1			perfect	always	N/A	N/A	N/A
	Samples	any	1			perfect	always	N/A	N/A	N/A
	DataStruct	any	1			perfect	always	N/A	N/A	N/A
Collision	TimeStamp	any	1			perfect	always	N/A	N/A	N/A
	IssuingEntitySite	any	1			perfect	always	N/A	N/A	N/A
	IssuingEntityHost	any	1			perfect	always	N/A	N/A	N/A
	IssuingEntityId	any	1			perfect	always	N/A	N/A	N/A
	CollidingEntitySite	any	1			perfect	always	N/A	N/A	N/A
	CollidingEntityHost	any	1			perfect	always	N/A	N/A	N/A
	CollidingEntityId	any	1			perfect	always	N/A	N/A	N/A
	EventIdSite	any	1			perfect	always	N/A	N/A	N/A
	EventIdHost	any	1			perfect	always	N/A	N/A	N/A
	EventIdEvent	any	1			perfect	always	N/A	N/A	N/A
	VelocityX	any	1			perfect	always	N/A	N/A	N/A
	VelocityY	any	1			perfect	always	N/A	N/A	N/A
	VelocityZ	any	1			perfect	always	N/A	N/A	N/A
	Mass	any	1			perfect	always	N/A	N/A	N/A
	RelLocationX	any	1			perfect	always	N/A	N/A	N/A

Updateable/ Reflectable
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A
N/A

Object/Interaction	Attribute/Parameter	Datatype	Cardinality	Units	Resolution	Accuracy	Accuracy Condition	Update Type	Update Condition	Transferable/ Acceptable
	RelLocationY	any	1			perfect	always	N/A	N/A	N/A
	RelLocationZ	any	1			perfect	always	N/A	N/A	N/A

Updateable/ Reflectable
N/A
N/A

Class/Interaction	Term	Definition
Entity	Entity_ID_site	
	Entity_ID_application	
	Entity_ID_entity	
	Force_ID	
	Orientation_Psi	
	Orientation_Theta	
	Orientation_Phi	
	Location_X	
	Location_Y	
	Location_Z	
	Velocity_X	
	Velocity_Y	
	Velocity_Z	
	Acceleration_X	
	Acceleration_Y	
	Acceleration_Z	
	Angular_Velocity_Psi	
	Angular_Velocity_Theta	
	Angular_Velocity_Phi	
	Entity_Type_Kind	
	Entity_Type_Domain	
	Entity_Type_Country	
	Entity_Type_Category	
	Entity_Type_Subcategory	
	Entity_Type_Specific	
	Entity_Type_Extra	
	Entity_Guise_Kind	
	Entity_Guise_Domain	
	Entity_Guise_Country	
	Entity_Guise_Category	
	Entity_Guise_Scategory	
	Entity_Guise_Specific	
	Entity_Guise_Extra	
	marking_charset	

Class/Interaction	Term	Definition
	marking_text	
	Appearance	
	Dead_Reckoning_Algorithm	
	TimeStamp	
	MarkingCharSet	
	Capabilities	
	ArticulatedStruct	
Detonation	Munition_ID_site	
	Munition_ID_application	
	Munition_ID_entity	
	Attacker_ID_site	
	Attacker_ID_application	
	Attacker_ID_entity	
	Target_ID_site	
	Target_ID_application	
	Target_ID_entity	
	Event_ID_site	
	Event_ID_application	
	Event_ID_entity	
	WorldLocationX	
	WorldLocationY	
	WorldLocationZ	
	EntityLocationX	
	EntityLocationY	
	EntityLocationZ	
	VelocityX	
	VelocityY	
	VelocityZ	
	BurstKind	
	BurstDomain	
	BurstCountry	
	BurstCategory	
	BurstSubcategory	
	BurstSpecific	

Class/Interaction	Term	Definition
	BurstExtra	
	BurstWarhead	
	BurstFuze	
	BurstQuantity	
	BurstRate	
	DetonationResult	
	TimeStamp	
	ArticulatedStruct	
Fire	Launch_Platform_ID_site	
	Launch_Platform_ID_applic	
	Launch_Platform_ID_entity	
	Target_ID_site	
	Target_ID_application	
	Target_ID_entity	
	Weapon_ID_site	
	Weapon_ID_application	
	Weapon_ID_entity	
	Event_ID_site	
	Event_ID_application	
	Event_ID_entity	
	LocationX	
	LocationY	
	LocationZ	
	VelocityX	
	VelocityY	
	VelocityZ	
	Range	
	TimeStamp	
	BurstKind	
	BurstDomain	
	BurstCountry	
	BurstCategory	
	BurstSubcategory	
	BurstSpecific	

Class/Interaction	Term	Definition
	BurstExtra	
	BurstWarhead	
	BurstFuze	
	BurstQuantity	
	BurstRate	
Signal	TimeStamp	
	EntitySite	
	EntityHost	
	EntityId	
	RadioId	
	EncodingScheme	
	TdlType	
	SampleRate	
	Samples	
	DataStruct	
Collision	TimeStamp	
	IssuingEntitySite	
	IssuingEntityHost	
	IssuingEntityId	
	CollidingEntitySite	
	CollidingEntityHost	
	CollidingEntityId	
	EventIdSite	
	EventIdHost	
	EventIdEvent	
	VelocityX	
	VelocityY	
	VelocityZ	
	Mass	
	RelLocationX	
	RelLocationY	
	RelLocationZ	

Appendix E – PHASE II FEDERATION WORKBOOK

Federation Execution Summary Table

Federation Execution Name: CCTT Federation

Number of Concurrent Federation Executions (total including this Federation Execution): 2

RTI Software Used (Version): 1.01

NOTE:
Complete one of these tables for each Federation execution

Federate Summary Information

	Name	API (C++,Ada, IDL, Java)	Time Management Switches		Host (assign # to each host) (List data on Host Table)	LAN (assign # to each LAN) (List data on LAN Tables)
			Regulating (y or n)	Constraining (y or n)		
Fed 1	Surrogate For CCTT Federation	Ada	N	N	1	1
Fed 2	dis federate	Ada	N	N	2	1
Fed 3	edb federate	Ada	N	N	2	1
Fed 4						
Fed 5						
Fed 6						
Fed 7						
Fed 8						
Fed 9						
Fed 10						
Fed 11						
Fed N						

Host Table

	Hardware	Operating System	Memory available to RTI (MB)	% CPU Available to RTI
Host $_1$	Motorola Powerstack	AIX 4.2		
Host $_2$	Motorola Powerstack	AIX 4.2		
Host $_3$				
Host $_4$				
Host $_5$				
Host $_6$				
Host $_7$				
Host $_n$				

NOTE:
Complete one of these tables for each Federation execution

LAN Tables

LAN Table 1: LAN Descriptions

	Physical Type (Ethernet, ATM, etc.)	Throughtput Available to FEDEX
LAN_1	Ethernet 10 Base-T	
LAN_2		
LAN_3		
LAN_4		
LAN_5		
LAN_6		
LAN_7		
LAN_n		

NOTE:
Complete one of these tables for each Federation execution

LAN Table 2: LAN to LAN Connectivity

	LAN_1	LAN_2	LAN_3	LAN_4	LAN_5	LAN_6	.	.	.	LAN_N
LAN_1										
LAN_2	1. _____ * 2. _____ * 3. _____ *									
LAN_3	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *								
LAN_4	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *							
LAN_5	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *						
LAN_6	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *					
...										
LAN_N	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *				

*

1. Device type means type of switch employed to connect the LANs
2. Throughput means the effective throughput available through the LAN connection for Federation execution, expressed in Mb
3. Latency contribution from devices connecting the LANs, expressed in milliseconds.

RTI Services Table

(Check if service to be used at least once during this Federation execution)

Service	IF Ref	Service Used?
Create Federation Execution	2.1	✓
Destroy Federation Execution	2.2	✓
Join Federation Execution	2.3	✓
Resign Federation Execution	2.4	✓
Request **Pause**	2.5	
Initiate **Pause**	2.6	
Pause Achieved	2.7	
Request **Resume**	2.8	
Initiate **Resume**	2.9	
Resume Achieved	2.10	
Request **Federation Save**	2.11	
Initiate **Federation Save**	2.12	
Federation Save Begun	2.13	
Federation Save Achieved	2.14	
Request **Restore**	2.15	
Initiate **Restore**	2.16	
Restore Achieved	2.17	
Publish Object Class	3.1	✓
Subscribe Object Class **Attributes**	3.2	✓
Publish Interaction	3.3	✓
Subscribe Interaction	3.4	✓
Control Updates	3.5	✓
Control Interactions	3.6	✓
Request ID	4.1	✓
Register Object	4.2	✓
Update Attribute Values	4.3	✓
Delete Object	4.8	
Remove Object	4.9	
Change Attribute Transportation Type	4.10	
Change Attribute Order Type	4.11	
Change Interaction Transportation Type	4.12	
Change Interaction Order Type	4.13	
Request Attribute Value Update	4.14	✓
Provide Attribute Value Update	4.15	✓
Retract	4.16	
Reflect Retract	4.17	
Request Attribute **Ownership Divestiture**	5.1	✓
Request Attribute **Ownership Assumption**	5.2	✓
Attribute Ownership **Divestiture Notification**	5.3	✓
Attribute Ownership **Acquisition Notification**	5.4	✓
Request Attribute **Ownership Acquisition**	5.5	✓
Request Attribute **Ownership Release**	5.6	✓
Query Attribute Ownership	5.7	
Inform Attribute Ownership	5.8	
Is **Attribute Owned by Federate?**	5.9	
Request **Federation Time**	6.1	
Request **LBTS**	6.2	
Request **Federate Time**	6.3	
Request **Min Next Event Time**	6.4	
Set **Lookahead**	6.5	
Request **Lookahead**	6.6	
Time Advance Request	6.7	

NOTE: Complete one of these tables for each Federation execution

Object/Interaction Table

NOTE: Complete one of these tables for each Federate

Federate,
Surrogate for CCTT Federate

					If Update = "y"					If Subscribe = y?	Ownership	
Object/ Interaction Class	**Attribute/ Parameter**	**Count**	**Size**	**Update? (y or n)**	**Update Rate # updates/unit time**	**Update Grouping (Assign same letter to attributes which will all be updated at the same time)**	**Transport R= Reliable B= Best Effort**	**Ordering TSO or FIFO**	**Subscribe? (y or n)**	**Maximum tolerable latency from any source (milliseconds)**	**Attribute Ownership Transfer Rate # times/unit time**	**Ownership Transfer Grouping (Assign same letter to attributes which will all be transferred together)**
Entity	FORCE_ID		2 bytes	y		A	R	FIFO	n			
	ENTITYID_SITE		2 bytes	y		A	R	FIFO	y			
	ENTITYID_APPLICATION		2 bytes	y		A	R	FIFO	y			
	ENTITYID_ENTITY		2 bytes	y		A	R	FIFO	y			
	LOCATION_X		8 bytes	y	15	A	R	FIFO	y			
	LOCATION_Y		8 bytes	y	15	A	R	FIFO	y			
	LOCATION_Z		8 bytes	y	15	A	R	FIFO	y			
	VELOCITY_X		4 bytes	n					n			
	VELOCITY_Y		4 bytes	n					n			
	VELOCITY_Z		4 bytes	n					n			
	ACCELERATION_X		4 bytes	n					n			
	ACCELERATION_Y		4 bytes	n					n			
	ACCELERATION_Z		4 bytes	n					n			
	ORIENTATION_PSI		4 bytes	n					n			
	ORIENTATION_THETA		4 bytes	n					n			
	ORIENTATION_PHI		4 bytes	n					n			
	APPEARANCE_PAINT_SCHEME		2 bytes	n					n			
Platform	DAMAGE_STATE_APPEARANCE		2 bytes	y	15	B	R	FIFO	y			
Munition	ANGULAR_VELOCITY_PSI		4 bytes	n					n			
	ANGULAR_VELOCITY_THETA		4 bytes	n					n			
	ANGULAR_VELOCITY_PHI		4 bytes	n					n			
	FIRING_ID		2 bytes	n					n			
Ground Vehicle	APPEARANCE_SMOKE		2 bytes	n					n			
	APPEARANCE_TRAILING		2 bytes	n					n			
	APPEARANCE_HATCH		2 bytes	n					n			
	APPEARANCE_LIGHTS		2 bytes	n					n			
	APPEARANCE_FLAMING		2 bytes	n					n			
	DAMAGE_STATE_MOBILITY		2 bytes	y	15	B	R	FIFO	y			
	DAMAGE_STATE_FIRE_POWER		2 bytes	y	15	B	R	FIFO	y			
Air Vehicle	ANGULAR_VELOCITY_PSI		4 bytes	n					n			
	ANGULAR_VELOCITY_THETA		4 bytes	n					n			
	ANGULAR_VELOCITY_PHI		4 bytes	n					n			

Tracked	TURRET_HEADING		4 bytes	n					n			
	MAIN_GUN_PITCH		4 bytes	n					n			
	COMMANDERS_GUN_PITCH		4 bytes	n					n			
	MACHINE_GUN_PITCH		4 bytes	n					n			
Armored Fighing Vehicle												
	APPEARANCE_LAUNCHER		2 bytes	n					n			
DETONATION	Munition_ID		2 bytes	y		C	R	FIFO	y			
	Target_ID		2 bytes	y		C	R	FIFO	y			
	Firing_ID		2 bytes	y		C	R	FIFO	y			
	Location_X		8 bytes	y	15	C	R	FIFO	y			
	Location_Y		8 bytes	y	15	C	R	FIFO	y			
	Location_Z		8 bytes	y	15	C	R	FIFO	y			
	Relative_Location_X		4 bytes	y	15	C	R	FIFO	y			
	Relative_Location_Y		4 bytes	y	15	C	R	FIFO	y			
	Relative_Location_Z		4 bytes	y	15	C	R	FIFO	y			
	Velocity_X		4 bytes	y	15	C	R	FIFO	y			
	Velocity_Y		4 bytes	y	15	C	R	FIFO	y			
	Velocity_Z		4 bytes	y	15	C	R	FIFO	y			
	Burst_Descriptor_Warhead		2 bytes	y	15	C	R	FIFO	y			
	Burst_Descriptor_Fuze		2 bytes	y	15	C	R	FIFO	y			
	Burst_Descriptor_Detonation_Result		2 bytes	y	15	C	R	FIFO	y			
	Quantity		2 bytes	y	15	C	R	FIFO	y			
	Rate		2 bytes	y	15	C	R	FIFO	y			
WEAPON FIRE	Firing_ID		2 bytes	y		D	R	FIFO	y			
	Target_ID		2 bytes	y		D	R	FIFO	y			
	Location_X		8 bytes	y	15	D	R	FIFO	y			
	Location_Y		8 bytes	y	15	D	R	FIFO	y			
	Location_Z		8 bytes	y	15	D	R	FIFO	y			
	Velocity_X		8 bytes	y	15	D	R	FIFO	y			
	Velocity_Y		8 bytes	y	15	D	R	FIFO	y			
	Velocity_Z		8 bytes	y	15	D	R	FIFO	y			
	Munition_Type		2 bytes	y	15	D	R	FIFO	y			
	Quantity		2 bytes	y	15	D	R	FIFO	y			
	Rate		2 bytes	y	15	D	R	FIFO	y			
WEAPON LAUNCH												
	Launch_Platform_ID		2 bytes	n					n			
	Munition_ID		2 bytes	n					n			
	Target_ID		2 bytes	n					n			
	Location_X		8 bytes	n					n			
	Location_Y		8 bytes	n					n			
	Location_Z		8 bytes	n					n			
	Velocity_X		8 bytes	n					n			
	Velocity_Y		8 bytes	n					n			
	Velocity_Z		8 bytes	n					n			
COLLISION	Issuing_ID		2 bytes	n					n			
	Colliding_ID		2 bytes	n					n			

	Mass		2 bytes	n					n			
	Relative_Location_X		4 bytes	n					n			
	Relative_Location_Y		4 bytes	n					n			
	Relative_Location_Z		4 bytes	n					n			
	Velocity_X		4 bytes	n					n			
	Velocity_Y		4 bytes	n					n			
	Velocity_Z		4 bytes	n					n			
AIR VEHICLE LAUNCH												
	Launch_Platform_ID		2 bytes	n					n			
	Air_Vehicle_ID		2 bytes	n					n			
	Location_X		4 bytes	n					n			
	Location_Y		4 bytes	n					n			
	Location_Z		4 bytes	n					n			
	Velocity_X		4 bytes	n					n			
	Velocity_Y		4 bytes	n					n			
	Velocity_Z		4 bytes	n					n			

CC TT

Object Class Structure Table

Class1	Class2	Class3	Class4	Class5	Class6
Entity (PS)	Platform (PS)	Ground_Vehicle (PS)	Tracked (PS)	Tank (PS)	M1 (PS)
					T72 (PS)
				Armored_Fighting_Vehicle (PS)	M2 (PS)
					BMP (PS)
		Air_Vehicle (PS)	Attack (PS)	FA18 (PS)	
		Human (PS)	INFANTRY (PS)		
	Munition (PS)	Guided (PS)	Anti_Ground (PS)	TOW (PS)	
				AT5 (PS)	
				LGB (PS)	
			Anti_Air (PS)	SA16 (PS)	

Term	Definition
DETONATION	Defines a detonation interaction
WEAPON_FIRE	Defines a weapon fire interaction.
WEAPON_LAUNCH	Defines a weapon launch interacion.
COLLISION	Defines a collision interaction.
AIR_VEHICLE_LAUNCH	Defines an air launch munition interaction.

Object	Attribute	Datatype	Cardinality	Units
Entity	FORCE_ID	short	1	enumeration
	ENTITYID_SITE	short	1	
	ENTITYID_APPLICATION	short	1	
	ENTITYID_ENTITY	short	1	
	LOCATION_X	double	1	meters
	LOCATION_Y	double	1	meters
	LOCATION_Z	double	1	meters
	VELOCITY_X	float	1	meters/sec
	VELOCITY_Y	float	1	meters/sec
	VELOCITY_Z	float	1	meters/sec
	ACCELERATION_X	float	1	meters/sec^2
	ACCELERATION_Y	float	1	meters/sec^2
	ACCELERATION_Z	float	1	meters/sec^2
	ORIENTATION_PSI	float	1	radians
	ORIENTATION_THETA	float	1	radians
	ORIENTATION_PHI	float	1	radians
	APPEARANCE_PAINT_SCHEME	short	1	enumeration
Platform	DAMAGE_STATE_APPEARANC	short	1	enumeration
Munition	FIRING_ID	long	1	
	ANGULAR_VELOCITY_PSI	float	1	radians
	ANGULAR_VELOCITY_THETA	float	1	radians
	ANGULAR_VELOCITY_PHI	float	1	radians
Ground_Vehicle	APPEARANCE_SMOKE	short	1	enumeration
	APPEARANCE_TRAILING	short	1	enumeration
	APPEARANCE_HATCH	short	1	enumeration
	APPEARANCE_LIGHTS	short	1	enumeration
	APPEARACNE_FLAMING	short	1	enumeration
	DAMAGE_STATE_MOBILITY	short	1	enumeration
	DAMAGE_STATE_FIRE_POWE	short	1	enumeration
Air_Vehicle	ANGULAR_VELOCITY_PSI	float	1	radians
	ANGULAR_VELOCITY_THETA	float	1	radians
	ANGULAR_VELOCITY_PHI	float	1	radians
Tracked	TURRET_HEADING	float	1	radians
	MAIN_GUN_PITCH	float	1	radians
	COMMANDERS_GUN_PITCH	float	1	radians
	MACHINE_GUN_PITCH	float	1	radians
Armored_Fighting_Vehicle	APPEARANCE_LAUNCHER	short	1	enumeration

Resolution	Accuracy	Accuracy Condition	Update Type	Update Condition	Transferable/Acceptable	Updateable/Reflectable
n/a	perfect	n/a	Static		N	UR
n/a	perfect	n/a	Static		N	UR
n/a	perfect	n/a	Static		N	UR
n/a	perfect	n/a	Static		N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
n/a	perfect	n/a	Static		N	UR
n/a	perfect	n/a	Conditional		N	UR
n/a	perfect	n/a	Static		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
n/a	perfect	n/a	Conditional		N	UR
n/a	perfect	n/a	Conditional		N	UR
n/a	perfect	n/a	Conditional		N	UR
n/a	perfect	n/a	Conditional		N	UR
n/a	perfect	n/a	Conditional		N	UR
n/a	perfect	n/a	Conditional		N	UR
n/a	perfect	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
n/a	perfect	n/a	Conditional		N	UR

Interaction	Parameter	Datatype	Cardinality	Units	Resolution	Accuracy	Accuracy Condition
DETONATION	Munition_ID	long	1			perfect	n/a
	Target_ID	long	1			perfect	n/a
	Firing_ID	long	1			perfect	n/a
	Location_X	double	1	meters		n/a	n/a
	Location_Y	double	1	meters		n/a	n/a
	Location_Z	double	1	meters		n/a	n/a
	Relative_Location_X	double	1	meters		n/a	n/a
	Relative_Location_Y	double	1	meters		n/a	n/a
	Relative_Location_Z	double	1	meters		n/a	n/a
	Velocity_X	float	1	meters/sec		n/a	n/a
	Velocity_Y	float	1	meters/sec		n/a	n/a
	Velocity_Z	float	1	meters/sec		n/a	n/a
	Burst_Descriptor_Warhead	short	1	enumeration		n/a	n/a
	Burst_Descriptor_Fuze	short	1	enumeration		n/a	n/a
	Burst_Descriptor_Detonation_Result	short	1	enumeration		n/a	n/a
	Quantity	short	1			n/a	n/a
	Rate	float	1			n/a	n/a
WEAPON_FIRE	Firing_ID	long	1			n/a	n/a
	Target_ID	long	1			n/a	n/a
	Location_X	double	1	meters		n/a	n/a
	Location_Y	double	1	meters		n/a	n/a
	Location_Z	double	1	meters		n/a	n/a
	Velocity_X	float	1	meters/sec		n/a	n/a
	Velocity_Y	float	1	meters/sec		n/a	n/a
	Velocity_Z	float	1	meters/sec		n/a	n/a
	Munition_Type	string	1			n/a	n/a
	Quantity	short	1			n/a	n/a
	Rate	float	1			n/a	n/a
WEAPON_LAUNCH	Launch_Platform_ID	long	1			perfect	n/a
	Munitiion_ID	long	1			perfect	n/a
	Target_ID	long	1			perfect	n/a
	Location_X	double	1	meters		n/a	n/a
	Location_Y	double	1	meters		n/a	n/a

Interaction	Parameter	Datatype	Cardinality	Units	Resolution	Accuracy	Accuracy Condition
	Location_Z	double	1	meters		n/a	n/a
	Velocity_X	float	1	meters/sec		n/a	n/a
	Velocity_Y	float	1	meters/sec		n/a	n/a
	Velocity_Z	float	1	meters/sec		n/a	n/a
COLLISION	Issuing_ID	long	1			perfect	n/a
	Colliding_ID	long	1			perfect	n/a
	Mass	float	1	kg		n/a	n/a
	Relative_Location_X	double	1	meters		n/a	n/a
	Relative_Location_Y	double	1	meters		n/a	n/a
	Relative_Location_Z	double	1	meters		n/a	n/a
	Velocity_X	float	1	meters/sec		n/a	n/a
	Velocity_Y	float	1	meters/sec		n/a	n/a
	Velocity_Z	float	1	meters/sec		n/a	n/a
AIR_VEHICLE_LAUNCH	Launch_Platform_ID	long	1			perfect	n/a
	Air_Vehicle_ID	long	1			perfect	n/a
	Location_X	double	1	meters		n/a	n/a
	Location_Y	double	1	meters		n/a	n/a
	Location_Z	double	1	meters		n/a	n/a
	Velocity_X	float	1	meters/sec		n/a	n/a
	Velocity_Y	float	1	meters/sec		n/a	n/a
	Velocity_Z	float	1	meters/sec		n/a	n/a

Term	Definition
Entity	The class representing the highest level. All the simulation entities are under this class.
Platform	Class defining platform
Munition	Class defining munition
INFANTRY	Class defining infantry warfighter
Ground_Vehicle	Class defining ground vehicle of a platform class.
Air_Vehicle	Class defining air vehicle of the platform class
Human	Class defining human warfighter.
Tracked	Class defining tracked ground vehicle
Attack	Class defining attack air vehicle
Tank	Class defining tank
Armored_Fighting_Vehicle	Class defining armored fighting vehicle
M1	Class defining M1 tank
T72	Class defining T72 tank
M2	Class defining M2 AFV
BMP	Class defining BMP AFV
FA18	Class defining FA18 attack air vehicle
Guided	Class defining guided munition
Anti_Ground	Class defining anti ground guided munition
Anti_Air	Class defining anti-air guided munition
SA16	Class defining SA16 missile
TOW	Class defining a TOW missile
AT5	Class defining AT5 missile
LGB	Class defining LG8 missile

Class/Interaction	Term	Definition
Entity	FORCE_ID	The ID of the force of the entity class
	ENTITYID_SITE	The site id of the entity class
	ENTITYID_APPLICATION	The application id of the entity class
	ENTITYID_ENTITY	The entity part of the entity id of the entity class
	LOCATION_X	The x coordinate of the location of the entity.
	LOCATION_Y	This defines the y coordinate of the entity location
	LOCATION_Z	This attribute defines the z coordinate of the location of the entity class
	VELOCITY_X	Defines the x coordinate of the velocity of the entity class.
	VELOCITY_Y	Defines the y coordinate of the velocity of the entity class
	VELOCITY_Z	Defines the z coordinate of the velocity of the entity class.
	ACCELERATION_X	Defines the x coordinate of the acceleration of the entity class.
	ACCELERATION_Y	Defines the y coordiante of the acceleration of the entity class.
	ACCELERATION_Z	Defines the z coordinate of the acceleration of the entity class,
	ORIENTATION_PSI	Defines the psi angle of the orientation of the entity class.
	ORIENTATION_THETA	Defines the theta angle of the orientation of the entity class.
	ORIENTATION_PHI	Defines the phi angle of the orientation of the entity class.
	APPEARANCE_PAINT_SCHEME	Defines the paint scheme of the entity class.
Platform	DAMAGE_STATE_APPEARANCE	Defines the damange state of the platform class.
Munition	FIRING_ID	Defines the entityid that fires this munition class.
	ANGULAR_VELOCITY_PSI	Defines the psi angle of velocity of the munition.
	ANGULAR_VELOCITY_THETA	Defines the theta angle of velocity of the munition.
	ANGULAR_VELOCITY_PHI	Defines the phi angle of velocity of the munition.
Ground_Vehicle	APPEARANCE_SMOKE	Defines the smoke appearance of the ground vehicle class.
	APPEARANCE_TRAILING	Defines the trailing appearance of the ground vehicle class.
	APPEARANCE_HATCH	Defines the hatch appearance of the ground vehicle class.
	APPEARANCE_LIGHTS	Defines the lights appearance of the ground vehicle class.
	APPEARACNE_FLAMING	Define the flaming appearance of the ground vehicle class.
	DAMAGE_STATE_MOBILITY	Defines the mobility state of the ground vehicle class.
	DAMAGE_STATE_FIRE_POWER	Defines the fire power state of the ground vehicle class.
Air_Vehicle	ANGULAR_VELOCITY_PSI	Defines the psi angle of velocity of the aircraft.
	ANGULAR_VELOCITY_THETA	Defines the theta angle of velocity of the aircraft.
	ANGULAR_VELOCITY_PHI	Defines the phi angle of velocity of the aircraft.
Tracked	TURRET_HEADING	Defines the turret heading.
	MAIN_GUN_PITCH	Defines the main gun pitch angle.

Class/Interaction	Term	Definition
	COMMANDERS_GUN_PITCH	Defines the commanders gun pitch angle.
	MACHINE_GUN_PITCH	Defines the machine gun pitch angle.
Armored_Fighting_Vehicle	APPEARANCE_LAUNCHER	Defines the launcher appearance of the Armored_Fighting_Vehicle class.
DETONATION	Munition_ID	Defines the id of the munition
	Target_ID	Defines the id of the target
	Firing_ID	Defines the id of the firing entity.
	Location_X	Defines the x coordiante of the detonation location
	Location_Y	Defines the y coordiante of the detonation location.
	Location_Z	Defines the z coordiante of the detonation location.
	Relative_Location_X	Defines the x coordinate of the detonation location relative to the center o
	Relative_Location_Y	Defines the y coordinate of the detonation location relative to the center o
	Relative_Location_Z	Defines the z coordinate of the detonation location relative to the center
	Velocity_X	Defines the impacting velocity x
	Velocity_Y	Defines the impacting velocity y
	Velocity_Z	Defines the impacting velocity z
	Burst_Descriptor_Warhead	Defines the warhead type
	Burst_Descriptor_Fuze	Defines the fuze type
	Burst_Descriptor_Detonation_Result	Defines the result of the detonation
	Quantity	Defines the quantity of the warhead.
	Rate	Defines the rate of the munition
WEAPON_FIRE	Firing_ID	Defines the id of the firing entity
	Target_ID	Defines the id of the target
	Location_X	Defines the x coordinate of the location where it is firing at.
	Location_Y	Defines the y coordinate of the location where it is firing at.
	Location_Z	Defines the z coordinate of the location where it is firing at.
	Velocity_X	Defines the x velocity of the munition when it leaves the weapon.
	Velocity_Y	Defines the y velocity of the munition when it leaves the weapon.
	Velocity_Z	Defines the z velocity of the munition when it leaves the weapon.
	Munition_Type	Defines the type of munition
	Quantity	Defines the number of the munition.
	Rate	Defines the rate of the munition.
WEAPON_LAUNCH	Launch_Platform_ID	Defines the id of the launch platorm.
	Munitiion_ID	Defines the id of the munition
	Target_ID	Defines the id of the target.

Class/Interaction	Term	Definition
	Location_X	Defines the x coordinate of the location where the weapon is targeted.
	Location_Y	Defines the y coordinate of the location the wepaon is targeting at.
	Location_Z	Defines the z coordinate of the location where the weapon is targeting at.
	Velocity_X	Defines the missile x velocity when it leaves the launcher.
	Velocity_Y	Defines the missile y velocity when it leaves the launcher.
	Velocity_Z	Defines the missile z velocity when the missile leaves the launcher.
COLLISION	Issuing_ID	Defines the id of the entity that issues this interaction.
	Colliding_ID	Defines the id which the entity has colided.
	Mass	Defines the mass of the entity which issues this interaction.
	Relative_Location_X	Defines the x coordinate of the relative location of the center of the issuin
	Relative_Location_Y	Defines the y coordinate of the location relative to the center of the issuin
	Relative_Location_Z	Defines the z coordiante of location relative to the center of the issuing en
	Velocity_X	Defines the x velocity of the issuing entity.
	Velocity_Y	Defines the y location of the issuing entity.
	Velocity_Z	Defines the z velocity of the issuing enitty.
AIR_VEHICLE_LAUNCH	Launch_Platform_ID	Defines the id of the launch platform.
	Air_Vehicle_ID	Defines the id of the aircraft that launched the munition.
	Location_X	Defines the x coordinate of the location where it is firing at.
	Location_Y	Defines the y coordinate of the location where it is firing at.
	Location_Z	Defines the z coordinate of the location where it is firing at.
	Velocity_X	Defines the x velocity of the munition when it leaves the weapon.
	Velocity_Y	Defines the y velocity of the munition when it leaves the weapon.
	Velocity_Z	Defines the z velocity of the munition when it leaves the weapon.

```
;; CCTTSAF FED
(fed
 (objects
 (class Entity
    (attribute FORCE_ID FED_RELIABLE FED_RECEIVE)
    (attribute ENTITYID_SITE FED_RELIABLE FED_RECEIVE)
    (attribute ENTITYID_APPLICATION FED_RELIABLE FED_RECEIVE)
    (attribute ENTITYID_ENTITY FED_RELIABLE FED_RECEIVE)
    (attribute LOCATION_X FED_RELIABLE FED_RECEIVE)
    (attribute LOCATION_Y FED_RELIABLE FED_RECEIVE)
    (attribute LOCATION_Z FED_RELIABLE FED_RECEIVE)
    (attribute VELOCITY_X FED_RELIABLE FED_RECEIVE)
    (attribute VELOCITY_Y FED_RELIABLE FED_RECEIVE)
    (attribute VELOCITY_Z FED_RELIABLE FED_RECEIVE)
    (attribute ACCELERATION_X FED_RELIABLE FED_RECEIVE)
    (attribute ACCELERATION_Y FED_RELIABLE FED_RECEIVE)
    (attribute ACCELERATION_Z FED_RELIABLE FED_RECEIVE)
    (attribute ORIENTATION_PSI FED_RELIABLE FED_RECEIVE)
    (attribute ORIENTATION_THETA FED_RELIABLE FED_RECEIVE)
    (attribute ORIENTATION_PHI FED_RELIABLE FED_RECEIVE)
    (attribute APPEARANCE_PAINT_SCHEME FED_RELIABLE FED_RECEIVE)
 (class Platform
    (attribute DAMAGE_STATE_APPEARANCE FED_RELIABLE FED_RECEIVE)
 (class Ground_Vehicle
    (attribute APPEARANCE_SMOKE FED_RELIABLE FED_RECEIVE)
    (attribute APPEARANCE_TRAILING FED_RELIABLE FED_RECEIVE)
    (attribute APPEARANCE_HATCH FED_RELIABLE FED_RECEIVE)
    (attribute APPEARANCE_LIGHTS FED_RELIABLE FED_RECEIVE)
    (attribute APPEARANCE_FLAMING FED_RELIABLE FED_RECEIVE)
    (attribute DAMAGE_STATE_MOBILITY FED_RELIABLE FED_RECEIVE)
    (attribute DAMAGE_STATE_FIRE_POWER FED_RELIABLE FED_RECEIVE)
 (class Tracked
    (attribute TURRET_HEADING FED_RELIABLE FED_RECEIVE)
    (attribute MAIN_GUN_PITCH FED_RELIABLE FED_RECEIVE)
    (attribute COMMANDERS_GUN_PITCH FED_RELIABLE FED_RECEIVE)
    (attribute MACHINE_GUN_PITCH FED_RELIABLE FED_RECEIVE)
 (class Tank
 (class M1
 )
 (class T72
 )
 )
 (class Armored_Fighting_Vehicle
    (attribute APPEARANCE_LAUNCHER FED_RELIABLE FED_RECEIVE)
 (class M2
 )
 (class BMP
 )
 )
 )
 )
 (class Air_Vehicle
    (attribute ANGULAR_VELOCITY_PSI FED_RELIABLE FED_RECEIVE)
    (attribute ANGULAR_VELOCITY_THETA FED_RELIABLE FED_RECEIVE)
    (attribute ANGULAR_VELOCITY_PHI FED_RELIABLE FED_RECEIVE)
```

```
(class Attack
(class FA18
)
)
)
(class Human
(class INFANTRY
)
)
)
(class Munition
   (attribute ANGULAR_VELOCITY_PSI FED_RELIABLE FED_RECEIVE)
   (attribute ANGULAR_VELOCITY_THETA FED_RELIABLE FED_RECEIVE)
   (attribute ANGULAR_VELOCITY_PHI FED_RELIABLE FED_RECEIVE)
   (attribute FIRING_ID FED_RELIABLE FED_RECEIVE)
(class Guided
(class Anti_Ground
(class TOW
)
(class AT5
)
(class LGB
)
)
(class Anti_Air
(class SA16
)
)
)
)
)
 (class Manager
     (class Federate
         (attribute FederateHost FED_RELIABLE FED_RECEIVE)
         (attribute FederateHandle FED_RELIABLE FED_RECEIVE)
         (attribute FederateState FED_RELIABLE FED_RECEIVE)
         (attribute FederateName FED_RELIABLE FED_RECEIVE)
         (attribute RTIversion FED_RELIABLE FED_RECEIVE)
         (attribute TimeManagerState FED_RELIABLE FED_RECEIVE)
         (attribute FederateLookahead FED_RELIABLE FED_RECEIVE)
         (attribute FederateTime FED_RELIABLE FED_RECEIVE)
         (attribute TimeConstrained FED_RELIABLE FED_RECEIVE)
         (attribute TimeRegulating FED_RELIABLE FED_RECEIVE)
         (attribute FIFOlength FED_RELIABLE FED_RECEIVE)
         (attribute TSOlength FED_RELIABLE FED_RECEIVE)
         (attribute DequeueFIFOasync FED_RELIABLE FED_RECEIVE)
         (attribute TotalObjectCount FED_RELIABLE FED_RECEIVE)
         (attribute HoldingTokensObjectCount FED_RELIABLE FED_RECEIVE)
         (attribute DeletedObjectCount FED_RELIABLE FED_RECEIVE)
         (attribute NumAttributes FED_RELIABLE FED_RECEIVE)
         (attribute NumParameters FED_RELIABLE FED_RECEIVE)
         )
     (class Federation
         (attribute FederationName FED_RELIABLE FED_RECEIVE)
         (attribute FederationState FED_RELIABLE FED_RECEIVE)
```

```
            (attribute FederatesInFederation FED_RELIABLE FED_RECEIVE)
            (attribute SaveIsScheduled FED_RELIABLE FED_RECEIVE)
            (attribute ScheduledSaveTime FED_RELIABLE FED_RECEIVE)
            (attribute RTIversion FED_RELIABLE FED_RECEIVE)
            )
        )
)
(interactions
 (class DETONATION FED_RELIABLE FED_RECEIVE
   (parameter Munition_ID)
    (parameter Target_ID)
    (parameter Firing_ID)
    (parameter Location_X)
    (parameter Location_Y)
    (parameter Location_Z)
    (parameter Relative_Location_X)
    (parameter Relative_Location_Y)
    (parameter Relative_Location_Z)
    (parameter Velocity_X)
    (parameter Velocity_Y)
    (parameter Velocity_Z)
    (parameter Burst_Descriptor_Warhead)
    (parameter Burst_Descriptor_Fuze)
    (parameter Burst_Descriptor_Detonation_Result)
    (parameter Quantity)
    (parameter Rate)
 )

 (class WEAPON_FIRE FED_RELIABLE FED_RECEIVE
   (parameter Firing_ID)
    (parameter Target_ID)
    (parameter Location_X)
    (parameter Location_Y)
    (parameter Location_Z)
    (parameter Velocity_X)
    (parameter Velocity_Y)
    (parameter Velocity_Z)
    (parameter Munition_Type)
    (parameter Quantity)
    (parameter Rate)
 )

 (class WEAPON_LAUNCH FED_RELIABLE FED_RECEIVE
   (parameter Launch_Platform_ID)
    (parameter Munitiion_ID)
    (parameter Target_ID)
    (parameter Location_X)
    (parameter Location_Y)
    (parameter Location_Z)
    (parameter Velocity_X)
    (parameter Velocity_Y)
    (parameter Velocity_Z)
 )

 (class COLLISION FED_RELIABLE FED_RECEIVE
```

```
  (parameter Issuing_ID)
  (parameter Colliding_ID)
  (parameter Mass)
  (parameter Relative_Location_X)
  (parameter Relative_Location_Y)
  (parameter Relative_Location_Z)
  (parameter Velocity_X)
  (parameter Velocity_Y)
  (parameter Velocity_Z)
)

(class AIR_VEHICLE_LAUNCH FED_RELIABLE FED_RECEIVE
  (parameter Launch_Platform_ID)
  (parameter Air_Vehicle_ID)
  (parameter Location_X)
  (parameter Location_Y)
  (parameter Location_Z)
  (parameter Velocity_X)
  (parameter Velocity_Y)
  (parameter Velocity_Z)
)

(class Manager FED_RELIABLE FED_RECEIVE
    (class Federate FED_RELIABLE FED_RECEIVE
        (parameter FromFederate)
        (class Alert FED_RELIABLE FED_RECEIVE
            (parameter AlertSeverity)
            (parameter AlertText)
            (parameter AlertID)
            )
        (class ServiceLog FED_RELIABLE FED_RECEIVE
            (parameter ServiceName)
            (parameter ServiceInitiator)
            (class ServiceLogArguments FED_RELIABLE FED_RECEIVE
                (parameter Handle1)
                (parameter Handle2)
                (parameter HandleSet)
                (parameter ObjectIDorCount)
                (parameter TagOrLabelOrName)
                (parameter Time)
                (parameter Enumeration)
                (parameter Boolean)
                )
            )
        (class ObjectInformation FED_RELIABLE FED_RECEIVE
            (parameter ObjectID)
            (parameter LockedAttributes)
            (parameter RegisteredClass)
            (parameter RepresentedClass)
            )
        (class PublishingClass FED_RELIABLE FED_RECEIVE
            (parameter ObjectClass)
            ;; format = ClassHandle:attrHandle,attrHandle,...,attrHandle
            (parameter InteractionClass)
            ;; format = ClassHandle
```

```
        )
    (class SubscribingClass FED_RELIABLE FED_RECEIVE
        (parameter ObjectClass)
        ;; format = ClassHandle:attrHandle,attrHandle,...,attrHandle
        (parameter InteractionClass)
        ;; format = ClassHandle
        )
    (class Action FED_RELIABLE FED_RECEIVE
        (parameter ToFederate)
        (class RequestPublicationTree FED_RELIABLE FED_RECEIVE
            )
        (class RequestSubscriptionTree FED_RELIABLE FED_RECEIVE
            )
        (class SetTiming FED_RELIABLE FED_RECEIVE
            (parameter FedReportPeriod)
            (parameter TimeReportPeriod)
            (parameter ObjectReportPeriod)
            )
        (class RequestObjectInformation FED_RELIABLE FED_RECEIVE
            (parameter ObjectID)
            )
        (class ModifyAttributeState FED_RELIABLE FED_RECEIVE
            (parameter ObjectID)
            (parameter AttributeID)
            (parameter TokenState)
            )
        (class RemoteServiceInvocation FED_RELIABLE FED_RECEIVE
            (class DoResignFederationExecution FED_RELIABLE FED_RECEIVE
                (parameter ResignAction)
                )
            (class DoDeleteObject FED_RELIABLE FED_RECEIVE
                (parameter ObjectID)
                (parameter Time)
                (parameter Tag)
                )
            (class DoSetLookahead FED_RELIABLE FED_RECEIVE
                (parameter Lookahead)
                )
            (class DoSetTimeConstrained FED_RELIABLE FED_RECEIVE
                (parameter State)
                )
            (class DoTurnRegulationOn FED_RELIABLE FED_RECEIVE
                )
            (class DoTurnRegulationOff FED_RELIABLE FED_RECEIVE
                )
            )
        (class Control FED_RELIABLE FED_RECEIVE
            (parameter SetServiceLogging)
            (parameter SetLogFile)
            (parameter DeleteObject)
            (parameter DequeueFIFO)
            )
        )
    )
)
```

Federation Execution Summary Table

Federation Execution Name: Specialty Federation

Number of Concurrent Federation Executions (total including this Federation Execution): 2

RTI Software Used (Version): 1.01

NOTE:
Complete one of these tables for each Federation execution

Federate Summary Information

	Name	API (C++,Ada, IDL, Java)	Time Management Switches		Host (assign # to each host) (List data on Host Table)	LAN (assign # to each LAN) (List data on LAN Tables)
			Regulating (y or n)	Constraining (y or n)		
Fed 1	Surrogate For Specialty Federation	Ada	N	N	1	1
Fed 2	weapon fire federate	Ada	N	N	2	1
Fed 3	damage assessment federate	Ada	N	N	2	1
Fed 4						
Fed 5						
Fed 6						
Fed 7						
Fed 8						
Fed 9						
Fed 10						
Fed 11						
Fed N						

Host Table

	Hardware	Operating System	Memory available to RTI (MB)	% CPU Available to RTI
Host $_1$	Motorola Powerstack	AIX 4.2		
Host $_2$	Motorola Powerstack	AIX 4.2		
Host $_3$				
Host $_4$				
Host $_5$				
Host $_6$				
Host $_7$				
Host $_n$				

NOTE:
Complete one of these tables for each Federation execution

LAN Tables

NOTE:
Complete one of these tables for each Federation execution

LAN Table 1: LAN Descriptions

	Physical Type (Ethernet, ATM,etc.)	Throughtput Available to FEDEX
LAN_1	Ethernet 10 Base-T	
LAN_2		
LAN_3		
LAN_4		
LAN_5		
LAN_6		
LAN_7		
.		
.		
LAN_n		

LAN Table 2: LAN to LAN Connectivity

	LAN_1	LAN_2	LAN_3	LAN_4	LAN_5	LAN_6	. . .	LAN_N
LAN_1								
LAN_2	1. ____ 2. ____ 3. ____							
LAN_3	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____						
LAN_4	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____					
LAN_5	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____				
LAN_6	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____			
...								
LAN_N	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____	1. ____ 2. ____ 3. ____		

1. Device type means type of switch employed to connect the LANs
2. Throughput means the effective throughput available through the LAN connection for Federation execution, expressed in Mb
3. Latency contribution from devices connecting the LANs, expressed in milliseconds

RTI Services Table

(Check if service to be used at least once during this Federation execution)

Service	IF Ref	Service Used?
Create Federation Execution	2.1	✓
Destroy Federation Execution	2.2	✓
Join Federation Execution	2.3	✓
Resign Federation Execution	2.4	✓
Request **Pause**	2.5	
Initiate **Pause**	2.6	
Pause Achieved	2.7	
Request **Resume**	2.8	
Initiate **Resume**	2.9	
Resume Achieved	2.10	
Request **Federation Save**	2.11	
Initiate **Federation Save**	2.12	
Federation Save Begun	2.13	
Federation Save Achieved	2.14	
Request **Restore**	2.15	
Initiate **Restore**	2.16	
Restore Achieved	2.17	
Publish Object Class	3.1	✓
Subscribe Object Class **Attributes**	3.2	✓
Publish Interaction	3.3	✓
Subscribe Interaction	3.4	✓
Control Updates	3.5	✓
Control Interactions	3.6	✓
Request ID	4.1	✓
Register Object	4.2	✓
Update Attribute Values	4.3	✓
Delete Object	4.8	
Remove Object	4.9	
Change Attribute Transportation Type	4.10	
Change Attribute Order Type	4.11	
Change Interaction Transportation Type	4.12	
Change Interaction Order Type	4.13	
Request Attribute Value Update	4.14	✓
Provide Attribute Value Update	4.15	✓
Retract	4.16	
Reflect Retract	4.17	
Request Attribute **Ownership Divestiture**	5.1	✓
Request Attribute **Ownership Assumption**	5.2	✓
Attribute Ownership **Divestiture Notification**	5.3	✓
Attribute Ownership **Acquisition Notification**	5.4	✓
Request Attribute **Ownership Acquisition**	5.5	✓
Request Attribute **Ownership Release**	5.6	✓
Query Attribute Ownership	5.7	
Inform Attribute Ownership	5.8	
Is **Attribute Owned by Federate?**	5.9	
Request **Federation Time**	6.1	
Request **LBTS**	6.2	
Request **Federate Time**	6.3	
Request **Min Next Event Time**	6.4	
Set **Lookahead**	6.5	
Request **Lookahead**	6.6	
Time Advance Request	6.7	

NOTE: Complete one of these tables for each Federation execution

Object/Interaction Table

NOTE: Complete one of these tables for each Federate

Federate ,
Surrogate for Specialty Federate

				If Update =" y"					If Subscribe = y?	Ownership	
Object/ Interaction Class	**Attribute/ Parameter**	**Count**	**Size**	**Update? (y or n)**	**Update Rate # updates/unit time**	**Update Grouping (Assign same letter to attributes which will all be updated at the same time)**	**Transport R= Reliable B= Best Effort**	**Ordering TSO or FIFO**	**Subscribe? (y or n)**	**Maximum tolerable latency from any source (milliseconds)**	**Attribute Ownership Transfer Rate # times/unit time**	**Ownership Transfer Grouping (Assign same letter to attributes which will all be transferred together)**
Entity	FORCE_ID		2 bytes	y		A	R	FIFO	y			
	ENTITYID_SITE		2 bytes	y		A	R	FIFO	y			
	ENTITYID_APPLICATION		2 bytes	y		A	R	FIFO	y			
	ENTITYID_ENTITY		2 bytes	y		A	R	FIFO	y			
	LOCATION_X		8 bytes	y	15	A	R	FIFO	y			
	LOCATION_Y		8 bytes	y	15	A	R	FIFO	y			
	LOCATION_Z		8 bytes	y	15	A	R	FIFO	y			
	VELOCITY_X		4 bytes	y	15	A	R	FIFO	y			
	VELOCITY_Y		4 bytes	y	15	A	R	FIFO	y			
	VELOCITY_Z		4 bytes	y	15	A	R	FIFO	y			
	ACCELERATION_X		4 bytes	y	15	A	R	FIFO	y			
	ACCELERATION_Y		4 bytes	y	15	A	R	FIFO	y			
	ACCELERATION_Z		4 bytes	y	15	A	R	FIFO	y			
	ORIENTATION_PSI		4 bytes	y	15	A	R	FIFO	y			
	ORIENTATION_THETA		4 bytes	y	15	A	R	FIFO	y			
	ORIENTATION_PHI		4 bytes	y	15	A	R	FIFO	y			
	APPEARANCE_PAINT_SCHEME		2 bytes	y	15	B	R	FIFO	y			
Platform	DAMAGE_STATE_APPEARANCE		2 bytes	y	15	B	R	FIFO	y			
Munition	ANGULAR_VELOCITY_PSI		4 bytes	y	15	C	R	FIFO	y			
	ANGULAR_VELOCITY_THETA		4 bytes	y	15	C	R	FIFO	y			
	ANGULAR_VELOCITY_PHI		4 bytes	y	15	C	R	FIFO	y			
	FIRING_ID		2 bytes	y		C	R	FIFO	y			
Ground Vehicle	APPEARANCE_SMOKE		2 bytes	y	15	B	R	FIFO	y			
	APPEARANCE_TRAILING		2 bytes	y	15	B	R	FIFO	y			
	APPEARANCE_HATCH		2 bytes	y	15	B	R	FIFO	y			
	APPEARANCE_LIGHTS		2 bytes	y	15	B	R	FIFO	y			
	APPEARANCE_FLAMING		2 bytes	y	15	B	R	FIFO	y			
	DAMAGE_STATE_MOBILITY		2 bytes	y	15	B	R	FIFO	y			
	DAMAGE_STATE_FIRE_POWER		2 bytes	y	15	B	R	FIFO	y			
Air Vehicle	ANGULAR_VELOCITY_PSI		4 bytes	y	15	D	R	FIFO	y			
	ANGULAR_VELOCITY_THETA		4 bytes	y	15	D	R	FIFO	y			
	ANGULAR_VELOCITY_PHI		4 bytes	y	15	D	R	FIFO	y			

Tracked	TURRET_HEADING		4 bytes	y	15	E	R	FIFO	y			
	MAIN_GUN_PITCH		4 bytes	y	15	E	R	FIFO	y			
	COMMANDERS_GUN_PITCH		4 bytes	y	15	E	R	FIFO	y			
	MACHINE_GUN_PITCH		4 bytes	y	15	E	R	FIFO	y			
Armored Fighing Vehicle												
	APPEARANCE_LAUNCHER		2 bytes	y	15	F	R	FIFO	y			
DETONATION	Munition_ID		2 bytes	y		G	R	FIFO	y			
	Target_ID		2 bytes	y		G	R	FIFO	y			
	Firing_ID		2 bytes	y		G	R	FIFO	y			
	Location_X		8 bytes	y	15	G	R	FIFO	y			
	Location_Y		8 bytes	y	15	G	R	FIFO	y			
	Location_Z		8 bytes	y	15	G	R	FIFO	y			
	Relative_Location_X		4 bytes	y	15	G	R	FIFO	y			
	Relative_Location_Y		4 bytes	y	15	G	R	FIFO	y			
	Relative_Location_Z		4 bytes	y	15	G	R	FIFO	y			
	Velocity_X		4 bytes	y	15	G	R	FIFO	y			
	Velocity_Y		4 bytes	y	15	G	R	FIFO	y			
	Velocity_Z		4 bytes	y	15	G	R	FIFO	y			
	Burst_Descriptor_Warhead		2 bytes	y	15	G	R	FIFO	y			
	Burst_Descriptor_Fuze		2 bytes	y	15	G	R	FIFO	y			
	Burst_Descriptor_Detonation_Result		2 bytes	y	15	G	R	FIFO	y			
	Quantity		2 bytes	y	15	G	R	FIFO	y			
	Rate		2 bytes	y	15	G	R	FIFO	y			
WEAPON FIRE	Firing_ID		2 bytes	y		H	R	FIFO	y			
	Target_ID		2 bytes	y		H	R	FIFO	y			
	Location_X		8 bytes	y	15	H	R	FIFO	y			
	Location_Y		8 bytes	y	15	H	R	FIFO	y			
	Location_Z		8 bytes	y	15	H	R	FIFO	y			
	Velocity_X		8 bytes	y	15	H	R	FIFO	y			
	Velocity_Y		8 bytes	y	15	H	R	FIFO	y			
	Velocity_Z		8 bytes	y	15	H	R	FIFO	y			
	Munition_Type		2 bytes	y	15	H	R	FIFO	y			
	Quantity		2 bytes	y	15	H	R	FIFO	y			
	Rate		2 bytes	y	15	H	R	FIFO	y			
WEAPON LAUNCH												
	Launch_Platform_ID		2 bytes	n					y			
	Munition_ID		2 bytes	n					y			
	Target_ID		2 bytes	n					y			
	Location_X		8 bytes	n					y			
	Location_Y		8 bytes	n					y			
	Location_Z		8 bytes	n					y			
	Velocity_X		8 bytes	n					y			
	Velocity_Y		8 bytes	n					y			
	Velocity_Z		8 bytes	n					y			
COLLISION	Issuing_ID		2 bytes	n					y			
	Colliding_ID		2 bytes	n					y			

	Mass		2 bytes	n					y				
	Relative_Location_X		4 bytes	n					y				
	Relative_Location_Y		4 bytes	n					y				
	Relative_Location_Z		4 bytes	n					y				
	Velocity_X		4 bytes	n					y				
	Velocity_Y		4 bytes	n					y				
	Velocity_Z		4 bytes	n					y				
AIR VEHICLE LAUNCH													
	Launch_Platform_ID		2 bytes	n					y				
	Air_Vehicle_ID		2 bytes	n					y				
	Location_X		4 bytes	n					y				
	Location_Y		4 bytes	n					y				
	Location_Z		4 bytes	n					y				
	Velocity_X		4 bytes	n					y				
	Velocity_Y		4 bytes	n					y				
	Velocity_Z		4 bytes	n					y				

Object/Interaction Table

NOTE: Complete one of these tables for each Federate

Federate #
Weapon Fire Federate

					If Update = " y"					If Subscribe = y?	Ownership	
Object/ Interaction Class	**Attribute/ Parameter**	**Count**	**Size**	**Update? (y or n)**	**Update Rate # updates/unit time**	**Update Grouping** (Assign same letter to attributes which will all be updated at the same time)	**Transport** R= Reliable B= Best Effort	**Ordering** TSO or FIFO	**Subscribe? (y or n)**	**Maximum tolerable latency from any source (milliseconds)**	**Attribute Ownership Transfer Rate # times/unit time**	**Ownership Transfer Grouping** (Assign same letter to attributes which will all be transferred together)
Entity	FORCE_ID		2 bytes	y		A	R	FIFO	n			
	ENTITYID_SITE		2 bytes	y		A	R	FIFO	n			
	ENTITYID_APPLICATION		2 bytes	y		A	R	FIFO	n			
	ENTITYID_ENTITY		2 bytes	y		A	R	FIFO	y			
	LOCATION_X		8 bytes	y	15	A	R	FIFO	y			
	LOCATION_Y		8 bytes	y	15	A	R	FIFO	y			
	LOCATION_Z		8 bytes	y	15	A	R	FIFO	y			
	VELOCITY_X		4 bytes	n					n			
	VELOCITY_Y		4 bytes	n					n			
	VELOCITY_Z		4 bytes	n					n			
	ACCELERATION_X		4 bytes	n					n			
	ACCELERATION_Y		4 bytes	n					n			
	ACCELERATION_Z		4 bytes	n					n			
	ORIENTATION_PSI		4 bytes	n					n			
	ORIENTATION_THETA		4 bytes	n					n			
	ORIENTATION_PHI		4 bytes	n					n			
	APPEARANCE_PAINT_SCHEME		2 bytes	n					n			
Platform	DAMAGE_STATE_APPEARANCE		2 bytes	n					n			
Munition	ANGULAR_VELOCITY_PSI		4 bytes	n					n			
	ANGULAR_VELOCITY_THETA		4 bytes	n					n			
	ANGULAR_VELOCITY_PHI		4 bytes	n					n			
	FIRING_ID		2 bytes	n					n			
Ground Vehicle	APPEARANCE_SMOKE		2 bytes	n					n			
	APPEARANCE_TRAILING		2 bytes	n					n			
	APPEARANCE_HATCH		2 bytes	n					n			
	APPEARANCE_LIGHTS		2 bytes	n					n			
	APPEARANCE_FLAMING		2 bytes	n					n			
	DAMAGE_STATE_MOBILITY		2 bytes	n					n			
	DAMAGE_STATE_FIRE_POWER		2 bytes	n					n			
Air Vehicle	ANGULAR_VELOCITY_PSI		4 bytes	n					n			
	ANGULAR_VELOCITY_THETA		4 bytes	n					n			
	ANGULAR_VELOCITY_PHI		4 bytes	n					n			

Tracked	TURRET_HEADING		4 bytes	n					n				
	MAIN_GUN_PITCH		4 bytes	n					n				
	COMMANDERS_GUN_PITCH		4 bytes	n					n				
	MACHINE_GUN_PITCH		4 bytes	n					n				
Armored Fighing Vehicle													
	APPEARANCE_LAUNCHER		2 bytes	n					n				
DETONATION	Munition_ID		2 bytes	y		B	R	FIFO	n				
	Target_ID		2 bytes	y		B	R	FIFO	n				
	Firing_ID		2 bytes	y		B	R	FIFO	n				
	Location_X		8 bytes	y	15	B	R	FIFO	n				
	Location_Y		8 bytes	y	15	B	R	FIFO	n				
	Location_Z		8 bytes	y	15	B	R	FIFO	n				
	Relative_Location_X		4 bytes	y	15	B	R	FIFO	n				
	Relative_Location_Y		4 bytes	y	15	B	R	FIFO	n				
	Relative_Location_Z		4 bytes	y	15	B	R	FIFO	n				
	Velocity_X		4 bytes	y	15	B	R	FIFO	n				
	Velocity_Y		4 bytes	y	15	B	R	FIFO	n				
	Velocity_Z		4 bytes	y	15	B	R	FIFO	n				
	Burst_Descriptor_Warhead		2 bytes	y	15	B	R	FIFO	n				
	Burst_Descriptor_Fuze		2 bytes	y	15	B	R	FIFO	n				
	Burst_Descriptor_Detonation_Result		2 bytes	y	15	B	R	FIFO	n				
	Quantity		2 bytes	y	15	B	R	FIFO	n				
	Rate		2 bytes	y	15	B	R	FIFO	n				
WEAPON FIRE	Firing_ID		2 bytes	y		C	R	FIFO	n				
	Target_ID		2 bytes	y		C	R	FIFO	n				
	Location_X		8 bytes	y	15	C	R	FIFO	n				
	Location_Y		8 bytes	y	15	C	R	FIFO	n				
	Location_Z		8 bytes	y	15	C	R	FIFO	n				
	Velocity_X		8 bytes	y	15	C	R	FIFO	n				
	Velocity_Y		8 bytes	y	15	C	R	FIFO	n				
	Velocity_Z		8 bytes	y	15	C	R	FIFO	n				
	Munition_Type		2 bytes	y	15	C	R	FIFO	n				
	Quantity		2 bytes	y	15	C	R	FIFO	n				
	Rate		2 bytes	y	15	C	R	FIFO	n				
WEAPON LAUNCH													
	Launch_Platform_ID		2 bytes	n					n				
	Munition_ID		2 bytes	n					n				
	Target_ID		2 bytes	n					n				
	Location_X		8 bytes	n					n				
	Location_Y		8 bytes	n					n				
	Location_Z		8 bytes	n					n				
	Velocity_X		8 bytes	n					n				
	Velocity_Y		8 bytes	n					n				
	Velocity_Z		8 bytes	n					n				
COLLISION	Issuing_ID		2 bytes	n					n				
	Colliding_ID		2 bytes	n					n				

	Mass		2 bytes	n					n			
	Relative_Location_X		4 bytes	n					n			
	Relative_Location_Y		4 bytes	n					n			
	Relative_Location_Z		4 bytes	n					n			
	Velocity_X		4 bytes	n					n			
	Velocity_Y		4 bytes	n					n			
	Velocity_Z		4 bytes	n					n			
AIR VEHICLE LAUNCH												
	Launch_Platform_ID		2 bytes	n					n			
	Air_Vehicle_ID		2 bytes	n					n			
	Location_X		4 bytes	n					n			
	Location_Y		4 bytes	n					n			
	Location_Z		4 bytes	n					n			
	Velocity_X		4 bytes	n					n			
	Velocity_Y		4 bytes	n					n			
	Velocity_Z		4 bytes	n					n			

SPECIALTY

Object Class Structure Table

Class1	Class2	Class3	Class4	Class5	Class6
Entity (PS)	Platform (PS)	Ground_Vehicle (PS)	Tracked (PS)	Tank (PS)	M1 (PS)
					T72 (PS)
				Armored_Fighting_Vehicle (PS)	M2 (PS)
					BMP (PS)
		Air_Vehicle (PS)	Attack (PS)	FA18 (PS)	
		Human (PS)	INFANTRY (PS)		
	Munition (PS)	Guided (PS)	Anti_Ground (PS)	TOW (PS)	
				AT5 (PS)	
				LGB (PS)	
			Anti_Air (PS)	SA16 (PS)	

Term	Definition
DETONATION	Defines a detonation interaction
WEAPON_FIRE	Defines a weapon fire interaction.
WEAPON_LAUNCH	Defines a weapon launch interacion.
COLLISION	Defines a collision interaction.
AIR_VEHICLE_LAUNCH	Defines an air launch munition interaction.

Object	Attribute	Datatype	Cardinality	Units
Entity	FORCE_ID	short	1	enumeration
	ENTITYID_SITE	short	1	
	ENTITYID_APPLICATION	short	1	
	ENTITYID_ENTITY	short	1	
	LOCATION_X	double	1	meters
	LOCATION_Y	double	1	meters
	LOCATION_Z	double	1	meters
	VELOCITY_X	float	1	meters/sec
	VELOCITY_Y	float	1	meters/sec
	VELOCITY_Z	float	1	meters/sec
	ACCELERATION_X	float	1	meters/sec^2
	ACCELERATION_Y	float	1	meters/sec^2
	ACCELERATION_Z	float	1	meters/sec^2
	ORIENTATION_PSI	float	1	radians
	ORIENTATION_THETA	float	1	radians
	ORIENTATION_PHI	float	1	radians
	APPEARANCE_PAINT_SCHEME	short	1	enumeration
Platform	DAMAGE_STATE_APPEARANCE	short	1	enumeration
Munition	FIRING_ID	long	1	
	ANGULAR_VELOCITY_PSI	float	1	radians
	ANGULAR_VELOCITY_THETA	float	1	radians
	ANGULAR_VELOCITY_PHI	float	1	radians
Ground_Vehicle	APPEARANCE_SMOKE	short	1	enumeration
	APPEARANCE_TRAILING	short	1	enumeration
	APPEARANCE_HATCH	short	1	enumeration
	APPEARANCE_LIGHTS	short	1	enumeration
	APPEARACNE_FLAMING	short	1	enumeration
	DAMAGE_STATE_MOBILITY	short	1	enumeration
	DAMAGE_STATE_FIRE_POWER	short	1	enumeration
Air_Vehicle	ANGULAR_VELOCITY_PSI	float	1	radians
	ANGULAR_VELOCITY_THETA	float	1	radians
	ANGULAR_VELOCITY_PHI	float	1	radians
Tracked	TURRET_HEADING	float	1	radians
	MAIN_GUN_PITCH	float	1	radians
	COMMANDERS_GUN_PITCH	float	1	radians
	MACHINE_GUN_PITCH	float	1	radians
Armored_Fighting_Vehicle	APPEARANCE_LAUNCHER	short	1	enumeration

Resolution	Accuracy	Accuracy Condition	Update Type	Update Condition	Transferable/Acceptable	Updateable/Reflectable
n/a	perfect	n/a	Static		N	UR
n/a	perfect	n/a	Static		N	UR
n/a	perfect	n/a	Static		N	UR
n/a	perfect	n/a	Static		N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
	n/a	DR	Conditional	DR	N	UR
n/a	perfect	n/a	Static		N	UR
n/a	perfect	n/a	Conditional		N	UR
n/a	perfect	n/a	Static		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
n/a	n/a	n/a	Conditional		N	UR
n/a	n/a	n/a	Conditional		N	UR
n/a	n/a	n/a	Conditional		N	UR
n/a	n/a	n/a	Conditional		N	UR
n/a	n/a	n/a	Conditional		N	UR
n/a	n/a	n/a	Conditional		N	UR
n/a	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
	n/a	n/a	Conditional		N	UR
n/a	perfect	n/a	Conditional		N	UR

Interaction	Parameter	Datatype	Cardinality	Units	Resolution	Accuracy	Accuracy Condition
DETONATION	Munition_ID	long	1			perfect	n/a
	Target_ID	long	1			perfect	n/a
	Firing_ID	long	1			perfect	n/a
	Location_X	double	1	meters		n/a	n/a
	Location_Y	double	1	meters		n/a	n/a
	Location_Z	double	1	meters		n/a	n/a
	Relative_Location_X	double	1	meters		n/a	n/a
	Relative_Location_Y	double	1	meters		n/a	n/a
	Relative_Location_Z	double	1	meters		n/a	n/a
	Velocity_X	float	1	meters/sec		n/a	n/a
	Velocity_Y	float	1	meters/sec		n/a	n/a
	Velocity_Z	float	1	meters/sec		n/a	n/a
	Burst_Descriptor_Warhead	short	1	enumeration		n/a	n/a
	Burst_Descriptor_Fuze	short	1	enumeration		n/a	n/a
	Burst_Descriptor_Detonation_Result	short	1	enumeration		n/a	n/a
	Quantity	short	1			n/a	n/a
	Rate	float	1			n/a	n/a
WEAPON_FIRE	Firing_ID	long	1			n/a	n/a
	Target_ID	long	1			n/a	n/a
	Location_X	double	1	meters		n/a	n/a
	Location_Y	double	1	meters		n/a	n/a
	Location_Z	double	1	meters		n/a	n/a
	Velocity_X	float	1	meters/sec		n/a	n/a
	Velocity_Y	float	1	meters/sec		n/a	n/a
	Velocity_Z	float	1	meters/sec		n/a	n/a
	Munition_Type	string	1			n/a	n/a
	Quantity	short	1			n/a	n/a
	Rate	float	1			n/a	n/a
WEAPON_LAUNCH	Launch_Platform_ID	long	1			perfect	n/a
	Munitiion_ID	long	1			perfect	n/a
	Target_ID	long	1			perfect	n/a
	Location_X	double	1	meters		n/a	n/a
	Location_Y	double	1	meters		n/a	n/a

Interaction	Parameter	Datatype	Cardinality	Units	Resolution	Accuracy	Accuracy Condition
	Location_Z	double	1	meters		n/a	n/a
	Velocity_X	float	1	meters/sec		n/a	n/a
	Velocity_Y	float	1	meters/sec		n/a	n/a
	Velocity_Z	float	1	meters/sec		n/a	n/a
COLLISION	Issuing_ID	long	1			perfect	n/a
	Colliding_ID	long	1			perfect	n/a
	Mass	float	1	kg		n/a	n/a
	Relative_Location_X	double	1	meters		n/a	n/a
	Relative_Location_Y	double	1	meters		n/a	n/a
	Relative_Location_Z	double	1	meters		n/a	n/a
	Velocity_X	float	1	meters/sec		n/a	n/a
	Velocity_Y	float	1	meters/sec		n/a	n/a
	Velocity_Z	float	1	meters/sec		n/a	n/a
AIR_VEHICLE_LAUNCH	Launch_Platform_ID	long	1			perfect	n/a
	Air_Vehicle_ID	long	1			perfect	n/a
	Location_X	double	1	meters		n/a	n/a
	Location_Y	double	1	meters		n/a	n/a
	Location_Z	double	1	meters		n/a	n/a
	Velocity_X	float	1	meters/sec		n/a	n/a
	Velocity_Y	float	1	meters/sec		n/a	n/a
	Velocity_Z	float	1	meters/sec		n/a	n/a

Term	Definition
Entity	The class representing the highest level. All the simulation entities are under this class.
Platform	Class defining platform
Munition	Class defining munition
INFANTRY	Class defining infantry warfighter
Ground_Vehicle	Class defining ground vehicle of a platform class.
Air_Vehicle	Class defining air vehicle of the platform class
Human	Class defining human warfighter.
Tracked	Class defining tracked ground vehicle
Attack	Class defining attack air vehicle
Tank	Class defining tank
Armored_Fighting_Vehicle	Class defining armored fighting vehicle
M1	Class defining M1 tank
T72	Class defining T72 tank
M2	Class defining M2 AFV
BMP	Class defining BMP AFV
FA18	Class defining FA18 attack air vehicle
Guided	Class defining guided munition
Anti_Ground	Class defining anti ground guided munition
Anti_Air	Class defining anti-air guided munition
SA16	Class defining SA16 missile
TOW	Class defining a TOW missile
AT5	Class defining AT5 missile
LGB	Class defining LG8 missile

Class/Interaction	Term	Definition
Entity	FORCE_ID	The ID of the force of the entity class
	ENTITYID_SITE	The site id of the entity class
	ENTITYID_APPLICATION	The application id of the entity class
	ENTITYID_ENTITY	The entity part of the entity id of the entity class
	LOCATION_X	The x coordinate of the location of the entity.
	LOCATION_Y	This defines the y coordinate of the entity location
	LOCATION_Z	This attribute defines the z coordinate of the location of the entity class
	VELOCITY_X	Defines the x coordinate of the velocity of the entity class.
	VELOCITY_Y	Defines the y coordinate of the velocity of the entity class
	VELOCITY_Z	Defines the z coordinate of the velocity of the entity class.
	ACCELERATION_X	Defines the x coordinate of the acceleration of the entity class.
	ACCELERATION_Y	Defines the y coordiante of the acceleration of the entity class.
	ACCELERATION_Z	Defines the z coordinate of the acceleration of the entity class,
	ORIENTATION_PSI	Defines the psi angle of the orientation of the entity class.
	ORIENTATION_THETA	Defines the theta angle of the orientation of the entity class.
	ORIENTATION_PHI	Defines the phi angle of the orientation of the entity class.
	APPEARANCE_PAINT_SCHEME	Defines the paint scheme of the entity class.
Platform	DAMAGE_STATE_APPEARANCE	Defines the damange state of the platform class.
Munition	FIRING_ID	Defines the entityid that fires this munition class.
	ANGULAR_VELOCITY_PSI	Defines the psi angle of velocity of the munition.
	ANGULAR_VELOCITY_THETA	Defines the theta angle of velocity of the munition.
	ANGULAR_VELOCITY_PHI	Defines the phi angle of velocity of the munition.
Ground_Vehicle	APPEARANCE_SMOKE	Defines the smoke appearance of the ground vehicle class.
	APPEARANCE_TRAILING	Defines the trailing appearance of the ground vehicle class.
	APPEARANCE_HATCH	Defines the hatch appearance of the ground vehicle class.
	APPEARANCE_LIGHTS	Defines the lights appearance of the ground vehicle class.
	APPEARACNE_FLAMING	Define the flaming appearance of the ground vehicle class.
	DAMAGE_STATE_MOBILITY	Defines the mobility state of the ground vehicle class.
	DAMAGE_STATE_FIRE_POWER	Defines the fire power state of the ground vehicle class.
Air_Vehicle	ANGULAR_VELOCITY_PSI	Defines the psi angle of velocity of the aircraft.
	ANGULAR_VELOCITY_THETA	Defines the theta angle of velocity of the aircraft.
	ANGULAR_VELOCITY_PHI	Defines the phi angle of velocity of the aircraft.
Tracked	TURRET_HEADING	Defines the turret heading.
	MAIN_GUN_PITCH	Defines the main gun pitch angle.

Class/Interaction	Term	Definition
	COMMANDERS_GUN_PITCH	Defines the commangers gun pitch angle.
	MACHINE_GUN_PITCH	Defines the machine gun pitch angle.
Armored_Fighting_Vehicle	APPEARANCE_LAUNCHER	Defines the launcher appearance of the Armored_Fighting_Vehicle class.
DETONATION	Munition_ID	Defines the id of the munition
	Target_ID	Defines the id of the target
	Firing_ID	Defines the id of the firing entity.
	Location_X	Defines the x coordiante of the detonation location
	Location_Y	Defines the y coordiante of the detonation location.
	Location_Z	Defines the z coordiante of the detonation location.
	Relative_Location_X	Defines the x coordinate of the detonation location relative to the center of t
	Relative_Location_Y	Defines the y coordinate of the detonation location relative to the center of t
	Relative_Location_Z	Defines the z coordinate of the detonation location relative to the center of t
	Velocity_X	Defines the impacting velocity x
	Velocity_Y	Defines the impacting velocity y
	Velocity_Z	Defines the impacting velocity z
	Burst_Descriptor_Warhead	Defines the warhead type
	Burst_Descriptor_Fuze	Defines the fuze type
	Burst_Descriptor_Detonation_Resu	Defines the result of the detonation
	Quantity	Defines the quantity of the warhead.
	Rate	Defines the rate of the munition
WEAPON_FIRE	Firing_ID	Defines the id of the firing entity
	Target_ID	Defines the id of the target
	Location_X	Defines the x coordinate of the location where it is firing at.
	Location_Y	Defines the y coordinate of the location where it is firing at.
	Location_Z	Defines the z coordinate of the location where it is firing at.
	Velocity_X	Defines the x velocity of the munition when it leaves the weapon.
	Velocity_Y	Defines the y velocity of the munition when it leaves the weapon.
	Velocity_Z	Defines the z velocity of the munition when it leaves the weapon.
	Munition_Type	Defines the type of munition
	Quantity	Defines the number of the munition.
	Rate	Defines the rate of the munition.
WEAPON_LAUNCH	Launch_Platform_ID	Defines the id of the launch platorm.
	Munitiion_ID	Defines the id of the munition
	Target_ID	Defines the id of the target.

Class/Interaction	Term	Definition
	Location_X	Defines the x coordinate of the location where the weapon is targeted.
	Location_Y	Defines the y coordinate of the location the wepaon is targeting at.
	Location_Z	Defines the z coordinate of the location where the weapon is targeting at.
	Velocity_X	Defines the missile x velocity when it leaves the launcher.
	Velocity_Y	Defines the missile y velocity when it leaves the launcher.
	Velocity_Z	Defines the missile z velocity when the missile leaves the launcher.
COLLISION	Issuing_ID	Defines the id of the entity that issues this interaction.
	Colliding_ID	Defines the id which the entity has colided.
	Mass	Defines the mass of the entity which issues this interaction.
	Relative_Location_X	Defines the x coordinate of the relative location of the center of the issuing
	Relative_Location_Y	Defines the y coordinate of the location relative to the center of the issuing
	Relative_Location_Z	Defines the z coordiante of location relative to the center of the issuing entit
	Velocity_X	Defines the x velocity of the issuing entity.
	Velocity_Y	Defines the y location of the issuing entity.
	Velocity_Z	Defines the z velocity of the issuing enitty.
AIR_VEHICLE_LAUNCH	Launch_Platform_ID	Defines the id of the launch platform.
	Air_Vehicle_ID	Defines the id of the aircraft that launched the munition.
	Location_X	Defines the x coordinate of the location where it is firing at.
	Location_Y	Defines the y coordinate of the location where it is firing at.
	Location_Z	Defines the z coordinate of the location where it is firing at.
	Velocity_X	Defines the x velocity of the munition when it leaves the weapon.
	Velocity_Y	Defines the y velocity of the munition when it leaves the weapon.
	Velocity_Z	Defines the z velocity of the munition when it leaves the weapon.

```
;; MODSAF FED
(fed
 ;; objects
 (objects
   (class Entity
        (attribute Entity_ID_site FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_ID_application FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_ID_entity FED_BEST_EFFORT FED_RECEIVE )
        (attribute Force_ID FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Type_Kind FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Type_Domain FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Type_Country FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Type_Category FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Type_Subcategory FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Type_Specific FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Type_Extra FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Guise_Kind FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Guise_Domain FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Guise_Country FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Guise_Category FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Guise_Scategory FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Guise_Specific FED_BEST_EFFORT FED_RECEIVE )
        (attribute Entity_Guise_Extra FED_BEST_EFFORT FED_RECEIVE )
        (attribute Orientation_Psi FED_BEST_EFFORT FED_RECEIVE )
        (attribute Orientation_Theta FED_BEST_EFFORT FED_RECEIVE )
        (attribute Orientation_Phi FED_BEST_EFFORT FED_RECEIVE )
        (attribute Location_X FED_BEST_EFFORT FED_RECEIVE )
        (attribute Location_Y FED_BEST_EFFORT FED_RECEIVE )
        (attribute Location_Z FED_BEST_EFFORT FED_RECEIVE )
        (attribute Velocity_X FED_BEST_EFFORT FED_RECEIVE )
        (attribute Velocity_Y FED_BEST_EFFORT FED_RECEIVE )
        (attribute Velocity_Z FED_BEST_EFFORT FED_RECEIVE )
        (attribute marking_charset FED_BEST_EFFORT FED_RECEIVE )
        (attribute marking_text FED_BEST_EFFORT FED_RECEIVE )
        (attribute Appearance FED_BEST_EFFORT FED_RECEIVE )
        (attribute Dead_Reckoning_Algorithm FED_BEST_EFFORT FED_RECEIVE )
        (attribute Acceleration_X FED_BEST_EFFORT FED_RECEIVE )
        (attribute Acceleration_Y FED_BEST_EFFORT FED_RECEIVE )
        (attribute Acceleration_Z FED_BEST_EFFORT FED_RECEIVE )
        (attribute Angular_Velocity_Psi FED_BEST_EFFORT FED_RECEIVE )
        (attribute Angular_Velocity_Theta FED_BEST_EFFORT FED_RECEIVE )
        (attribute Angular_Velocity_Phi FED_BEST_EFFORT FED_RECEIVE )
        (attribute TimeStamp FED_BEST_EFFORT FED_RECEIVE )
        (attribute MarkingCharSet FED_BEST_EFFORT FED_RECEIVE )
        (attribute Capabilities FED_BEST_EFFORT FED_RECEIVE )
        (attribute ArticulatedStruct FED_BEST_EFFORT FED_RECEIVE )
  )
  (class Manager

   (class Federate
    (attribute FederateHost FED_RELIABLE FED_RECEIVE)
    (attribute FederateHandle FED_RELIABLE FED_RECEIVE)
    (attribute FederateState FED_RELIABLE FED_RECEIVE)
    (attribute FederateName FED_RELIABLE FED_RECEIVE)
    (attribute RTIversion FED_RELIABLE FED_RECEIVE)
```

```
    (attribute TimeManagerState FED_RELIABLE FED_RECEIVE)
    (attribute FederateLookahead FED_RELIABLE FED_RECEIVE)
    (attribute FederateTime FED_RELIABLE FED_RECEIVE)
    (attribute TimeConstrained FED_RELIABLE FED_RECEIVE)
    (attribute TimeRegulating FED_RELIABLE FED_RECEIVE)
    (attribute FIFOlength FED_RELIABLE FED_RECEIVE)
    (attribute TSOlength FED_RELIABLE FED_RECEIVE)
    (attribute DequeueFIFOasync FED_RELIABLE FED_RECEIVE)
    (attribute TotalObjectCount FED_RELIABLE FED_RECEIVE)
    (attribute HoldingTokensObjectCount FED_RELIABLE FED_RECEIVE)
    (attribute DeletedObjectCount FED_RELIABLE FED_RECEIVE)
    (attribute NumAttributes FED_RELIABLE FED_RECEIVE)
    (attribute NumParameters FED_RELIABLE FED_RECEIVE)
   )
   (class Federation
    (attribute FederationName FED_RELIABLE FED_RECEIVE)
    (attribute FederationState FED_RELIABLE FED_RECEIVE)
    (attribute FederatesInFederation FED_RELIABLE FED_RECEIVE)
    (attribute SaveIsScheduled FED_RELIABLE FED_RECEIVE)
    (attribute ScheduledSaveTime FED_RELIABLE FED_RECEIVE)
    (attribute RTIversion FED_RELIABLE FED_RECEIVE)
   )


  )
)
;; interactions
(interactions
 (class Detonation FED_BEST_EFFORT FED_RECEIVE
 ;;(class Detonation FED_BEST_EFFORT FED_TIMESTAMP
  (parameter Munition_ID_site)
  (parameter Munition_ID_application)
  (parameter Munition_ID_entity)
  (parameter Attacker_ID_site)
  (parameter Attacker_ID_application)
  (parameter Attacker_ID_entity)
  (parameter Target_ID_site)
  (parameter Target_ID_application)
  (parameter Target_ID_entity)
  (parameter Event_ID_site)
  (parameter Event_ID_application)
  (parameter Event_ID_entity)
  (parameter WorldLocationX)
  (parameter WorldLocationY)
  (parameter WorldLocationZ)
  (parameter EntityLocationX)
  (parameter EntityLocationY)
  (parameter EntityLocationZ)
  (parameter VelocityX)
  (parameter VelocityY)
  (parameter VelocityZ)
  (parameter BurstKind)
  (parameter BurstDomain)
  (parameter BurstCountry)
  (parameter BurstCategory)
```

```
 (parameter BurstSubcategory)
 (parameter BurstSpecific)
 (parameter BurstExtra)
 (parameter BurstWarhead)
 (parameter BurstFuze)
 (parameter BurstQuantity)
 (parameter BurstRate)
 (parameter DetonationResult)
 (parameter TimeStamp)
 (parameter ArticulatedStruct)
)
;;(class Fire FED_BEST_EFFORT FED_TIMESTAMP
(class Fire FED_BEST_EFFORT FED_RECEIVE
 (parameter Launch_Platform_ID_site )
 (parameter Launch_Platform_ID_application )
 (parameter Launch_Platform_ID_entity )
 (parameter Target_ID_site )
 (parameter Target_ID_application )
 (parameter Target_ID_entity )
 (parameter Weapon_ID_site )
 (parameter Weapon_ID_application )
 (parameter Weapon_ID_entity )
 (parameter Event_ID_site )
 (parameter Event_ID_application )
 (parameter Event_ID_entity )
 (parameter LocationX )
 (parameter LocationY )
 (parameter LocationZ )
 (parameter VelocityX )
 (parameter VelocityY )
 (parameter VelocityZ )
 (parameter Range)
 (parameter BurstKind)
 (parameter BurstDomain)
 (parameter BurstCountry)
 (parameter BurstCategory)
 (parameter BurstSubcategory)
 (parameter BurstSpecific)
 (parameter BurstExtra)
 (parameter BurstWarhead)
 (parameter BurstFuze)
 (parameter BurstQuantity)
 (parameter BurstRate)
 (parameter TimeStamp)
)
(class Signal FED_BEST_EFFORT FED_RECEIVE
 (parameter TimeStamp)
 (parameter EntitySite)
 (parameter EntityHost)
 (parameter EntityId)
 (parameter RadioId)
 (parameter EncodingScheme)
 (parameter TdlType)
 (parameter SampleRate)
 (parameter Samples)
```

```
  (parameter DataStruct)
)
(class Collision FED_BEST_EFFORT FED_RECEIVE
  (parameter TimeStamp)
  (parameter IssuingEntitySite)
  (parameter IssuingEntityHost)
  (parameter IssuingEntityId)
  (parameter CollidingEntitySite)
  (parameter CollidingEntityHost)
  (parameter CollidingEntityId)
  (parameter EventIdSite)
  (parameter EventIdHost)
  (parameter EventIdEvent)
  (parameter VelocityX)
  (parameter VelocityY)
  (parameter VelocityZ)
  (parameter Mass)
  (parameter RelLocationX)
  (parameter RelLocationY)
  (parameter RelLocationZ)
)

(class Manager FED_RELIABLE FED_RECEIVE
  (class Federate FED_RELIABLE FED_RECEIVE
    (parameter FromFederate)
    (class Alert FED_RELIABLE FED_RECEIVE
      (parameter AlertSeverity)
      (parameter AlertText)
      (parameter AlertID)
    )
    (class ServiceLog FED_RELIABLE FED_RECEIVE
      (parameter ServiceName)
      (parameter ServiceInitiator)
      (class ServiceLogArguments FED_RELIABLE FED_RECEIVE
        (parameter Handle1)
        (parameter Handle2)
        (parameter HandleSet)
        (parameter ObjectIDorCount)
        (parameter TagOrLabelOrName)
        (parameter Time)
        (parameter Enumeration)
        (parameter Boolean)
      )
    )
    (class ObjectInformation FED_RELIABLE FED_RECEIVE
      (parameter ObjectID)
      (parameter LockedAttributes)
      (parameter RegisteredClass)
      (parameter RepresentedClass)
    )
    (class PublishingClass FED_RELIABLE FED_RECEIVE
      (parameter ObjectClass)
      ;; format = ClassHandle:attrHandle,attrHandle,...,attrHandle
      (parameter InteractionClass)
      ;; format = ClassHandle
```

```
  )
  (class SubscribingClass FED_RELIABLE FED_RECEIVE
   (parameter ObjectClass)
   ;; format = ClassHandle:attrHandle,attrHandle,...,attrHandle
   (parameter InteractionClass) .
   ;; format = ClassHandle
  )
  (class Action FED_RELIABLE FED_RECEIVE
   (parameter ToFederate)
   (class RequestPublicationTree FED_RELIABLE FED_RECEIVE
   )
   (class RequestSubscriptionTree FED_RELIABLE FED_RECEIVE
   )
   (class SetTiming FED_RELIABLE FED_RECEIVE
    (parameter FedReportPeriod)
    (parameter TimeReportPeriod)
    (parameter ObjectReportPeriod)
   )
   (class RequestObjectInformation FED_RELIABLE FED_RECEIVE
    (parameter ObjectID)
   )
   (class ModifyAttributeState FED_RELIABLE FED_RECEIVE
    (parameter ObjectID)
    (parameter AttributeID)
    (parameter TokenState)
   )
   (class RemoteServiceInvocation FED_RELIABLE FED_RECEIVE
    (class DoResignFederationExecution FED_RELIABLE FED_RECEIVE
     (parameter ResignAction)
    )
    (class DoDeleteObject FED_RELIABLE FED_RECEIVE
     (parameter ObjectID)
     (parameter Time)
     (parameter Tag)
    )
    (class DoSetLookahead FED_RELIABLE FED_RECEIVE
     (parameter Lookahead)
    )
    (class DoSetTimeConstrained FED_RELIABLE FED_RECEIVE
     (parameter State)
    )
    (class DoTurnRegulationOn FED_RELIABLE FED_RECEIVE
    )
    (class DoTurnRegulationOff FED_RELIABLE FED_RECEIVE
    )
   )
   (class Control FED_RELIABLE FED_RECEIVE
    (parameter SetServiceLogging)
    (parameter SetLogFile)
    (parameter DeleteObject)
    (parameter DequeueFIFO)
   )
  )
 )
)
```

Object/Interaction Table

NOTE: Complete one of these tables for each Federate

Federate,
dis Federate

					If Update = " y"					If Subscribe = y?	Ownership	
Object/ Interaction Class	**Attribute/ Parameter**	**Count**	**Size**	**Update?** (y or n)	**Update Rate** # updates/unit time	**Update Grouping** (Assign same letter to attributes which will all be updated at the same time)	**Transport** R= Reliable B= Best Effort	**Ordering** TSO or FIFO	**Subscribe?** (y or n)	**Maximum tolerable latency from any source** (milliseconds)	**Attribute Ownership Transfer Rate** # times/unit time	**Ownership Transfer Grouping** (Assign same letter to attributes which will all be transferred together)
Entity	FORCE_ID		2 bytes	n					n			
	ENTITYID_SITE		2 bytes	n					n			
	ENTITYID_APPLICATION		2 bytes	n					n			
	ENTITYID_ENTITY		2 bytes	n					n			
	LOCATION_X		8 bytes	n					n			
	LOCATION_Y		8 bytes	n					n			
	LOCATION_Z		8 bytes	n					n			
	VELOCITY_X		4 bytes	n					n			
	VELOCITY_Y		4 bytes	n					n			
	VELOCITY_Z		4 bytes	n					n			
	ACCELERATION_X		4 bytes	n					n			
	ACCELERATION_Y		4 bytes	n					n			
	ACCELERATION_Z		4 bytes	n					n			
	ORIENTATION_PSI		4 bytes	n					n			
	ORIENTATION_THETA		4 bytes	n					n			
	ORIENTATION_PHI		4 bytes	n					n			
	APPEARANCE_PAINT_SCHEME		2 bytes	n					n			
Platform	DAMAGE_STATE_APPEARANCE		2 bytes	n					n			
Munition	ANGULAR_VELOCITY_PSI		4 bytes	n					n			
	ANGULAR_VELOCITY_THETA		4 bytes	n					n			
	ANGULAR_VELOCITY_PHI		4 bytes	n					n			
	FIRING_ID		2 bytes	n					n			
Ground Vehicle	APPEARANCE_SMOKE		2 bytes	n					n			
	APPEARANCE_TRAILING		2 bytes	n					n			
	APPEARANCE_HATCH		2 bytes	n					n			
	APPEARANCE_LIGHTS		2 bytes	n					n			
	APPEARANCE_FLAMING		2 bytes	n					n			
	DAMAGE_STATE_MOBILITY		2 bytes	n					n			
	DAMAGE_STATE_FIRE_POWER		2 bytes	n					n			
Air Vehicle	ANGULAR_VELOCITY_PSI		4 bytes	n					n			
	ANGULAR_VELOCITY_THETA		4 bytes	n					n			
	ANGULAR_VELOCITY_PHI		4 bytes	n					n			

Tracked	TURRET_HEADING		4 bytes	n					n			
	MAIN_GUN_PITCH		4 bytes	n					n			
	COMMANDERS_GUN_PITCH		4 bytes	n					n			
	MACHINE_GUN_PITCH		4 bytes	n					n			
Armored Fighing Vehicle												
	APPEARANCE_LAUNCHER		2 bytes	n					n			
DETONATION	Munition_ID		2 bytes	y		A	R	FIFO	y			
	Target_ID		2 bytes	y		A	R	FIFO	y			
	Firing_ID		2 bytes	y		A	R	FIFO	y			
	Location_X		8 bytes	y	15	A	R	FIFO	y			
	Location_Y		8 bytes	y	15	A	R	FIFO	y			
	Location_Z		8 bytes	y	15	A	R	FIFO	y			
	Relative_Location_X		4 bytes	y	15	A	R	FIFO	y			
	Relative_Location_Y		4 bytes	y	15	A	R	FIFO	y			
	Relative_Location_Z		4 bytes	y	15	A	R	FIFO	y			
	Velocity_X		4 bytes	y	15	A	R	FIFO	y			
	Velocity_Y		4 bytes	y	15	A	R	FIFO	y			
	Velocity_Z		4 bytes	y	15	A	R	FIFO	y			
	Burst_Descriptor_Warhead		2 bytes	y	15	A	R	FIFO	y			
	Burst_Descriptor_Fuze		2 bytes	y	15	A	R	FIFO	y			
	Burst_Descriptor_Detonation_Result		2 bytes	y	15	A	R	FIFO	y			
	Quantity		2 bytes	y	15	A	R	FIFO	y			
	Rate		2 bytes	y	15	A	R	FIFO	y			
WEAPON FIRE	Firing_ID		2 bytes	y		B	R	FIFO	y			
	Target_ID		2 bytes	y		B	R	FIFO	y			
	Location_X		8 bytes	y	15	B	R	FIFO	y			
	Location_Y		8 bytes	y	15	B	R	FIFO	y			
	Location_Z		8 bytes	y	15	B	R	FIFO	y			
	Velocity_X		8 bytes	y	15	B	R	FIFO	y			
	Velocity_Y		8 bytes	y	15	B	R	FIFO	y			
	Velocity_Z		8 bytes	y	15	B	R	FIFO	y			
	Munition_Type		2 bytes	y	15	B	R	FIFO	y			
	Quantity		2 bytes	y	15	B	R	FIFO	y			
	Rate		2 bytes	y	15	B	R	FIFO	y			
WEAPON LAUNCH												
	Launch_Platform_ID		2 bytes	y		C	R	FIFO	y			
	Munition_ID		2 bytes	y		C	R	FIFO	y			
	Target_ID		2 bytes	y		C	R	FIFO	y			
	Location_X		8 bytes	y	15	C	R	FIFO	y			
	Location_Y		8 bytes	y	15	C	R	FIFO	y			
	Location_Z		8 bytes	y	15	C	R	FIFO	y			
	Velocity_X		8 bytes	y	15	C	R	FIFO	y			
	Velocity_Y		8 bytes	y	15	C	R	FIFO	y			
	Velocity_Z		8 bytes	y	15	C	R	FIFO	y			
COLLISION	Issuing_ID		2 bytes	y		D	R	FIFO	y			
	Colliding_ID		2 bytes	y		D	R	FIFO	y			

	Mass		2 bytes	y	15	D	R	FIFO	y			
	Relative_Location_X		4 bytes	y	15	D	R	FIFO	y			
	Relative_Location_Y		4 bytes	y	15	D	R	FIFO	y			
	Relative_Location_Z		4 bytes	y	15	D	R	FIFO	y			
	Velocity_X		4 bytes	y	15	D	R	FIFO	y			
	Velocity_Y		4 bytes	y	15	D	R	FIFO	y			
	Velocity_Z		4 bytes	y	15	D	R	FIFO	y			
AIR VEHICLE LAUNCH												
	Launch_Platform_ID		2 bytes	n					n			
	Air_Vehicle_ID		2 bytes	n					n			
	Location_X		4 bytes	n					n			
	Location_Y		4 bytes	n					n			
	Location_Z		4 bytes	n					n			
	Velocity_X		4 bytes	n					n			
	Velocity_Y		4 bytes	n					n			
	Velocity_Z		4 bytes	n					n			

Object/Interaction Table

NOTE: Complete one of these tables for each Federate

Federate,
edb Federate

					If Update = "y"					If Subscribe = y?	Ownership	
Object/ Interaction Class	**Attribute/ Parameter**	**Count**	**Size**	**Update? (y or n)**	**Update Rate # updates/unit time**	**Update Grouping (Assign same letter to attributes which will all be updated at the same time)**	**Transport R= Reliable B= Best Effort**	**Ordering TSO or FIFO**	**Subscribe? (y or n)**	**Maximum tolerable latency from any source (milliseconds)**	**Attribute Ownership Transfer Rate # times/unit time**	**Ownership Transfer Grouping (Assign same letter to attributes which will all be transferred together)**
Entity	FORCE_ID		2 bytes	y		A	R	FIFO	y			
	ENTITYID_SITE		2 bytes	y		A	R	FIFO	y			
	ENTITYID_APPLICATION		2 bytes	y		A	R	FIFO	y			
	ENTITYID_ENTITY		2 bytes	y		A	R	FIFO	y			
	LOCATION_X		8 bytes	y	15	A	R	FIFO	y			
	LOCATION_Y		8 bytes	y	15	A	R	FIFO	y			
	LOCATION_Z		8 bytes	y	15	A	R	FIFO	y			
	VELOCITY_X		4 bytes	y	15	A	R	FIFO	y			
	VELOCITY_Y		4 bytes	y	15	A	R	FIFO	y			
	VELOCITY_Z		4 bytes	y	15	A	R	FIFO	y			
	ACCELERATION_X		4 bytes	y	15	A	R	FIFO	y			
	ACCELERATION_Y		4 bytes	y	15	A	R	FIFO	y			
	ACCELERATION_Z		4 bytes	y	15	A	R	FIFO	y			
	ORIENTATION_PSI		4 bytes	y	15	A	R	FIFO	y			
	ORIENTATION_THETA		4 bytes	y	15	A	R	FIFO	y			
	ORIENTATION_PHI		4 bytes	y	15	A	R	FIFO	y			
	APPEARANCE_PAINT_SCHEME		2 bytes	y	15	B	R	FIFO	y			
Platform	DAMAGE_STATE_APPEARANCE		2 bytes	y	15	B	R	FIFO	y			A
Munition	ANGULAR_VELOCITY_PSI		4 bytes	y	15	C	R	FIFO	y			
	ANGULAR_VELOCITY_THETA		4 bytes	y	15	C	R	FIFO	y			
	ANGULAR_VELOCITY_PHI		4 bytes	y	15	C	R	FIFO	y			
	FIRING_ID		2 bytes	n					y			
Ground Vehicle	APPEARANCE_SMOKE		2 bytes	y	15	B	R	FIFO	y			
	APPEARANCE_TRAILING		2 bytes	y	15	B	R	FIFO	y			
	APPEARANCE_HATCH		2 bytes	y	15	B	R	FIFO	y			
	APPEARANCE_LIGHTS		2 bytes	y	15	B	R	FIFO	y			
	APPEARANCE_FLAMING		2 bytes	y	15	B	R	FIFO	y			
	DAMAGE_STATE_MOBILITY		2 bytes	y	15	B	R	FIFO	y			A
	DAMAGE_STATE_FIRE_POWER		2 bytes	y	15	B	R	FIFO	y			A
Air Vehicle	ANGULAR_VELOCITY_PSI		4 bytes	y	15	D	R	FIFO	y			
	ANGULAR_VELOCITY_THETA		4 bytes	y	15	D	R	FIFO	y			
	ANGULAR_VELOCITY_PHI		4 bytes	y	15	D	R	FIFO	y			

Tracked	TURRET_HEADING		4 bytes	y	15	D	R	FIFO	y			
	MAIN_GUN_PITCH		4 bytes	y	15	D	R	FIFO	y			
	COMMANDERS_GUN_PITCH		4 bytes	y	15	D	R	FIFO	y			
	MACHINE_GUN_PITCH		4 bytes	y	15	D	R	FIFO	y			
Armored Fighing Vehicle												
	APPEARANCE_LAUNCHER		2 bytes	y	15	E	R	FIFO	y			
DETONATION	Munition_ID		2 bytes	n					n			
	Target_ID		2 bytes	n					n			
	Firing_ID		2 bytes	n					n			
	Location_X		8 bytes	n					n			
	Location_Y		8 bytes	n					n			
	Location_Z		8 bytes	n					n			
	Relative_Location_X		4 bytes	n					n			
	Relative_Location_Y		4 bytes	n					n			
	Relative_Location_Z		4 bytes	n					n			
	Velocity_X		4 bytes	n					n			
	Velocity_Y		4 bytes	n					n			
	Velocity_Z		4 bytes	n					n			
	Burst_Descriptor_Warhead		2 bytes	n					n			
	Burst_Descriptor_Fuze		2 bytes	n					n			
	Burst_Descriptor_Detonation_Result		2 bytes	n					n			
	Quantity		2 bytes	n					n			
	Rate		2 bytes	n					n			
WEAPON FIRE	Firing_ID		2 bytes	n					n			
	Target_ID		2 bytes	n					n			
	Location_X		8 bytes	n					n			
	Location_Y		8 bytes	n					n			
	Location_Z		8 bytes	n					n			
	Velocity_X		8 bytes	n					n			
	Velocity_Y		8 bytes	n					n			
	Velocity_Z		8 bytes	n					n			
	Munition_Type		2 bytes	n					n			
	Quantity		2 bytes	n					n			
	Rate		2 bytes	n					n			
WEAPON LAUNCH												
	Launch_Platform_ID		2 bytes	n					n			
	Munition_ID		2 bytes	n					n			
	Target_ID		2 bytes	n					n			
	Location_X		8 bytes	n					n			
	Location_Y		8 bytes	n					n			
	Location_Z		8 bytes	n					n			
	Velocity_X		8 bytes	n					n			
	Velocity_Y		8 bytes	n					n			
	Velocity_Z		8 bytes	n					n			
COLLISION	Issuing_ID		2 bytes	n					n			
	Colliding_ID		2 bytes	n					n			

	Mass		2 bytes	n					n			
	Relative_Location_X		4 bytes	n					n			
	Relative_Location_Y		4 bytes	n					n			
	Relative_Location_Z		4 bytes	n					n			
	Velocity_X		4 bytes	n					n			
	Velocity_Y		4 bytes	n					n			
	Velocity_Z		4 bytes	n					n			
AIR VEHICLE LAUNCH													
	Launch_Platform_ID		2 bytes	n					n			
	Air_Vehicle_ID		2 bytes	n					n			
	Location_X		4 bytes	n					n			
	Location_Y		4 bytes	n					n			
	Location_Z		4 bytes	n					n			
	Velocity_X		4 bytes	n					n			
	Velocity_Y		4 bytes	n					n			
	Velocity_Z		4 bytes	n					n			

Object/Interaction Table

NOTE: Complete one of these tables for each Federate

Federate ,
Damage Assessment Federate

					If Update =" y"					If Subscribe = y?	Ownership	
Object/ Interaction Class	**Attribute/ Parameter**	**Count**	**Size**	**Update?** (y or n)	**Update Rate** # updates/unit time	**Update Grouping** (Assign same letter to attributes which will all be updated at the same time)	**Transport** R= Reliable B= Best Effort	**Ordering** TSO or FIFO	**Subscribe?** (y or n)	**Maximum tolerable latency from any source** (milliseconds)	**Attribute Ownership Transfer Rate** # times/unit time	**Ownership Transfer Grouping** (Assign same letter to attributes which will all be transferred together)
Entity	FORCE_ID		2 bytes	n					n			
	ENTITYID_SITE		2 bytes	n					y			
	ENTITYID_APPLICATION		2 bytes	n					y			
	ENTITYID_ENTITY		2 bytes	y		A	R	FIFO	y			
	LOCATION_X		8 bytes	n					n			
	LOCATION_Y		8 bytes	n					n			
	LOCATION_Z		8 bytes	n					n			
	VELOCITY_X		4 bytes	n					n			
	VELOCITY_Y		4 bytes	n					n			
	VELOCITY_Z		4 bytes	n					n			
	ACCELERATION_X		4 bytes	n					n			
	ACCELERATION_Y		4 bytes	n					n			
	ACCELERATION_Z		4 bytes	n					n			
	ORIENTATION_PSI		4 bytes	n					n			
	ORIENTATION_THETA		4 bytes	n					n			
	ORIENTATION_PHI		4 bytes	n					n			
	APPEARANCE_PAINT_SCHEME		2 bytes	n					n			
Platform	DAMAGE_STATE_APPEARANCE		2 bytes	y	15	B	R	FIFO	y		n/a	A
Munition	ANGULAR_VELOCITY_PSI		4 bytes	n					n			
	ANGULAR_VELOCITY_THETA		4 bytes	n					n			
	ANGULAR_VELOCITY_PHI		4 bytes	n					n			
	FIRING_ID		2 bytes	n					n			
Ground Vehicle	APPEARANCE_SMOKE		2 bytes	n					n			
	APPEARANCE_TRAILING		2 bytes	n					n			
	APPEARANCE_HATCH		2 bytes	n					n			
	APPEARANCE_LIGHTS		2 bytes	n					n			
	APPEARANCE_FLAMING		2 bytes	n					n			
	DAMAGE_STATE_MOBILITY		2 bytes	y	15	B	R	FIFO	y		n/a	A
	DAMAGE_STATE_FIRE_POWER		2 bytes	y	15	B	R	FIFO	y		n/a	A
Air Vehicle	ANGULAR_VELOCITY_PSI		4 bytes	n					n			
	ANGULAR_VELOCITY_THETA		4 bytes	n					n			
	ANGULAR_VELOCITY_PHI		4 bytes	n					n			

Tracked	TURRET_HEADING		4 bytes	n					n			
	MAIN_GUN_PITCH		4 bytes	n					n			
	COMMANDERS_GUN_PITCH		4 bytes	n					n			
	MACHINE_GUN_PITCH		4 bytes	n					n			
Armored Fighing Vehicle												
	APPEARANCE_LAUNCHER		2 bytes	n					n			
DETONATION	Munition_ID		2 bytes	n					y			
	Target_ID		2 bytes	n					y			
	Firing_ID		2 bytes	n					y			
	Location_X		8 bytes	n					y			
	Location_Y		8 bytes	n					y			
	Location_Z		8 bytes	n					y			
	Relative_Location_X		4 bytes	n					y			
	Relative_Location_Y		4 bytes	n					y			
	Relative_Location_Z		4 bytes	n					y			
	Velocity_X		4 bytes	n					y			
	Velocity_Y		4 bytes	n					y			
	Velocity_Z		4 bytes	n					y			
	Burst_Descriptor_Warhead		2 bytes	n					y			
	Burst_Descriptor_Fuze		2 bytes	n					y			
	Burst_Descriptor_Detonation_Result		2 bytes	n					y			
	Quantity		2 bytes	n					y			
	Rate		2 bytes	n					y			
WEAPON FIRE	Firing_ID		2 bytes	n					y			
	Target_ID		2 bytes	n					y			
	Location_X		8 bytes	n					y			
	Location_Y		8 bytes	n					y			
	Location_Z		8 bytes	n					y			
	Velocity_X		8 bytes	n					y			
	Velocity_Y		8 bytes	n					y			
	Velocity_Z		8 bytes	n					y			
	Munition_Type		2 bytes	n					y			
	Quantity		2 bytes	n					y			
	Rate		2 bytes	n					y			
WEAPON LAUNCH												
	Launch_Platform_ID		2 bytes	n					n			
	Munition_ID		2 bytes	n					n			
	Target_ID		2 bytes	n					n			
	Location_X		8 bytes	n					n			
	Location_Y		8 bytes	n					n			
	Location_Z		8 bytes	n					n			
	Velocity_X		8 bytes	n					n			
	Velocity_Y		8 bytes	n					n			
	Velocity_Z		8 bytes	n					n			
COLLISION	Issuing_ID		2 bytes	n					n			
	Colliding_ID		2 bytes	n					n			

	Mass		2 bytes	n					n				
	Relative_Location_X		4 bytes	n					n				
	Relative_Location_Y		4 bytes	n					n				
	Relative_Location_Z		4 bytes	n					n				
	Velocity_X		4 bytes	n					n				
	Velocity_Y		4 bytes	n					n				
	Velocity_Z		4 bytes	n					n				
AIR VEHICLE LAUNCH													
	Launch_Platform_ID		2 bytes	n					n				
	Air_Vehicle_ID		2 bytes	n					n				
	Location_X		4 bytes	n					n				
	Location_Y		4 bytes	n					n				
	Location_Z		4 bytes	n					n				
	Velocity_X		4 bytes	n					n				
	Velocity_Y		4 bytes	n					n				
	Velocity_Z		4 bytes	n					n				

```
.//FOMAT FILE
 (cctt_federation
  (class M1
    (attribute FORCE_ID integer32)
    (attribute ENTITYID_SITE integer32 )
    (attribute ENTITYID_APPLICATION integer32 )
    (attribute ENTITYID_ENTITY integer32 )
    (attribute LOCATION_X integer32 )
    (attribute LOCATION_Y integer32 )
    (attribute LOCATION_Z integer32 )
    (attribute VELOCITY_X integer32 )
    (attribute VELOCITY_Y integer32 )
    (attribute VELOCITY_Z integer32 )
    (attribute ACCELERATION_X integer32 )
    (attribute ACCELERATION_Y integer32 )
    (attribute ACCELERATION_Z integer32 )
    (attribute ORIENTATION_PSI integer32 )
    (attribute ORIENTATION_THETA integer32 )
    (attribute ORIENTATION_PHI integer32 )
    (attribute APPEARANCE_PAINT_SCHEME integer32 )
    (attribute DAMAGE_STATE_APPEARANCE integer32 )
    (attribute APPEARANCE_SMOKE integer32 )
    (attribute APPEARANCE_TRAILING integer32 )
    (attribute APPEARANCE_HATCH integer32 )
    (attribute APPEARANCE_LIGHTS integer32 )
    (attribute APPEARANCE_FLAMING integer32 )
    (attribute DAMAGE_STATE_MOBILITY integer32 )
    (attribute DAMAGE_STATE_FIRE_POWER integer32 )
  )
  (class T72
    (attribute FORCE_ID integer32)
    (attribute ENTITYID_SITE integer32 )
    (attribute ENTITYID_APPLICATION integer32 )
    (attribute ENTITYID_ENTITY integer32 )
    (attribute LOCATION_X integer32 )
    (attribute LOCATION_Y integer32 )
    (attribute LOCATION_Z integer32 )
    (attribute VELOCITY_X integer32 )
    (attribute VELOCITY_Y integer32 )
    (attribute VELOCITY_Z integer32 )
    (attribute ACCELERATION_X integer32 )
    (attribute ACCELERATION_Y integer32 )
    (attribute ACCELERATION_Z integer32 )
    (attribute ORIENTATION_PSI integer32 )
    (attribute ORIENTATION_THETA integer32 )
    (attribute ORIENTATION_PHI integer32 )
    (attribute APPEARANCE_PAINT_SCHEME integer32 )
    (attribute DAMAGE_STATE_APPEARANCE integer32 )
    (attribute APPEARANCE_SMOKE integer32 )
    (attribute APPEARANCE_TRAILING integer32 )
    (attribute APPEARANCE_HATCH integer32 )
    (attribute APPEARANCE_LIGHTS integer32 )
    (attribute APPEARANCE_FLAMING integer32 )
    (attribute DAMAGE_STATE_MOBILITY integer32 )
    (attribute DAMAGE_STATE_FIRE_POWER integer32 )
  )
```

```
 (interaction DETONATION
    (parameter Munition_ID integer32)
    (parameter Target_ID integer32)
    (parameter Firing_ID integer32)
    (parameter Location_X integer32)
    (parameter Location_Y integer32)
    (parameter Location_Z integer32)
    (parameter Relative_Location_X integer32)
    (parameter Relative_Location_Y integer32)
    (parameter Relative_Location_Z integer32)
    (parameter Velocity_X integer32)
    (parameter Velocity_Y integer32)
    (parameter Velocity_Z integer32)
    (parameter Burst_Descriptor_Warhead integer32)
    (parameter Burst_Descriptor_Fuze integer32)
    (parameter Burst_Descriptor_Detonation_Result integer32)
    (parameter Quantity integer32)
    (parameter Rate integer32)
 )
 (interaction WEAPON_FIRE
    (parameter Firing_ID integer32)
    (parameter Target_ID integer32)
    (parameter Location_X integer32)
    (parameter Location_Y integer32)
    (parameter Location_Z integer32)
    (parameter Velocity_X integer32)
    (parameter Velocity_Y integer32)
    (parameter Velocity_Z integer32)
    (parameter Munition_Type integer32)
    (parameter Quantity integer32)
    (parameter Rate integer32)
 )
)

(specialty_federation
 (class M1
    (attribute FORCE_ID integer32)
    (attribute ENTITYID_SITE integer32 )
    (attribute ENTITYID_APPLICATION integer32 )
    (attribute ENTITYID_ENTITY integer32 )
    (attribute LOCATION_X integer32 )
    (attribute LOCATION_Y integer32 )
    (attribute LOCATION_Z integer32 )
    (attribute VELOCITY_X integer32 )
    (attribute VELOCITY_Y integer32 )
    (attribute VELOCITY_Z integer32 )
    (attribute ACCELERATION_X integer32 )
    (attribute ACCELERATION_Y integer32 )
    (attribute ACCELERATION_Z integer32 )
    (attribute ORIENTATION_PSI integer32 )
    (attribute ORIENTATION_THETA integer32 )
    (attribute ORIENTATION_PHI integer32 )
    (attribute APPEARANCE_PAINT_SCHEME integer32 )
    (attribute DAMAGE_STATE_APPEARANCE integer32 )
    (attribute APPEARANCE_SMOKE integer32 )
    (attribute APPEARANCE_TRAILING integer32 )
```

```
    (attribute APPEARANCE_HATCH integer32 )
    (attribute APPEARANCE_LIGHTS integer32 )
    (attribute APPEARANCE_FLAMING integer32 )
    (attribute DAMAGE_STATE_MOBILITY integer32 )
    (attribute DAMAGE_STATE_FIRE_POWER integer32 )
  )
  (class T72
    (attribute FORCE_ID integer32)
    (attribute ENTITYID_SITE integer32 )
    (attribute ENTITYID_APPLICATION integer32 )
    (attribute ENTITYID_ENTITY integer32 )
    (attribute LOCATION_X integer32 )
    (attribute LOCATION_Y integer32 )
    (attribute LOCATION_Z integer32 )
    (attribute VELOCITY_X integer32 )
    (attribute VELOCITY_Y integer32 )
    (attribute VELOCITY_Z integer32 )
    (attribute ACCELERATION_X integer32 )
    (attribute ACCELERATION_Y integer32 )
    (attribute ACCELERATION_Z integer32 )
    (attribute ORIENTATION_PSI integer32 )
    (attribute ORIENTATION_THETA integer32 )
    (attribute ORIENTATION_PHI integer32 )
    (attribute APPEARANCE_PAINT_SCHEME integer32 )
    (attribute DAMAGE_STATE_APPEARANCE integer32 )
    (attribute APPEARANCE_SMOKE integer32 )
    (attribute APPEARANCE_TRAILING integer32 )
    (attribute APPEARANCE_HATCH integer32 )
    (attribute APPEARANCE_LIGHTS integer32 )
    (attribute APPEARANCE_FLAMING integer32 )
    (attribute DAMAGE_STATE_MOBILITY integer32 )
    (attribute DAMAGE_STATE_FIRE_POWER integer32 )
  )
  (interaction DETONATION
    (parameter Munition_ID integer32)
    (parameter Target_ID integer32)
    (parameter Firing_ID integer32)
    (parameter Location_X integer32)
    (parameter Location_Y integer32)
    (parameter Location_Z integer32)
    (parameter Relative_Location_X integer32)
    (parameter Relative_Location_Y integer32)
    (parameter Relative_Location_Z integer32)
    (parameter Velocity_X integer32)
    (parameter Velocity_Y integer32)
    (parameter Velocity_Z integer32)
    (parameter Burst_Descriptor_Warhead integer32)
    (parameter Burst_Descriptor_Fuze integer32)
    (parameter Burst_Descriptor_Detonation_Result integer32)
    (parameter Quantity integer32)
    (parameter Rate integer32)
  )
  (interaction WEAPON_FIRE
    (parameter Firing_ID integer32)
    (parameter Target_ID integer32)
    (parameter Location_X integer32)
```

```
    (parameter Location_Y integer32)
    (parameter Location_Z integer32)
    (parameter Velocity_X integer32)
    (parameter Velocity_Y integer32)
    (parameter Velocity_Z integer32)
    (parameter Munition_Type integer32)
    (parameter Quantity integer32)
    (parameter Rate integer32)
  )
)

object_mappings
(
  ((cctt_federation M1 (FORCE_ID)) (specialty_federation M1 (FORCE_ID)) identity N)
  ((cctt_federation M1 (ENTITYID_SITE )) (specialty_federation M1 (ENTITYID_SITE ))identity N)
  ((cctt_federation M1 (ENTITYID_APPLICATION )) (specialty_federation M1
(ENTITYID_APPLICATION )) identity N)
  ((cctt_federation M1 (ENTITYID_ENTITY)) (specialty_federation M1 (ENTITYID_ENTITY)) identity
N)
  ((cctt_federation M1 (LOCATION_X)) (specialty_federation M1 (LOCATION_X)) identity N)
  ((cctt_federation M1 (LOCATION_Y)) (specialty_federation M1 (LOCATION_Y)) identity N)
  ((cctt_federation M1 (LOCATION_Z)) (specialty_federation M1 (LOCATION_Z)) identity N)
  ((cctt_federation M1 (VELOCITY_X)) (specialty_federation M1 (VELOCITY_X)) identity N)
  ((cctt_federation M1 (VELOCITY_Y)) (specialty_federation M1 (VELOCITY_Y)) identity N)
  ((cctt_federation M1 (VELOCITY_Z)) (specialty_federation M1 (VELOCITY_Z)) identity N)
  ((cctt_federation M1 (ACCELERATION_X)) (specialty_federation M1 (ACCELERATION_X)) identity
N)
  ((cctt_federation M1 (ACCELERATION_Y)) (specialty_federation M1 (ACCELERATION_Y)) identity
N)
  ((cctt_federation M1 (ACCELERATION_Z)) (specialty_federation M1 (ACCELERATION_Z)) identity
N)
  ((cctt_federation M1 (ORIENTATION_PSI)) (specialty_federation M1 (ORIENTATION_PSI))  identity
N)
  ((cctt_federation M1 (ORIENTATION_THETA))  (specialty_federation M1
(ORIENTATION_THETA))identity N)
  ((cctt_federation M1 (ORIENTATION_PHI)) (specialty_federation M1 (ORIENTATION_PHI))
identity N)
  ((cctt_federation M1 (APPEARANCE_PAINT_SCHEME ))(specialty_federation M1
(APPEARANCE_PAINT_SCHEME )) identity N)
  ((cctt_federation M1 (DAMAGE_STATE_APPEARANCE ))(specialty_federation M1
(DAMAGE_STATE_APPEARANCE )) identity N)
  ((cctt_federation M1 (APPEARANCE_SMOKE ))(specialty_federation M1 (APPEARANCE_SMOKE ))
identity N)
  ((cctt_federation M1 (APPEARANCE_TRAILING ))(specialty_federation M1
(APPEARANCE_TRAILING )) identity N)
  ((cctt_federation M1 (APPEARANCE_HATCH )) (specialty_federation M1 (APPEARANCE_HATCH
))identity N)
  ((cctt_federation M1 (APPEARANCE_LIGHTS ))(specialty_federation M1 (APPEARANCE_LIGHTS ))
identity N)
  ((cctt_federation M1 (APPEARANCE_FLAMING )) (specialty_federation M1
(APPEARANCE_FLAMING ))identity N)
  ((cctt_federation M1 (DAMAGE_STATE_MOBILITY)) (specialty_federation M1
(DAMAGE_STATE_MOBILITY))identity N)
  ((cctt_federation M1 (DAMAGE_STATE_FIRE_POWER))(specialty_federation M1
(DAMAGE_STATE_FIRE_POWER)) identity N)
  ((specialty_federation M1 (FORCE_ID)) (cctt_federation M1 (FORCE_ID)) identity N)
```

((specialty_federation M1 (ENTITYID_SITE)) (cctt_federation M1 (ENTITYID_SITE))identity N)
((specialty_federation M1 (ENTITYID_APPLICATION)) (cctt_federation M1 (ENTITYID_APPLICATION)) identity N)
((specialty_federation M1 (ENTITYID_ENTITY)) (cctt_federation M1 (ENTITYID_ENTITY)) identity N)
((specialty_federation M1 (LOCATION_X)) (cctt_federation M1 (LOCATION_X)) identity N)
((specialty_federation M1 (LOCATION_Y)) (cctt_federation M1 (LOCATION_Y)) identity N)
((specialty_federation M1 (LOCATION_Z)) (cctt_federation M1 (LOCATION_Z)) identity N)
((specialty_federation M1 (VELOCITY_X)) (cctt_federation M1 (VELOCITY_X)) identity N)
((specialty_federation M1 (VELOCITY_Y)) (cctt_federation M1 (VELOCITY_Y)) identity N)
((specialty_federation M1 (VELOCITY_Z)) (cctt_federation M1 (VELOCITY_Z)) identity N)
((specialty_federation M1 (ACCELERATION_X)) (cctt_federation M1 (ACCELERATION_X)) identity N)
((specialty_federation M1 (ACCELERATION_Y)) (cctt_federation M1 (ACCELERATION_Y)) identity N)
((specialty_federation M1 (ACCELERATION_Z)) (cctt_federation M1 (ACCELERATION_Z)) identity N)
((specialty_federation M1 (ORIENTATION_PSI)) (cctt_federation M1 (ORIENTATION_PSI)) identity N)
((specialty_federation M1 (ORIENTATION_THETA)) (cctt_federation M1 (ORIENTATION_THETA))identity N)
((specialty_federation M1 (ORIENTATION_PHI)) (cctt_federation M1 (ORIENTATION_PHI)) identity N)
((specialty_federation M1 (APPEARANCE_PAINT_SCHEME))(cctt_federation M1 (APPEARANCE_PAINT_SCHEME)) identity N)
((specialty_federation M1 (DAMAGE_STATE_APPEARANCE))(cctt_federation M1 (DAMAGE_STATE_APPEARANCE)) identity N)
((specialty_federation M1 (APPEARANCE_SMOKE))(cctt_federation M1 (APPEARANCE_SMOKE)) identity N)
((specialty_federation M1 (APPEARANCE_TRAILING))(cctt_federation M1 (APPEARANCE_TRAILING)) identity N)
((specialty_federation M1 (APPEARANCE_HATCH)) (cctt_federation M1 (APPEARANCE_HATCH))identity N)
((specialty_federation M1 (APPEARANCE_LIGHTS))(cctt_federation M1 (APPEARANCE_LIGHTS)) identity N)
((specialty_federation M1 (APPEARANCE_FLAMING)) (cctt_federation M1 (APPEARANCE_FLAMING))identity N)
((specialty_federation M1 (DAMAGE_STATE_MOBILITY)) (cctt_federation M1 (DAMAGE_STATE_MOBILITY))identity N)
((specialty_federation M1 (DAMAGE_STATE_FIRE_POWER))(cctt_federation M1 (DAMAGE_STATE_FIRE_POWER)) identity N)
((specialty_federation T72 (FORCE_ID))(cctt_federation T72 (FORCE_ID)) identity N)
((specialty_federation T72 (ENTITYID_SITE)) (cctt_federation T72 (ENTITYID_SITE))identity N)
((specialty_federation T72 (ENTITYID_APPLICATION)) (cctt_federation T72 (ENTITYID_APPLICATION))identity N)
((specialty_federation T72 (ENTITYID_ENTITY)) (cctt_federation T72 (ENTITYID_ENTITY))identity N)
((specialty_federation T72 (LOCATION_X)) (cctt_federation T72 (LOCATION_X)) identity N)
((specialty_federation T72 (LOCATION_Y)) (cctt_federation T72 (LOCATION_Y)) identity N)
((specialty_federation T72 (LOCATION_Z)) (cctt_federation T72 (LOCATION_Z)) identity N)
((specialty_federation T72 (VELOCITY_X)) (cctt_federation T72 (VELOCITY_X))identity N)
((specialty_federation T72 (VELOCITY_Y)) (cctt_federation T72 (VELOCITY_Y)) identity N)
((specialty_federation T72 (VELOCITY_Z)) (cctt_federation T72 (VELOCITY_Z)) identity N)
((specialty_federation T72 (ACCELERATION_X)) (cctt_federation T72 (ACCELERATION_X)) identity N)

((specialty_federation T72 (ACCELERATION_Y)) (cctt_federation T72 (ACCELERATION_Y)) identity N)
((specialty_federation T72 (ACCELERATION_Z)) (cctt_federation T72 (ACCELERATION_Z)) identity N)
((specialty_federation T72 (ORIENTATION_PSI)) (cctt_federation T72 (ORIENTATION_PSI)) identity N)
((specialty_federation T72 (ORIENTATION_THETA)) (cctt_federation T72 (ORIENTATION_THETA)) identity N)
((specialty_federation T72 (ORIENTATION_PHI)) (cctt_federation T72 (ORIENTATION_PHI)) identity N)
((specialty_federation T72 (APPEARANCE_PAINT_SCHEME)) (cctt_federation T72 (APPEARANCE_PAINT_SCHEME)) identity N)
((specialty_federation T72 (DAMAGE_STATE_APPEARANCE)) (cctt_federation T72 (DAMAGE_STATE_APPEARANCE)) identity N)
((specialty_federation T72 (APPEARANCE_SMOKE)) (cctt_federation T72 (APPEARANCE_SMOKE)) identity N)
((specialty_federation T72 (APPEARANCE_TRAILING)) (cctt_federation T72 (APPEARANCE_TRAILING)) identity N)
((specialty_federation T72 (APPEARANCE_HATCH)) (cctt_federation T72 (APPEARANCE_HATCH)) identity N)
((specialty_federation T72 (APPEARANCE_LIGHTS)) (cctt_federation T72 (APPEARANCE_LIGHTS)) identity N)
((specialty_federation T72 (APPEARANCE_FLAMING)) (cctt_federation T72 (APPEARANCE_FLAMING)) identity N)
((specialty_federation T72 (DAMAGE_STATE_MOBILITY)) (cctt_federation T72 (DAMAGE_STATE_MOBILITY)) identity N)
((specialty_federation T72 (DAMAGE_STATE_FIRE_POWER)) (cctt_federation T72 (DAMAGE_STATE_FIRE_POWER)) identity N)
((cctt_federation T72 (FORCE_ID))(specialty_federation T72 (FORCE_ID)) identity N)
((cctt_federation T72 (ENTITYID_SITE)) (specialty_federation T72 (ENTITYID_SITE))identity N)
((cctt_federation T72 (ENTITYID_APPLICATION)) (specialty_federation T72 (ENTITYID_APPLICATION))identity N)
((cctt_federation T72 (ENTITYID_ENTITY)) (specialty_federation T72 (ENTITYID_ENTITY))identity N)
((cctt_federation T72 (LOCATION_X)) (specialty_federation T72 (LOCATION_X)) identity N)
((cctt_federation T72 (LOCATION_Y)) (specialty_federation T72 (LOCATION_Y)) identity N)
((cctt_federation T72 (LOCATION_Z)) (specialty_federation T72 (LOCATION_Z)) identity N)
((cctt_federation T72 (VELOCITY_X)) (specialty_federation T72 (VELOCITY_X))identity N)
((cctt_federation T72 (VELOCITY_Y)) (specialty_federation T72 (VELOCITY_Y)) identity N)
((cctt_federation T72 (VELOCITY_Z)) (specialty_federation T72 (VELOCITY_Z)) identity N)
((cctt_federation T72 (ACCELERATION_X)) (specialty_federation T72 (ACCELERATION_X)) identity N)
((cctt_federation T72 (ACCELERATION_Y)) (specialty_federation T72 (ACCELERATION_Y)) identity N)
((cctt_federation T72 (ACCELERATION_Z)) (specialty_federation T72 (ACCELERATION_Z)) identity N)
((cctt_federation T72 (ORIENTATION_PSI)) (specialty_federation T72 (ORIENTATION_PSI)) identity N)
((cctt_federation T72 (ORIENTATION_THETA)) (specialty_federation T72 (ORIENTATION_THETA)) identity N)
((cctt_federation T72 (ORIENTATION_PHI)) (specialty_federation T72 (ORIENTATION_PHI)) identity N)
((cctt_federation T72 (APPEARANCE_PAINT_SCHEME)) (specialty_federation T72 (APPEARANCE_PAINT_SCHEME)) identity N)
((cctt_federation T72 (DAMAGE_STATE_APPEARANCE)) (specialty_federation T72 (DAMAGE_STATE_APPEARANCE)) identity N)

```
 ((cctt_federation T72 (APPEARANCE_SMOKE )) (specialty_federation T72
(APPEARANCE_SMOKE )) identity N)
 ((cctt_federation T72 (APPEARANCE_TRAILING )) (specialty_federation T72
(APPEARANCE_TRAILING )) identity N)
 ((cctt_federation T72 (APPEARANCE_HATCH )) (specialty_federation T72 (APPEARANCE_HATCH
)) identity N)
 ((cctt_federation T72 (APPEARANCE_LIGHTS )) (specialty_federation T72
(APPEARANCE_LIGHTS )) identity N)
 ((cctt_federation T72 (APPEARANCE_FLAMING )) (specialty_federation T72
(APPEARANCE_FLAMING )) identity N)
 ((cctt_federation T72 (DAMAGE_STATE_MOBILITY )) (specialty_federation T72
(DAMAGE_STATE_MOBILITY )) identity N)
 ((cctt_federation T72 (DAMAGE_STATE_FIRE_POWER)) (specialty_federation T72
(DAMAGE_STATE_FIRE_POWER)) identity N)
)

inter_mappings
(
 ((cctt_federation DETONATION (Munition_ID)) (specialty_federation DETONATION (Munition_ID))
identity )
 ((cctt_federation DETONATION (Target_ID)) (specialty_federation DETONATION (Target_ID))
identity)
 ((cctt_federation DETONATION (Firing_ID)) (specialty_federation DETONATION (Firing_ID))
identity)
 ((cctt_federation DETONATION (Location_X)) (specialty_federation DETONATION (Location_X))
identity)
 ((cctt_federation DETONATION (Location_Y)) (specialty_federation DETONATION (Location_Y))
identity)
 ((cctt_federation DETONATION (Location_Z)) (specialty_federation DETONATION (Location_Z))
identity)
 ((cctt_federation DETONATION (Velocity_X)) (specialty_federation DETONATION (Velocity_X))
identity )
 ((cctt_federation DETONATION (Velocity_Y)) (specialty_federation DETONATION (Velocity_Y))
identity )
 ((cctt_federation DETONATION (Velocity_Z)) (specialty_federation DETONATION (Velocity_Z))
identity )
 ((cctt_federation DETONATION (Relative_Location_X)) (specialty_federation DETONATION
(Relative_Location_X)) identity )
 ((cctt_federation DETONATION (Relative_Location_Y)) (specialty_federation DETONATION
(Relative_Location_Y)) identity )
 ((cctt_federation DETONATION (Relative_Location_Z)) (specialty_federation DETONATION
(Relative_Location_Z)) identity )
 ((cctt_federation DETONATION (Burst_Descriptor_Warhead)) (specialty_federation DETONATION
(Burst_Descriptor_Warhead))
identity)
 ((cctt_federation DETONATION (Burst_Descriptor_Fuze)) (specialty_federation DETONATION
(Burst_Descriptor_Fuze)) identity)
 ((cctt_federation DETONATION (Burst_Descriptor_Detonation_Result)) (specialty_federation
DETONATION
(Burst_Descriptor_Detonation_Result)) identity)
 ((cctt_federation DETONATION (Quantity)) (specialty_federation DETONATION (Quantity)) identity )
 ((cctt_federation DETONATION (Rate)) (specialty_federation DETONATION (Rate)) identity)

 ((specialty_federation DETONATION (Munition_ID)) (cctt_federation DETONATION (Munition_ID))
identity)
```

((specialty_federation DETONATION (Firing_ID)) (cctt_federation DETONATION (Firing_ID)) identity)
((specialty_federation DETONATION (Target_ID)) (cctt_federation DETONATION (Target_ID)) identity)
((specialty_federation DETONATION (Location_X)) (cctt_federation DETONATION (Location_X)) identity)
((specialty_federation DETONATION (Location_Y)) (cctt_federation DETONATION (Location_Y)) identity)
((specialty_federation DETONATION (Location_Z)) (cctt_federation DETONATION (Location_Z)) identity)
((specialty_federation DETONATION (Velocity_X)) (cctt_federation DETONATION (Velocity_X)) identity)
((specialty_federation DETONATION (Velocity_Y)) (cctt_federation DETONATION (Velocity_Y)) identity)
((specialty_federation DETONATION (Velocity_Z)) (cctt_federation DETONATION (Velocity_Z)) identity)
((specialty_federation DETONATION (Relative_Location_X)) (cctt_federation DETONATION (Relative_Location_X)) identity)
((specialty_federation DETONATION (Relative_Location_Y)) (cctt_federation DETONATION (Relative_Location_Y)) identity)
((specialty_federation DETONATION (Relative_Location_Z)) (cctt_federation DETONATION (Relative_Location_Z)) identity)
((specialty_federation DETONATION (Burst_Descriptor_Warhead)) (cctt_federation DETONATION (Burst_Descriptor_Warhead))
identity)
((specialty_federation DETONATION (Burst_Descriptor_Fuze)) (cctt_federation DETONATION (Burst_Descriptor_Fuze)) identity)
((specialty_federation DETONATION (Quantity)) (cctt_federation DETONATION (Quantity)) identity)
((specialty_federation DETONATION (Rate)) (cctt_federation DETONATION (Rate)) identity)
((specialty_federation DETONATION (Burst_Descriptor_Detonation_Result)) (cctt_federation DETONATION
(Burst_Descriptor_Detonation_Result)) identity)

((cctt_federation WEAPON_FIRE (Firing_ID)) (specialty_federation WEAPON_FIRE (Firing_ID)) identity)
((cctt_federation WEAPON_FIRE (Target_ID)) (specialty_federation WEAPON_FIRE (Target_ID)) identity)
((cctt_federation WEAPON_FIRE (Location_X)) (specialty_federation WEAPON_FIRE (Location_X)) identity)
((cctt_federation WEAPON_FIRE (Location_Y)) (specialty_federation WEAPON_FIRE (Location_Y)) identity)
((cctt_federation WEAPON_FIRE (Location_Z)) (specialty_federation WEAPON_FIRE (Location_Z)) identity)
((cctt_federation WEAPON_FIRE (Velocity_X)) (specialty_federation WEAPON_FIRE (Velocity_X)) identity)
((cctt_federation WEAPON_FIRE (Velocity_Y)) (specialty_federation WEAPON_FIRE (Velocity_Y)) identity)
((cctt_federation WEAPON_FIRE (Velocity_Z)) (specialty_federation WEAPON_FIRE (Velocity_Z)) identity)
((cctt_federation WEAPON_FIRE (Munition_Type)) (specialty_federation WEAPON_FIRE (Munition_Type)) identity)
((cctt_federation WEAPON_FIRE (Quantity)) (specialty_federation WEAPON_FIRE (Quantity)) identity)
((cctt_federation WEAPON_FIRE (Rate)) (specialty_federation WEAPON_FIRE (Rate)) identity)

((specialty_federation WEAPON_FIRE (Firing_ID)) (cctt_federation WEAPON_FIRE (Firing_ID)) identity)
((specialty_federation WEAPON_FIRE (Target_ID)) (cctt_federation WEAPON_FIRE (Target_ID)) identity)
((specialty_federation WEAPON_FIRE (Location_X)) (cctt_federation WEAPON_FIRE (Location_X)) identity)
((specialty_federation WEAPON_FIRE (Location_Y)) (cctt_federation WEAPON_FIRE (Location_Y)) identity)
((specialty_federation WEAPON_FIRE (Location_Z)) (cctt_federation WEAPON_FIRE (Location_Z)) identity)
((specialty_federation WEAPON_FIRE (Velocity_X)) (cctt_federation WEAPON_FIRE (Velocity_X)) identity)
((specialty_federation WEAPON_FIRE (Velocity_Y)) (cctt_federation WEAPON_FIRE (Velocity_Y)) identity)
((specialty_federation WEAPON_FIRE (Velocity_Z)) (cctt_federation WEAPON_FIRE (Velocity_Z)) identity)
((specialty_federation WEAPON_FIRE (Munition_Type)) (cctt_federation WEAPON_FIRE (Munition_Type)) identity)
((specialty_federation WEAPON_FIRE (Quantity)) (cctt_federation WEAPON_FIRE (Quantity)) identity)
((specialty_federation WEAPON_FIRE (Rate)) (cctt_federation WEAPON_FIRE (Rate)) identity)
)

Appendix F – OMDDS WORKBOOK

Federation Execution Summary Table

Federation Execution Name: OMDD Experiment - Bottom Up

Number of Concurrent Federation Executions (total including this Federation Execution): 1

RTI Software Used (Version): 1.0 Release 2

NOTE:
Complete one of these tables for each Federation execution

Federate Summary Information

	Name	API (C++,Ada, IDL, Java)	Time Management Switches		Host (assign # to each host) (List data on Host Table)	LAN (assign # to each LAN) (List data on LAN Tables)
			Regulating (y or n)	Constraining (y or n)		
Fed 1	CCIT	Ada	N	N	1	1
Fed 2	ModSAF	Ada	N	N	2	1
Fed 3						
Fed 4						
Fed 5						
Fed 6						
Fed 7						
Fed 8						
Fed 9						
Fed 10						
Fed 11						
Fed N						

Host Table

	Hardware	Operating System	Memory available to RTI (MB)	% CPU Available to RTI
Host $_1$	IBM 43P	AIX 4.1		
Host $_2$	IBM 43P	AIX 4.1		
Host $_3$	IBM 43P	AIX 4.1		
Host $_4$	SUN	SunOS5.5.2		
Host $_5$	SGI	IRIX64		
Host $_6$				
Host $_7$				
Host $_n$				

NOTE:
Complete one of these tables for each Federation execution

LAN Tables

LAN Table 1: LAN Descriptions

	Physical Type (Ethernet, ATM,etc.)	Throughtput Available to FEDEX
LAN_1	Ethernet 10 Base-T	
LAN_2		
LAN_3		
LAN_4		
LAN_5		
LAN_6		
LAN_7		
LAN_n		

NOTE:
Complete one of these tables for each Federation execution

LAN Table 2: LAN to LAN Connectivity

	LAN_1	LAN_2	LAN_3	LAN_4	LAN_5	LAN_6	.	.	.	LAN_N
LAN_1										
LAN_2	1. _____ * 2. _____ * 3. _____ *									
LAN_3	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *								
LAN_4	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *							
LAN_5	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *						
LAN_6	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *					
...										
LAN_N	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *				

*

1. Device type means type of switch employed to connect the LANs
2. Throughput means the effective throughput available through the LAN connection for Federation execution, expressed in Mb
3. Latency contribution from devices connecting the LANs, expressed in milliseconds.

RTI Services Table (Check if service to be used at least once during this Federation execution)		
Service	**IF Ref**	**Service Used?**
Create Federation Execution	2.1	✓
Destroy Federation Execution	2.2	✓
Join Federation Execution	2.3	✓
Resign Federation Execution	2.4	✓
Request **Pause**	2.5	
Initiate **Pause**	2.6	
Pause Achieved	2.7	
Request **Resume**	2.8	
Initiate **Resume**	2.9	
Resume Achieved	2.10	
Request **Federation Save**	2.11	
Initiate **Federation Save**	2.12	
Federation Save Begun	2.13	
Federation Save Achieved	2.14	
Request **Restore**	2.15	
Initiate **Restore**	2.16	
Restore Achieved	2.17	
Publish Object Class	3.1	✓
Subscribe Object Class **Attributes**	3.2	✓
Publish Interaction	3.3	✓
Subscribe Interaction	3.4	✓
Control Updates	3.5	✓
Control Interactions	3.6	✓
Request ID	4.1	✓
Register Object	4.2	✓
Update Attribute Values	4.3	✓
Delete Object	4.8	
Remove Object	4.9	
Change Attribute Transportation Type	4.10	
Change Attribute Order Type	4.11	
Change Interaction Transportation Type	4.12	
Change Interaction Order Type	4.13	
Request Attribute Value Update	4.14	✓
Provide Attribute Value Update	4.15	✓
Retract	4.16	
Reflect Retract	4.17	
Request Attribute **Ownership Divestiture**	5.1	
Request Attribute **Ownership Assumptio**	5.2	
Attribute Ownership **Divestiture Notifica**	5.3	
Attribute Ownership **Acquisition Notifica**	5.4	
Request Attribute **Ownership Acquisitio**	5.5	
Request Attribute **Ownership Release**	5.6	
Query Attribute Ownership	5.7	
Inform Attribute Ownership	5.8	
Is **Attribute Owned by Federate?**	5.9	
Request **Federation Time**	6.1	
Request **LBTS**	6.2	
Request **Federate Time**	6.3	
Request **Min Next Event Time**	6.4	
Set **Lookahead**	6.5	
Request **Lookahead**	6.6	
Time Advance Request	6.7	

NOTE: Complete one of these tables for each Federation execution

Object/Interaction Table

NOTE: Complete one of these tables for each Federate

Federate,
Bottom Up

					If Update =" y"					If Subscribe = y?	Ownership	
Object/ Interaction Class	**Attribute/ Parameter**	**Count**	**Size**	**Update?**	**Update Rate**	**Update Grouping**	**Transport**	**Ordering**	**Subscribe?**	**Maximum tolerable latency from any source**	**Attribute Ownership Transfer Rate**	**Ownership Transfer Grouping**
				(y or n)	**# updates/unit time**	(Assign same letter to attributes which will all be updated at the same time)	**R= Reliable B= Best Effort**	**TSO or FIFO**	**(y or n)**	**(milliseconds)**	**# times/unit time**	(Assign same letter to attributes which will all be transferred together)
Munition												
	Munition_Type			y	15	A	R	FIFO	y			
Military_Platform												
	Acceleration			y	15	B	R	FIFO	y			
	Acceleration_Vector			y	15	B	R	FIFO	y			
	Articulated_Parameter_Type			y	15	B	R	FIFO	y			
	Articulated_Parameters_Count			y	15	B	R	FIFO	y			
	Combat_Status			y	15	B	R	FIFO	y			
	Concealed_Indicator			y	15	B	R	FIFO	y			
	Country_Code			y	15	B	R	FIFO	y			
	Damage_Status			y	15	B	R	FIFO	y			
	Dead_Reckoning_Algorithm			y	15	B	R	FIFO	y			
	Distance			y	15	B	R	FIFO	y			
	Domain			y	15	B	R	FIFO	y			
	Event_Identifier			y	15	B	R	FIFO	y			
	Flames_Present_Indicator			y	15	B	R	FIFO	y			
	Friend_Foe_Code			y	15	B	R	FIFO	y			
	Immobilized_Indicator			y	15	B	R	FIFO	y			
	Mass			y	15	B	R	FIFO	y			
	Speed			y	15	B	R	FIFO	y			
	Velocity_Vector			y	15	B	R	FIFO	y			
	World_Coordinate			y	15	B	R	FIFO	y			
	World_Position			y	15	B	R	FIFO	y			
Military_Ground_Vehicle												
	Equipment_Type			y	15	C	R	FIFO	y			
Military_Aircraft												
	Aircraft_Type			y	15	D	R	FIFO	y			
	Altitude			y	15	D	R	FIFO	y			
	Angle_of_Rotation			y	15	D	R	FIFO	y			
	Angular_Velocity_Vector			y	15	D	R	FIFO	y			
	Angular_Velocity			y	15	D	R	FIFO	y			
	Mission			y	15	D	R	FIFO	y			
	Operation_Status			y	15	D	R	FIFO	y			

	Relative_Position			y	15	D	R	FIFO	y			
Detonate_Munition												
	Detonation_Result_Code			y	15	E	R	FIFO	y			
	Fuse_Type			y	15	E	R	FIFO	y			
	World_Position			y	15	E	R	FIFO	y			
	Warhead_Type			y	15	E	R	FIFO	y			
	Velocity_Vector			y	15	E	R	FIFO	y			
	Velocity_Magnitude			y	15	E	R	FIFO	y			
	Munition_Type			y	15	E	R	FIFO	y			
Fire_Weapon												
	Fire_Rate			y	15	F	R	FIFO	y			
	Munition_Type			y	15	F	R	FIFO	y			
	Relative_Position			y	15	F	R	FIFO	y			
	Warhead_Type			y	15	F	R	FIFO	y			
	World_Position			y	15	F	R	FIFO	y			
	Missile_Type			y	15	F	R	FIFO	y			
	Fuse_Type			y	15	F	R	FIFO	y			
Collide												
	Collision_Type			y	15	G	R	FIFO	y			

Federation Execution Summary Table

Federation Execution Name: OMDD Experiment - Merge SOM using CCTT as baseline

Number of Concurrent Federation Executions (total including this Federation Execution): 1

RTI Software Used (Version): 1.0 Release 2

NOTE:
Complete one of these tables for each Federation execution

Federate Summary Information

	Name	API (C++,Ada, IDL, Java)	Time Management Switches		Host (assign # to each host) (List data on Host Table)	LAN (assign # to each LAN) (List data on LAN Tables)
			Regulating (y or n)	Constraining (y or n)		
Fed 1	CCTT	Ada	N	N	1	1
Fed 2	ModSAF	Ada	N	N	2	1
Fed 3						
Fed 4						
Fed 5						
Fed 6						
Fed 7						
Fed 8						
Fed 9						
Fed 10						
Fed 11						
Fed N						

Host Table

	Hardware	Operating System	Memory available to RTI (MB)	% CPU Available to RTI
Host $_1$	IBM 43P	AIX 4.1		
Host $_2$	IBM 43P	AIX 4.1		
Host $_3$	IBM 43P	AIX 4.1		
Host $_4$	SUN	SunOS5.5.2		
Host $_5$	SGI	IRIX64		
Host $_6$				
Host $_7$				
Host $_n$				

NOTE:
Complete one of these tables for each Federation execution

LAN Tables

LAN Table 1: LAN Descriptions

	Physical Type (Ethernet, ATM,etc.)	Throughtput Available to FEDEX
LAN_1	Ethernet 10 Base-T	
LAN_2		
LAN_3		
LAN_4		
LAN_5		
LAN_6		
LAN_7		
LAN_n		

NOTE:
Complete one of these tables for each Federation execution

LAN Table 2: LAN to LAN Connectivity

	LAN_1	LAN_2	LAN_3	LAN_4	LAN_5	LAN_6	.	.	.	LAN_N
LAN_1										
LAN_2	1. _____ * 2. _____ * 3. _____ *									
LAN_3	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *								
LAN_4	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *							
LAN_5	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *						
LAN_6	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *					
...										
LAN_N	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *				

*

1. Device type means type of switch employed to connect the LANs
2. Throughput means the effective throughput available through the LAN connection for Federation execution, expressed in Mb
3. Latency contribution from devices connecting the LANs, expressed in milliseconds.

RTI Services Table (Check if service to be used at least once during this Federation execution)		
Service	**IF Ref**	**Service Used?**
Create Federation Execution	2.1	✓
Destroy Federation Execution	2.2	✓
Join Federation Execution	2.3	✓
Resign Federation Execution	2.4	✓
Request **Pause**	2.5	
Initiate **Pause**	2.6	
Pause Achieved	2.7	
Request **Resume**	2.8	
Initiate **Resume**	2.9	
Resume Achieved	2.10	
Request **Federation Save**	2.11	
Initiate **Federation Save**	2.12	
Federation Save Begun	2.13	
Federation Save Achieved	2.14	
Request **Restore**	2.15	
Initiate **Restore**	2.16	
Restore Achieved	2.17	
Publish Object Class	3.1	✓
Subscribe Object Class **Attributes**	3.2	✓
Publish Interaction	3.3	✓
Subscribe Interaction	3.4	✓
Control Updates	3.5	✓
Control Interactions	3.6	✓
Request ID	4.1	✓
Register Object	4.2	✓
Update Attribute Values	4.3	✓
Delete Object	4.8	
Remove Object	4.9	
Change Attribute Transportation Type	4.10	
Change Attribute Order Type	4.11	
Change Interaction Transportation Type	4.12	
Change Interaction Order Type	4.13	
Request Attribute Value Update	4.14	✓
Provide Attribute Value Update	4.15	✓
Retract	4.16	
Reflect Retract	4.17	
Request Attribute **Ownership Divestiture**	5.1	
Request Attribute **Ownership Assumptio**	5.2	
Attribute Ownership **Divestiture Notifica**	5.3	
Attribute Ownership **Acquisition Notifica**	5.4	
Request Attribute **Ownership Acquisitio**	5.5	
Request Attribute **Ownership Release**	5.6	
Query Attribute Ownership	5.7	
Inform Attribute Ownership	5.8	
Is **Attribute Owned by Federate?**	5.9	
Request **Federation Time**	6.1	
Request **LBTS**	6.2	
Request **Federate Time**	6.3	
Request **Min Next Event Time**	6.4	
Set **Lookahead**	6.5	
Request **Lookahead**	6.6	
Time Advance Request	6.7	

NOTE: Complete one of these tables for each Federation execution

Object/Interaction Table

NOTE: Complete one of these tables for each Federate

Federate #
CCTT

				If Update = " y"					If Subscribe = y?	Ownership	
Object/ Interaction Class	**Attribute/ Parameter**	**Count**	**Size**	**Update?**	**Update Rate**	**Update Grouping**	**Transport**	**Ordering**	**Subscribe?**	**Maximum tolerable latency from any source**	**Attribute Ownership Transfer Rate**	**Ownership Transfer Grouping**
				(y or n)	**# updates/unit time**	(Assign same letter to attributes which will all be updated at the same time)	**R= Reliable B= Best Effort**	**TSO or FIFO**	**(y or n)**	**(milliseconds)**	**# times/unit time**	(Assign same letter to attributes which will all be transferred together)
Enitity												
	FORCE_ID					A	R	FIFO	y			
	ENTITYID_SITE					A	R	FIFO	y			
	ENTITYID_APPLICATION					A	R	FIFO	y			
	ENTITYID_ENTITY					A	R	FIFO	y			
	LOCATION_X			y	15	A	R	FIFO	y			
	LOCATION_Y			y	15	A	R	FIFO	y			
	LOCATION_Z			y	15	A	R	FIFO	y			
	VELOCITY_X			y	15	A	R	FIFO	y			
	VELOCITY_Y			y	15	A	R	FIFO	y			
	VELOCITY_Z			y	15	A	R	FIFO	y			
	Angular_Velocity			y	15	A	R	FIFO	y			
	ACCELERATION_X			y	15	A	R	FIFO	y			
	ACCELERATION_Y			y	15	A	R	FIFO	y			
	ACCELERATION_Z			y	15	A	R	FIFO	y			
	ORIENTATION_PSI			y	15	A	R	FIFO	y			
	ORIENTATION_THETA			y	15	A	R	FIFO	y			
	ORIENTATION_PHI			y	15	A	R	FIFO	y			
	APPEARANCE_PAINT_SCHEME			y	15	A	R	FIFO	y			
Platform												
	DAMAGE_STATE_APPEARANCE			y	15	B	R	FIFO	y			
Munition												
	FIRING_ID			y	15	C	R	FIFO	y			
Ground_Vehicle												
	APPEARANCE_SMOKE			y	15	D	R	FIFO	y			
	APPEARANCE_TRAILING			y	15	D	R	FIFO	y			
	APPEARANCE_HATCH			y	15	D	R	FIFO	y			
	APPEARANCE_LIGHTS			y	15	D	R	FIFO	y			
	APPEARACNE_FLAMING			y	15	D	R	FIFO	y			
	DAMAGE_STATE_MOBILITY			y	15	D	R	FIFO	y			
	DAMAGE_STATE_FIRE_POWER			y	15	D	R	FIFO	y			

Armored_Fighting_Vehicle											
	APPEARANCE_LAUNCHER		y	15	E	R	FIFO	y			
Detonation											
	Munition_ID		y	15	F	R	FIFO	y			
	Target_ID		y	15	F	R	FIFO	y			
	Firing_ID		y	15	F	R	FIFO	y			
	Location_X		y	15	F	R	FIFO	y			
	Location_Y		y	15	F	R	FIFO	y			
	Location_Z		y	15	F	R	FIFO	y			
	Relative_Location_X		y	15	F	R	FIFO	y			
	Relative_Location_Y		y	15	F	R	FIFO	y			
	Relative_Location_Z		y	15	F	R	FIFO	y			
	Velocity_X		y	15	F	R	FIFO	y			
	Velocity_Y		y	15	F	R	FIFO	y			
	Velocity_Z		y	15	F	R	FIFO	y			
	Burst_Descriptor_Warhead		y	15	F	R	FIFO	y			
	Burst_Descriptor_Fuze		y	15	F	R	FIFO	y			
	Burst_Descriptor_Detonation_Result		y	15	F	R	FIFO	y			
	Quantity		y	15	F	R	FIFO	y			
	Rate		y	15	F	R	FIFO	y			
Weapon_Fire											
	Firing_ID		y	15	G	R	FIFO	y			
	Target_ID		y	15	G	R	FIFO	y			
	Location_X		y	15	G	R	FIFO	y			
	Location_Y		y	15	G	R	FIFO	y			
	Location_Z		y	15	G	R	FIFO	y			
	Velocity_X		y	15	G	R	FIFO	y			
	Velocity_Y		y	15	G	R	FIFO	y			
	Velocity_Z		y	15	G	R	FIFO	y			
	Munition_Type		y	15	G	R	FIFO	y			
	Quantity		y	15	G	R	FIFO	y			
	Rate		y	15	G	R	FIFO	y			
Weapon_Launch											
	Launch_Platform_ID		y	15	H	R	FIFO	y			
	Munition_ID		y	15	H	R	FIFO	y			
	Target_ID		y	15	H	R	FIFO	y			
	Location_X		y	15	H	R	FIFO	y			
	Location_Y		y	15	H	R	FIFO	y			
	Location_Z		y	15	H	R	FIFO	y			
	Velocity_X		y	15	H	R	FIFO	y			
	Velocity_Y		y	15	H	R	FIFO	y			
	Velocity_Z		y	15	H	R	FIFO	y			
Collision											
	Issuing_ID		y	15	I	R	FIFO	y			
	Colliding_ID		y	15	I	R	FIFO	y			
	Mass		y	15	I	R	FIFO	y			
	Relative_Location_X		y	15	I	R	FIFO	y			

	Relative_Location_Y			y	15		I	R	FIFO	y			
	Relative_Location_Z			y	15		I	R	FIFO	y			
	Velocity_X			y	15		I	R	FIFO	y			
	Velocity_Y			y	15		I	R	FIFO	y			
	Velocity_Z			y	15		I	R	FIFO	y			

Federation Execution Summary Table

Federation Execution Name: OMDD Experiment - Merge SOM using ModSAF as baseline

Number of Concurrent Federation Executions (total including this Federation Execution): 1

RTI Software Used (Version): 1.0 Release 2

NOTE:
Complete one of these tables for each Federation execution

Federate Summary Information

	Name	API (C++,Ada, IDL, Java)	Time Management Switches		Host (assign # to each host) (List data on Host Table)	LAN (assign # to each LAN) (List data on LAN Tables)
			Regulating (y or n)	Constraining (y or n)		
Fed 1	CCTT	Ada	N	N	1	1
Fed 2	ModSAF	Ada	N	N	2	1
Fed 3						
Fed 4						
Fed 5						
Fed 6						
Fed 7						
Fed 8						
Fed 9						
Fed 10						
Fed 11						
Fed N						

Host Table

	Hardware	Operating System	Memory available to RTI (MB)	% CPU Available to RTI
Host $_1$	IBM 43P	AIX 4.1		
Host $_2$	IBM 43P	AIX 4.1		
Host $_3$	IBM 43P	AIX 4.1		
Host $_4$	SUN	SunOS5.5.2		
Host $_5$	SGI	IRIX64		
Host $_6$				
Host $_7$				
Host $_n$				

NOTE:
Complete one of these tables for each Federation execution

LAN Tables

LAN Table 1: LAN Descriptions

	Physical Type (Ethernet, ATM,etc.)	Throughtput Available to FEDEX
LAN_1	Ethernet 10 Base-T	
LAN_2		
LAN_3		
LAN_4		
LAN_5		
LAN_6		
LAN_7		
LAN_n		

NOTE:
Complete one of these tables for each Federation execution

LAN Table 2: LAN to LAN Connectivity

	LAN_1	LAN_2	LAN_3	LAN_4	LAN_5	LAN_6	.	.	.	LAN_N
LAN_1										
LAN_2	1. _____ * 2. _____ * 3. _____ *									
LAN_3	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *								
LAN_4	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *							
LAN_5	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *						
LAN_6	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *					
...										
LAN_N	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *	1. _____ * 2. _____ * 3. _____ *				

*

1. Device type means type of switch employed to connect the LANs
2. Throughput means the effective throughput available through the LAN connection for Federation execution, expressed in Mb
3. Latency contribution from devices connecting the LANs, expressed in milliseconds.

RTI Services Table (Check if service to be used at least once during this Federation execution)		
Service	**IF Ref**	**Service Used?**
Create Federation Execution	2.1	✓
Destroy Federation Execution	2.2	✓
Join Federation Execution	2.3	✓
Resign Federation Execution	2.4	✓
Request **Pause**	2.5	
Initiate **Pause**	2.6	
Pause Achieved	2.7	
Request **Resume**	2.8	
Initiate **Resume**	2.9	
Resume Achieved	2.10	
Request **Federation Save**	2.11	
Initiate **Federation Save**	2.12	
Federation Save Begun	2.13	
Federation Save Achieved	2.14	
Request **Restore**	2.15	
Initiate **Restore**	2.16	
Restore Achieved	2.17	
Publish Object Class	3.1	✓
Subscribe Object Class **Attributes**	3.2	✓
Publish Interaction	3.3	✓
Subscribe Interaction	3.4	✓
Control Updates	3.5	✓
Control Interactions	3.6	✓
Request ID	4.1	✓
Register Object	4.2	✓
Update Attribute Values	4.3	✓
Delete Object	4.8	
Remove Object	4.9	
Change Attribute Transportation Type	4.10	
Change Attribute Order Type	4.11	
Change Interaction Transportation Type	4.12	
Change Interaction Order Type	4.13	
Request Attribute Value Update	4.14	✓
Provide Attribute Value Update	4.15	✓
Retract	4.16	
Reflect Retract	4.17	
Request Attribute **Ownership Divestiture**	5.1	
Request Attribute **Ownership Assumptio**	5.2	
Attribute Ownership **Divestiture Notificat**	5.3	
Attribute Ownership **Acquisition Notifica**	5.4	
Request Attribute **Ownership Acquisitio**	5.5	
Request Attribute **Ownership Release**	5.6	
Query Attribute Ownership	5.7	
Inform Attribute Ownership	5.8	
Is **Attribute Owned by Federate?**	5.9	
Request **Federation Time**	6.1	
Request **LBTS**	6.2	
Request **Federate Time**	6.3	
Request **Min Next Event Time**	6.4	
Set **Lookahead**	6.5	
Request **Lookahead**	6.6	
Time Advance Request	6.7	

NOTE: Complete one of these tables for each Federation execution

Object/Interaction Table

NOTE: Complete one of these tables for each Federate

Federate
ModSAF

				If Update = " y"					If Subscribe = y?	Ownership	
Object/ Interaction Class	**Attribute/ Parameter**	**Count**	**Size**	**Update?** (y or n)	**Update Rate** # updates/unit time	**Update Grouping** (Assign same letter to attributes which will all be updated at the same time)	**Transport** R= Reliable B= Best Effort	**Ordering** TSO or FIFO	**Subscribe?** (y or n)	**Maximum tolerable latency from any source** (milliseconds)	**Attribute Ownership Transfer Rate** # times/unit time	**Ownership Transfer Grouping** (Assign same letter to attributes which will all be transferred together)
Enitity												
	Entity_ID_site					A	R	FIFO	y			
	Entity_ID_application					A	R	FIFO	y			
	Entity_ID_entity					A	R	FIFO	y			
	Force_ID					A	R	FIFO	y			
	Orientation_Psi			y	15	A	R	FIFO	y			
	Orientation_Theta			y	15	A	R	FIFO	y			
	Orientation_Phi			y	15	A	R	FIFO	y			
	Location_X			y	15	A	R	FIFO	y			
	Location_Y			y	15	A	R	FIFO	y			
	Location_Z			y	15	A	R	FIFO	y			
	Velocity_X			y	15	A	R	FIFO	y			
	Velocity_Y			y	15	A	R	FIFO	y			
	Velocity_Z			y	15	A	R	FIFO	y			
	Acceleration_X			y	15	A	R	FIFO	y			
	Acceleration_Y			y	15	A	R	FIFO	y			
	Acceleration_Z			y	15	A	R	FIFO	y			
	Angular_Velocity_Psi			y	15	A	R	FIFO	y			
	Angular_Velocity_Theta			y	15	A	R	FIFO	y			
	Angular_Velocity_Phi			y	15	A	R	FIFO	y			
	Entity_Type_Kind			y	15	A	R	FIFO	y			
	Entity_Type_Domain			y	15	A	R	FIFO	y			
	Entity_Type_Country			y	15	A	R	FIFO	y			
	Entity_Type_Category			y	15	A	R	FIFO	y			
	Entity_Type_Subcategory			y	15	A	R	FIFO	y			
	Entity_Type_Specific			y	15	A	R	FIFO	y			
	Entity_Type_Extra			y	15	A	R	FIFO	y			
	marking_charset			y	15	A	R	FIFO	y			
	marking_text			y	15	A	R	FIFO	y			
	Dead_Reckoning_Algorithm			y	15	A	R	FIFO	y			
	TimeStamp			y	15	A	R	FIFO	y			
	MarkingCharSet			y	15	A	R	FIFO	y			
	Capabilities			y	15	A	R	FIFO	y			
	APPEARANCE_SMOKE			y	15	A	R	FIFO	y			
	APPEARANCE_TRAILING			y	15	A	R	FIFO	y			
	APPEARANCE_HATCH			y	15	A	R	FIFO	y			

	APPEARANCE_LIGHTS			y	15	A	R	FIFO	y			
	APPEARANCE_FLAMING			y	15	A	R	FIFO	y			
Detonation												
	Munition_ID_site			y	15	B	R	FIFO	y			
	Munition_ID_application			y	15	B	R	FIFO	y			
	Munition_ID_entity			y	15	B	R	FIFO	y			
	Attacker_ID_site			y	15	B	R	FIFO	y			
	Attacker_ID_application			y	15	B	R	FIFO	y			
	Attacker_ID_entity			y	15	B	R	FIFO	y			
	Target_ID_site			y	15	B	R	FIFO	y			
	Target_ID_application			y	15	B	R	FIFO	y			
	Target_ID_entity			y	15	B	R	FIFO	y			
	Event_ID_site			y	15	B	R	FIFO	y			
	Event_ID_application			y	15	B	R	FIFO	y			
	Event_ID_entity			y	15	B	R	FIFO	y			
	WorldLocationX			y	15	B	R	FIFO	y			
	WorldLocationY			y	15	B	R	FIFO	y			
	WorldLocationZ			y	15	B	R	FIFO	y			
	EntityLocationX			y	15	B	R	FIFO	y			
	EntityLocationY			y	15	B	R	FIFO	y			
	EntityLocationZ			y	15	B	R	FIFO	y			
	VelocityX			y	15	B	R	FIFO	y			
	VelocityY			y	15	B	R	FIFO	y			
	VelocityZ			y	15	B	R	FIFO	y			
	BurstKind			y	15	B	R	FIFO	y			
	BurstDomain			y	15	B	R	FIFO	y			
	BurstCountry			y	15	B	R	FIFO	y			
	BurstCategory			y	15	B	R	FIFO	y			
	BurstSubcategory			y	15	B	R	FIFO	y			
	BurstSpecific			y	15	B	R	FIFO	y			
	BurstExtra			y	15	B	R	FIFO	y			
	BurstWarhead			y	15	B	R	FIFO	y			
	BurstFuze			y	15	B	R	FIFO	y			
	BurstQuantity			y	15	B	R	FIFO	y			
	BurstRate			y	15	B	R	FIFO	y			
	DetonationResult			y	15	B	R	FIFO	y			
	TimeStamp			y	15	B	R	FIFO	y			
	ArticulatedStruct			y	15	B	R	FIFO	y			
Fire												
	Launch_Platform_ID_site			y	15	C	R	FIFO	y			
	Launch_Platform_ID_application			y	15	C	R	FIFO	y			
	Launch_Platform_ID_entity			y	15	C	R	FIFO	y			
	Target_ID_site			y	15	C	R	FIFO	y			
	Target_ID_application			y	15	C	R	FIFO	y			
	Target_ID_entity			y	15	C	R	FIFO	y			
	Weapon_ID_site			y	15	C	R	FIFO	y			
	Weapon_ID_application			y	15	C	R	FIFO	y			
	Weapon_ID_entity			y	15	C	R	FIFO	y			
	Event_ID_site			y	15	C	R	FIFO	y			
	Event_ID_application			y	15	C	R	FIFO	y			

	Event_ID_entity			y	15	C	R	FIFO	y				
	LocationX			y	15	C	R	FIFO	y				
	LocationY			y	15	C	R	FIFO	y				
	LocationZ			y	15	C	R	FIFO	y				
	VelocityX			y	15	C	R	FIFO	y				
	VelocityY			y	15	C	R	FIFO	y				
	VelocityZ			y	15	C	R	FIFO	y				
	Range			y	15	C	R	FIFO	y				
	TimeStamp			y	15	C	R	FIFO	y				
	BurstKind			y	15	C	R	FIFO	y				
	BurstDomain			y	15	C	R	FIFO	y				
	BurstCountry			y	15	C	R	FIFO	y				
	BurstCategory			y	15	C	R	FIFO	y				
	BurstSubcategory			y	15	C	R	FIFO	y				
	BurstSpecific			y	15	C	R	FIFO	y				
	BurstExtra			y	15	C	R	FIFO	y				
	BurstWarhead			y	15	C	R	FIFO	y				
	BurstFuze			y	15	C	R	FIFO	y				
	BurstQuantity			y	15	C	R	FIFO	y				
	BurstRate			y	15	C	R	FIFO	y				
Collision													
	TimeStamp			y	15	D	R	FIFO	y				
	IssuingEntitySite			y	15	D	R	FIFO	y				
	IssuingEntityHost			y	15	D	R	FIFO	y				
	IssuingEntityId			y	15	D	R	FIFO	y				
	CollidingEntitySite			y	15	D	R	FIFO	y				
	CollidingEntityHost			y	15	D	R	FIFO	y				
	CollidingEntityId			y	15	D	R	FIFO	y				
	EventIdSite			y	15	D	R	FIFO	y				
	EventIdHost			y	15	D	R	FIFO	y				
	EventIdEvent			y	15	D	R	FIFO	y				
	VelocityX			y	15	D	R	FIFO	y				
	VelocityY			y	15	D	R	FIFO	y				
	VelocityZ			y	15	D	R	FIFO	y				
	Mass			y	15	D	R	FIFO	y				
	RelLocationX			y	15	D	R	FIFO	y				
	RelLocationY			y	15	D	R	FIFO	y				
	RelLocationZ			y	15	D	R	FIFO	y				

Federation Execution Summary Table

Federation Execution Name: OMDD Experiment - RPR FOM

Number of Concurrent Federation Executions (total including this Federation Execution): 1

RTI Software Used (Version): 1.0 Release 2

NOTE:
Complete one of these tables for each Federation execution

Federate Summary Information

	Name	API (C++,Ada, IDL, Java)	Time Management Switches		Host (assign # to each host) (List data on Host Table)	LAN (assign # to each LAN) (List data on LAN Tables)
			Regulating (y or n)	Constraining (y or n)		
Fed 1	CCTT	Ada	N	N	1	1
Fed 2	ModSAF	Ada	N	N	2	1
Fed 3						
Fed 4						
Fed 5						
Fed 6						
Fed 7						
Fed 8						
Fed 9						
Fed 10						
Fed 11						
Fed N						

Host Table

	Hardware	Operating System	Memory available to RTI (MB)	% CPU Available to RTI
Host $_1$	IBM 43P	AIX 4.1		
Host $_2$	IBM 43P	AIX 4.1		
Host $_3$	IBM 43P	AIX 4.1		
Host $_4$	SUN	SunOS5.5.2		
Host $_5$	SGI	IRIX64		
Host $_6$				
Host $_7$				
Host $_n$				

NOTE:
Complete one of these tables for each Federation execution

LAN Tables

LAN Table 1: LAN Descriptions

	Physical Type (Ethernet, ATM,etc.)	Throughtput Available to FEDEX
LAN_1	Ethernet 10 Base-T	
LAN_2		
LAN_3		
LAN_4		
LAN_5		
LAN_6		
LAN_7		
LAN_n		

NOTE:
Complete one of these tables for each Federation execution

LAN Table 2: LAN to LAN Connectivity

	LAN_1	LAN_2	LAN_3	LAN_4	LAN_5	LAN_6	.	.	.	LAN_N
LAN_1										
LAN_2	1. ____ * 2. ____ * 3. ____ *									
LAN_3	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *								
LAN_4	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *							
LAN_5	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *						
LAN_6	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *					
...										
LAN_N	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *	1. ____ * 2. ____ * 3. ____ *				

*

1. Device type means type of switch employed to connect the LANs
2. Throughput means the effective throughput available through the LAN connection for Federation execution, expressed in Mb
3. Latency contribution from devices connecting the LANs, expressed in milliseconds.

RTI Services Table (Check if service to be used at least once during this Federation execution)		
Service	**IF Ref**	**Service Used?**
Create Federation Execution	2.1	✓
Destroy Federation Execution	2.2	✓
Join Federation Execution	2.3	✓
Resign Federation Execution	2.4	✓
Request **Pause**	2.5	
Initiate **Pause**	2.6	
Pause Achieved	2.7	
Request **Resume**	2.8	
Initiate **Resume**	2.9	
Resume Achieved	2.10	
Request **Federation Save**	2.11	
Initiate **Federation Save**	2.12	
Federation Save Begun	2.13	
Federation Save Achieved	2.14	
Request **Restore**	2.15	
Initiate **Restore**	2.16	
Restore Achieved	2.17	
Publish Object Class	3.1	✓
Subscribe Object Class **Attributes**	3.2	✓
Publish Interaction	3.3	✓
Subscribe Interaction	3.4	✓
Control Updates	3.5	✓
Control Interactions	3.6	✓
Request ID	4.1	✓
Register Object	4.2	✓
Update Attribute Values	4.3	✓
Delete Object	4.8	
Remove Object	4.9	
Change Attribute Transportation Type	4.10	
Change Attribute Order Type	4.11	
Change Interaction Transportation Type	4.12	
Change Interaction Order Type	4.13	
Request Attribute Value Update	4.14	✓
Provide Attribute Value Update	4.15	✓
Retract	4.16	
Reflect Retract	4.17	
Request Attribute **Ownership Divestiture**	5.1	
Request Attribute **Ownership Assumptio**	5.2	
Attribute Ownership **Divestiture Notifica**	5.3	
Attribute Ownership **Acquisition Notifica**	5.4	
Request Attribute **Ownership Acquisitio**	5.5	
Request Attribute **Ownership Release**	5.6	
Query Attribute Ownership	5.7	
Inform Attribute Ownership	5.8	
Is **Attribute Owned by Federate?**	5.9	
Request **Federation Time**	6.1	
Request **LBTS**	6.2	
Request **Federate Time**	6.3	
Request **Min Next Event Time**	6.4	
Set **Lookahead**	6.5	
Request **Lookahead**	6.6	
Time Advance Request	6.7	

NOTE: Complete one of these tables for each Federation execution

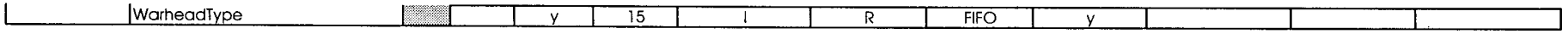
Object/Interaction Table

NOTE: Complete one of these tables for each Federate

Federate #
RPR FOM

				If Update = " y"					If Subscribe = y?	Ownership	
Object/ Interaction Class	**Attribute/ Parameter**	**Count**	**Size**	**Update?**	**Update Rate**	**Update Grouping**	**Transport**	**Ordering**	**Subscribe?**	**Maximum tolerable latency from any source**	**Attribute Ownership Transfer Rate**	**Ownership Transfer Grouping**
				(y or n)	**# updates/unit time**	(Assign same letter to attributes which will all be updated at the same time)	**R= Reliable B= Best Effort**	**TSO or FIFO**	**(y or n)**	**(milliseconds)**	**# times/unit time**	(Assign same letter to attributes which will all be transferred together)
BaseEnitity												
	AccelerationVector			y	15	A	R	FIFO	y			
	AngularVelocityVector			y	15	A	R	FIFO	y			
	DRAlgorithm			y	15	A	R	FIFO	y			
	EntityType			y	15	A	R	FIFO	y			
	FederateID					A	R	FIFO	y			
	IsFrozen			y	15	A	R	FIFO	y			
	Orientation			y	15	A	R	FIFO	y			
	Position			y	15	A	R	FIFO	y			
	VelocityVector			y	15	A	R	FIFO	y			
MilitaryEntity												
	AlternateEntityType			y	15	B	R	FIFO	y			
	CamouflageType			y	15	B	R	FIFO	y			
	FirePowerDisabled			y	15	B	R	FIFO	y			
	ForceID			y	15	B	R	FIFO	y			
	IsConcealed			y	15	B	R	FIFO	y			
MilitaryPlatformEntity												
	AfterburnerOn			y	15	C	R	FIFO	y			
	HasAmmunitionSupplyCap			y	15	C	R	FIFO	y			
	LauncherRaised			y	15	C	R	FIFO	y			
MunitionEntity												
	LauncherFlashPresent			y	15	D	R	FIFO	y			
PhysicalEntity												
	ArticulatedParametersArray			y	15	E	R	FIFO	y			
	ArticulatedParametersCount			y	15	E	R	FIFO	y			
	DamageState			y	15	E	R	FIFO	y			
	EngineSmokeOn			y	15	E	R	FIFO	y			
	FlamesPresent			y	15	E	R	FIFO	y			
	HasFuelSupplyCap			y	15	E	R	FIFO	y			
	HasRecoveryCap			y	15	E	R	FIFO	y			

	HasRepairCap			y	15	E	R	FIFO	y			
	HatchState			y	15	E	R	FIFO	y			
	Immobilized			y	15	E	R	FIFO	y			
	LifeformState			y	15	E	R	FIFO	y			
	LightsState			y	15	E	R	FIFO	y			
	Marking			y	15	E	R	FIFO	y			
	PowerPlantOn			y	15	E	R	FIFO	y			
	RampDeployed			y	15	E	R	FIFO	y			
	SmokePlumePresent			y	15	E	R	FIFO	y			
	TentDeployed			y	15	E	R	FIFO	y			
	TrailState			y	15	E	R	FIFO	y			
Collision												
	CollidingObjectID			y	15	F	R	FIFO	y			
	CollidingObjectMass			y	15	F	R	FIFO	y			
	CollidingObjectVelocity			y	15	F	R	FIFO	y			
	CollisionType			y	15	F	R	FIFO	y			
	CollisionLocation			y	15	F	R	FIFO	y			
	EventID			y	15	F	R	FIFO	y			
	IssuingObjectID			y	15	F	R	FIFO	y			
MunitionDetonation												
	ArticulatedPartsArray			y	15	G	R	FIFO	y			
	ArticulatedPartsCount			y	15	G	R	FIFO	y			
	DetonationLocation			y	15	G	R	FIFO	y			
	DetonationResult			y	15	G	R	FIFO	y			
	EventID			y	15	G	R	FIFO	y			
	FiringObjectID			y	15	G	R	FIFO	y			
	FinalVelocityVector			y	15	G	R	FIFO	y			
	FuseType			y	15	G	R	FIFO	y			
	MunitionObjectID			y	15	G	R	FIFO	y			
	MunitionType			y	15	G	R	FIFO	y			
	QuantityFired			y	15	G	R	FIFO	y			
	RateOfFire			y	15	G	R	FIFO	y			
	RelativeDetonationLocation			y	15	G	R	FIFO	y			
	TargetObjectID			y	15	G	R	FIFO	y			
	WarheadType			y	15	G	R	FIFO	y			
WeaponFire												
	EventID			y	15	I	R	FIFO	y			
	FireControlSolutionRange			y	15	I	R	FIFO	y			
	FireMissionIndex			y	15	I	R	FIFO	y			
	FiringLocation			y	15	I	R	FIFO	y			
	FiringObjectID			y	15	I	R	FIFO	y			
	FuseType			y	15	I	R	FIFO	y			
	InitialVelocityVector			y	15	I	R	FIFO	y			
	MunitionObjectID			y	15	I	R	FIFO	y			
	MunitionType			y	15	I	R	FIFO	y			
	QuantityFired			y	15	I	R	FIFO	y			
	RateOfFire			y	15	I	R	FIFO	y			
	TargetObjectID			y	15	I	R	FIFO	y			


WarheadType
y
15
I
R
FIFO
y

Appendix G – SIMULATION INTEROPERABILITY WORKSHOP (SIW) PAPERS

FEDEP Phase III Examination: Federation Testing, Execution and Feedback

Christina Bouwens
Science Applications International Corporation
12479 Research Parkway
Orlando, FL 32826-3248
(407) 207-2742
Chris_Bouwens@cpqm.saic.com

Rhonda Freeman
TASC, Inc.
12443 Research Parkway
Orlando, FL 32826
(407) 275-8755 x-233
rlfreeman@tasc.com

Susan Harkrider
U.S. Army Stricom / AMSTI-ET
12350 Research Parkway
Orlando, FL 32826-3926
(407) 384-3830
harkrids@stricom.army.mil

Philomena Zimmerman
Naval Air Warfare Center - Aircraft Division/ACETEF
48150 Shaw Rd, Unit 5, Bldg. 2109
Patuxent River, MD 20670-1907
zimmermanp%am7@mr.nawcad.navy.mil

Keywords: FEDEP, Federation, Federation Execution, Federation Testing, Federation Feedback

ABSTRACT: *Since the time of its inception, the current Federation Development Process (FEDEP) Model (V1.0) of the High Level Architecture (HLA) has been a product of the early experiences of the protofederations. Little if any attention has been placed on examining the later phases of the FEDEP. Indeed, even the FEDEP document on the Defense Modeling and Simulation Office (DMSO) home page alludes to the fact that the descriptions of the processes which occur later in the FEDEP will be added at a later date. This paper will highlight the third phase of the FEDEP lifecycle: Federation Testing, Execution and Feedback. During this examination, limited protofederation experiences will be expanded by technical expectations from the broader M&S community. These expanded experiences will be applied to the current FEDEP model to determine modifications which need to be made, highlight issues within phase III which warrant further exploration, examine supporting resources required to execute the third phase, and/or determine the adequacy of the outputs of each step of phase III. This paper presents information representing the HLA development process underway for the DMSO and the DoD Architecture Management Group (AMG).*

INTRODUCTION

To many, phase III of the FEDEP is the reason for the existence of the High Level Architecture. It is in this final phase that all of the work associated with planning and preparation, simulator/model creation, selection and modification, and research and requirements definition is realized through component interoperability. Once the federation reaches this phase, ideally all of the decisions have been made, and all of the kinks have been worked out. In reality, this is rarely the case. While careful attention in the first two phases of FEDEP can significantly reduce the trouble encountered in this last phase, there is no way, at least for now, to eliminate trouble completely. This paper will attempt to show how careful attention to the parts of this

phase will result in a federation execution which as closely as possible meets the User/Sponsor requirements, with as few anomalies in the parts as possible.

Phase III of the FEDEP currently consists of the last four parts: Federation Test, Federation Execution, Results, and Feedback (Figure 1). These four steps will finish the federation execution, but do not necessarily complete the federation life. There may be [redacted] the federation for [redacted]

Add [redacted] these four step [redacted] netimes, this is n [redacted] olerant of the [redacted] ugh this ada [redacted] om the cur [redacted]

Figure 1 - FEDEP Phase III

PART I: FEDERATION TEST

Description of Federation Test

The purpose of Federation Test is to determine the degree of readiness of the federation for Federation Execution. Federation Test consists of three parts (Figure 2). Basic HLA Compliance Testing, Integration Testing, and Federation Testing.

Compliance Testing

Compliance Testing for HLA is described in the HLA Compliance Checklist [1]. The compliance testing is performed on a federate level and is intended to ensure that the federate software correctly implements the HLA requirements as documented in the Compliance Checklist. Compliance Testing tests individual federates against the HLA specifications and the Federation's Object Model.

Integration Testing

The purpose of Integration Testing is to bring all the pieces of the federation together and assess their ability to interoperate on a basic level. This includes observing the ability of the federates to exchange data and to interact with other federates. Testing on this level is concerned with correct HLA implementation and, to a small extent, the User/Sponsor's study objectives.

Federation Testing

The purpose of Federation Testing is to assess the federation's ability to interoperate as described in the User/Sponsor's study objectives (defined in Part I of the FEDEP). This includes observing the ability of federates to interact according to the defined scenario and to the level of fidelity defined for the Federation.

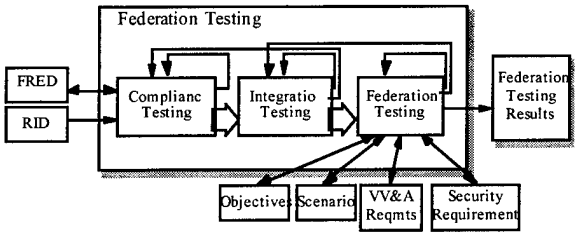


Figure 2 - Federation Test Part of the FEDEP

It is important to note that, although the three parts are described independently, they have generally been carried out together, especially the second and third parts. In addition, experience from the Engineering Protofederation has shown the most common approach to testing to be iterative. "The events may be thought of as a test-fix-test paradigm wherein the first tentative prototype testing provides results to direct the development and subsequent testing in an iterative design approach." [2]. It is generally believed that successive iterations of this part will move the federation towards proper federation execution. Likewise it was shown during the protofederation executions, that failure to rectify the problems encountered during this readiness will result in the inability to obtain the complete set of desired results. (For further details, see Engineering Proto-Federation Evaluation of the High Level

Architecture Developed by the Defense Modeling and Simulation Office (DMSO), dated January 1997).

It is during this part that tests for Validation, Verification and Accreditation, and security will be done. Validation, Verification and Accreditation (VV&A) tests at this stage are used to determine credibility of the entire federation; verify that the resulting interoperation has not compromised the VV&A-ness of any federate. In addition, VV&A can be used to establish the means to trace the test results from this phase to the requirements of the user. This test procedure will, in essence, determine the credibility of the federation results.

For security, the federation and its accompanying documentation will be reviewed and evaluated during this phase in order to reach a favorable accreditation decision. This decision will be based on the information contained in the System Security Authorization Agreement (SSAA) [3]. It is important to note that much like VV&A, security is not limited to this phase of the FEDEP, but has been integral in the decisions regarding formation of the federation since phase I of the FEDEP.

Federation Test Inputs

There are two major inputs to the Federation Test stage of the FEDEP: Federation Required Execution Data (FRED) and RTI Initialization Data (RID). These two inputs combine to provide the environment-required for the RTI to execute properly, regardless of the part or phase of the testing. The FRED and RID, as listed in the HLA glossary [4], are described below:

> FRED
> According to the HLA Glossary, the FRED contains the information required by the RTI to properly execute the federation. It is used to establish the digital contract between the federates. In essence, it has become the 'catch-all' for the details of the federation.
>
> RID
> The RID contains the vendor-specific RTI data needed to execute the federation.

Little can be done to modify the execution environment at this stage; however, that does not mean that decisions about the selection of the execution environment can be made adhoc. Proper selection of components of the execution environment (such as RTI selection, properly validated federates, federates in the correct security domain) can impact the operation of the federation.

Federation Test Execution

Conducting Federation Tests involves following an agreed upon testing approach and collecting the appropriate data to determine if the test was successful. Data collection may consist of data logging network traffic to ensure correct data was sent out under the right conditions (as with Compliance Testing). Data collection may also be coordinated through use of the Management Object Model (MOM). As part of Integration and Federation Testing, the MOM may be used to provide valuable information about the status of federates, objects and other aspects of the federation. [5]

Federation Test Outputs

Depending on the type of Federation Test, a number of outputs will result. Compliance Testing and Integration Testing will result in iterative revision on the part of the federate developer until the federate's operation is correct. Federation testing will result in feedback to the FOM development. In rare instances, new federates will need to be chosen. It is more likely that federates will adapt, through customized interfaces and adjustments in the FOM. Since the RID and the FRED are both dependent on the FOM, changes in the FOM will likely result in changes in the RID and the FRED. As the inputs change, the outcome of all the tests (to include VV&A and security) run during this portion will need to be re-evaluated. If it is determined that the tests are impacted by the new RID and FRED, they will have to be rerun.

Recommendations For Part I

To support the expanded functionality described in this section, the following activities are recommended to update the FEDEP:

Indicate feedback/input between Federation Test and FRED, FOM, RID in FEDEP chart, and support with text.

Include sections in MSRR in FEDEP chart for results of VV&A and security testing, and support with text.

Show feedback/input between objectives and scenario and the test process in the FEDEP chart and support with text.

PART II: FEDERATION EXECUTION

Description of Federation Execution

The purpose of Federation Execution is to utilize the designed federation to support the User/Sponsor Objectives for which it was designed. Two functions (Figure 3) performed during Federation Execution are Execution Management and Data Collection.

Execution Management

Execution Management includes control of the Federation execution, including hardware / system monitoring, federation and federate monitoring. The purpose of these activities is to ensure that the federation execution proceeds as intended.

Data Collection

As determined through earlier stages of the FEDEP, selected data is collected during the Federation Execution for purposes of analysis following the Federation Execution.

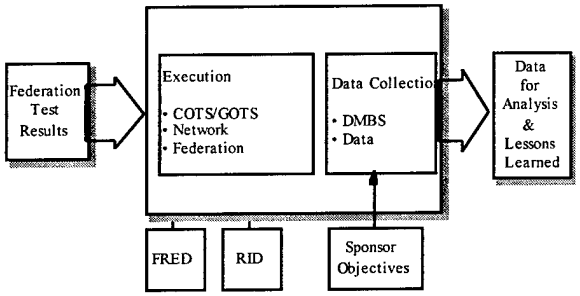


Figure 3 - Federation Execution Component of the FEDEP

It is highly recommended that a Federation Execution Planning Workbook, found on the DMSO home page, be completed before federation execution begins. This workbook currently serves as the framework to document HLA performance. It is a structured mechanism to describe the characteristics of a federation which impact its performance, including the performance of the RTI. The workbook is an Excel workbook composed of five tables: Federation Execution Summary Table; Host Table; LAN Tables; RTI Services Tables; and Object/Interaction Table. A completed workbook is a description of a federation execution (fedex). While the workbook requests information about a range of aspects of the fedex to provide the key dimensions for assessing performance of the fedex, these are all aspects that a fedex developer would necessarily need to understand in order to operate a fedex. Therefore, the Federation Execution Planning Workbook provides a first step in defining the Federation Required Execution Details (FRED). All of the tables are completed for each fedex, and are described below.

Federation Execution Summary Table - This table is completed for each fedex. It includes information about the fedex (name, number of concurrent fedexes) and summary information about each federate in the fedex (API used, time management switches).

Host Table - This table requests information on the hardware, software, and capacity of the hosts which support the federates in the fedex.

LAN Table - This table requests information describing each LAN/WAN used in the fedex, and the network connections.

RTI Services Table - This table identifies the current RTI services used and requests made by the fedex. The service/request calls are used at least once in each fedex, but not necessarily by all federates.

Object/Interaction Table - This table identifies which attributes of objects are updated by each federate, how often, and in what groupings. Likewise, the table identifies which attributes a federate subscribes to and the update latency constraints. The table also records similar data for interactions. One table is completed for each federate in a fedex.

Once the FRED and the RID have been sufficiently modified according to the readiness testing and practice performed in Part I, a determination is made by the User/Sponsor as to the readiness of the Federation to execute as intended to meet the desired objectives. It is critical that changes to the federation in this part of phase III of the FEDEP be strongly discouraged. Once the decision to proceed is made, it becomes harder to justify any modification to the RID, FRED and/or anything which feeds them. It is the recommendation of this paper that at this point, the RID and the FRED be tightly managed. This will reduce any compromise to VV&A or security posture of the federation. Therefore, the decision to proceed must not be taken lightly.

However, there is a chance that, in the execution of the federation, problems may be encountered which were not identified in the Federation Testing part. As problems do arise, however, it may be necessary to make corrections as required. Strict configuration

management of the RID and the FRED does not preclude changes in the RID or the FRED (or anything else which is to be controlled). It provides the means to document the reason for the change and the change itself, to mitigate any compromise to VV&A or security posture of the federation.

Federation Execution Inputs

Inputs to the Federation Execution portion of the FEDEP include the FRED and RID for operation of the RTI during execution. The results of Federation Test are also used to verify federation readiness for execution.

Federation Execution 'Execution'

While this may seem redundant, the actual execution of the federation is more than just starting the execution. It includes the use of COTS/GOTS tools available for network monitoring and federation/federate monitoring. Use of these monitoring tools can assist in identifying problems early in the federation execution, so that they can be corrected, or at worst, so that the execution can be aborted and restarted before losing too much time and resources. If monitoring tools are not available for federation monitoring for some reason, it may be necessary to construct them to encourage monitoring. However, this practice should be discouraged; as it may lead to stove-pipe solutions which may undo some of the interoperability achieved by the very federation it was used to monitor. For example, one of the objectives of the Platform Proto-Federation (PPF) was to assess the functional and execution performance aspects of the RTI prototype as a distributed communication medium. In order to accomplish this, the federates of the PPF monitored their own functional and execution performance by using the Simple Network Management Protocol (SNMP) tools. This allowed monitoring of the network for data such as updates per second and network usage. This tool was very limited however, which made data reduction difficult. [7]

Data Collection

Data collection includes the ability to collect data, analyze, and replay federation executions. In prior federation executions, data collection was accomplished by using COTS/GOTS tools available or by developing federate specific data collection methods. This was intended to allow a federation to assess how well it met the objectives of the User/Sponsor, but often became a point solution for each federate. For the PPF, assessment of objectives included data collection on performance, both federate/federation and RTI performance. As mentioned previously, data collection was accomplished using the SNMP for federation data, while federate data, such as CPU usage and RTI message traffic, was collected manually at each node. Today, data collection within the HLA is clouded by two factors: the use of the RTI services for data exchange; and filtering based on the Data Management/Data Distribution Management (DM/DDM) services. These two factors require a data collection paradigm shift for traditional simulations with fixed format protocols. As yet, a standard approach for data collection has not been defined, however many efforts have been initiated. And, in fact, the Management Object Model provides some mechanisms for the management of a federation, but these services are initiated within each federate.

Federation Execution Outputs

The successful conduct of the Federation Execution will produce data which will be used to support the User's/Sponsor's objectives and to provide insight for conduct of such federations in the future.

RECOMMENDATIONS FOR PART 2:

To support the expanded functionality described in this section, the following activities are recommended to update the FEDEP:

Rename Execution Environment. The new title should be Federation Execution Planning, to allow documentation via the Federation Execution Planning Workbook.

Add arrow from Other Resources to Federation Execution. This will allow use of the MSRR to collect recommendations on COTS/GOTS/Custom tools.

Add arrow into Federation Execution to allow use of MOM services.

PART III: RESULTS AND FEEDBACK

Description of Results and Feedback

The results and feedback part of the FEDEP is used by the federation developers to both return experiences gained during the federation process (start to finish), and to capture the outcomes of the execution for use in analysis. Two functions (Figure 4) are performed during this part: Generation of Results, and Feedback on this Federation Process

Generation of Results

This is where the measurable objectives, as defined in the initial phases of FEDEP, are

compared and contrasted against the measured results.

Feedback on this Federation Process
This will result in the capturing of the lessons learned throughout the entire federation lifecycle, and is dependent on the results generated, as well as input from every part of the FEDEP.

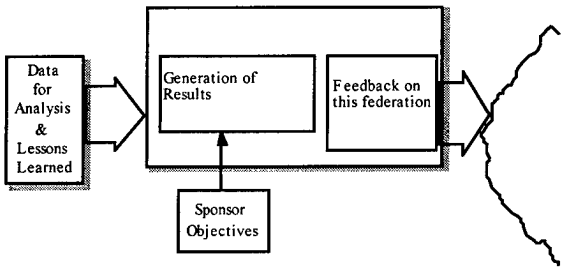


Figure 4 - Results and Feedback Part of the FEDEP

Jeff Rothenberg has defined three (3) critical elements of any model or simulation:

- The purpose of the simulation,
- The referent, or real world thing being modeled, and
- The cost-effectiveness of the model or simulation, as you never want to pay more for a simulation than it costs to use the real thing. [6]

The purpose of the federation is established by the User/Sponsor using the guidelines above, in the earliest phase of the FEDEP. It answers the questions, "What do you want to know or study during this FEDEP?" The Results and Feedback part of the FEDEP provides the information needed to answer the question, or to indicate why the question cannot be fully answered

The referent can be described in terms of the scenario, the types and numbers of objects, attributes, interactions, etc. Other considerations to be included are the length of time the simulation needs to cover to be realistic, the number of runs necessary for statistical validation, the numbers of factors and the levels of the factors critical to the outcome and measures of merit. These are discrete, quantifiable elements of the simulation. Required fidelity and resolution is expressly specified for the Objectives phase of the FEDEP, and is used during the Results and Feedback part to measure the outcome of the federation execution. A significant concern in this area is that there is currently no agreed upon method for expressing fidelity in quantifiable terms. Several methods have been discussed and proposed, but no agreed upon method has been decided upon. Fidelity, is of course, one of the main cost drivers of any model or simulation. With such a glaring absence of a method for measuring fidelity, defining an agreed upon method should be a top priority for the community. This point cannot be over emphasized.

The cost-effectiveness of the simulation can be describe in such objective terms as number of labor hours required for the FEDEP and the type and numbers of equipment used in the FEDEP. This information can be used to provide an unambiguous determination of the cost-effectiveness of the FEDEP. Cost-effectiveness is an important measure providing feedback to the User/Sponsor on the viability of the federation execution for similar study objectives in the future. Comparing this cost figure to the cost figure for using the real-world referent would provide an indication of the benefit of simulation over the use of real-world assets.

Results and Feedback Inputs
In order to properly evaluate the federation execution, a quantifiable set of objectives must be established. The FEDEP model specifies three (3) classes of information to be included for the federation objectives:

1. A specification of the problem domain, including a formalized problem statement, high-level descriptions of critical systems of interests, and required Measures of Merit. Coarse indications of required fidelity and resolution for simulated entities should also be included.

2. A specification of operational context requirements, such as geographic conditions, environmental conditions, threat conditions, and required tactics.

3. A specification of management considerations, such as cost constraints, schedule constraints, facility constraints, and security requirements.

Obviously, some of the required elements for the objectives definition lend themselves more readily to quantitative descriptions than others. That not withstanding, it is highly recommended that ALL objectives are stated in quantifiable terms at the outset of the FEDEP. If qualitative terms are used to describe FEDEP objectives, then some

"standardized" method of transformation to quantitative, measurable, objectives must be instituted. These FEDEP objectives are the driving force of any federation execution analysis.

Another challenge in this part is the gathering of all data from the actual execution needed to support the User/Sponsor objectives. The collection activity takes place during the Federation Execution (described earlier). Then, the data must be collected for analysis. Once challenge is the need for continued data logging for the federation. During the protofederation operations, some chose to use RTI data-loggers at each federate. This added complexity to the data gathering, and prohibited using the data loggers for real-time analysis. One method for reducing the complexity is linking the data logger to the RTI service calls. Logging in this manner helps to ensure that the data logged at each federate was identical in format. However, this method may not be the total data collection effort, as the data collected is limited to what is described in the FOM. If this is the only method, then that may force the federation to publish data that would not normally be published during the federation execution. The composition of the proper toolset for data collection has yet to be determined, and may even be driven by the needs of the federation.

Results and Feedback Execution

The results portion of this part includes the analysis of the raw data and formation of derived results to help meet the objectives of the federation. Reporting these results is also critical and should include the appropriate statistical measure and suitable treatment of error cause by experimental. In a statistical context, the term error is a technical and emotionally neutral term. It is variation that is unavoidable, not associated with blame. Many source contribute to experimental error, in addition to erroneous measurement, analysis, and sampling. This experimental error cannot be ignored.

Details on the context of the results are also critical to proper evaluation of the outcome. The process currently lacks the tools and recommended metrics for consistent and proper reporting of results

Feedback is essential for gaining the most from any federation execution. It takes various forms. Feedback to the User/Sponsor helps to improve their application of the FEDEP to their particular objectives for future analysis. Feedback to the community helps to improve the process itself by providing lessons learned and resulting products into the MSRR.

It is recommended that the feedback section include an analysis of variance. This would allow for the determination of the sources of variation of the average, between treatments, and within treatments. Of course, this assumes that efficient methods of experimental design have been employed in the initial federation design (phase I), which allow the FEDEP designer to obtain answers to their questions that are as unequivocal and as little effected by experimental error as possible. If experimental design is wisely chosen, a great deal of information in a readily extractable form is usually available, and no elaborate analysis may be necessary. Experimental error, confusion or correlation with causation, and complexity of the effects studied are all difficulties which can be mitigated by use of proper statistical methods. Additional improvements would include a provision for more standardized method of providing this feedback.

Results and Feedback Outputs

There are two very different outputs which result from this part: the data and/or analysis of the actual federation execution, for which the User/Sponsor has paid for, and the lessons learned from the actual process of implementing the federation. Data gathered during execution (such as network statistics, RTI performance, CPU usage, etc.) can help the federation developer improve the federation development process and execution. Such data analysis would contribute toward the evolution of the process and technology for future applications.

In addition, both types of data can be further analyzed by communities influenced by, but not directly involved in the actual Phase III of the federation lifecycle: VV&A Agents, and Security accreditors. In some cases, this analysis is required; in others, it is a nice-to-know so mistakes can be identified and corrected in future, similar situations.

RECOMMENDATIONS FOR PART III:

To support the expanded functionality described in this section, the following activities are recommended to update the FEDEP:

Add arrow from Other Resources; using MSRR to accumulate information of tools of use. Does not have to be a separate repository; could be part of required information from federation

Recommendations for information to be gathered: VV&A certifications, HLA compliance certifications, Security accreditations and supporting documentation, POCs for federates; software/hardware tools developed; FOMs; SOMs

CONCLUSION

The HLA Federation Development and Execution Process is a necessary component in the HLA. While it is not mandated for use in adherence with HLA, it serves as a valuable resource in describing the federation lifecycle. As such, it's continued existence demands that it be expanded by current events, and future needs. In addition, its applicability to present day needs constant attention as the sphere of experience with HLA grows.

Experience is not the only driver of change in this process. As the HLA expands beyond the realm of DoD, the FEDEP will need to be monitored and modified to include the processes of other communities as they develop their federations.

This paper has shown the processes taken by the current HLA community in the final phase of the FEDEP. Given that no text exists in the current FEDEP model, this paper has explained the rationale behind Phase III, and suggested modifications to clarify and correct the current FEDEP model. It is the recommendation of this paper that some or all of the text in this paper be used in the model to complete the description of the federation lifecycle contained in the FEDEP Model.

REFERENCES

[1] Defense Modeling and Simulation Office, "HLA Compliance Checklist", version 1.1, 26 March 1997

[2] Engineering Proto-Federation Evaluation of the High Level Architecture Developed by the Defense Modeling and Simulation Office (DMSO), January 1997, para 1.1.1.1

[3] Landrum, E Taylor Jr, and Filsinger, Jarrellann, "Security Engineering Process for HLA Federations", 1997 Spring Simulation Interoperability Workshop, Workshop Papers, Vol 1, 97S-SIW-041

[4] Defense Modeling and Simulation Office, "HLA Glossary"

[5] Management Object Model, Defense Modeling and Simulation Office, 17 October 1996

[6] The Nature of Modeling, by Jeff Rothenberg, Rand Corporation, Santa Monica, CA

[7] Platform Proto-Federations Lessons Learned Document, 8 October 1996

BIOGRAPHIES

CHRISTINA BOUWENS is a Senior Simulation Systems Engineer and a member of SAIC Orlando's Advanced Simulation Research Team. Ms. Bouwens is currently the Project Director for the ADST II HLA Support Experiments delivery order to research in support of HLA 1.0 development and its current evolution. Ms. Bouwens has been an active part of the interoperability standards development community since 1989 and is currently a member of the SISO Transition Team and the new Executive Committee. Ms. Bouwens received her M.S. degree in Mathematical Science from the University of Central Florida.

RHONDA FREEMAN is the project leader for the TASC Object Model Development Toolkit, and a technical lead on the Distributed Exercise Control Tool Project. She is completing her Ph.D. in Industrial Engineering, specialty of Modeling and Simulation, at the University of Central Florida, and is an employee of TASC, Inc.

SUSAN HARKRIDER is a Systems Engineer in the Engineering Directorate at the U.S. Army Simulation, Training and Instrumentation Command (STRICOM). Ms. Harkrider is currently providing technical support to the Defense Modeling and Simulation Office for several projects, including the procurement of RTI version 2.0. She was the lead government engineer for the HLA Platform Proto-Federation project. Ms. Harkrider holds a B.S.E. from the University of Central Florida, and is a M.S. student in Simulation Systems at UCF.

PHILOMENA ZIMMERMAN is the Distributed Simulation Project Lead for the Warfare Simulation Branch of Air Combat Environment Test and Evaluation Facility (ACETEF) at the Naval Air Warfare Center - Aircraft Division (NAWC-AD). She is currently involved in the HLA as a member of the Engineering Federation, a member of the HLA TSTCore, and lead of the HLA Security IPT. She is also the technical lead for the Warfare Simulation

Branch in the JADS-EW Test. Ms. Zimmerman has been involved in the interoperability standards effort since 1990, and is currently a member of the PRP for the Federation Development Process Forum. She received her B.S. in Mathematics from St. John Fisher College.

BUILDING HLA INTERFACES FOR FOM FLEXIBILITY: FIVE CASE STUDIES

Brian Beebe
Science Applications International Corporation
Brian.W.Beebe@cpmx.saic.com
(937) 431-2258

Chris Bouwens
Science Applications International Corporation
12479 Research Parkway
Orlando, Florida 32826-3248
Christina.Bouwens@cpmx.saic.com
(407) 207-2742

Wesley Braudaway
Science Applications International Corporation
12479 Research Parkway
Orlando, FL 32826
Wesley.Braudaway@cpmx.saic.com
(407) 207-2743

Susan Harkrider
Simulation, Training, and Instrumentation Command
12350 Research Parkway
Orlando, FL 32826-3276
harkrids@stricom.army.mil
(407) 384-3926

Jack Ogren
The MITRE Corporation
11493 Sunset Hills Drive
Reston, VA 22090
jogren@mitre.org
(913) 684-9240

Dana Paterson
Naval Air Warfare Center - Aircraft Division/ACETEF
48150 Shaw Rd, Unit 5, Bldg. 2109
Patuxent River, MD 20670-1907
pattersonda%am6@mr.nawcad.navy.mil

Russ Richardson
Science Applications International Corporation
4301 N. Fairfax Drive, Suite 370
Arlington, VA 22203
rrichardson@std.saic.com
(703) 907-2540

Philomena Zimmerman
Naval Air Warfare Center - Aircraft Division/ACETEF
48150 Shaw Rd, Unit 5, Bldg. 2109
Patuxent River, MD 20670-1907
zimmermanp%am7@mr.nawcad.navy.mil

Keywords:

Federation, Federation Object Model (FOM), Simulation Object Model (SOM)

ABSTRACT: *To be successful, the HLA needs to create a set of conditions under which optimal reuse is a natural and assured outcome. Reuse in this case means utilizing the same federate in different federations. Different federations have different Federation Object Models (FOMs), therefore federates need to be built to support different FOMs. To facilitate federation formation, each federate is required to develop a Simulation Object Model (SOM). The SOM provides a means for federation developers to determine the suitability of simulation systems to assume specific roles within a federation. Once the federation has been formed, the FOM is developed. A FOM is an identification of the essential classes of objects, object attributes, and object interactions that are supported by a HLA federation. In short, simulations develop SOMs, SOMs support the development of FOMs, and FOMs support the creation and execution of federations. The issues of federates supporting different FOMs is described within the five case studies presented in this paper.*

INTRODUCTION

As a part of the High Level Architecture (HLA) baseline effort, the proto-federations developed Federation Object Models (FOMs). Each FOM contained qualities specific to that federation, whether it was a class hierarchy used for filtering, or a set of interactions only that federation could utilize. Since the HLA baseline was declared in August 1996, prototyping efforts have produced other sets of FOMs, also with characteristics specific to that federation. However, during this phase of implementing the HLA, federates are participating in multiple federations, rather than concentrating on one federation. This paper discusses the types of FOMs developed to support federates playing in multiple federations, the motivation for the development of each type of FOM, and the lessons learned in multiple FOM implementation. The case studies in this paper include descriptions of the Close Combat Tactical Trainer (CCTT) Semi-Automated Forces (SAF), the Air Combat Environment Test and Evaluation Facility (ACETEF), the Joint Precision Strike Demonstration (JPSD), Joint Modeling and Simulation System (JMASS), and Eagle, and the results they observed while implementing more than one FOM.

1. CCTT SAF

1.1 Description of CCTT SAF

The STRICOM/PM CATT Close Combat Tactical Trainer (CCTT) is the first fully DIS compliant training system that will train armor, cavalry, and mechanized infantry platoons through battalion/task forces on their doctrinal mission training plan collective tasks. The system consists of networked vehicle simulator manned-modules in combination with Semi-Automated Forces (SAF), and other workstation systems. The SAF system has the capacity to create a with variety of friendly or opposing force units and vehicles that exhibit highly realistic behaviors to support the CCTT training objectives. CCTT SAF was modified as a case study within a federation of three other DIS, platform-level simulation systems to demonstrate and evaluate the HLA approach.

1.2 CCTT SAF FOM Developments

CCTT SAF was a participant in the Platform Proto-Federation (PPF) and in a Multi-language/Multi-platform experiment. The FOMs for these two federations were designed with different goals in mind.

1.3 PPF FOM

CCTT SAF was modified as a case study within the PPF to evaluate and expose issues associated with HLA implementation. One of the goals of the PPF was to diverge from DIS where HLA provides potential benefits. The FOM developed for this study was based upon the sole objective of using the current capabilities of the four platform-oriented simulation systems to test the concepts of the HLA. As a DIS-based federation, the FOM represented a small portion of the data model defined by the DIS 2.0.4 standard. The object class structure (Figure 1) was essentially equivalent to the DIS Entity enumeration for those entities identified in the scenario. The class hierarchy, which was several levels deep, was used to filter data. The attribute table featured specialized attributes by class type, e.g., explicit articulation for appropriate classes. This allowed the federation to take advantage of inheritance. Specialized interaction events such as weapons fire versus weapons launch were used. And interactions used RTI IDs rather than DIS IDs. The Collision, Detonation, and Fire PDUs were translated into FOM interactions.

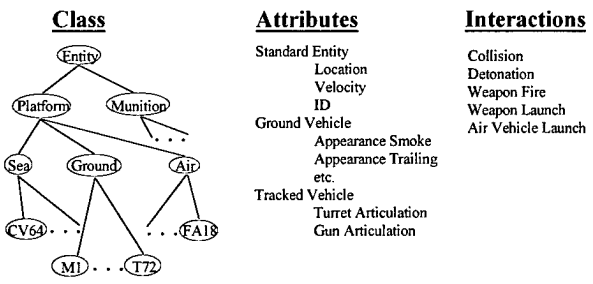


Figure 1. PPF FOM

1.4 Multi-language/Multi-platform FOM

In a second exercise, the CCTT SAF participated in a multi-language/multi-platform federation. This federation focused on the RTI version F.0 ports and testing, with the goal of minimizing divergence from DIS. For the multi-language, multi-platform experiment, the FOM closely resembled the DIS Enumerations. The class hierarchy (Figure 2) was based on the entity class, which eliminated the ability to filter data based on class. For this exercise, the CCTT SAF maintained its leaf class types purely to reduce code modifications. The attributes were based on the Entity State PDU and bit packing. The interactions were basically recoded DIS Event PDUs, where Entity IDs, rather than RTI IDs were used.

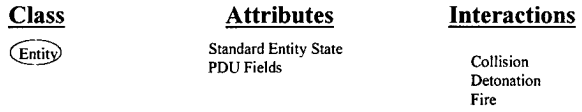


Figure 2. Multi-language/Multi-platform FOM

1.5 CCTT SAF FOM Implementation

The HLA implementation for CCTT SAF (Figure 3) involved the application, entity database, and PDU queues operating much as they did in their DIS configuration, with minimal changes. To support HLA, a translation between the DIS data model of CCTT and the FOM defined data model was required. Translated data was sent to other federates through the RTI / Federate Interface services to the RTI. Changes to the FOM required changes to the FOM Data Model translation.

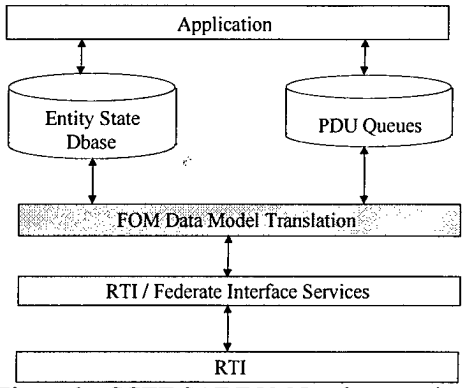


Figure 3. CCTT SAF FOM Implementation

2. ModSAF as a Part of the JPSD

2.1 Description of JPSD.

The goals of the JPSD experiment were to integrate an existing federation of various simulation types using the HLA, specifically using the RTI version x.3. With the number of different federates and short integration schedule, a common set of software, known as the federate common software (FCS), was developed. This development allowed the integration of the RTI at the output API, but within the same process space of the various federates. Although this approach did not take advantage of the complete set of HLA services, it allowed the verification of the RTI API and its capability to support a complex mixed federate type federation. The federate common software (built in C++ using OO design methodologies) is composed of several SW objects grouped in three functional areas: the RTI API interface services, the federate interface services, and the FOM evaluation services. The software objects were designed to be reusable, i.e., only extensions to base classes are necessary to adapt the software to various federates and FOMs.

- Each ModSAF Federate is an Executable

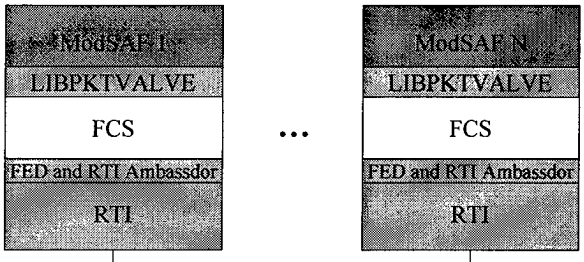


Figure 4. HLA ModSAF Architecture

2.2 JPSD FOM Developments

ModSAF participated in two federations. The first federation supported JPSD. The goal of this development was to provide the JPSD functionality using HLA. The second federation was the Multi-language / Multi-federation experiment discussed in previous paragraphs.

2.3 JPSD FOM

For the JPSD Experiment, the FOM development was based on an existing federation concept currently implemented in DIS. The scenario was already well established. The existing JPSD Interface Requirement Specification (IRS) was translated to the Object Model Template (OMT) format, and the JPSD interest management scheme was refined from multicast groups to the HLA interest management scheme. The resulting FOM was a DIS-like data representation, where class attributes were a minimal field of the Entity State PDU for each entity type. Interactions were used for sporadic PDUs, tactical messages, hand-off to engineering models, and aggregation / disaggregation. The JPSD FOM needed to support both entity level and aggregate level object representations. The component table specified mapping between aggregate and entity representation. The data structure table defined the complex attributes.

2.4 Multi-language/Multi-platform FOM

For the Multi-language/Multi-platform experiment, the FOM development process began by examining the existing PPF and JPSD FOMs previously implemented for CCTT SAF and ModSAF. In order to minimize development, both parties agreed to a merged FOM. The Multi-language/Multi-platform FOM thus defined the naming conventions and class structures, and used the DIS 2.0.4 Enumerations and PDU contents to form the basis for attributes and classes. As discussed earlier, the Multi-language/Multi-platform FOM contained a flat structure, i.e., all the attributes were specified at the entity level. The sub-classes were defined for class based filtering at the entity type level.

2.5 JPSD FOM Implementation

The JPSD Testbed utilized Federation Common Software (FCS) (Figure 5) in order to facilitate the integration of Corps Level Computer Generated Forces (CLCGF) and HLA Testbed simulations with the RTI. The FCS provides many functions: encapsulation and automation of services all simulations must exercise; FOM management and RTI services; a framework for translation between simulation and FOM data representation; and a common instrumentation for performance analysis. The JPSD specifically uses simulation translation actors within this FCS to allow for multiple FOMs. The actors are FOM specific.

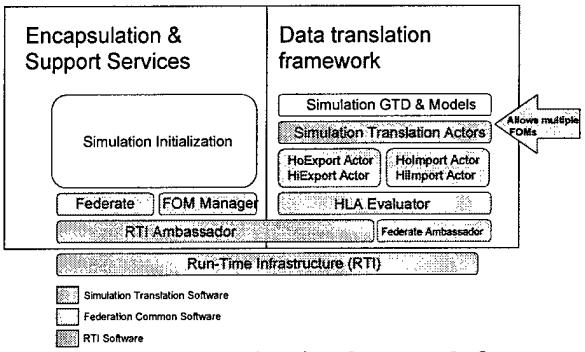


Figure 5. JPSD Federation Common Software

3. Air Combat Environment Test and Evaluation Facility (ACETEF)

3.1 Description of ACETEF

The Air Combat Environment Test and Evaluation Facility (ACETEF) (Figure 5) is a fully integrated ground test facility that supports test and evaluation of multi-spectral offensive and defensive aircraft systems in a secure and scientifically controlled engineering environment. ACETEF is located within the Naval Air Warfare Center, Aircraft Division (NAWC-AD) at Patuxent River, Maryland, and is an enterprise team consisting of the Warfare Simulation, Aircraft Simulation, Electronic Warfare Stimulation, and the Electromagnetic Environmental Effects (E3) Stimulation teams. ACETEF led the Engineering Proto-Federation, which successfully completed its mission to "determine the suitability of the HLA to support detailed, high-fidelity simulations in the context of a realistic engineering situation". The Engineering Proto-Federation developed a FOM based on the capabilities and projected needs of its federates and the requirements of a classified, JCS-approved scenario of the type demanded for simulation-based acquisition processes. Most of the requirements were centered in the Electronic Combat (EC) test arena.

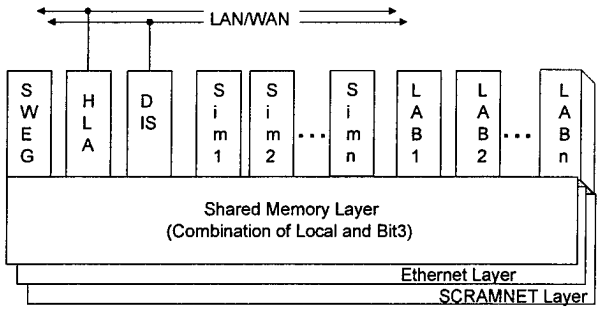


Figure 6. ACETEF Architecture

3.2 ACETEF System Architecture

ACETEF's main simulation integrator is the Simulated Warfare Environment Generator (SWEG). The primary function of this software object is to manage a dynamic representation of the state of the total simulation (much like a stealth observer does for humans). This representation is accessed by all of the computers on the facilities secure network. This state data is called SWEDAT and is physically mapped to shared memory which is reflected in every host through VME-based SCRAMNET® and BIT-3® memory. All ACETEF hosts use an instance of the low-level SWEDAT access routines. These routines are collectively known as the SWEG Interoperability Software Object (SISO). HLA and/or DIS software objects are loaded into their own host and are interfaced, through local shared memory to an instance SISO. If necessary, SISO and the HLA software objects can be loaded into two processors of the same host. The function of the HLA software object is to provide input/output between the SISO-controlled local shared memory and the RTI. Incoming objects are mapped onto SWEG Semantic codes by the HLA software object using the "Entity File" (Figure 6). Outgoing objects are mapped onto HLA object classes, or DIS enumerations, using the same file.

SWEG Semantic Code	DIS Enumeration	HLA Object Class
710006	1.2.225.1.1.0.0	Airplane
710007	1.3.225.1.1.1.0	Boat
710006	1.2.225.1.1.0.0	MilitaryPlatform
710007	1.3.225.1.1.1.0	MilitaryPlatform

Figure 7. Entity File Layout

3.3 ACETEF FOM Implementation

The chief difficulty to overcome when moving from federation to federation is the fact that, by definition, different federations use different FOMs. In order to utilize different FOMs, the "Entity File" must contain all of the classes that ACETEF subscribes to and/or published. ACETEF's approach to this is simple. New, previously unknown classes used by other federations will be added to the Entity File. Old object classes would be left in the entity file so that a capability to operate in either (or both) federation(s) is preserved. This architecture will theoretically allow ACETEF to operate simultaneously in multiple federations while preserving support for the pre-HLA DIS protocols currently in use by some of their customers. ACETEF is required to support multiple, back-to-back, and sometimes simultaneous, customer operations using different protocols. The investment in and development of a multi-protocol support capability such as ACETEF's is not appropriate for every situation by may, in many cases, be an applicable and cost-effective approach.

4. JMASS/JPSD

4.1 Description

Joint Modeling and Simulation System (JMASS) is a modeling and simulation system that provides a software environment for the development, execution, and post-processing of simulations. JMASS implements a standards based set of tools that assist in the development, maintenance, and use of reusable models and model components. JMASS is designed specifically to fulfill the simulation architecture role as defined in the emerging standard test processes from within the Air Force and DDT&E organizations. JMASS implements a single thread of functionality through all the major activities found in modeling and simulation tasks. During testing, the simulation executed under the JMASS architecture provided the Engineering Proto-Federation the simulation of threat weapons, a surrogate interface to the ADARS, and the constructive simulation of weapon engagement outcomes.

4.2 JMASS FOM Development

JMASS participated in two federations, the Engineering Proto-Federation (EPF) and the JPSD Federation. The EPF FOM development was based on the capabilities of the participants and the resulting

scenario. The JPSD federation, as described earlier, had a well-defined scenario and was focused on providing an existing capability using HLA.

4.3 Engineering Proto-Federation FOM

For the EPF, the FOM developed contained hierarchically based entities. The attribute details were contained in the EPF Interface Control Document. The FOM was largely DIS entity based, which significantly reduced the complexity. During the EPF exercise, JMASS used Port Objects as standard JMASS inter-model data interfaces. This provided models access to/from EPF FOM data. The federates interfaced to the RTI through the EPF middleware (Figure 8), which was developed mostly to perform data logging. Little, if any, data translation was performed by this layer.

4.4 JPSD FOM

The JPSD FOM also used hierarchically based entities, however the hierarchy was at a different level than the EPF FOM. The details were contained in the attribute/parameter tables, and were defined across and down the hierarchy. The FOM again was largely DIS entity based, with full PDU complexity. In order to support the JPSD FOM with the JMASS EPF models, JMASS used the same EPF FOM based port objects. However, this time the EPF middleware was modified to provide translation to and from the EPF to JPSD FOMs, particularly for the missile and aircraft models.

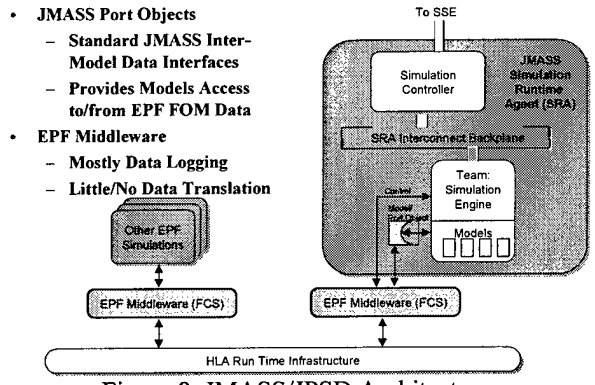


Figure 8. JMASS/JPSD Architecture

5. Eagle

5.1 Eagle Description

Distributed Eagle, i.e., Eagle-to-Eagle, was developed to eliminate the single processor constraint of no more than 220 units at a run speed of six to one. Initially, the Aggregate Level Simulation Protocol (ALSP) was used for the Eagle-to-Eagle confederation. Multiple copies of the simulation hosted on multiple processors were connected through the ALSP. Each processor was capable of representing 220 units at six to one run time. Theoretically, an unlimited number of units could be represented using Distributed Eagle. Combat units hosted on each processor appeared as 'ghost units' on all other processors. The 'ghost units' have full interaction with hosted units to include detection, receiving attrition, and causing attrition. In 1996, Distributed Eagle was moved from ALSP and fully implemented under the HLA. The multiple processors now communicate through the RTI. Overhead demand by HLA on the processors is minimal and has almost no effect on run time speed. Current applications include the Joint Precision Strike Demonstration (JPSD). The 1996 version of JPSD ran with 600 units using two processors under HLA. By contrast, the 1995 JPSD was limited to only 200 units using Eagle on a single processor.

5.2 FOM Experiment

The Eagle system has participated in three federations: the Early Eagle Analysis Experiment, the Joint Training Federation (JTFp), and the Eagle to MCSP/Beta-WarLab (Figure 9). Since it was constructed to support multiple situations, the Eagle SOM was used in all three cases to support development of a federation FOM. Each of the FOMs developed are characterized by class structures that feed into a bottom level class. The attributes are used to define reflected units. A typical ground combat unit of Eagle is defined by approximately 450 attributes.

5.3 Distributed Eagle FOM

In distributed Eagle, the number of attributes to reflect units is 43 to define, and 31 to update. In the Distributed Eagle FOM, the main interactions included shooting, communications, data base consistency, DIS interactions, and surrogate C4I. The interaction structure included 23 total interaction types.

5.4 Joint Training Federation FOM

In a like manner, the number of attributes to reflect aggregate ground units in JTFp is 29 to define and 17 to update. To support the JTFp object class and attributes, Eagle required some modification. Specifically, new classes were created and the Eagle combat units and attributes were translated to conform to the JTFp FOM. The JTFp FOM included federation

management, shooting, communications, and environmental interactions. This FOM defined 24 interactions, and Eagle subscribed to 10.

5.5 MCSP/Beta-WarLab FOM

The Eagle MCSP Beta FOM included only two types of interactions, simulation management and communications. The FOM defined 10 interactions from Eagle to MCSP.

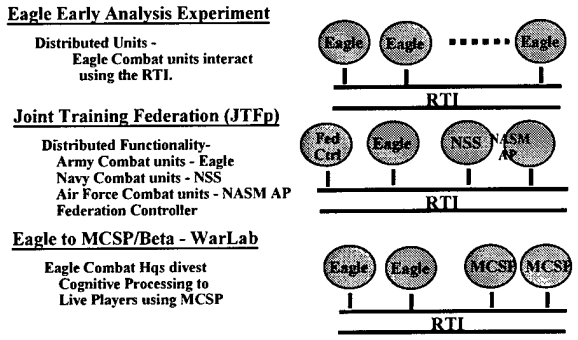


Figure 9. Eagle Federate Architecture

6. Conclusion

Based on the HLA experience cited, it can be concluded that HLA provides the conditions under which reuse can be gained. Specifically, HLA facilitates the participation of federates in more than one federation by allowing flexibility in defining attributes, interactions, and parameters. It is still true that federations should be built for specific purposes, but the ability of a federate to move from one federation to another will depend on the similarity of the origin and destination federations. It is possible, however, for federates to mitigate risk associated with being required to utilize a FOM that their SOM may not have designed to support.

Experience has shown that federates are able to switch between fundamentally different federations, yet this switch is not easily accomplished, nor is it always recommended. Ideally, a federate does whatever its SOM indicates it can do, and candidate federates are chosen by the content of their SOMs. Initially, a federation's FOM is created as a composition of the individual federate SOMs. Thus rework is accounted for and minimized at that point, resulting in a federation with a FOM to which everyone has agreed. Thus, when like federates join federations, FOMs are often similar from one exercise to the next, minimizing development impact. However, when federates are required to interoperate with a member or members of a federation built for purposes entirely different than that federate's purpose, new approaches need to be considered.

Federates, in general, should develop a software mechanism to flexibly map their SOMs into differing FOMs. This mechanism may take the form of a parameterized set of middleware, new specifications, or new object types. However, any mechanism chosen should allow a federate to participate in multiple federations, and should require the minimum in software code changes.

7. Author's Biographies

BRIAN W. BEEBE is an Assistant Vice President and Chief Scientist for Science Applications International Corporation (SAIC), has been involved with the development of software intensive systems, particularly Electronic Warfare simulation and training systems for over 20 years. Mr. Beebe graduated from Wright State University with a degree in Computer Science. His software background includes systems and software engineering for object-oriented computer modeling and distributed simulation systems and architectures, as well as interactive computer graphics, communications systems, and real-time cockpit and interpretive computer simulations. Involved with the Joint Modeling and Simulation System (JMASS) program since 1991, Mr. Beebe was the industry chair for the JMASS Architectural Design Team, that developed the JMASS architecture concept now being pursued by the JMASS Program Office (ASC/SMJ). Currently, Mr. Beebe is the chief technology engineer for SAIC's JMASS Technology efforts with the Air Force's Wright Laboratory and managed SAIC's development of the JMASS High Level Architecture (HLA) Federate prototype portion of the DMSO AMG Engineering Proto-Federation.

CHRISTINA L. BOUWENS is a Senior Simulation Systems Engineer and a member of SAIC Orlando's Advanced Simulation Research Team. Ms. Bouwens is currently the Project Director for the ADST II HLA Support Experiments delivery order to conduct development and experimentation in support of HLA 1.0 development and its current evolution. Ms. Bouwens has been an active part of the interoperability standards development community since 1989 and is currently a member of the SISO Transition Team and the new Executive Committee. Ms. Bouwens received her M.S. degree in Mathematical Science from the University of Central Florida.

WESLEY BRAUDAWAY, Ph.D. is a member and technical lead of the Advanced Simulation Research Team within SAIC Orlando's Operation. Dr. Braudaway has been the lead engineer for the ADST II HLA delivery order to conduct development and experimentation in support of the HLA 1.0 development and its current evolution. Dr. Braudaway received his Ph.D. from Rutgers University's Computer Science Department and has been actively involved in the M&S community since 1991.

SUSAN M. HARKRIDER is a Systems Engineer in the Engineering Directorate at the U.S. Army Simulation, Training, and Instrumentation Command (STRICOM). Ms. Harkrider is currently providing technical support to the Defense Modeling and Simulation Office for several projects, including the procurement of RTI version 2.0. She was the lead Government engineer for the HLA Platform Proto-Federation project. Ms. Harkrider holds a B.S.E. from the University of Central Florida, and is a M.S. student in Simulation Systems at UCF.

JOHN W. OGREN is a Simulation Modeling Engineer with the MITRE Corporation, and is responsible for the integration of the combat model Eagle with the HLA. He is currently working at the Ft. Leavenworth site supporting the TRADOC Analysis Center, the National Simulation Center, and the Battle Command Battle Laboratory.

DANA A. PATERSON is a U.S. Government Computer Scientist with 15 years experience in Trajectory Measurement Systems as applied to Aerial Targets Development and Threat Simulation for Test & Training. Mr. Paterson is the Distributed Simulation Projects Coordinator at Naval Air Warfare Center - Aircraft Divison in the Air Combat Environment Test & Evaluation Facility (ACETEF) and is a member the Simulation Interoperability Standards Organization's Coordinating Committee (SISO - CC). Mr. Paterson holds a B.S. in Computer & Information Science from the University of Oregon, 1985. He is a U.S. Navy Operations Specialist.

RUSS RICHARDSON is presently a member of the DMSO RTI Integrated Product Team and is the RTI testbed manager. In addition Mr. Richardson is the technical director for the Joint Precision Strike Program. In that capacity he is responsible for the development and utilization of a wide variety of simulations and C2 systems for analysis, prototyping, testing, and training. Mr. Richardson has been actively involved with simulation technologies for 15 years and with the definition of the HLA.

PHILOMENA ZIMMERMAN is the Distributed Simulation Project Lead for the Warfare Simulation Branch of Air Combat Environment Test and Evaluation Facility (ACETEF) at the Naval Air Warfare Center - Aircraft Division (NAWC-AD). She is currently involved in the HLA as a member of the Engineering Federation, a member of the HLA TSTCore, and lead of the HLA Security IPT. She is also the technical lead for the Warfare Simulation Branch in the JADS-EW Test. Ms. Zimmerman has been involved in the interoperability standards effort since 1990, and is currently a member of the PRP for the Federation Development Process Forum. She received her B.S. in Mathematics from St. John Fisher College.

8. Acknowledgments

The authors would like to acknowledge DMSO for sponsoring much of the work discussed in this paper, including the Proto-Federations and the JPSD Experiment.

Object Model Development: Tools and Techniques

Christina Bouwens
Science Applications International Corporation
12479 Research Parkway
Orlando, Florida 32826-3248
Christina.Bouwens@cpmx.saic.com
(407) 207-2742

Raymond L. Miller
Science Applications International Corporation
12479 Research Parkway
Orlando, Florida 32826-3248
millerr@orl.saic.com
(407) 207-2785

Roy Scrudder
Applied Research Laboratories:
The University of Texas at Austin
P.O. Box 8029
Austin, Texas 78713-8029
scrudder@arlut.utexas.edu
(512) 835-3857

Robert Lutz
John Hopkins University / Applied Physics Laboratory

Johns Hopkins Road
Laurel, Maryland 20723
robert.lutz@jhuapl.edu
(301) 953-5000

Keywords:
Federation, Federation Object Model (FOM), FEDEP, Object Model Data Dictionary (OMDD), Object Model Library (OML), Object Model Development Tools (OMDT)

ABSTRACT: *(This paper presents information representing the HLA development process underway by the Defense Modeling and Simulation Office (DMSO) and the Department of Defense (DoD) Architecture Management Group (AMG)).*
The development of a Federation Object Model (FOM) is a critical activity within the Federation Development and Execution Process Model (FEDEP) and represents much of the work in assembling a Federation Execution for a particular sponsor's objectives. To support FOM development, the Defense Modeling and Simulation Office (DMSO) has been sponsoring the development of an integrated object model tools suite to support development and management of HLA object models. A key component of the object model tool suite is the Object Model Data Dictionary System (OMDDS). The FEDEP recommends an object model development approach that begins with the Conceptual Model and a standardized data dictionary from which a federation developer can build their FOM from the "bottom up." Another approach is to utilize existing FOMs or "Reference FOMs" developed by a particular community or simulation domain and tailor the FOM for the specific simulation needs of the developing federation. This tailoring represents more of a "top-down" approach to FOM development. This paper reports on a recent study that examined several approaches to FOM development while utilizing components of the object model tool suite. The paper provides guidance to help a federation developer determine which FOM development tools and techniques would best suit their particular needs. It also discusses the use and utility of the object model tool suite: the OMDDS in providing content for the creation of FOMs, the Object Model Library (OML) in providing existing FOMs and SOMs for use in creating new FOMs, and the Object Model Development Tools (OMDTs) in using OMDDS content to build FOMs.

1. Introduction

The development of a FOM is a critical activity within the Federation Development and Execution Process (FEDEP) Model and represents much of the work in assembling a Federation Execution. A good understanding of the development process and available tools provides the federation developer with the means to develop a FOM that is well designed for federation use and reusable for other federation executions.

This paper discusses the FOM development process, its place within the FEDEP and the tools used throughout. It also provides information on the various tools used during the FOM development process. The OMDD experiment is discussed along with its results. The paper concludes with some developer guidance and references for where to get more information for this process.

2. FOM Development and The Federation Development and Execution Process

The FOM development process is part of a larger process called the FEDEP. The FOM development process receives valuable inputs from the FEDEP and provides critical output to the FEDEP. It is important to understand the place of the FOM development process within the FEDEP.

2.1 FEDEP Process

The FEDEP document [1] describes a recommended process for development and execution of an HLA federation execution. The five-step process recommended in FEDEP v1.1 is shown in **Figure 2-1**, and is summarized below:

Step 1: The federation sponsor and federation development team must define and agree on a set of objectives and document what must be accomplished to achieve those objectives.

Step 2: A representation of the real world domain of interest (entities and tasks) is developed, and described in terms of a set of required objects and interactions.

Step 3: Federation participants are determined (if not previously identified) and a FOM is developed to explicitly document information exchange requirements and responsibilities.

Step 4: All necessary federation implementation activities are performed, and testing is conducted to ensure interoperability requirements are being met.

Step 5: The federation is executed, outputs analyzed, and feedback provided to the federation sponsor.

It is critical Steps 1 and 2 are accomplished before FOM development begins as part of Federation Design.

2.2 Description of the Federation Object Model Development Process

A full description of the OM development process may be found in [2]. The process is summarized below:

2.2.1 Summary of FOM Development Approaches

There are several different fundamental approaches to the development of HLA FOMs. In addition to bottoms-up approaches, in which one builds the FOM from "darkness" based on the federation conceptual model and the Object Model Data Dictionary (OMDD), other approaches may be appropriate which provide more emphasis on the reuse of existing object models. Possible alternative approaches include:

- Merge together Simulation Object Models (SOMs) of participating federates, removing aspects of SOMs that do not apply to the domain of interest.
- Begin with SOM that most closely aligns with desired FOM, remove aspects of that SOM which do not apply to the domain of interest, and merge-in elements of other SOMs to fully represent domain.
- Begin with FOM from previous, but similar application. Modify and/or augment as required.
- Begin with Reference FOM. Remove elements of Reference FOM that are not required for the immediate application. Modify and/or augment only if necessary.

The starting point for the HLA FOM development process is the Federation Development phase of the HLA FEDEP model. Implicit to this assumption is that the federation requirements have been clearly delineated, the scenario(s) have been defined, the real world objects and interactions that are to be represented explicitly in the federation have been identified (and, where possible, mapped to OMDD entries), and the federation participants have been both identified and fully described according to an HLA SOM.

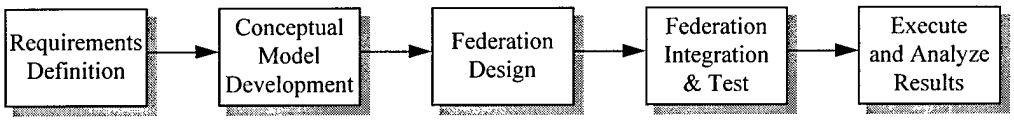


Figure 2-1 Five-Step Process

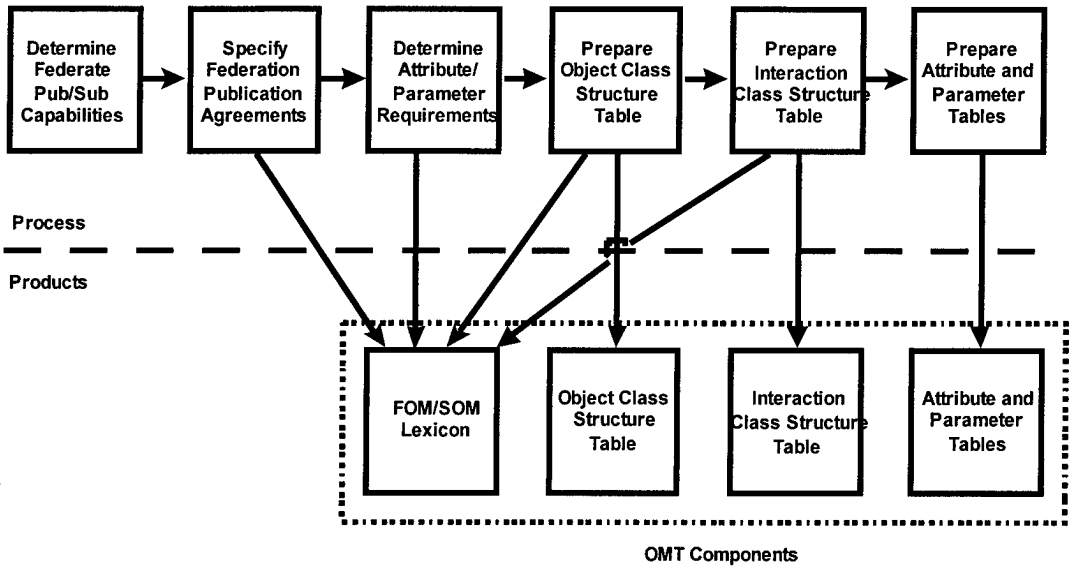


Figure 2-2 FOM Development Process

2.2.2 FOM Development Process

While each of the above FOM development approaches may represent a somewhat more efficient FOM development strategy (relative to starting entirely from scratch) under certain circumstances, all will require some use and appropriate tailoring of the essential activities described in the current HLA Object Model (OM) Development Process. A summary of these activities is provided in Figure 2-2. Federation security personnel must always maintain knowledge of any classified information associated with applicable entries in each federates SOM, and the implications when this data is combined into a single FOM.

The use of automated tools to facilitate the object model development process is strongly encouraged and the subject of this paper. The HLA Object Model Library (OML) provides user access to libraries of reusable object models that can be used either as a starting framework or as individual "piece parts" for a new FOM. In addition, Object Model Development Tools (OMDTs) may be used to modify or extend an existing object model, or to build an entirely new object model from scratch. Other OMDT features include consistency checking, syntax checking, Federation Execution Data (FED) file generation, external interfaces to commercial object model development tools, and an on-line user's manual.

In addition to FOM development, there are other important activities that take place in this phase of the FEDEP. Negotiations must take place among federation members and agreements reached regarding supporting resources that will be common across the federation. For instance, decisions must be made regarding any databases (e.g., environmental) or algorithms (e.g., line-of-sight) that must be common across all federation participants. Another important activity is to transition the functional description of the scenario (developed in an earlier phase) to an actual scenario instance (or set of instances) based on authoritative data sources. The output of this last activity permits federation testing to be conducted directly within the context of the domain of interest, and also drives the execution of the federation later in the FEDEP.

3. Object Model Tool Suite

3.1 Description of the Tools and Their Use

From the early days of HLA experiments with the protofederations, the need for automated tools to support the Federation Development and Execution Process (FEDEP) has been evident [3]. To meet this need, the Defense Modeling and Simulation Office (DMSO) sponsored the development of the HLA Tools Architecture, which defines a core set of tools to support the FEDEP and interfaces between these tool sets [4]. Also under DMSO sponsorship, an initial suite of HLA support tools has been developed, specifically to support the object modeling phase of the FEDEP. Components of the initial Object Model Tools Suite and their

information exchange relationships are shown in Figure 3-1.

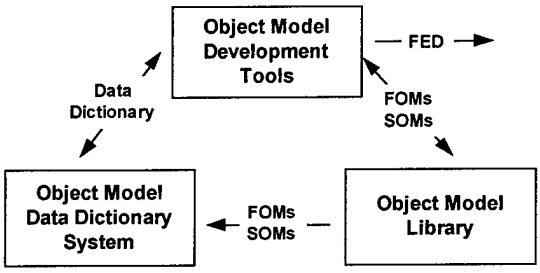


Figure 3-1. Object Model Tools Suite

The Object Model Development Tool used in the OMDD Experiment was developed by AEgis Research. It is a Microsoft Windows 95/NT 4.0 application which allows users to create and edit FOMs and SOMs compliant with the Object Model Template [5]. The core set of functionality includes data entry and modification, syntax checking, automated access to the Object Model Library and the Object Model Data Dictionary System, and automated generation of Federation Execution Data necessary for the initialization of an HLA Run Time Infrastructure.

The interface between the OMDT and the OMDDS is unique to the version of the OMDT used for the OMDD Experiment. The version of the AEgis OMDT released to the HLA community in October 1997 does not have this feature. The OMDT's interface to the OMDD allows users to import one or more data dictionary files and pick from the contents of the data dictionary files when populating or modifying an object model. Data dictionary choices are provided in a "pick list" fashion, within the context of the object model table a user is editing. For example, when a user is editing the object class table of a FOM, only the object classes exported from the OMDD are available for inclusion in this table, not other components such as complex data types or interaction classes.

The Object Model Library is a central repository of reusable FOMs and SOMs. It is available through a web-based interface and provides mechanisms to search and browse object models in the library, download copies of object models, and submit new object models for inclusion in the library.

The Object Model Data Dictionary System (OMDDS) is the newest component of the Object Model Tools Suite. It is a central repository of reusable components for the construction of FOMs and SOMs and is available through a Web-based interface. The OMDDS provides users with the capability to browse and search OMDD components and select subsets for export and use with the OMDT. User selections for export are maintained persistently, so a user can build up a list of items to export from the OMDD over the span of multiple web access sessions.

The OMDD contents accessible through the OMDDS are separated into five component types: object classes, interaction classes, generic elements, complex data types, and enumerated data types. All of these component types, with the exception of generic elements, map directly to Object Model Template concepts. Generic elements correspond to both attributes and parameters in the Object Model Template. In effect, both attributes and parameters are the same type of component, i.e., a property of a class. The only distinction between attributes and parameters is whether they are a property of an object class or of an interaction class. A generic element from the OMDD (e.g., velocity) could be used as either an attribute or a parameter. Each generic element in the OMDD has one or more representations that specify a unit of measure (if applicable) and a data type. Data types for generic elements may be any of the base data types from the OMT or one of the enumerated or complex data types also contained in the OMDD. For complex data types, the OMDD specifies a list of fields and the corresponding generic element for each field.

OMDD contents available at the time of the OMDD Experiment were developed in a bottom-up fashion based on components found in three widely used FOMs:

- Real-time Platform Reference FOM - A FOM widely used in the transition of legacy systems from the Distributed Interactive Simulation environment to HLA.
- Joint Training Confederation FOM - A FOM developed for the transition of legacy systems from the Aggregate Level Simulation Protocol environment to HLA.
- Engineering Federation FOM – A FOM evolved from the Engineering Protofederation experiment that has been used in the federation of several engineering-level simulations.

Components from these FOMs were merged into a common set of OMDD components. These OMDD components expanded upon the definitions and representations specified in the FOMs by using information from authoritative sources including the Defense Data Dictionary System and the JCS Pub 1-02, Dictionary of Military and Associated Terms.

4. Object Model Development Experiment

The OMDD experiment sought to prototype the FOM development approaches discussed in Section 2.2.1 utilizing the tools in the OM Tool Suite. The experiment approach was to step through the FEDEP and the FOM Process Model, prototype FOMs using several different approaches, and provide feedback and lessons learned concerning the processes and tools employed.

4.1 FEDEP Preliminaries

Before prototyping the FOMs, it is necessary to go through the first two steps of the FEDEP. These define the problem space for which the FOMs will be built.

4.1.1 Requirements Definition

As part of Requirements Definition the project team developed the following **Problem Statement**:

> With the proliferation of SAF systems, there is a concern in the community that the resulting simulations may not be "fair" due to varying modeling techniques and resolution. The problem is how to determine whether or not the fight is fair. This begins with the determination of whether the results of a particular engagement are the same for two different SAF systems.
>
> For this particular experiment, ModSAF and CCTT SAF will be utilized at platform-level resolution. Proposed measures of merit (MOM): Timing at various points of the engagement, results of the engagement.

A set of **Objectives** were then developed for the experiment:

1. Develop a scenario that will allow opposing forces to engage in an attack and defense situation. This scenario must be usable by SAF.
2. Run the scenarios in a variety of configurations of the SAF systems to examine the systems individually as well as engaged against one another for comparison.

The **Environmental Context** for the experiment would be the use of the Grafenfels database (normal weather and environment conditions). Hasty Attack and Hasty Defense with M1 and T72 tanks supported by Hinds and Apaches were examined.

There were no **Security Issues** for our experiment. The experiment would be conducted as Unclassified.

The **Equipment, Facilities and Data** to be used are shown in Table 4-1.

No **VV&A** was to be performed for this experiment. **Test Plans** included unit testing for interfaces and consistency testing for developed object models.

Table 4-1 Equipment & Facilities

Federates	ModSAF and CCTT SAF
Facilities	SAIC SAFLab utilizing four Motorola machines, one Sun Ultra workstation and one SGI Indy2
Terrain Database	Grafenfels
RTI	Version 1.0, release 2
Other tools to be used	OMDDS, OMDD, OMDT, OML, MSRR, Federation Developers Workbook, HLA documentation

4.1.2 Conceptual Model Development

Once the requirements for our federation are identified, the next step is to develop the Conceptual Model. This part of the process includes Scenario Development and the Conceptual Analysis.

4.1.2.1 Scenario Development

The **Scenario Development** process began with the search for existing scenarios that could be reused for the experiment. The use of scenarios from the A2ATD experiment was explored. The experiment borrowed two scenarios, slightly modified, for the purposes of the experiment. It should also be noted that the problem being addressed by the OMDD experiment differed from that of the scenario's original experiment. The Modeling and Simulation Resource Repository (MSRR) was also searched for other possible scenarios, but no specific scenario could be identified for our experiment. The selected scenario was defined as follows:

1. Hasty attack involving a company of BLUFOR (M1s) and a Platoon of OPFOR (T72s) with close air support provided by Apaches (for the BLUFOR) and Hinds (for the OPFOR).
2. Hasty defense involving a company BLUFOR (M1s) and a Bn of OPFOR (T72s).

The scenario requires the use of M1s, T72s, Apaches and Hinds, which will need to move, shoot and take damage during the scenario. No filtering requirements (DM) were defined at the time of the scenario development.

Both scenarios will be run using the following conditions: 1) CCTT Blue SAF against CCTT Red SAF, 2) ModSAF Blue SAF against ModSAF Red SAF, 3) CCTT Blue SAF against ModSAF Red SAF, and 4) CCTT Red SAF against ModSAF Blue SAF. To obtain a good sampling, each scenario with each condition will be run five times for a total of forty runs.

4.1.2.2 Conceptual Analysis

Class Analysis

Based on the scenario described above, the simplest class structure to support our experiment was developed for our Conceptual Model. The Conceptual Model was developed without regard to the federates participating in the federation. The goal was to create a Conceptual Model that met the requirements of the scenario. Federate specific details would be addressed during Federation Design. A rough model was developed which represented tanks and helicopters as primary classes. The resulting rough model included base classes of Tank and Aircraft with M1 and T72 as subclasses to tank and Apache and Hind as subclasses to Aircraft. Identified interactions included Fire, Detonation and Collision.

Once these elements were identified, the OMDDS was utilized to access the class elements in the OMDD. During this process, elements of the rough model were used to search and browse through the OMDD (using the OMDDS) and selected elements which supported the Conceptual Model. These elements were exported from the OMDD and brought into the OMDT, which was used to document the final Conceptual Model. The resulting OM developed with the OMDT is called the "Ideal FOM." Our Conceptual Analysis yielded the following Class structure:

IDEAL FOM CLASSES

Military Platform (PS)	Military Ground Vehicle (PS)
	Rotary Wing Aircraft (PS)

The type of ground vehicle and RWA would be captured within the attributes through enumerated types.

It should be noted that when browsing the OMDD, multiple elements were identified that could be used to support the Conceptual Model. For example, Apache and Hind aircraft were found under Rotary Wing Aircraft and under Aircraft. Selection of either element would have satisfied our requirements.

Interaction Analysis

Three interactions were identified for our federation: Fire Weapon, Detonate Munition and Collide (vehicle). These three interactions were found in the OMDD and exported to our Ideal FOM:

INTERACTIONS

Fire (IR)
Detonate (IR)
Collide (IR)

Attribute / Parameter Analysis

Although a minimum set of attributes and parameters had been identified during the conceptual analysis, the final selection of parameters would be related to the particular FOM development approach being explored and some basic assumptions about the systems to drive the tradeoffs. For the purposes of our conceptual model, we included the minimum attribute set as a starting point. Other related attributes / parameters were identified and brought into the Ideal FOM during its development. Analysis of specific requirements would come during the FOM development process.

For the various classes, minimum attributes included: type, location, velocity, appearance (state), side (Blue or Red). For the three interactions, the following minimum parameters are defined:

	Minimum Required Parameters
Fire	-who fired -what was fired - when was it fired - intended target
Detonate	- where detonated - who was affected - type of round (what) - corresponding fire event - resulting appearance
Collision	- who collided - resulting damage (appearance)

Using the requirements above, elements in the OMDD were identified and exported to our OMDT for inclusion in the Ideal FOM.

The Conceptual Model is now complete and documented using the OMDT in conjunction with the OMDD. Our product is an Ideal FOM that will be used as a starting point for our bottoms-up approach and reference for other approaches.

4.1.3 Federation Design

Federate Selection

Although the plan was to use CCTT SAF and ModSAF for the experiment, the Object Model

Library (OML) was checked to identify any other any federates that might qualify for the experiment. Once the Conceptual Model was completed, the project team logged into the Object Model Library (OML) to other federates (in the list of SOMs) which had the kind of capabilities needed for our experiment. Our two federates' SOMs were found in the OML along with one FOM to be used as our starting point for the Reference FOM approach: CCTT SAF SOM, ModSAF FCS SOM and the RPR FOM. These Object Models were easily downloaded from the OML to our local PC.

Although the class structures were quite different between the SOMs (and even with the RPR FOM), the leaf nodes proved to be nearly identical. This was due, of course, to the fact that the three object models had their legacy in DIS.

Because of the capabilities of the federates, we determined that both would subscribe and publish to all the objects identified in the scenario except for the RWA. For the Apache and Hinds, CCTT would simply subscribe and not publish. ModSAF would publish and subscribe.

4.2 FOM Prototype Development

As discussed in Section 2, there are a number of approaches to FOM development. These approaches focus on whether the developer begins from scratch (bottoms-up approach) or begins with an existing FOM or SOM. Since several of the approaches are similar in their implementation, we reduced the number of cases prototyped to three:

1. Bottoms-up Approach
2. Single SOM / Merge SOM Approach
3. Similar FOM / Reference FOM Approach

The following sub-paragraphs discuss the manner in which each approach was implemented along with the results of that approach.

It should be noted that a number of tradeoffs drive the selection of certain attributes, parameters, class structure, etc. Such decisions typically take place during federation design meetings where all involved federate developers coordinate the choices. For the purposes of this experiment, we made some simple assumptions regarding our requirements.

4.2.1 Bottoms-Up Approach

The Bottoms-Up approach to FOM development begins with the Ideal FOM developed during Conceptual Analysis. For this approach, the SOMs for the federates to participate would be used to augment the FOM with additional details. When new elements are added to the FOM, the OMDD is consulted and its terminology adopted for the FOM. This approach did not take into consideration the impact that FOM changes that would have on federates in order to support the FOM structure.

Class Structure

This approach involved beginning with the Ideal FOM and modifying it as required to support the experiment with the selected federates. The result was a slightly modified FOM from the Ideal FOM with the addition of one level to the class hierarchy in order to support munitions. This was based on the need to add a class for munition entities to support Fire and Detonation Interactions. We also changed the RWA subclass to Aircraft to be more consistent with the generality of Military Ground Vehicle.

Interactions

Utilizing the SOMs / FOMs we had downloaded and selected from the OML, an analysis of the Interactions was performed. Analysis was very simple since both federates supported essentially the same interactions. The resulting interactions were the same as included in the Ideal FOM: Fire, Detonate and Collide.

Both participating federates would Interact /React to all three interactions.

Attributes / Parameters

For Attributes selection in this FOM development approach, the project team chose to favor the CCTT specifications but added Angular Velocity. Guise related attributes and Articulated Parts found in the ModSAF SOM would not be supported.

For the most part, the federates and the RPR-FOM agreed on the attributes and parameters that should be included for the objects and interactions to be used since their legacy was DIS. There were some subtle differences. One particular difference is the information included in CCTT SAF's class structure is found in ModSAF FCS attributes. To explore the various FOM approaches, we made some assumptions about the federate information to be used. This is pointed out throughout the FOM development descriptions in this section.

The following observations resulted from this approach:

1. Having the Conceptual Model expressed in the Ideal FOM gave us an excellent starting point for FOM development.

2. Utilizing elements from the OMDD made assembling the FOM easier. All elements of the FOM were found in the OMDD.
3. The resulting FOM did not take into consideration the changes that may be required of the participating federates.

4.2.2 Merge / Single SOM Approach

For the Merge / Single SOM approach, the starting point for FOM development is an existing SOM. The one SOM is modified according to the requirements defined in the Conceptual Model and is augmented using information from other SOMs. In the case that information is not available in the SOMs, elements of the Ideal FOM or the OMDD would be used to augment the FOM. For the experiment this approach was performed twice. The first was based on using CCTT SAF SOM as the baseline with modifications made to accommodate the federation and ModSAF FCS capabilities. The second FOM used the ModSAF FCS SOM as the starting point. Our assumptions for modification were:

- Angular velocity needed to be supported
- Appearance would be expressed as a series of enumerated types (as opposed to a bit encoded value)
- Class hierarchy would match the SOM used as a starting point

This approach proved to be more difficult than other FOM development approaches for this particular experiment because of the differences in SOM hierarchy. Identifying attributes in one SOM that mapped to classes in the other was not straight forward.

The following observations resulted from this approach:

1. The modifications on the base FOM (SOM used for starting point) were few. This can be attributed to the similarities between federates (their DIS legacy).
2. The resulting FOM did not have OMDD elements for its representation since its source consisted of existing SOMs.
3. The FOMs resulting from the two approaches were as different as the SOMs they were based on. No assessment was made as to which FOM might be better or worse to use. As an expression of the Object Model they were equally valid. As a point of departure for federate implementation, a particular FOM may be desirable based which FOM is more easily supported by the participating federates.

4.2.3 Existing FOM / Reference FOM Approach

For the Existing FOM / Reference FOM Approach, the starting point for FOM development is an existing FOM. The FOM is modified according to the requirements of the Conceptual Model, primarily through the removal of elements that are not required for the target federation. Any elements required but not found in the existing FOM are obtained from the Ideal FOM (or the OMDD). The use of an existing FOM which closely matches the federation being assembled represents the easiest approach for FOM development, especially if the participating federates have implemented the existing FOM in past federation executions. If an existing FOM does not exist that is relevant to the target federation, the use of a "reference" FOM has also been explored. The RPR FOM has been developed as a starting point for DIS-based simulation for developing their FOMs. The experiment used the RPR FOM as its starting point for this FOM development approach because of the use of DIS-based systems.

The following observations resulted from this approach:

1. Although the approach resulted in a relatively large number of changes to the baseline FOM to get the target FOM, these changes were primarily deletions that are quickly and easily accomplished in the OMDT.
2. Because the RPR FOM served as a basis for the OMDD elements, the resulting FOM was largely similar to the Ideal FOM. The Reference FOM approach resulted in a deeper hierarchy, but once again the leaf nodes were the same. This approach, however, may result in FOMs with an artificially deep class hierarchy.
3. The depth of the hierarchy made the removing of the classes and interactions very easy without destroying the integrity of the FOM. This is something that could not be done with a flat hierarchy.

5. Experiment Summary

The following is a summary of the results of process and tool use for our experiment.

5.1 Process Summary

The FEDEP provided the project team with an easy to follow and complete process which provided guidance which allowed for the development of a FOM. The products of the early part of the process were critical in providing FOMs that met the objectives of the federation development.

The Object Model development process provides an easy to follow, more detailed approach to FOM

development. Although geared for a Bottoms-up approach, the same considerations must be made to support any of the other FOM development processes.

5.2 Object Model Tool Suite Implementation

The HLA OMDD Experiment has provided the first opportunity to test the entire object model tool suite in an integrated fashion. Prior experiments and object modeling experiences concentrated only on the use of the Object Model Library and the Object Model Development Tools. The OMDD Experiment provided the first use case where OMDD contents were used to develop a new FOM.

The OMDD Experiment was also structured to provide early feedback about the OMDT, OMDDS, and OMDD in several key areas. First, the experiment provided a forum for gauging the applicability and usefulness of the OMDD contents. The proportion of elements in the FOMs developed in the experiment that were taken directly from the OMDD is a key performance measure of OMDD usefulness. Second, the experiment provided the first practical use of the OMDDS user interface for identifying and exporting OMDD components for the FOM or SOM creation. The experiment required exercising the full range of OMDDS capabilities—browsing and searching the contents and exporting a set of components for use in FOM creation. Finally, the OMDD Experiment provided valuable information about the integration of the OMDDS with the OMDT and the OMDT user interface for accessing and including OMDD data in an object model.

Table 5-1 provides a summary of how the tools were used in the FOM development process.

The object model tool suite provided an invaluable resource for development of our FOMs. It reduced the time for FOM development by providing a means to reuse existing SOMs, FOMs and OM elements without the need to manually enter the desired information.

6. Guidance for Federation Developers

This paper explored several methods for development of FOMs utilizing a suite of tools that have been developed for the modeling and simulation community. Based on the results of our work, the following guidance is offered for federation developers:

GENERAL GUIDANCE - TOOL AND PROCESS USE

1. A federation development process such as the FEDEP should be utilized as a guide for ensuring that all relevant information is captured early in the development process. The detail at which such a process should be followed will depend on the particular federation being assembled. In smaller experimental situations (such as that featured in this paper) the process takes only a day or so and allows the developers to completely define the problem space. In the case of a larger federation development, the process could take weeks or even months depending on the requirements of the federation.
2. Use of the OMDT early in the process (during conceptual analysis) provides an easy way to clearly define the Conceptual Model and provides reuse for Bottoms-up FOM development efforts.
3. Consulting various resources documenting past developments as well as the use of the OML could save time and effort in federation development by providing the user with a starting point for federation design and development. Please see the end of this paper for how to access these resources.
4. A key to reuse is the OMDDS used in conjunction with the OMDD. Utilizing a common resource for OM development (both FOM and SOM) helps to build consistency across a community of users.

FOM DEVELOPMENT - METHOD AND TOOLS

Table 6-1 provides some guidelines on which FOM Development approach should be employed and which tools are involved:

Table 5-1 Summary of Object Model Tool Suite Use

	Tool		
Process / Experiment	OML	OMDDS/OMDD	OMDT
Conceptual Analysis		√	√
Federation Design	√		
Federation Development	√	√	√
Select Baseline OM	√		

Changes to Baseline		√	√
Addition of elements		√	√
Bottoms-up Approach		√	√
Merge/Single SOM Approach	√	√	√
Existing / Reference FOM Approach	√	√	√

Table 6-1 FOM Development Approach Guidance and Tools

FOM Development Method	When to Use	Tools to Use
Bottoms-Up	- Capturing the Conceptual Model - Development of a Model/Simulation's SOM - Standardized data use is desired - No existing FOMs/SOMs can be utilized	OMDDS / OMDD OMDT
Merge SOM	- Focus is on meeting needs of a few federates - Number of participating federates is small - Federates are similar in purpose and domain	OML OMDDS / OMDD OMDT
Single SOM	- One federate dominates the federation (others play minor role) - Number of participating federates is small	OML OMDDS / OMDD OMDT
Existing FOM - Similar Application	- Federation under construction has similar purpose / objectives to existing federation - Federates participating are same as in previous federation	OML OMDDS / OMDD OMDT
Reference FOM	- Participating federates are similar in function and domain supported - Large number of federates participating	OML OMDT (rarely - OMDDS / OMDD)

7. Conclusions

The FEDEP provided a useful guideline for capturing the information relevant to federation design. Its use did not require an excessive amount of time for the development of the federation in the subject experiment. The object model tool suite was a critical resource for quickly developing our FOMs and served to be very useful in Conceptual Analysis and in Federation Design. These easy to use tools saved a tremendous amount of time in the definition and design of our federation.

Many issues remain with regard to FOM development. These include:

1. Reference FOMs
2. Building federations with disparate federates.
3. The availability of existing FOMs/SOMs and information on how they have been used to assist the developer on identifying similarity of objectives.
4. FOM flexible federates and how FOM development time may be reduced.

Acknowledgments

The authors would like to acknowledge DMSO and STRICOM for sponsoring the work presented in this paper.

References

[1] Defense Modeling and Simulation Office, "HLA Federation Development and Execution Process (FEDEP) Model, Version 1.1 (DRAFT)," 22 November 1997.

[2] Lutz, R., "HLA Object Model Development: A Process View," 1997 Spring Simulation Interoperability Workshop, March 1997.

[3] Lutz, R.; Hooks, M.; and Hunt, K., "Automation in the HLA FOM Development Process," 15th DIS Workshop, 16-20 September 1996.

[4] Scrudder, R. and Lutz R., "High Level Architecture (HLA) Object Model Tool Development," Simulation Interoperability Workshop, 8-12 September 1997.

[5] Defense Modeling and Simulation Office, "HLA Object Model Template, Version 1.2," 13 August 1997.

[6] Department of Defense, "High Level Architecture, Federation Development and Execution Process (FEDEP) Model", Version 1.1, 21 November 1997.

Useful Websites

High Level Architecture: http://hla.dmso.mil

Provides information on the FEDEP and the object model tool suite

Object Model Library:
http://www.omlibrary.epgc4i.com

Simulation Interoperability Standards Organization: http://www.sisostds.org

Provides many of the paper references above from the 1997 Simulation Interoperability Workshops.

Author's Biographies

CHRISTINA L. BOUWENS is a Senior Simulation Systems Engineer and a member of SAIC Orlando's Advanced Simulation Research Team. Ms. Bouwens is currently the Project Director for the ADST II HLA Support Experiments delivery order to conduct development and experimentation in support of HLA 1.0 development and its current evolution. Ms. Bouwens has been an active part of the interoperability standards development community since 1989 and is currently a member of the SISO Transition Team and the new Executive Committee. Ms. Bouwens received her MS degree in Mathematical Science from the University of Central Florida.

RAYMOND L. MILLER is a software engineer on the Advanced Simulation Research Team within SAIC Orlando's Operation. Mr. Miller has designed and developed software for operational man-in-the-loop trainers like the CCTT manned modules since 1993. Mr. Miller received his BS degree in Computer Science from Kansas State University and has been involved in the M&S community since 1989.

ROBERT LUTZ is a senior scientist at the Johns Hopkins University Applied Physics Laboratory. He is a member of the HLA Technical Support Team (TSTcore), and is the DMSO focal point for HLA object modeling activities. He is also a technical consultant to the JWARS program in the areas of architecture and decision modeling, and currently serves as the chair of the SIW Federation Development Process Forum.

ROY SCRUDDER is a Program Manager at the Applied Research Laboratories, The University of Texas at Austin. He is a member of the HLA Core Technical Support Team. Mr. Scrudder is currently providing support to the HLA community through DMSO's Data Engineering program. This includes management of the HLA OML, OMDD, and OMDDS efforts and the development of supporting data interchange formats (DIF).

Implementing Ownership Management Services With a Bridge Federate

Christina Bouwens, Daphne Hurrell & David Shen
Science Applications International Corporation
12479 Research Parkway
Orlando, Florida 32826-3248
Christina.Bouwens@cpmx.saic.com
Daphne.S.Hurrell@cpmx.saic.com
David.T.Shen@cpmx.saic.com
(407) 207-2700

Keywords:
HLA, Multiple Federations, Bridge Federate, Ownership Management

Major concepts & topics:
Federation Bridging, Ownership Management

Forums:
RDE, IMPL

ABSTRACT: *(This paper presents information representing the HLA development process underway by the Defense Modeling and Simulation Office (DMSO) and the Department of Defense (DoD) Architecture Management Group (AMG).)*

As part of an ongoing effort focused on defining the functionalities of a Bridge Federate, prototyping of bridge federate concepts and application experiments are used to verify these functionalities with real simulation systems. In a previous paper on the Bridge Federate, the general concept of the Bridge Federate was presented. This paper follows by presenting how Ownership Management services are implemented. To explore the concept of Ownership Management across the Bridge Federate, a scenario was created in which one federation contains a tank federate simulating a M1 tank platoon. The other federation includes a Weapon Fire federate and a Damage Assessment federate. The Bridge Federate links the two federation executions together. During the course of the experiment, the ownership of a tank's damage level is transferred to the Damage Assessment federate on the other side of the Bridge Federate. The Weapon Fire system then engages the tank. The Damage Assessment federate assesses the resulting damage from the firing interaction and publishes the damage level on behalf of the tank federate. This paper describes the applications experiment in further detail along with some findings and recommendations in the area of ownership transfer.

1. Introduction & Background

A previous paper [1] discussed the developing concept of a bridge federate, which is used to link two federation executions that may be operating under different Federation Object Models (FOMs). That paper presented the concepts of the Bridge Federate and its capability relative to Federation Management, Declaration Management, and Object Management. As part of an ongoing effort to continue definition of HLA functions across a Bridge, further work has been done in the area of Ownership Management.

This paper provides a discussion of how ownership management is implemented across a bridge federate along with relevant technical issues. An application experiment is described which exercises this capability using an actual simulation system.

2. Bridge Federate Component Revisit

A Bridge Federate is a federate that participates in multiple federations. It is comprised of several internal components: surrogate federates (one for each participating federation), transformation manager and FOM Mappings/Transformations specification (FOMAT). Figure 1 shows two federation executions , FEDEX A and FEDEX B, interoperating through a Bridge Federate.

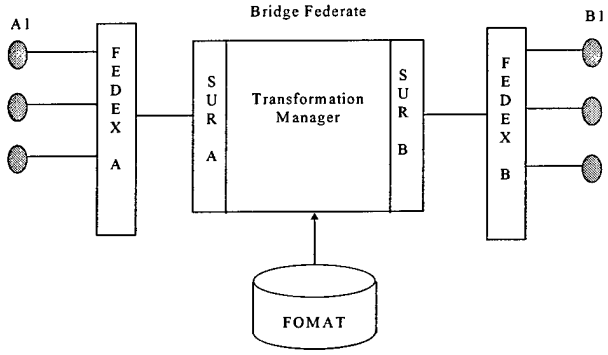


Figure 1 Bridge Federate

The function of the Bridge Federate is to link multiple federation executions by propagating the necessary HLA service invocations from the originating federation execution to all other federation executions outside that federation, which effectively forms a federation that is composed of all the federations linked by the bridge federate. Through the bridge federate, one federate perceives all other federates as if they were in the same federation execution.

3. Design of Ownership Management for the Bridge Federate

The RTI Ownership Management$_2$ services support the transfer of ownership of attributes of specific objects from one federate to another. The event trace in Figure 2 illustrates a scenario of divesting attribute ownership. This scenario shows Federate A1 issuing a request for attribute ownership divestiture.

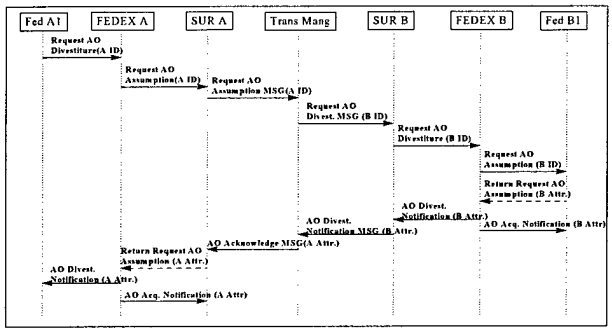


Figure 2 Divesting Attribute Ownership

1. Federate A1 issues a Request Attribute Ownership Divestiture of a specific FEDEX A object instance's attributes. Federate A1 may either explicitly request Surrogate Federate A (SUR A) as the Federate to take the ownership or request for a negotiation to take place.
2. SUR A receives a request for Attribute Ownership Assumption for the object instance's attributes. Note that in the HLA Interface Specification version 1.1, this RTI initiated services is a blocking two-way call, which could potentially impact time sensitive processes. The HLA I/F Specification version 1.3 has replaced this service by two one-way calls to avoid such impact.
3. SUR A sends a Request Attribute Ownership Assumption message to the Transformation Manager.
4. The Transformation Manager sends a Request Attribute Ownership Divestiture message to SUR B indicating the specific FEDEX B object instance's attributes corresponding to those provided by FEDEX A.
5. SUR B issues a Request Attribute Ownership Divestiture for the given FEDEX B object instance's attributes.
6. Federate B1 receives a request for Attribute Ownership Assumption for the object instance's attributes.
7. Federate B1 issues an Attribute Ownership Acknowledge indicating the list of attributes that it wishes to take ownership. Note that Federate B1 does not actually own these attributes at that time. Also, Federate B1 will issue an Attribute Ownership Acknowledge with a NULL attribute list if it chooses not to accept ownership for any of the requested attributes.
8. SUR B receives an Attribute Ownership Divestiture Notification indicating the list of attributes that have

been accepted by some federate in federation FEDEX B. Note that there may be multiple such calls received.

9. Federate B1 receives an Attribute Ownership Acquisition Notification indicating which attributes that it has acknowledged it will actually own.

10. SUR B issues an Attribute Ownership Divestiture Notification message to the Transformation Manager indicating the list of attributes now owned by federation FEDEX B.

11. The transformation manager issues an Attribute Ownership Acknowledge message to SUR A indicating the list of federation A attributes that FEDEX B wishes to own.

12. SUR A sends a Return Request Attribute Ownership Assumption to FEDEX A indicating the list of federation A attributes that it wishes to own on behalf of FEDEX B.

13. Federate A1 receives an Attribute Ownership Divestiture Notification indicating the list of attributes that have been accepted by some federate in federation FEDEX A.

14. SUR A receives an Attribute Ownership Acquisition Notification indicating which attributes it will actually own.

3.1 Ownership Negotiation Among Multiple Federations

Currently, the RTI conducts the negotiation among multiple federates wishing to take ownership of the same attributes. When negotiation has to take place among multiple federations, several potential problems have been identified. The functional diagram in Figure 3 and subsequent paragraphs describe these problems in further detail.

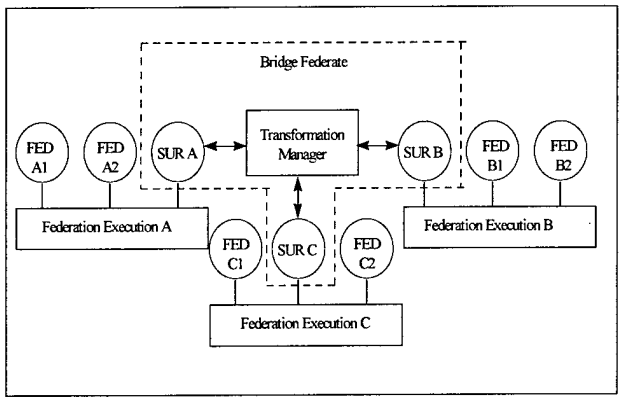


Figure 3 Ownership Negotiations in Multiple Federations

For example, Federate A1 (Fed A1) issues a Request Attribute Ownership Divestiture of a specific FEDEX A object instance's attributes. Through the Bridge Federate, this Request Attribute Ownership Divestiture message is sent to Surrogate Federate B (SUR B) and SUR C. FEDEX A, B and C send a Request Attribute Ownership Assumption to their own federates, i.e. FEDEX A to FED A2 and SUR A; FEDEX B to B1 and B2; FEDEX C to C1 and C2. The problem occurs when there is more than one federate from different federation executions attempting to take over the ownership of the offered attributes. The following paragraphs examine two possible cases :

Case 1 : If both Fed C1 and Fed B1 want to take over the ownership, Fed C1 will send an Attribute Ownership Acknowledge to FEDEX C, FEDEX C will grant that ownership to Fed C1 with an Attribute Ownership Acquisition Notification. At the same time, Fed B1 will also send a Attribute Ownership Acknowledge to FEDEX B, and FEDEX B will grant the ownership to Fed B1. The result is that Fed C1 and Fed B1 have the ownership of the attributes in their respective federation executions, but the same set of attributes is now owned by both federates in the larger combined federation execution.

Case 2 : If Fed A2 and Fed B2 both wants to take over the ownership, Fed B2 will send an Attribute Ownership Acknowledge to FEDEX B, and FEDEX B will grant the ownership to Fed B2. Surrogate B will get the Attribute Ownership Divestiture Notification from FEDEX B and passes the message to the Transformation Manager and to Surrogate A. Surrogate A then sends an Attribute Ownership Acknowledge to FEDEX A to ask for the ownership. However, suppose Fed A2 also has informed FEDEX A that it wants to take over the ownership. FEDEX A now conducts negotiation and FEDEX A grants the ownership to Fed A2. The result is that both Fed A2 and Fed B2 own the same set of attributes.

A possible solution for Case 1 is to have the Transformation Manager propagate the Request Attribute Ownership Assumption to the other Federations one at a time. For example, the Transformation Manager will propagate the Request Attribute Ownership Assumption to Federation B first, wait for the result, if no federate in Federation B wants the ownership, then the Transformation Manager will propagate the Request Attribute Ownership Assumption to Federation C. If a federate in Federation B wants to assume the ownership of the attributes, the Transformation Manager will not propagate the request to Federation C. The short fall for this solution is that this process can take a long time if a

large number of Federations are involved in the negotiation.

For Case 2, a possible solution is to use RTI interactions to perform pre-ownership transfer negotiation before the actual ownership transfer takes place. This method will require a common ownership negotiation protocol that is understood by all the participating federates from all the bridged federation executions.

4. Application Experiment Description

An application experiment was conducted to verify the requirements defined for ownership transfer across a bridge federate. The scenario for the ownership transfer application experiment involves two federation executions as shown in Figure 4. One is the CCTT federation where the CCTT SAF system is simulating a platoon of M1 tanks moving along a designated route. The CCTT system joins that federation through its Network Process and Entity Database federates. The same federation is also joined by its federate surrogate from the bridge federate. The other federation is the Specialty federation joined by the Damage Assessment federate, the Weapon Fire federate and its surrogate from the bridge federate.

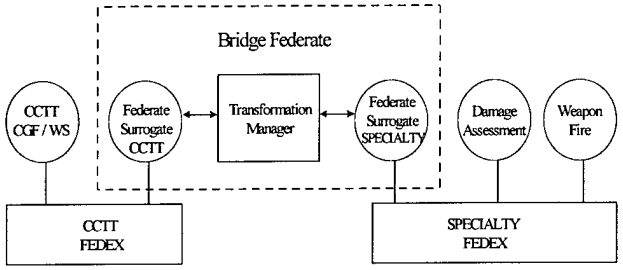


Figure 4 Ownership Transfer Application Experiment

For the purpose of this experiment, the two federation executions have the same federation object model (FOM), although federation executions using different FOMs can be bridged. The difference between them is only notional in the sense that the information flow from one federation execution to another is through the bridge federate. In this experiment, the bridge federate performs minimal information transformation to map the object identifications from one federation to the other. All other information is propagated without any data transformation.

The Weapon Fire federate's firing action is controlled through user inputs. The Weapon Fire federate plays the role of a weapon system whose damage effects to its targets are only known to the Damage Assessment federate. It is, therefore, the responsibility of the Damage Assessment federate in the Specialty federation execution to assess any damage on the M1 tanks caused by the Fire and Detonation interactions originating from the Weapon Fire federate on behalf of the CCTT federate.

The application experiment is composed of the following players:

1. A blufor M1 tank platoon.
2. A weapon system firing "special" munitions at the M1 tanks.
3. A damage assessment system to assess the damage suffered by the M1 tanks caused by the special munitions fired by the weapon fire system.

The scenario for the experiment unfolds as follows:

The M1 tank platoon marches along a designated route while its tanks are fired upon by the Weapon Fire system. The tank platoon continues to follow the route until it is completely disabled, meaning that all of its tanks have suffered catastrophic kills. The Weapon Fire system is depicted as a stationary T72 tank, which does not suffer any damage. The Damage Assessment system is not depicted visually because it is not a battlefield entity.

For this experiment, the Appearance attributes were selected to portray the damage state of the M1 tanks on the battlefield. There are three attributes that describe the damage state of a vehicle, namely, the Damage State Appearance, the Damage State Mobility and the Damage State Fire Power attributes. The transfer of these attributes signifies the transfer of the damage assessment responsibilities from one federate to another.

A typical sequence of events for the appearance attribute ownership transfer is described in Figure 5:

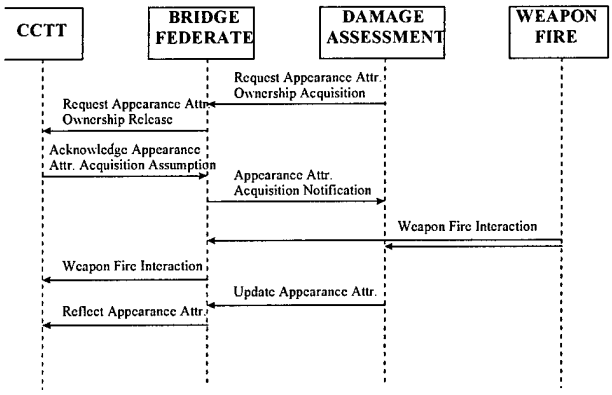


Figure 5 Attribute Ownership Transfer Across a Bridge Federate

1. The Damage Assessment federate invokes the Request Attribute Ownership Acquisition service to obtain the ownership of the appearance attributes from the CCTT federate with regard to the M1 tanks.
2. The CCTT federate acknowledges the request for divestiture release invocation and makes the appearance attributes ownership available.
3. The Damage Assessment federate is informed of the ownership acquisition notification. This concludes the appearance attribute ownership transfer from the CCTT federate to the Damage Assessment federate.
4. The user commands the Weapon Fire federate to fire special munitions at a specific M1 tank.
5. Both the CCTT federate and the Damage Assessment federate receive the interaction corresponding to the firing of the special munition.
6. The CCTT federate ignores the interaction because it no longer owns the appearance attributes.
7. The Damage Assessment federate, verifies that it owns the appearance attributes, assesses the damage, updates its local damage state of the M1 tank and updates the M1 tank's appearance attributes on behalf of the CCTT federate.
8. The CCTT federate receives the reflection of the appearance attributes from the Damage Assessment federate and updates its internal state of the M1 tank with regard to its damage level.

In the scenario above, the transfer of appearance attributes ownership can occur at any time at the discretion of the user of the Damage Assessment federate. The damage assessment responsibility is initially assigned to the CCTT federate. The transfer of the appearance attributes ownership is initiated by the Damage Assessment federate using the Request Attribute Ownership Acquisition service. These same attributes are transferred back to the CCTT federate from the Damage Assessment federate using the Request Attribute Ownership Divestiture service.

The problem cases discussed in section 3.1 manifest themselves only when there is more than one federate candidate from different federation execution wanting to own the divested attributes. To avoid this problem, the application experiment is designed to have a single candidate federate to acquire the divested attributes' ownership.

5. Conclusions and Recommendations

A prototype bridge federate with the HLA Ownership Management services is implemented and an initial experiment which exercise the transfer of appearance attributes across federation executions is described. The experiment demonstrates the feasibility of distributed processing across federation executions of a simulation object using the Ownership Management services. This approach opens the possibility for specialized applications to cooperate with simulation applications from other domains to form an integrated simulation system across federation executions.

The experiment conducted so far exercised only the ownership transfer without regarding to the sensitivity of the information passed through the bridge federate. The next step is to characterize the information flow across the bridge federate as input to analysis supporting the security Guard Federate.

6. Acknowledgments

The authors would like to acknowledge DMSO and STRICOM for sponsoring the work presented in this paper. The authors would also like to recognize the technical support of Wes Braudaway and Reed Little in the development of the Bridge Federate concepts.

References

1. Braudaway, Wesley; Little, Reed: "The High Level Architecture Bridge Federate", Simulation Interoperability Workshop, Spring 1997.

2. "High Level Architecture Interface Specification" Version 1.2, February 1997.

Author Biographies

CHRISTINA L. BOUWENS is a Senior Simulation Systems Engineer and a member of SAIC Orlando's Advanced Simulation Research Team. Ms. Bouwens is currently the Project Director for the ADST II HLA Support Experiments delivery order to conduct development and experimentation in support of HLA 1.0 development and its current evolution. Ms. Bouwens has been an active part of the interoperability standards development community since 1989 and is currently Vice-Chair of the Executive Committee. Ms. Bouwens received her MS degree in Mathematical Science from the University of Central Florida.

DAPHNE HURRELL is a Software Engineer with SAIC supporting the ADST II program. Ms. Hurrell has been a developer of simulation systems for the past ten years. She is currently technical lead for the analysis and

requirements definition of the bridge federate for the ADST II HLA Support Experiments delivery order. Ms. Hurrell holds a M.S.E. in Electrical Engineering from the University of Central Florida.

DAVID SHEN is a Software Engineer with SAIC supporting the ADST II program. Mr. Shen has been active in the modeling and simulation community for the past 8 years. He is currently involved in the research and experimentation of HLA/RTI concepts employing CCTT SAF and ModSAF for the ADST II HLA Support Experiments delivery order. Mr. Shen holds a M.S. in Computer Engineering from the University of Central Florida.

Appendix H – FOM FLEXIBILITY QUESTIONNAIRES

QUESTIONNAIRES

JMASS Questionnaire

1. What language was the federate written in and on what platform?

   C++ and C, on a Sun Sparc 20 platform running SunOS. The federate was a JMASS simulation which had been built specifically for the Engineering Protofederation.

2. What language was the HLA Interface written in and on what platform?

   C++, for Sun Sparc under SunOS (specifically RTI v0.33).

3. What type of model was used: (Constructive (Analysis, Aggregate), Virtual, Live, Others)

   The model was simulating a Surface to Air radar systems and its missile systems. The model was constructive in natural, intended to provide a threat environment for a virtual cockpit system.

4. Was there already a DIS (or network) interface before the HLA was added? If so, how did that effect the FOM flexibility of the interface and what kind of interface was used (e.g. VR Link, CIU, custom)?

   No.

5. How is time management performed with your federate? Did the federate have to change its time management scheme in order to utilize either FOM implementation? If so, what were the changes?

   JMASS models are time-based, event-driven. The first HLA application for the federate was within the Engineering Protofederation (EPF), which was real-time and did not use an HLA time management scheme. So, first the JMASS event synchronizer was modified to pace the events to the system clock. Then, the EPF used an independent time management scheme were all federates, distributed across the USA, synchronized to a GPS-aligned time source, to align their individual system clocks. Thus the federation was synchronized.

   To transition the federate to the JPSD facility, no changes were necessary. That is, the JMASS event synchronizer was still paced to the real-time system clock. The fact that the JSPD systems were not aligned to a GPS time source was not important.

6. Did using HLA require changes to the model (simulator) code (i.e. Behavior models, etc.)?

   No. JMASS provides a mechanism, called a Port Object, for insulating the models from data transmission
   details. The Port Objects a re unique for each data type, so we simply invoked the RTI interface from inside the Port Objects. The models therefore remained unchanged.

7. How is the object state maintained (hash table, shared memory, etc.)?

   Not sure that I understand this question.

   The JMASS model is responsible for its own object state, while the JMASS Data Manager provides some services to assist the model in this regard. Further, the JMASS Spatial system maintains TSPI level information on the other objects within the federations. These states are updated by a special player added to the JMASS player set just for HLA-based executions.

8. How is the object state received/sent to the "outside" - via network (hash table, shared memory, etc.)?

   The JMASS model "sent" new state values to the rest of the federation by "updating" the Port Object that represented that attribute set (e.g., invoking the update method of the Port Object with the new data). The modified code within the Port Object then interfaces with the RTI to "update" the object state. To the model, the sending and reception of data is via method parameters.

9. What constraints were used in the development of the HLA implementation (software language, hardware platform, utilization of existing interfaces, minimal changes to model, etc.)?

   JMASS Rel 3.0 Beta was used for this effort, so Sun Sparc SunOS, suing C++ was required. As we were doing basic research for the JMASS project as well as supporting the DMSO AMG efforts to define the HLA, we constrained ourselves to make not changes to the JMASS models themselves. All HLA or federation knowledge was to be in either the JMASS simulation engine or federation middleware. We were successful in this goal.

10. What are the main differences between the first FOM implemented and later FOM(s) (class structure, data type, adding or deleting attribute, attribute, etc.)?

Relatively little, since both FOM were largely DIS based. Still, we have numerous attribute representation changes to make and some changes in basic entity types (e.g., helicopters vs. aircraft, etc.)

11. How did your HLA implementation change with the change FOM (Interface changes, model changes - type and LOC)? Can we get some sample code which reflects the changes?

    Both federations used RTI v0.33. FOM changes were handled in middleware, which is available on request.

12. If you had a chance to do it all over again what would you do differently?

    Not sure that we would do anything differently, as our approach was largely successful. In this case, we only had two (2) federations to interface the federate to. If interfacing to more federations was required, we would definitely explore the application of some adaptive middleware concepts to handle the various FOMs.

13. Do you have a more detailed design diagram than that featured in the SIW paper?

    SIW paper details a suggested approach for making adaptive middleware. This approach was not used during the interfacing of the EPF model to JSPD. I will look to see if I have a good picture of the EPF/JPSD approach. Please note that the engineer involved with that effort has left SAIC, making things somewhat harder to find.

14. Any other information that would help our analysis?

CCTT Questionnaire

1. What language was the federate written in and on what platform?

   The implementation was in Ada83 on PowerPC running AIX.

2. What language was the HLA Interface written in and on what platform?

   The implementation was in Ada95 on PowerPC running AIX.

3. What type of model was used: (Constructive (Analysis, Aggregate), Virtual, Live, Others)

   The federate used a virtual model (CCTT legacy code).

4. Was there already a DIS (or network) interface before the HLA was added? If so, how did that effect the FOM flexibility of the interface and what kind of interface was used (e.g. VR Link, CIU, custom)?

   There was a DIS interface. The new HLA interface was implemented based on the Platform Proto-federation experiment. The legacy implementation dictated much of the new HLA FOM. Many of the services in the interface were not implemented because they were not supported by the legacy system. The New HLA FOM is very much DIS-like.

5. How is time management performed with your federate? Did the federate have to change its time management scheme in order to utilize either FOM implementation? If so, what were the changes?

   The legacy system in intended to run in real-time. There is no support for non-real-time modes.

6. Did using HLA require changes to the model (simulator) code (i.e. Behavior models, etc.)?

   The required changes were made to the network interface. No simulation model change was necessary.

7. How is the object state maintained (hash table, shared memory, etc.)?

The state of the objects is maintained in shared memory and consumed by the simulation models.

8. How is the object state received/sent to the "outside" - via network (hash table, shared memory, etc.)?

The state of the objects is communicated via RTI?

9. What constraints were used in the development of the HLA implementation (software language, hardware platform, utilization of existing interfaces ,minimal changes to model, etc.)?

In order to integrate the HLA interface, Ada95 was used to implement the the net work interface and it communicated with the simulation models implemented in Ada83 through shared memory.

10. What are the main differences between the first FOM implemented and later FOM(s) (class structure, data type, adding or deleting attribute, attribute, etc.)?

The main difference was to depart from the object model "imposed" by the DIS and move to a more generic FOM where the same functionalities were preserved with better classes and attribute definitions. For instance:

- Change bit encoded variables to individual attributes.
- Group entities into more elaborated class hierarchies.
- Remove explicit entity id from the class attribute definition.

11. How did your HLA implementation change with the change FOM (Interface changes, model changes - type and LOC)? Can we get some sample code which reflects the changes?

Most of the required changes were associated with translating from something that was more generic to more specific. Many lookup tables were necessary to map elements from one FOM to the other.

12. If you had a chance to do it all over again what would you do differently?

Not much, because the adopted strategy was the least impacting.

13. Do you have a more detailed design diagram than that featured in the SIW paper?

No.

14. Any other information that would help our analysis?

To achieve FOM flexibility, one needs to isolate and group FOM dependent elements in a organized manner to facilitate code change and update. It is important to think about possible systems that the federate could potentially interoperate with and design the interfaces that would isolate any potential changes. Adopting commonly accepted data representation for the class attributes is crucial. A good class hierarchy will allow specializations to suit new requirements without affecting much of the existing code.

EAGLE Questionnaire

1. What language was the federate written in and on what platform?

   Eagle is basically written in LISP and currently supports sun and hp platforms.

2. What language was the HLA Interface written in and on what platform?

   The Eagle HLA Interface has two parts a LISP part that provides direct access to the model and the Eagle common module (ECM) which is written in C++. The ECM is run on a sun (for .33 development is was running on an HP – I have not recompiled for RTI 1.0)

3. What type of model was used: (Constructive (Analysis, Aggregate), Virtual, Live, Others)

   Eagle is classified as an aggregate constructive model.

4. Was there already a DIS (or network) interface before the HLA was added? If so, how did that effect the FOM flexibility of the interface and what kind of interface was used (e.g. VR Link, CIU, custom)?

   Your bias for DIS shows here. Eagle had a much better interface using ALSP prior to interfacing with HLA. The interface was my own custom design.

5. How is time management performed with your federate? Did the federate have to change its time management scheme in order to utilize either FOM implementation? If so, what were the changes?

   Time management is explicitly performed in Eagle. I use both time advance requests and next event requests to coordinate time. The key element that had to be changed is subordinating Eagles event queue to that of the RTI. In other words, The event queue must have permission from the rti before it can execute an event. The changes are to numerous to explain – see papers.

6. Did using HLA require changes to the model (simulator) code (i.e. Behavior models, etc.)?

   Yes, yet the amount of changes depends on the federation. For distributed Eagle I could not change the code that simulated the physical or cognitive activities of a unit. These algorithms have been V&V and to change them would invalidate the model. Change was required so that they could share information and most of that change was at the model management level. Eagle has ground truth managers that are activated to be honest brokers between combatants. These managers had to be changed. In the JSIMS protofederation, where I share functionality with the Navy and Air Force models, my behavior models had to be supplemented to support the translation between my representation of a unit and theirs. For example, they expected the model to dead reckon their airplanes. Eagle did not explicitly dead reckon units so new code was implemented that gave Eagle the ability to dead reckon their airplanes and provide the location information in a form that the Eagle model normally requires for its air ground play.

7. How is the object state maintained (hash table, shared memory, etc.)?

   All units in Eagle whether actual or reflected are LISP Objects (using KEE Frames). Reflected units maintained in their own class structure, which has the required attributes and methods to make them functional in Eagle.

8. How is the object state received/sent to the "outside" - via network (hash table, shared memory, etc.)?

   The Eagle event queue has been modified to add new events to output and receive this information based on its own internal time step event processing order. Each actual unit maintains its current reflect state (as an attribute to its object – LISP List structure) and will update it as necessary. The information is provided via. sockets to the ECM, which packages it up in the correct RTI format. The ECM is always available for receiving RTI calls. These calls are stored on the socket stream until the Eagle event queue decides that the model is in a state such that it can accept outside input.

9. What constraints were used in the development of the HLA implementation (software language, hardware platform, utilization of existing interfaces, minimal changes to model, etc.)?

   All of the above.

10. What are the main differences between the first FOM implemented and later FOM(s) (class structure, data type, adding or deleting attribute, attribute, etc.)?

    Eagle's interface to the outside world has changed very little since its ASLP design. With each federation new class structure and attribute/parameter translations are required but for Eagle this is all controlled with a configuration file.

11. How did your HLA implementation change with the change FOM (Interface changes, model changes - type and LOC)? Can we get some sample code which reflects the changes?

The key is that it has changed very little. The bulk on my interface code is in the ECM, which is very generic. It maintains very little model state information and is just basically a conduit between the model and the RTI. It has required some change such as in the HLA C2 experiment where I had to add the notion of CCSIL Messages and a process to get these messages from the C++ code to the lisp code (In this case files). The basic design is based on the original Test Federate code, replacing the standard in from the menus (tickle code) with a socket stream.

12. If you had a chance to do it all over again what would you do differently?

NO.

13. Do you have a more detailed design diagram than that featured in the SIW paper?

As indicated above, I have written two papers that go into much more detail on explaining my interface.
At the Spring SISO conference – Paper #17: The Analytical Community and the High Level Architecture - A Happy Marriage. This paper presents an overview of the technical approach taken to incorporate the RTI into the legacy Eagle model code, the development of the Eagle Software and Federation Object Models, the time management scheme used to ensure causality and consistency among the federates and a summary of the lesson learned and results.

At the Fall SISO conference – Paper #73: Eagle Time Management. This paper will describe the unified approach implemented in the combat simulation Eagle in its use of time management to simultaneously coordinate its execution with other simulations using time step updates and/or events for coordination, running in real time, scaled real time or as fast as possible modes.

14. Any other information that would help our analysis?

Please call for further info if desired – recommend you read the paper above first.

IST HLA GATEWAY Questionnaire

1. What language was the federate written in and on what platform?

   In the case of the Gateway, just what constitute the "federate" has to be stated. The Gateway has a built-in CGF system, which might be considered the "federate"; it was written in C and is compiled with a C++ compiler (to get stricter type checking, etc.). However, we assume that more typically the Gateway is running as a gateway, rather than a CGF system, in which case the "federate" is whatever DIS-compliant simulation the Gateway is translating for. The federate's implementation in that case varies, of course.

2. What language was the HLA Interface written in and on what platform?

   The HLA interface of the Gateway was written in C++. It makes full and extensive use of the object oriented features of C++. It runs on SGI workstations under Irix 6.2.

3. What type of model was used?

   Virtual real-time (i.e., DIS style).

4. Was there already a DIS (or network) interface before the HLA was added?

   Yes, the CGF system that the Gateway was built on had a DIS protocol stack and interface. That DIS interface is still present, because the Gateway sends and receives both DIS and HLA data. The DIS interface really didn't affect the FOM flexibility on the HLA side, as the latter is separate from the former. Our HLA interface is a custom implementation. The Gateway uses the RPR FOM. The RTI and FOM interfaces are at the same level in the code, i.e., there is no "wrapper" layer separating them.

5. How is time management performed with your federate?

   There is no time management in the Gateway (as is customary for DIS); the Gateway assumes that wall clock time = simulation time (with offset).

6. Did using HLA require changes to the model (simulator) code (i.e. Behavior models, etc.)?

The reason for the Gateway is that using HLA requires NO changes to the federate, if the federate is a DIS-compliant legacy simulator. As it turned out, we had no need to change the Gateway's internal CGF code either.

7. How is the object state maintained (hash table, shared memory, etc.)?

Even though it is a translator, the Gateway maintains object state information. That data is needed when translating from HLA to DIS to fill in DIS PDU fields not included in an HLA attribute update, and when translating from DIS to HLA to avoid sending HLA attribute updates for DIS PDU fields that have not changed. That object state information is kept in a table of data structures, accessed by hashing on object ID (for HLA updates) or vehicle ID (for DIS updates).

8. How is the object state received/sent to the "outside" - via network (hash table, shared memory, etc.)?

When a DIS PDU or HLA attribute update is received, the corresponding HLA attribute update or DIS PDU respectively is assembled using the data in the object state hash tables, and then sent using the RTI services or DIS protocol stack respectively.

9. What constraints were used in the development of the HLA implementation (software language, hardware platform, utilization of existing interfaces, minimal changes to model, etc.)?

We chose to use SGI, Irix, and C++ as our implementation environment so as to match the version of the RTI available at that time. By doing so we avoided certain technical obstacles. We also decided to build the Gateway on top of our CGF system to take advantage of that system's DIS protocol stack, internal control executive, and the testing capabilities inherent in a CGF system. You might also say that our basic decision to build a Gateway was a constraint imposed on us, as our mission for the Platform Proto-Federation (PPF) experiment was to bring an existing legacy simulator (a SIMNET M1) into the PPF without change to the simulator.

10. What are the main differences between the first FOM implemented and later FOM(s) (class structure, data type, adding or deleting attribute, attribute, etc.)?

The initial prototype of the Gateway used the PPF FOM. Large portions of that prototype, including the FOM interface, were discarded when we began work on the production Gateway. The production Gateway has used the RPR FOM, so we have experienced precisely the changes that have been made to the RPR FOM through its various versions. Those changes have included all of the items

mentioned: changes to the class structure, changes to data types, additions and deletions of the attributes, etc.

11. How did your HLA implementation change with the changed FOM? Can we get some sample code which reflects the changes?

The Gateway's HLA interface can be thought of as having two sides: the RTI interface and the FOM interface. In the RTI interface, the RTI is represented by an object class with public methods corresponding to the RTI services. That interface does not change at all with FOM changes. In the FOM interface, the FOM is represented by a C++ class hierarchy that corresponds to the object class hierarchy in the FOM. The C++ classes are called "object actors". During a federation execution, when a simulation object (e.g., a tank) is discovered, an instance of the object actor C++ class that corresponds to the tank's FOM class is instantiated. That object actor instance holds the data for the specific instance and either implements or inherits the implementation of the translation routines for the attributes of that class. Now the point is that changes to the FOM are reflected, almost directly, in the corresponding C++ object actor classes. If a FOM class changes, its corresponding object actor class must be changed in the Gateway code, and no other classes need to be changed (all of this takes inheritance into account, of course). So, on the one hand FOM changes do require code changes in the Gateway; on the other hand, the code changes are by design confined to the corresponding object actor classes, so it is very easy to determine where and how to make the changes. Please note, it is the object class specific translation code that must be changed; the general data communication "paths" are part of the RTI interface, which does not change. Much more detail is available in [2].

As an aside, this discussion has focused on FOM changes, but because this is a Gateway PDU changes might also trigger modifications. However, PDU changes are rarer than FOM changes at this point. As for getting sample code, SAIC is already in possession of the source code for the Gateway. If you would like to be pointed to a specific sample of the object actor classes, please let me know.

12. If you had a chance to do it all over again what would you do differently?

The best way for me to answer this is to say that we DID do it all over again. Our prototype Gateway for the PPF experiment did not have anything like the FOM interface just described, and changes to it in response to FOM changes were quite tedious. It also used a "middleware" software layer to interface with the RTI. The current production Gateway integrates directly with the RTI and has the FOM reconfigurability approach described above and in [2].

13. Do you have a more detailed design diagram than that featured in the SIW paper?

Yes, see both [1] and [2]. I will be happy to provide those figures in PowerPoint form if you so request and Susan Harkrider approves.

14. Any other information that would help our analysis?

I offer two final ideas. First, we see that FOM reconfigurability can be categorized in three broad levels:

1. Run-time reconfigurability; the federate reads a FOM (possibly from a fed file) and reconfigures itself to that FOM at start up.
2. Code-time reconfigurability; FOM changes require code changes, but the code was designed and written to make those changes reasonable.
3. Design-time reconfigurability (or non-reconfigurability); FOM changes require code changes, and the code was NOT written with FOM changes in mind, so it might be quite difficult.

At the time we designed the Gateway, we believed that level 2 was the best available compromise between FOM reconfigurability and execution performance. We conjectured that a run-time reconfigurable federate could be significantly slower than a code-time reconfigurable federate, and we could not risk slow execution and still meet the performance requirements for our Gateway. We see the Gateway design as very good example of level 2. Now, with that example in hand, we hope to turn to developing some techniques for run-time reconfigurability that do not impose an excessive execution performance penalty.

Second, everything about the Gateway and its approach to FOM reconfigurability is based on the assumption that the Gateway will be faced with FOMs that resemble to some reasonable extent the DIS protocol (it is a DIS-HLA gateway, after all). It is not known how well the Gateway's FOM interface would adapt to a constructive, logical-time, ALSP style FOM and federation. That remains to be seen.

References

An overall description of the Gateway can be found in paper [1]; much more detail about its RTI and FOM interface can be found in [2]. The latter is especially recommended as it is relevant to the subject of
your study and was selected for the Fall 1997 SIW "Recommended Reading List".

[1] Wood, D. D., Petty, M. D., Cox, A., Hofer, R., and Harkrider, S. (1997). "HLA Gateway Status and Future Plans", Proceedings of the 1997 Spring Simulation Interoperability Workshop, Orlando FL, March 3-7 1997, pp. 807-814.

[2] Wood, D. D. (1997). "An Object Oriented RTI Interface", Proceedings of the 1997 Fall Simulation Interoperability Workshop, Orlando FL, September 8-12 1997, pp. 477-486.

SMOC Questionnaire

1. What language was the federate written in and on what platform?

   SMOC was converted from fortran to C++. The platforms are Windows NT, Linux, Junos, Irix, and Hrux.

2. What language was the HLA Interface written in and on what platform?

   The interface is written in C++ on the same platforms as above.

3. What type of model was used: (Constructive (Analysis, Aggregate), Virtual, Live, Others)

   SMOC is Virtual

4. Was there already a DIS (or network) interface before the HLA was added? If so, how did that effect the FOM flexibility of the interface and what kind of interface was used (e.g. VR Link, CIU, custom)?

   Yes. DIS made it easier because we had developed the DIS NIU for it.

5. How is time management performed with your federate? Did the federate have to change its time management scheme in order to utilize either FOM implementation? If so, what were the changes?

   No HLA time management is used, only wall clock.

6. Did using HLA require changes to the model (simulator) code (i.e. Behavior models, etc.)?

   Using HLA required no changes to the model code.

7. How is the object state maintained (hash table, shared memory, etc.)?

   Hash table is used to maintain object state.

8. How is the object state received/sent to the "outside" - via network (hash table, shared memory, etc.)?

   On the DIS side the object state is maintained by shared memory. On the HLA side the object state is maintained by the RTI ambassador.

9. What constraints were used in the development of the HLA implementation (software language, hardware platform, utilization of existing interfaces, minimal changes to model, etc.)?

   The only constraints are that no changes be made to the model and the model interoperate with both DIS and HLA simulating as middleware or as a gateway.

10. What are the main differences between the first FOM implemented and later FOM(s) (class structure, data type, adding or deleting attribute, attribute, etc.)?

    Used a modified RPR FOM but designed the FOM for flexibility using the Aegis Object Model Development Tool.

11. How did your HLA implementation change with the change FOM (Interface changes, model changes - type and LOC)? Can we get some sample code which reflects the changes?

    Only interface changes. Software code hopefully will be available by March or April.

12. If you had a chance to do it all over again what would you do differently?

    Nothing

13. Do you have a more detailed design diagram than that featured in the SIW paper?

    No

14. Any other information that would help our analysis?

    We are evolving SMOC to provide cross-FOM translation. Also adding more functionality (originally entity state, fire detonate, siman on DIS side only so far) to change all DIS PDUs, add DDM, MOM, and exercise control.

ModSAF Questionnaire

1. What language was the federate written in and on what platform?

   Language – C
   Platform - Sun and SGI

2. What language was the HLA Interface written in and on what platform?

   Language - C++
   Platform - Sun

What type of model was used: (Constructive (Analysis, Aggregate), Virtual, Live, Others)

Others (ModSAF and EADTB)

4. Was there already a DIS (or network) interface before the HLA was added? If so, how did that effect the FOM flexibility of the interface and what kind of interface was used (e.g. VR Link, CIU, custom)?

   A DIS interface existed prior to the addition of the HLA middleware. We wrote a Gateway that acts as middleware between two DIS applications. The FOM supports the DIS IEEE spec and does not send any additional information.

5. How is time management performed with your federate? Did the federate have to change its time management scheme in order to utilize either FOM implementation? If so, what were the changes?

6. Did using HLA require changes to the model (simulator) code (i.e. Behavior models, etc.)?

   No.

7. How is the object state maintained (hash table, shared memory, etc.)?

   Maintained in memory.

8. How is the object state received/sent to the "outside" - via network (hash table, shared memory, etc.)?

Sent/received from federate via TCP sockets sent/received via network on multicast ports

9. What constraints were used in the development of the HLA implementation (software language, hardware platform, utilization of existing interfaces, minimal changes to model, etc.)?

C++, POSIX threads, Solaris 2.5.x operating system, flexibility to work with multiple FOMs.

10. What are the main differences between the first FOM implemented and later FOM(s) (class structure, data type, adding or deleting attribute, attribute, etc.)?

None.

11. How did your HLA implementation change with the change FOM (Interface changes, model changes - type and LOC)? Can we get some sample code which reflects the changes?

Hasn't changed.

12. If you had a chance to do it all over again what would you do differently?

Would not use multi-threaded processes. Would design for fewer queues and better queue management

13. Do you have a more detailed design diagram than that featured in the SIW paper?

Yes.

14. Any other information that would help our analysis?

ACETEF Questionnaire

1. The federate was implemented using ACETEF's Simulated Warfare Environment Generator, SWEG, version 6.4.3.12.2.1 , was written in C++ and compiled on an SGI Onyx running IRIX 5.2.

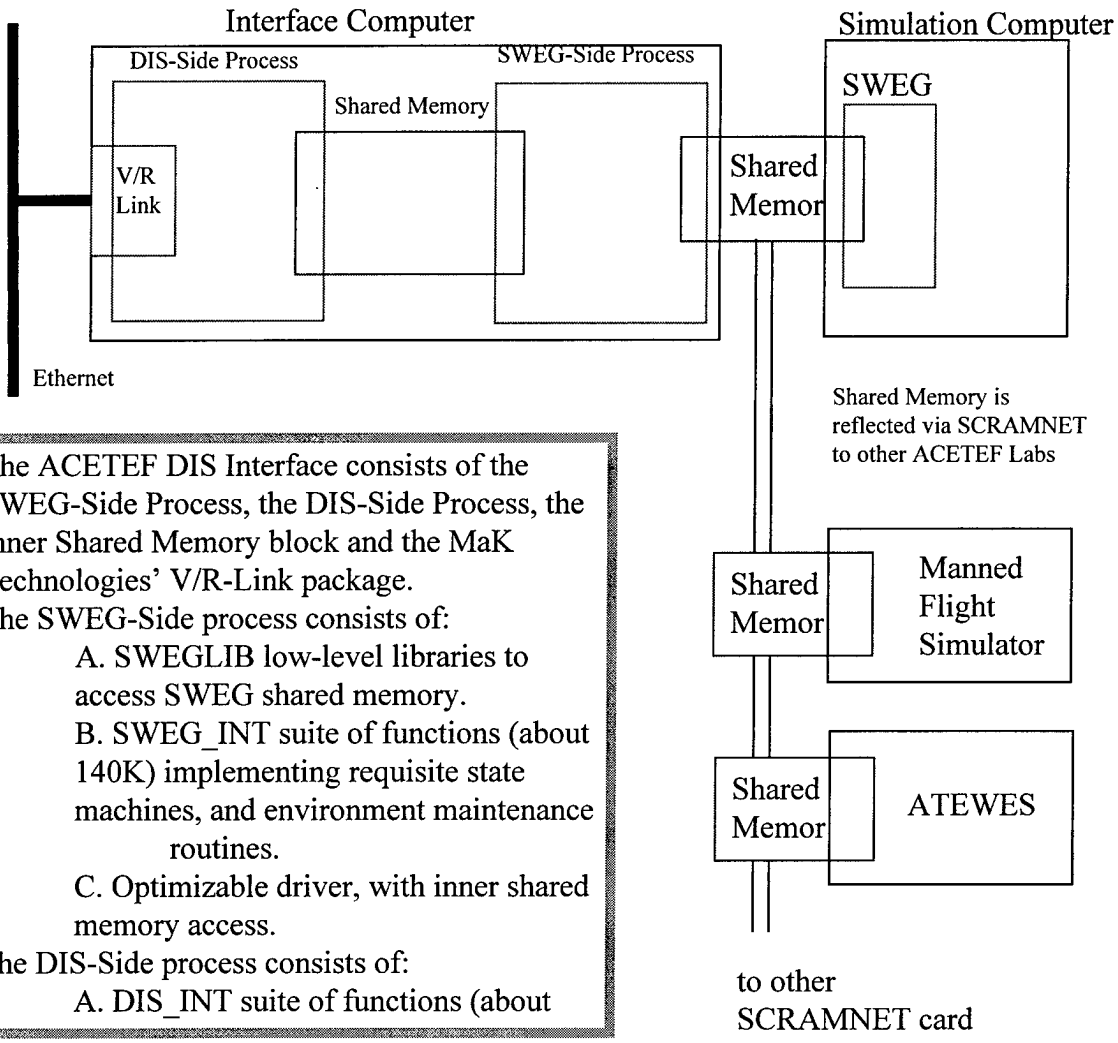
2. The Interface was written in C and C++, on an SGI 440 running IRIX 5.2.

3. The SWEG environment allows the implementation of diverse constructive entities (it is not bound to warfare domain entities, but allows the user to model anything that moves, shoots, senses, communicates or disrupts, in any combinations). It's principle strength is the ability break out components of the constructives (i.e., entities or parts of entities), and give control of these, or replace them with, external components (hardware, live players, virtual players or even other constructives), via a shared memory interface.

The HLA EPF FOM included constructive, live and virtual players, as well as hardware-in-the-loop. SWEG was configured to integrate a live a/c (was not used to to funding/time constraints), a virtual player (Manned Flight Simulator), and hardware-the-loop (an ALR-67, with indications sent to MFS cockpit) locally, via the shared memory interface (not via the HLA interface). The attributes and interactions of these aforementioned assets were published via the HLA interface. Classes, attributes and interactions of the other federates of the EPF were subscribed to via the HLA interface and appeared in the SWEG environment, accordingly.

4. ACETEF already had a DIS interface for SWEG, prior to the HLA-EPF. I designed this interface to be built in re-usable suites, and parts of it were employed in the HLA Interface. The block diagrams of the DIS interface and the HLA interface used in the EPF, appear below:

DIS Interface, and its integration with ACETEF architecture:



The ACETEF DIS Interface consists of the SWEG-Side Process, the DIS-Side Process, the Inner Shared Memory block and the MaK Technologies' V/R-Link package.

The SWEG-Side process consists of:

A. SWEGLIB low-level libraries to access SWEG shared memory.

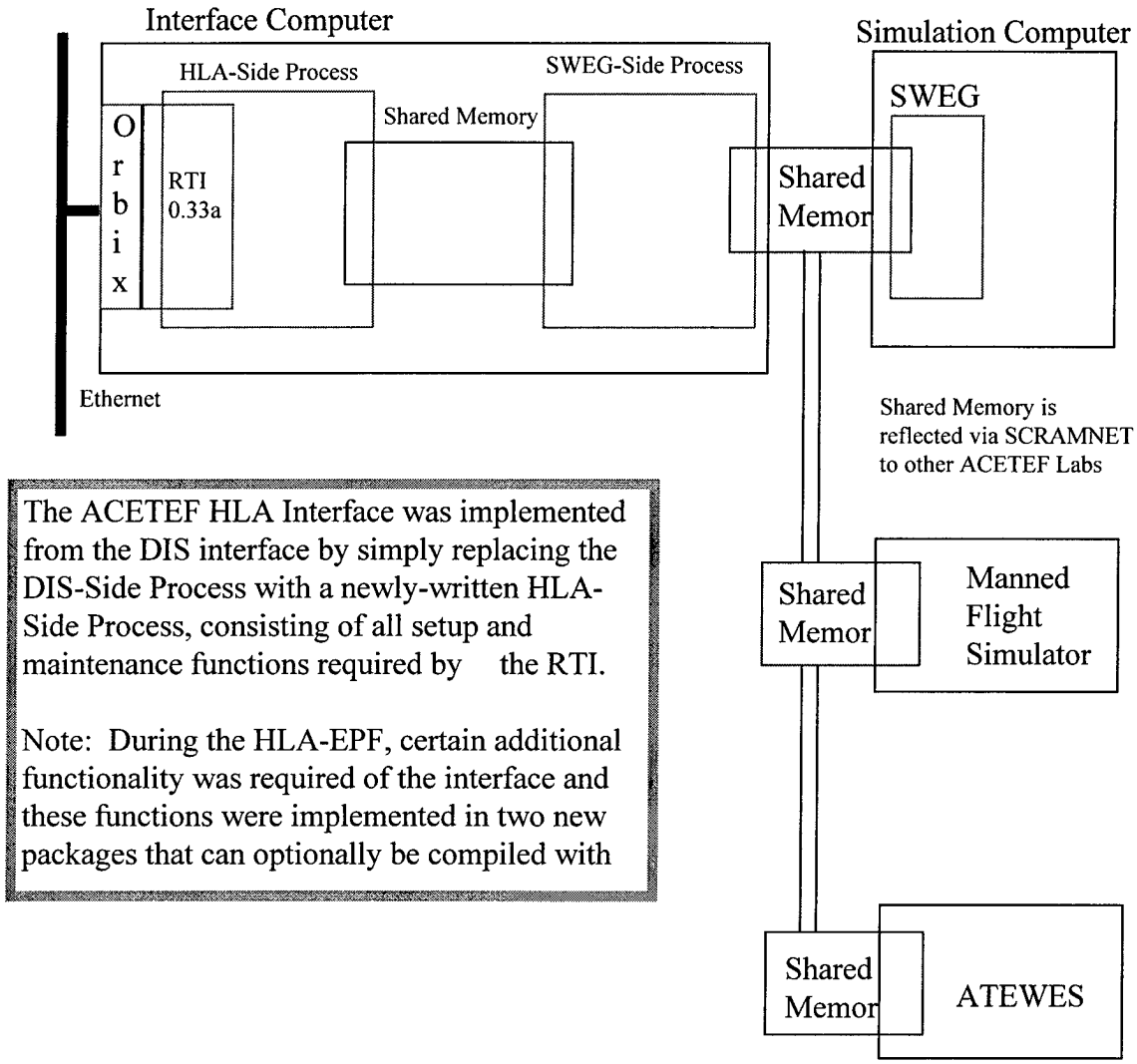
B. SWEG_INT suite of functions (about 140K) implementing requisite state machines, and environment maintenance routines.

C. Optimizable driver, with inner shared memory access.

The DIS-Side process consists of:

A. DIS_INT suite of functions (about

HLA interface (for RTI 0.33a and earlier) and its integration with the ACETEF Architecture:



The ACETEF HLA Interface was implemented from the DIS interface by simply replacing the DIS-Side Process with a newly-written HLA-Side Process, consisting of all setup and maintenance functions required by the RTI.

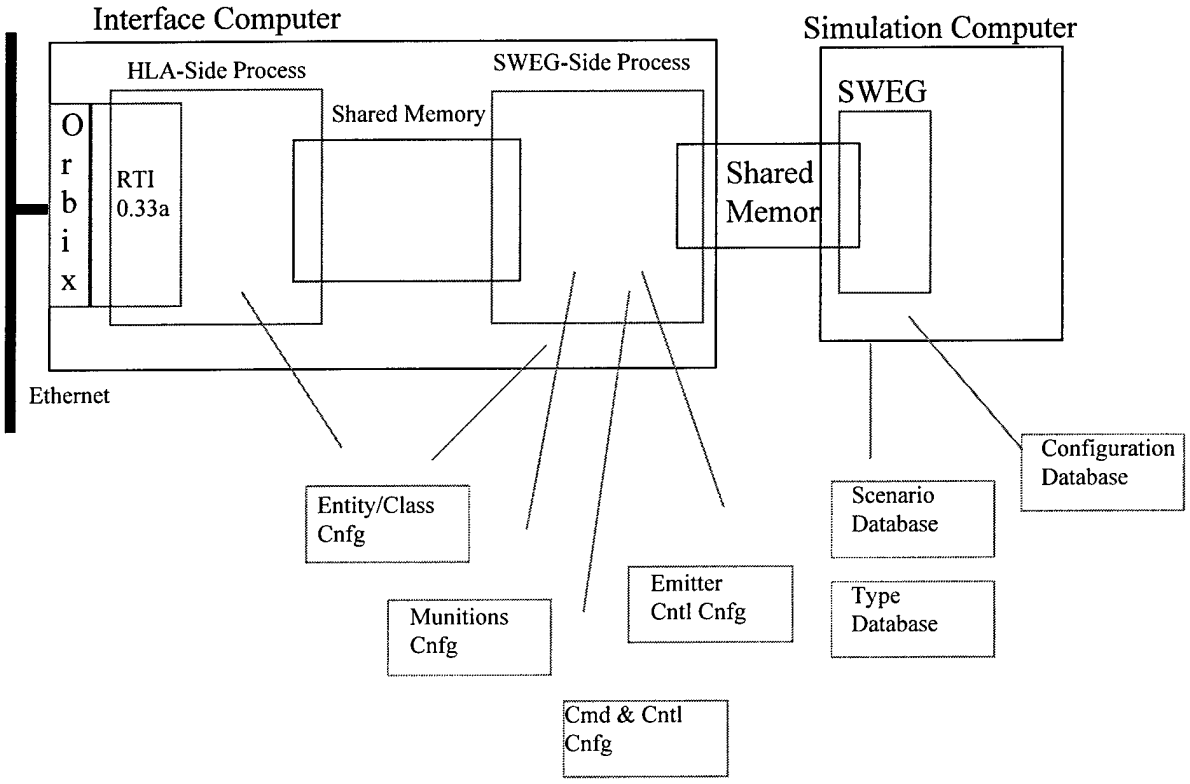
Note: During the HLA-EPF, certain additional functionality was required of the interface and these functions were implemented in two new packages that can optionally be compiled with

The flexibility required by the FOM did not substantially change the interface design and did not substantially impact the simulation engine. On the contrary, SOM development for our environment, initially appeared to be a severe limitation upon our innate flexibility - this was later found to be false, and in fact, the full development of the

ACETEF SOM will prove to be a superior vehicle for displaying and employing our flexibility.

The following diagram illustrates the configuration elements of the interface and the SWEG environment. Note that each ACETEF lab/asset has its own set of configuration elements, and these are not shown.

Configuration Elements of the ACETEF HLA Interface and SWEG environment



Entity/Class Cnfg - Maps classes and instance identifiers to SWEG semantic codes (classes) and global ID numbers (instances). Also indicates publication/subscription.

Munitions Cnfg - Maps weapon identifiers to SWEG ordnance codes.

Emitter Cntl Cnfg - (optional) Identifies and maps federation codes to SWEG codes for any emitters whose status and/or pointing is published and/or subscribed.

Cmd & Cntl Cnfg - (optional) Identifies and maps federation codes to SWEG codes for any SWEG simulation entities that will respond to external commanders.

Type Database - (SWEG Conflict Language) Defines the nature and capability of classes of simulation entities in the SWEG environment.

Scenario Database - (SWEG Conflict Language) Creates instances of the classes defined in the SWEG environment, gives them locations and initial states, puts them on command chains and sides (can be more than two sides).

The FOM will impact any or all of the configuration files noted above. In the case where new structured attributes or interactions must be implemented, C-language source must be added to both the HLA-side process and the SWEG-side process, and the interface must be re-compiled. Most interactions and attributes can be dealt with very effectively utilizing the primitives of the SWEG shared memory interface - the SWEG_INT suite of the SWEG-side process, may require extension, if a new interaction must be implemented, but only in rare cases would the simulation engine have to be modified.

5. Time management was not implemented, in the strict sense; the interface tags attribute updates and interactions with an IRIG code, but this is used only for dead-reckoning and analysis. The SWEG environment is event-driven, and therefore updates and interactions were processed as received.

Admittedly, this could result in inconsistent results, although the EPF did not suffer from them. A more effective time management scheme is under consideration.

6. HLA did not require any changes to the SWEG model.

7. For the most part, object state is maintained in the shared memory which SWEG uses to communicate with the interface, in the form of a "current-values table". In certain cases, a portion of the object's state is maintained by interface code.

For instance - a SWEG constructive SAM Battalion maintains its set of perceptions of the warfare environment internally, without a "current-values table" representation of its air picture in shared memory. If such a constructive is configured to report to an external commanded, the SWEG configuration database would indicate the player's perceptions, as an input to the external asset, and the "decision to fire" as an output from the external asset.

At runtime, SWEG would put a report of each of that player's perceptions (new, updated and last) in a shared memory area called a "mailbox", every 2 seconds or so.. The interface to the external asset is required to read it immediately (it will be overwritten with the next perception). When the external asset wants the SAM to fire on one of its tracks, the interface must place a "decision to fire" message, with necessary data, into another mailbox area, so that the constructive battalion will act on it.

The interface must therefore maintain a dynamic list of that SAM Battalion's perceptions adding, updating, or deleting, as SWEG's perception reports indicate. The interface must also respond to queries from the external asset (acting as the SAM Battalion's commander), requesting information on its air picture, and the interface will do so, using the dynamic list of perceptions it maintain.

In contrast, the position, orientation and velocity attributes of a simulation entity are updated directly into a persistent block in the shared memory between SWEG and the interface. To understand the role played by the interface processing in these cases, one

must recognize that the "anything goes" nature of DIS simulations (and early HLA federations) is in conflict with the detailed pre-definition and rigorous specification required by the SWEG environment - SWEG is designed for serious simulations, with clearly defined analytical objectives.

When developing the DIS interface, it was found that "mishaps" of a distributed simulation (a site going down, then coming up again, incorrect codes on an entity, munition or warhead, etc.) played havoc with the SWEG engine. Updating the attributes of an entity, directly from V/R-Link routines, or from an RTI callback, into the SWEG shared memory results in frequent core dumps. Since SWEG integrates various assets at ACETEF, it was determined that the interface must maintain consistency of distributed participants updates and interactions, to minimize the possiblity of crashes, and delays associated with restarting multiple lab's assets.

9. The SWEG configuration database indentifies how often a simulation entity's state will be updated in SWEG shared memory. The interface configuration files indicate that the particular entity must be updated, causing the SWEG-side process to place the update data in inner shared memory whenever it is updated. The HLA-side process reads this data from inner shared memory as often as the RTI is required to update it.

10. ACETEF has participated in two FOM developments, one of which has been executed (EPF), and one that will be executed in the near future (JADS-EW). The major difference is that the EPF FOM required attribute updates of even fixed sites, a legacy, possibly, of the DIS days. The JADS-EW FOM is being developed with a keener eye towards the objectives of the federation, and therefore is "leaner". Another difference is that the JADS-EW FOM specifies higher update rates since it anticipates a more practical RTI than the 0.33a version used in the EPF.

11. The implementation changes were driven primarily by the changes in the RTI versions being delivered and integrated during the testing. The FOM changes could be easily implemented in the configuration files described above. Sample code can be provided; please identify what part of the interface, you want samples from.

12. Using the present HLA Interface Specification, and the ACETEF SOM (implemented at a higher level of abstraction than was used in the EPF), I would re-design the interface to implement the SOM in its entirety, with an integrated configuration tool, that would have a user-interface much like the OMDT. This is not an unattainable goal.

13. See diagrams included. A detailed design document for the DIS interface is also available. Detailed documentation on the shared memory structures used between SWEG and external assets are available.

14. Many of the generalization/flexibility problems of simulation asset integration and object definition have already been conquered in SWEG and its interface. SWEG works, it is real, and it can be used for demanding RDT&E applications. While it certainly has

shortcomings, the approaches it employs in its environment development and in its integration of external assets, parallel the efforts of HLA, though on a laboratory scale. They deserve consideration.

Notes: The above responses are in the context of the HLA Engineering Protofederation FOM, which utilized RTI 0.33a and earlier (the CORBA versions of 1996). ACETEF has not yet built an interface for the most recent, non-CORBA -based RTI's, but it will most likely follow the same architecture.

Appendix I – ACRONYMS

A2ATD – Anti-Armor Advanced Technology Demonstration
ACETEF – Air Combat Environment Test and Evaluation Facility
ADST – Advanced Distributed Simulation Technology
ALSP – Aggregate Level Simulation Protocol
AMG – Architecture Management Group
API – Applications Program Interface
ARL – Applied Research Laboratory
ATEWES – Advanced Tactical Electronic Warfare Environment Simulator
BFTT – Battle Force Tactical Trainer
BLUFOR – BLUe FORces
C4I – Command, Control, Communications, Computers, and Intelligence
CATT – Combined Arms Tactical Trainer
CCSIL – Command and Control Simulation Interface Language
CCTT – Close Combat Tactical Trainer
CDRL – Contract Data Requirements List
CGF – Computer Generated Forces
CIU – Cell Interface Unit
CLCGF – Corps Level Computer Generated Forces
COR – Contracting Officer Representative
COTS – Commercial Off The Shelf
CPU – Central Processing Unit
CRDA – Cooperative Research & Development Agreement
DD – Data Dictionary
DIS – Distributed Interactive Software
DDM – Data Distribution Management
DDT&E – Director, Defense Test & Evaluation
DM – Data Management
DMSO – Defense Modeling and Simulation Office
DO – Delivery Order
DOD – Department Of Defense
E3 – Electromagnetic Environmental Effects
EC – Electronic Combat
ECM – Eagle Common Module
EPF – Engineering ProtoFederation
FCS – Federation Common Software
FDDI – Fiber Digital Data Interface
FED – Federation Execution Data
FEDEP – Federation Development and Execution Process Model
FEDEX – Federation Executive
FOM – Federation Object Model

FOMAT - FOM Mappings / Transformations
FRED – Federation Required Execution Details
GOTS – Government Off The Shelf
GTRI – Georgia Tech Research Institute
HLA – High Level Architecture
HP – Hewlett Packard
HSE – HLA Support Experiments
I/F – InterFace
I/O – Input/Output
IBM – International Business Machines
ID – Identification
IEEE – Institute of Electrical and Electronic Engineers
IPC –
IRS – Interface Requirement Specification
IST – Institute for Simulation and Training
JADS-EW – Joint Advanced Distributed Simulation - Electronic Warfare
JMASS – Joint Modeling and Simulation System
JHAPL – John Hopkins Applied Research Laboratory
JPSD – Joint Precision Strike Demonstration
JSIMS – Joint Simulations System
JTF – Joint Training Federation
LAN – Local Area Network
LOC – Lines Of Code
LMC – Lockheed Martin Corporation
M&S – Modeling & Simulation
MFS – Manned Flight Simulator
MOM – Management Object Model
MOM – Measures Of Merit
MSRR – Modeling & Simulations Resource Repository
NAWC-AD – Naval Air Warfare Center, Aircraft Division
NAWC-TSD – Naval Air Warfare Center, Training Systems Division
NIU – Network Interface Unit
OM – Object Model
OMDD – Object Model Data Dictionary
OMDDS – Object Model Data Dictionary System
OMDT – Object Model Development Tools
OML – Object Model Library
OMT – Object Model Template
OO – Object Oriented
OPFOR – OPposing FORces
OS – Operating System
PC – Personnel Computer
PDU – Protocol Data Unit
PM – Program Manager
POC – Point Of Contact

POSIX – Portable Operating System Interface for Computer Environments
PPF – Platform Proto-Federation
PVD – Plan View Display
RDT&E – Research, Development, Test & Evaluation
RID – RTI Initialization Data
RPR – Real-time Platform Reference
RTI – Run Time Infrastructure
RWA – Rotary Wing Aircraft
SAF – Simulated Armed Forces
SAIC – Science Applications International Corporation
SAM – Surface to Air Missile
SEI – Software Engineering Institute
SEOD – SAF Entity Object Database
SFI – Surrogate Federate Interface
SGI – Silicon Graphics Incorporation
SIMNET – Simulation Network
SISO – SWEG Interoperability Software Object
SIW – Simulation Interoperability Workshop
SMOC – Simulation Middleware Object Classes
SNMP – Simple Network Management Protocol
SOM – Simulation Object Model
SOW – Statement Of Work
SSAA – System Security Authorization Agreement
STRICOM – Simulation TRaining and Instrumentation Command
SUR – SURrogate
SWEG – Simulated Warfare Environment Generator
TM – Time Management
UT – University of Texas
VME – Virtual Memory Extender
VV&A – Validation, Verification & Acceptance
WAN – Wide Area Network
WARSIM – Warfighters Simulation
WS – Work Station

Federation Execution Summary Table

Federation Execution Name: CCTT Federation

Number of Concurrent Federation Executions (total including this Federation Execution): 2

RTI Software Used (Version): 1.0

NOTE:
Complete one of these tables for each Federation execution

Federate Summary Information

	Name	API (C++,Ada, IDL, Java)	Time Management Switches		Host (assign # to each host) (List data on Host Table)	LAN (assign # to each LAN) (List data on LAN Tables)
			Regulating (y or n)	Constraining (y or n)		
Fed 1	Surrogate For cctt_federation	Ada	N	N	1	1
Fed 2	dis_federate	Ada	N	N	2	1
Fed 3	edb_federate	Ada	N	N	3	1
Fed 4						
Fed 5						
Fed 6						
Fed 7						
Fed 8						
Fed 9						
Fed 10						
Fed 11						
Fed N						