Annex C – NETN Federation Agreements and FOM Reference Document v1.0

DRAFT 4

MSG-068 NATO Education and Training Network
Federation Architecture and FOM Design Technical Subgroup

Table of Contents

NETN Federation Agreements and FOM Reference Document v1.0 ..... *1*

Table of Contents *2*

1 Introduction *6*

1.1 Purpose **6**

1.2 Use **6**

1.3 Background **6**

1.4 Acknowledgements **6**

1.5 References **7**

2 Design Agreements *8*

3 Infrastructure Agreements *9*

3.1 CFBL-Net **9**

3.2 Booster Network **9**

3.3 Sites **9**

3.4 IP Addresses **9**

3.4.1 IP addresses and Site Identifiers in CFBL-Net 9

3.4.2 Site Identifier Reference 10

3.5 Simulation Infrastructure **10**

3.5.1 Federations and RTIs 11

3.6 Federates **11**

3.7 Supporting Software and Services **12**

4 Information Exchange Agreements *14*

4.1 Information Exchange Data Models **14**

4.2 Platforms and Aggregate Units **15**

4.2.1 Entity Identifier 15

4.2.2 Entity Types 16

4.2.3 Symbols 16

4.3 Modeling Responsibilities **16**

4.3.1 Services Modeling 16

4.3.2 Dynamic Transfer of Modeling Responsibilities 16

4.4 Radio Simulation Agreements **17**

5 RPR-FOM v2.0 D17 (r2) FOM Module *18*

6 Link 16 FOM Module *19*

7 NETN Service Consumer-Provider FOM Module *20*

7.1 Introduction **20**

7.2 General Approach **20**

7.2.1 Service Consumer ... 22
7.2.2 Service Provider ... 23

7.3 Interaction Classes ... 25
7.3.1 NETN_Service ... 25
7.3.2 NETN_RequestService ... 26
7.3.3 NETN_OfferService ... 26
7.3.4 NETN_AcceptOffer ... 26
7.3.5 NETN_RejectOffer ... 26
7.3.6 NETN_CancelService ... 27
7.3.7 NETN_ReadyToReceiveService ... 27
7.3.8 NETN_ServiceStarted ... 27
7.3.9 NETN_ServiceCompleted ... 27
7.3.10 NETN_ServiceReceived ... 27

7.4 Simple Datatypes ... 27

7.5 Arrays ... 28

7.6 Fixed Records ... 28
7.6.1 NETN_EventIdentifier ... 28

7.7 Enumerated Datatypes ... 28
7.7.1 NETN_ServiceTypeEnum8 ... 28

8 NETN Logistics FOM Module v1.0 ... 29

8.1 Scope ... 29

8.2 Definitions ... 29
8.2.1 Materials ... 30
8.2.2 Means of transport ... 30
8.2.3 Transport resources ... 31
8.2.4 Facilities ... 31

8.3 Transfer of Control ... 31
8.3.1 Principles of Transfer of Control ... 31
8.3.2 Use of an Active / Inactive status parameter ... 32
8.3.3 Disaggregation of units during transportation ... 32
8.3.4 Representation of “Inactive” units ... 33
8.3.5 Representation of “Active” units ... 33
8.3.6 Example: Transport by one single transporter in one travel with ToC ... 33
8.3.7 Example: Transport by several transporters in several travels with ToC ... 33
8.3.8 Illustration ... 34

8.4 Supply Pattern ... 35
8.4.1 Supply Service ... 35
8.4.2 Storage Service ... 37

8.5 Maintenance Pattern ... 37
8.5.1 Repair Service ... 37
8.5.2 Repair Types ... 38

8.6 Convoy Pattern ... 39
8.6.1 Transport service ... 40
8.6.2 Embarkment service ... 41
8.6.3 Disembarkment service ... 43
8.6.4 Convoy Services and Attrition ... 44
8.6.5 Convoy datatypes ... 45

8.6.6 Scenario preparation phase ... 46
8.6.7 Convoy Pattern Exception ... 47

8.7 RPR-FOM Platform Objects Class Extensions ... 48

8.8 Facility object class ... 48

8.9 Interaction Classes ... 50
8.9.1 NETN_RequestConvoy ... 50
8.9.2 NETN_OfferConvoy ... 51
8.9.3 NETN_CancelConvoy ... 51
8.9.4 NETN_ RejectOfferConvoy ... 52
8.9.5 NETN_ ConvoyEmbarkmentStatus ... 52
8.9.6 NETN_ ConvoyDisembarkmentStatus ... 52
8.9.7 NETN_ConvoyDestroyedEntities ... 53
8.9.8 NETN_RequestSupply ... 53
8.9.9 NETN_OfferSupply ... 54
8.9.10 NETN_AcceptOffer/NETN_RejectOffer ... 55
8.9.11 NETN_ReadyToReceiveSupply ... 55
8.9.12 NETN_SupplyComplete ... 55
8.9.13 NETN_RequestStorage ... 56
8.9.14 NETN_OfferStorage ... 56
8.9.15 NETN_StorageStarted ... 57
8.9.16 NETN_RequestRepair ... 57
8.9.17 NETN_OfferRepair ... 57

8.10 Fixed Record Datatypes ... 58
8.10.1 NETN_ObjectDefinitiontStruct ... 58
8.10.2 NETN_ObjectDescription ... 58
8.10.3 NETN_HumanDescription ... 58
8.10.4 NETN_EquipDescription ... 59
8.10.5 NETN_PlatformDescription ... 59
8.10.6 NETN_DataTStruct ... 59
8.10.7 NETN_DataEDStruct ... 59
8.10.8 NETN_AppointmentStruct ... 59
8.10.9 RepairStruct ... 60
8.10.10 SupplyStruct ... 60

8.11 Array Datatypes ... 60

8.12 Enumerated Datatypes ... 60
8.12.1 ServiceTypeEnum8 datatype ... 60
8.12.2 NETN_ConvoyTypeEnum32 ... 61
8.12.3 NETN_OfferTypeEnum32 ... 61
8.12.4 NETN_FeatureLevelEnum32 ... 61

8.13 Variant Record Datatypes ... 61
8.13.1 NETN_TransportStruct ... 61
8.13.2 NETN_ObjectFeatureStruct ... 62

9 NETN Aggregate Unit FOM Module v1.0 ... 63

9.1 Introduction ... 63

9.2 NETN Aggregated Unit Object Classes ... 63
9.2.1 NETN_Aggregate ... 63
9.2.2 Federation Agreements ... 66

9.3 Enumerated Datatypes ... 66

9.3.1 ActiveStatusEnum8 66
9.3.2 AggregateMissionEnum16 66
9.3.3 CaptureStatus8 74
9.3.4 ConcealmentEnum8 74
9.3.5 CoverEnum8 75
9.3.6 DamageStatusEnhancedEnum32 75
9.3.7 EchelonEnum8 (from JC3IEDM echelon-size-code) 75
9.3.8 EntityCategoryEnum8 76
9.3.9 SensorStateEnum8 76
9.3.10 SupportRelationshipEnum8 76
9.3.11 UpdateTypeEnum8 76
9.3.12 WeaponsControlOrderEnum8 76

9.4 Complex Datatypes 76
9.4.1 CoverStatusStruct 76
9.4.2 ElectronicSignatureStruct 77
9.4.3 EntityListStruct 77
9.4.4 EntityStruct 77
9.4.5 EntityListVariableArrayStruct 77
9.4.6 HUMINTSignatureStruct 77
9.4.7 MissionStruct 77
9.4.8 NetworkList 78
9.4.9 ResourceStatusNumber 78
9.4.10 SensorStructArray1 78
9.4.11 SensorStruct 78
9.4.12 SupportRelationshipStruct 78
9.4.13 VisualSignatureStruct 78
9.4.14 WorldLocationStructArray2 79
9.4.15 WorldLocationStruct 79

9.5 Entity Object Class Extensions 79
9.5.1 RPR-FOM Platform Object Class Extension 79
9.5.2 RPR-FOM Lifeform Object Class Extension 79

9.6 Combat adjudication 80
9.6.1 Combat Adjudication Service Federate (CASF) 81
9.6.2 Area-effects Munitions 82

9.7 Interaction Extensions 83
9.7.1 NETN_CombatResult 83
9.7.2 EngagementResultListStruct 83
9.7.3 EngagementResultStruct 83
9.7.4 SupplyStructArray1 83
9.7.5 SupplyStruct 83
9.7.6 NETN_MunitionDetonation 83

1 Introduction

1.1 Purpose

The purpose of this document is to provide a common reference federation agreements document (FAD) for all federations in the NATO Education and Training Network (NETN). Agreements that are common to all NETN based federations are specified in this document. Templates for documenting required federation specific agreements are also provided. Principles and format for information exchange between federates in a NETN based federation is defined in the FAD. As part of the federation agreements a module based HLA reference Federation Object Model (FOM) is provided.

1.2 Use

This document is intended to be used as a template and/or reference when developing federation specific agreements. In any specific federation more detailed and other types of agreements are almost always required. This reference agreement document is not intended to replace the need for developing federation specific agreements.

1.3 Background

This version of the NETN Reference FAD was developed by NATO Modeling and Simulation Group (NMSG) Task Group MSG-068 NETN. This task group was initiated to support the ACT Snow Leopard Program with M&S recommendations for establishing a NATO wide network for education and training (NETN), a.k.a. Snow Leopard.

A technical subgroup of MSG-068, Federation Agreements and FOM Design (FAFD) subgroup was created with representatives from the participating NATO and partner nations. This group represented a broad community of practice with respect to federation architecture and design. Major systems, federations and training networks were represented in the FAFD group. The input provided and the harmonization of federation architecture and design agreements forms the basis of this document.

Key input to the development of this FAD includes:

- ALLIANCE FOM
- CASIOPEA FOM
- JLVC FOM
- JMRM FOM
- KOSI FOM
- P2SN FOM
- RPR-FOM v2.0

1.4 Acknowledgements

The following individuals participated in the MSG-068 FAFD subgroup and contributed to the initial development of the NETN Reference FAD.

Amy Grom, André Geiger, Andy Bowers, Angel San Jose, Anthony Jones, Bharat Patel, Birol Güvenç, Björn Löfstrand, Brian Gregg, Cem Kumsal, Christian Mårtensson, Clive Wood, Darren Mc Farlane, Dieter Steinkamp, Duncan E Rogers, Elena Bravo, Ercan Atalay, Erdal Cayirci, Erich Schmid, Farshad Moradi, Frank Bertling, Franz Schubert, Fredrik Jonsson, Gareth M Pugh, Gokay Sursal, Gunnar Öhlund, Gustav Schulz, Göran Bergström,

Hans Jense, Heliodoro Ruiperez, Herbert Tietje, Ionel Vlasie, Jacek Sumislawski, James Boulet, Jan van Geest, Jean-Pierre Faye, Jochen Siebeneicher, John Loughhead, Jose Mimbrero, Jose Ruiz, Juan José Ruiz Pérez, Karl-Heinz Neumann, Klaus Greiwe, Konradin Keller, Lars Lindberg, Laurent Lesage, Leif Almgren, Lennart Olsson, Lesley Jacobs, Manuel Dogaru, Mark Shelford, Martin Eklöf, Miles Patterson, Mimi Nguyen, Ola Wall, Oliver Henne, Orlin Nikolov, Patricio Jimenez, Per- Philip Sollin, Petar Savkov Petrov, Peter Meyer zu Drewer, Pontus Svenson, Rachid El Abdouni Khayari, Robert B. Kean, Robert Paledau, Roger Jansen, Ron Caprio, Sergio Galán, Stephane Devaud, Steven Blackstone, Søren Larsen, Thomas Orichel, Tom van den Berg, Torbjörn Hultén, Ulf Björkman, Ulf Jinnestrand, Vladimir Manda, Wim Huiskamp, Xavier Coste

1.5 References

While this document is intended to be sufficiently complete to be read stand alone, time does not allow for all concepts to be explained fully. Please refer to the documents referenced below for more details. This document borrows especially heavily from the RPR-FOM v2.0 D17 documentation. In the case of differences between this document and the references, this document is primary.

The list of references indicates relevant standards that should be considered during the development of distributed simulation systems. Some of these are not referenced in the main body of the document.

- **Distributed Interactive Simulation (DIS)**
  IEEE 1278.1-1995 Application Protocols
  IEEE 1278.1a-1998 Supplement to Application Protocols - Enumeration and Bit-encoded Values
  IEEE 1278.2 - Communication Services and Profiles
  IEEE 1278.3 - Exercise Management & Feedback (EMF) - Recommended Practice
  SISO-REF-010-2006 DIS Enumerations

- **High Level Architecture (HLA)**
  IEEE 1516-2000 Framework and Rules
  IEEE 1516.1-2000 Federate Interface Specification
  IEEE 1516.2-2000 Object Model Template (OMT) Specification
  IEEE 1516.3-2003 Federation Development and Execution Process (FEDEP)
  IEEE 1516.4-2007 Verification, Validation, and Accreditation of a Federation
  IEEE 1516-2010 HLA "Evolved" Framework and Rules
  IEEE 1516.1-2010 HLA "Evolved" Federate Interface Specification
  IEEE 1516.2-2010 HLA "Evolved" Object Model Template (OMT) Specification

- **Real-time Platform Reference Federation Object Model (RPR-FOM)**
  SISO-STD-001-1999: Guidance, Rationale, & Interoperability Modalities for the RPR FOM (GRIM 1.0)
  SISO-STD-001.1-1999: Real-time Platform Reference Federation Object Model (RPR FOM 1.0)
  RPR FOM v2.0 D17 FOM
  RPR FOM v2.0 D17 GRIM

- SISO-STD-004-2004: Dynamic Link Compatible HLA API Standard for the HLA Interface
- Specification Version 1.3
- SISO-STD-004.1-2004: Dynamic Link Compatible HLA API Standard for the HLA Interface Specification (IEEE 1516.1 Version)
- NATO STANAG 4603
- IEEE P1703 Distributed Simulation Engineering and Execution Process (DSEEP)
- SISO-STD-002-2006: Standard for: Link16 Simulations
- SISO-STD-003-2006; Base Object Model (BOM) Template Specification (approved 8 May 06)
- SISO-STD-003.1-2006; Guide for BOM Use and Implementation (approved 8 May 06)
- SISO-STD-007-2008: Military Scenario Definition Language (MSDL)

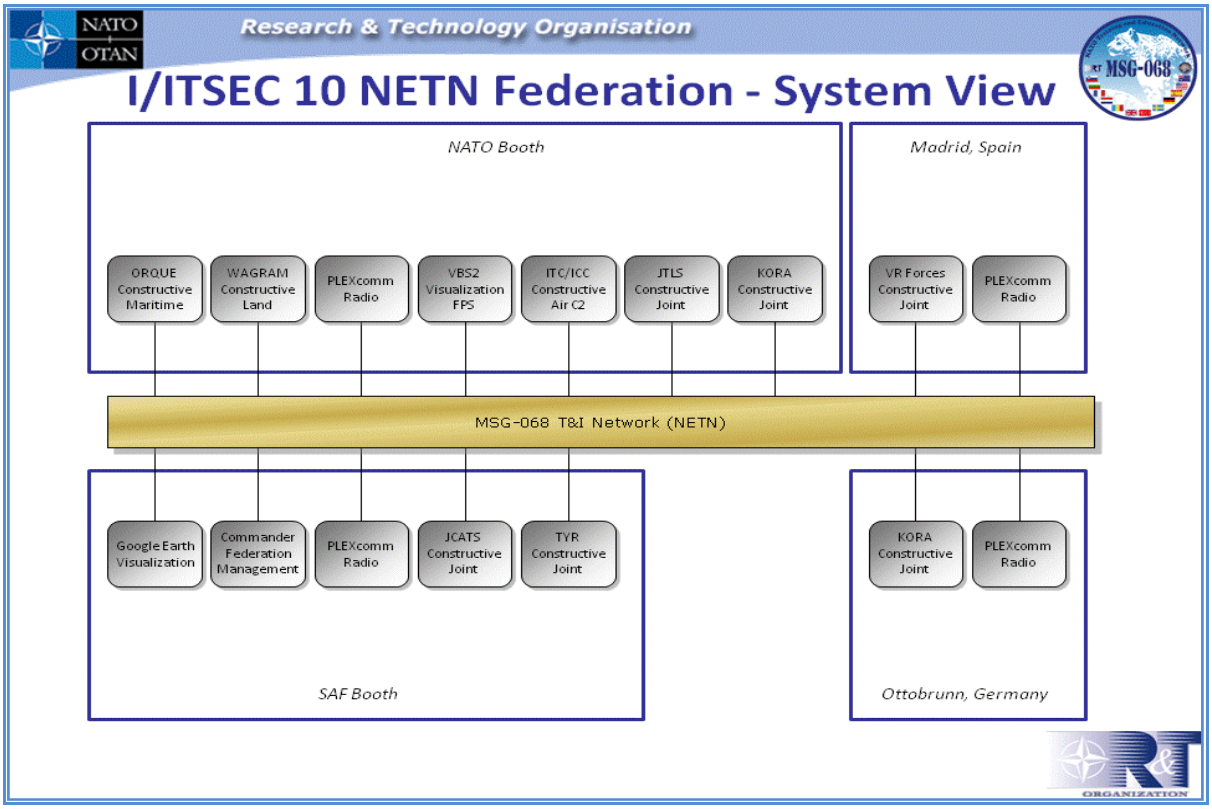
2 Design Agreements

Agreements related to the overall design of the federation shall be documented in the federation agreements. This includes information about the connected systems, their role and their purpose in the federation. A system can consist of several hardware and software components and each system can run in multiple instances.

System Name	Hardware	Software	Description	Purpose	POC
<Name and version>	<List of HW>	<List of SW>	<System Description>	<Purpose/Role in the Federation>	<POC information>
…					

For clarity, the federation agreements should also contain figures describing the main systems of the federation.

Example:



3 Infrastructure Agreements

3.1 CFBL-Net

The Combined Federated Battle-Lab Network (CFBL-net) is a core network component of the NETN Federation Architecture. This network provides a managed secure IP network used to connect accredited sites and national networks. However, the NETN Reference Federation Agreements document can also be applied to federations running on-top of other IP based networks including the Internet.

3.2 Booster Network

The NETN Federation Architecture recommends using a simulation overlay network ("Booster Network") on-top of existing IP network to create a persistent capability to access and connect to the various simulation resources without the need for a specific technical setup or configuration. The Booster Network hides the complexity of the physical networking infrastructure, simplifies the federation setup and optimizes simulation performance over WAN by providing an HLA-aware software router. The NETN Reference Federation Agreements document does not explicitly require this technology and can also be applied to federations running without Booster Network.

Agreements on how the booster network is configured shall be documented in a table describing each node.

Name	Address	Location
<Logical Name of Booster>	<Public IP>	<Geographical Location or address>
…		

3.3 Sites

In NETN federations the unique identification of sites is a vital part of agreements related to the representation of simulated entities. A list of site IDs is therefore required to be completed as part of a federation specific FAD. The unique site identifier is also used as part of unique identifiers of simulated entities in the federation as defined in the RPR-FOM datatype "*EntityIdentifier*".

3.4 IP Addresses

All systems connected to the network infrastructure shall be assigned IP addresses. The method for assigning these addresses can vary depending on the underlying network policies and procedures. A complete list of all IP addresses of all hosts involved in a specific experiment and their purpose must be listed in the Federation Specific FAD.

3.4.1 IP addresses and Site Identifiers in CFBL-Net

The CFBL-Net formula for assigning IPs is straightforward and typically in the following format. From a CFBL-Net view each nation is given a 16 address block and National (NNN) id.

Format: III.NNN.SSS.###

- III is initiative id
- NNN is the national id
- SSS is the CFBL-Net site identifier

Each site shall correspond with a fixed IP range. The hosts on each site needed for the experiment shall get an IP address from this range. By using VPN technology all hosts in the network can be accessed by their IP address.

3.4.2 Site Identifier Reference

Sites connecting through CFBL-net are given a unique site identifier corresponding to the CFBL-Net IP setup. For all other federation agreements all sites identifiers should use the reference table below as the basis for assigning site identifiers. Only series of Site ID for nations participating in the development of this Reference Document have been included in this version of the reference agreements.

Site IDs should be in the range 1 – 65534

Site Name	Site ID	Site Description	Site Location
CFBL-net	0-255	Reserved for CFBL-Net sites allocated by CFBL-net authorities during initiative setup	
JWC	301	NATO Joint Warfare Center	Stavanger, Norway
JFTC	302	NATO Joint Force Training Center	Bydgoszcz, Poland
NC3A	303	NATO Command, Control and Consultation Agency	the Hague, the Netherlands
ESP NETN HUB	400 - 499	Series allocated for Spanish Sites	Spain
ITM Simulation Lab.	401	Institute of Technology "La Marañosa" Simulation Laboratory	Madrid, Spain
The Netherlands	500-599	Series allocated for Dutch Sites	The Netherlands
TNO	501	TNO	The Hague, Netherlands
Germany	600-699	Series allocated for German Sites	Germany
Sweden	700-799	Series allocated for Swedish Sites	Sweden
USA	800 - 899	Series allocated for US sites	USA
USJFCOM	801	US Joint Forces Command	Suffolk VA, USA
France	900-999	Series allocated to France	France
UK	1000-1099	Series allocated to UK	UK
Bulgaria	1100-1199	Series allocated to Bulgaria	Bulgaria
Hungary	1200-1299	Series allocated to Hungary	Hungary
Australia	1300-1399	Series allocated to Australia	Australia
Turkey	1400-1499	Series allocated to Turkey	Turkey
Romania	1500-1599	Series allocated to Romania	Romania

3.5 Simulation Infrastructure

NETN based federations are based on STANAG 4603 which states that High-Level Architecture (HLA) IEEE 1516 shall be used as the standard for developing and federating simulation systems. The NETN Reference Federation Agreements allows non-HLA or legacy HLA (i.e. HLA 1.3) federates to participate in the simulation using appropriate bridging and/or adapter technologies. NETN based federations use the latest version of IEEE 1516 (currently IEEE 1516-2010, a.k.a. HLA Evolved).

3.5.1 Federations and RTIs

A NETN based federation shall run a core federation based on an IEEE 1516 compliant and certified Runtime Infrastructure (RTI). Any bridging required in order to adapt federates to IEEE 1516 or the selected RTI shall be the responsibility of integrating federate.

This Reference FAD does not specify a specific RTI implementation for use in NETN based federations. Several certified RTI implementations exist that can provide the IEEE 1516 services to participating federates. In some cases multiple NETN based federations may exist and information between them exchanged using bridges/filters/guards etc.

All federations that exist to support an NETN federation execution must be clearly defined. Agreements related to the naming of federations, hosting of RTIs and specific details with respect to RTI settings must be declared as part of the federation specific agreements.

The following template shall be used to document all relevant federations including at minimum the primary HLA IEEE 1516 based NETN federation and supporting RTI.

Federation Name	RTI Version	RTI Host/Name	RTI Port	Comment	RID
<Name>	<Version>	<IP or Name of CRC>	<CRC Port>	<Description>	<RTI settings>

3.6 Federates

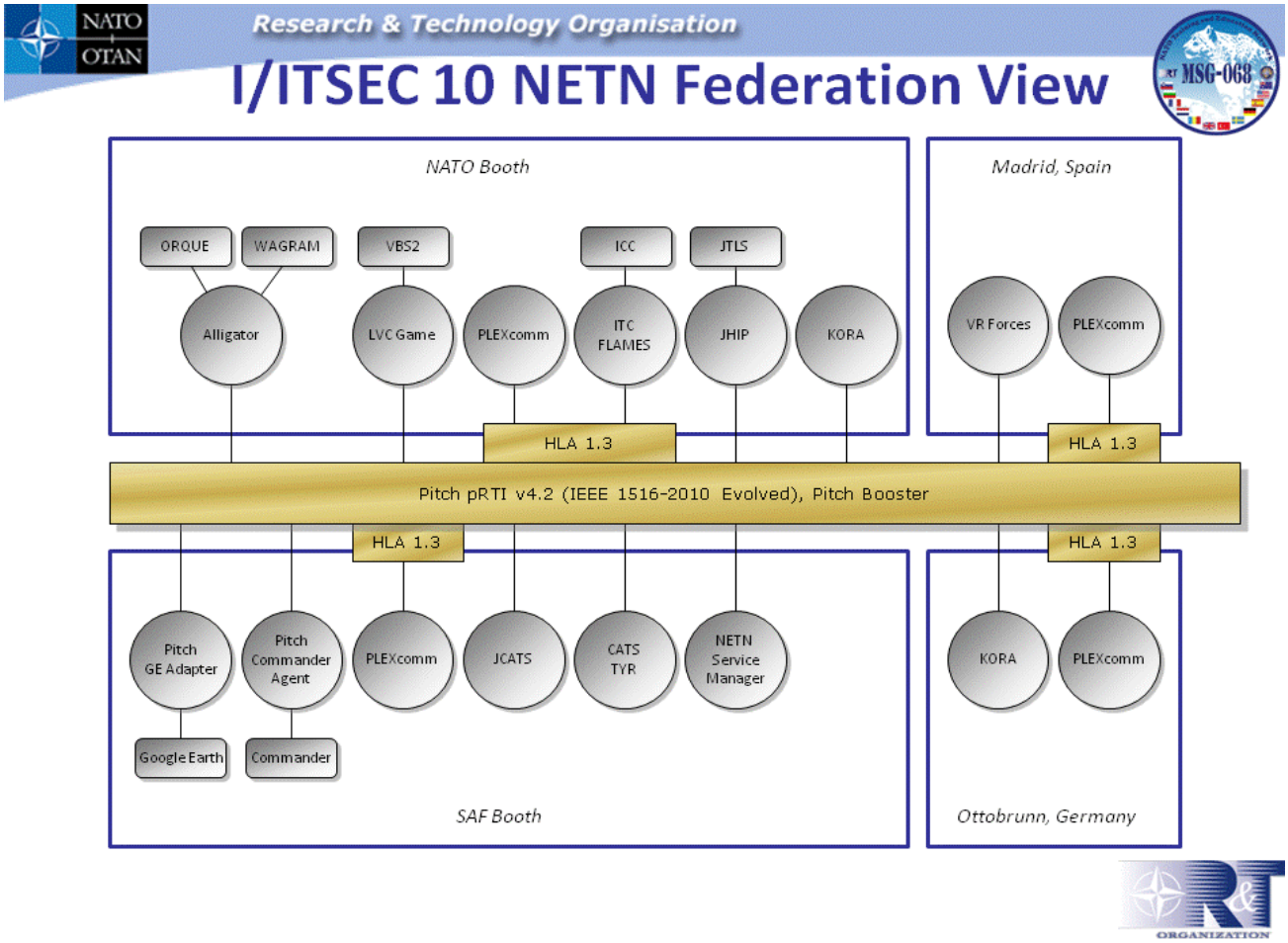
All federates participating in a NETN Federation must be clearly identified and described. Agreements with respect to federates include:

- Federate Name (used in HLA IEEE 1516-2010 based federations and may be different from federate type)
- Federate Type
- Federate Application Name and Id (name of the application hosting the federate, note that the same application can host several federates, e.g. a bridge application)
- Application ID – The application Id is used when exchanging information to create unique identifiers assigned to entities. The ID is used in the RPR-FOM datatype "*EntityIdentifier*".
- Interface used to connect to the RTI (e.g. HLA 1.3, IEEE 1516-2010 …)
- Federation Name (the name of the federation which the federate joins)
- Description (Information about the main role in the exercise and any additional important information)
- Federate POC (Org or person responsible for the federate)

Federate Name	Federate Type	Federate Application	Application ID	Interface	Federation Name	Description	POC
<Name>	<Version>	<App Name>	ID	<Interface>	<Federation>	<Description>	<POC info>

For clarity, a figure describing all federates connected in a federation (so-called lollipop picture) should be included in the federation agreements.

Example:



Each federate should also document their HLA interface in terms of which HLA services are used and which information is exchange in the federation. A Simulation Object Model (SOM), SOM Modules and/or other descriptions of the object model used to exchange information in the federation shall be clearly documented. HLA federate certification is recommended and may be required.

3.7 Supporting Software and Services

Simulation Support Services are processes (software) which must be executed in parallel to the federate processes to enable a federation execution or which is required to support individual simulations in the federation to enable them to participate in a federation according agreements.

NETN Reference Architecture does not include any specific Simulation Support services.

- All federates/simulations in a NETN based federation must specify any Simulation Support services used.
- The federation specific FAD shall document all Simulation Support services for all participating federates

Examples

- Local RTI Component (LRC) is an integral part of the Federate Application and is usually started in the same process space as the federate itself. Usually this service is not documented explicitly in the FAD unless the LRC does more than expected like loading a plug-in for SOM to FOM translation.
- Central RTI Component (CRC). Unless running a connectionless RTI mode this component represents the RTI Executive and is the initial point of access to a federation.
- Web Service Provider RTI Component (WSPRC) is an RTI component used when offering the RTI
- Services using the standard IEEE 1516 Web-Service API.

- Execution Control / starter daemons for remote start up of the CRC, federates / simulation applications, or the various database services that can be required to run a simulation.
- Bridging/gateway/adapter services (either as a bi-directional transfer or as a data diode.)
    - HLA 1.3 <-> HLA IEEE 1516
    - FOM X <-> FOM Y
    - DIS <-> HLA
    - TENA <-> HLA
    - RTI X <-> RTI Y
    - SIMPLE <-> HLA LINK 16 BOM
- Databases to provide initialization data
- Web-services to provide initialization data
- Databases to provide weapon-system-parameters or material data

4 Information Exchange Agreements

4.1 Information Exchange Data Models

NETN Federations use the modular FOM concept defined in IEEE 1516-2010. The modules describe how data is represented and encoded/decoded when exchanged in an HLA federation. The modular concept allows federates to load only those modules they are aware of and use. In addition the modules can be extended with more detailed representation by creating new modules and sub-classing/using information from other modules.

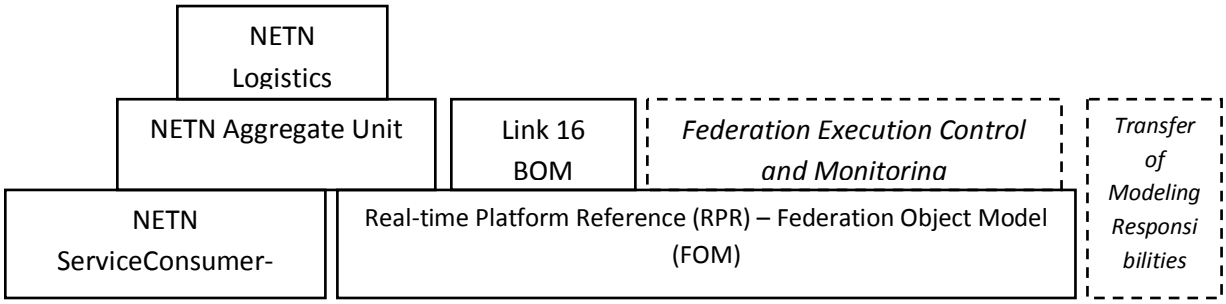
In an NETN federation agreement all federates shall be documented with respect to which FOM modules they use. In addition each federate shall also document a Simulation Object Model (SOM) describing in detail what parts of the FOM modules are used.

The NETN Reference Federation Agreements Document have developed and identified a set of FOM Modules to support some specific aspects of information exchange between federates in an NETN federation. Future versions of this document may include references to more and/or updated modules that represent other aspects currently not included in the scope of NETN Reference Architecture.

The NETN Reference FOM is a set of independent and dependent FOM Modules. Each FOM Module is either an already established standard maintained by other organizations/communities or defined as a NETN FOM Module.

The following FOM Modules versions constitute the current version 1.0 of the NETN Reference FOM:

FOM Module	Dependencies	Comment
RPR-FOM v2.0 D17 (r2)	Standalone	The RPR-FOM Module is based on the SISO RPR-FOM PDG release of RPR-FOM v2.0 D17.
Link 16 FOM Module v1.0 D2 (r2)	RPR-FOM v2.0 D17	The Link 16 FOM Module is based on the SISO Link16 PDG work and release of Link 16 BOM v1.0 Draft2.
NETN Service Consumer-Provider FOM Module v1.0	Standalone	The NETN Service Consumer-Provider FOM Module is a base-module intended to be extended to support the modeling of different types of services.
NETN Logistics FOM Module v1.0	NETN Service Consumer-Provider FOM Module v 1.0, NETN Aggregate FOM Module v1.0, RPR-FOM v2.0 D17	The NETN Logistics Module is used to model logistics services such as transport, supply and repair.
NETN Aggregate FOM Module v1.0	RPR-FOM v2.0 D17 (r2) NETN Service Consumer-Provider FOM Module v 1.0	The NETN Aggregate FOM Module extends the RPR-FOM representation of Aggregate and Platform entities.



All FOM modules can also be obtained as HLA IEEE-1516-2010 OMT Data Interchange Format files.

Federations may extend the reference FOM with additional FOM Modules when appropriate. The basic FOM Module rules as defined in IEEE 1516-2010 shall be applied.

The name of object- and interaction classes in FOM Modules developed specifically for NETN are prefixed **NETN_**. When extending the FOM with additional modules the naming of classes, datatypes and other identifiers should be de-conflicted.

Registered objects and interactions are always discovered/received at the most specific subscribed class level. Extending a FOM Module with additional subclasses provides the possibility to add extra attributes/ parameters at the more specific class level. Exchange of information using this more specific level can take place between federates publishing and subscribing to this level. However, to become compatible with and receive information from federates only publishing on the more general level, the receiving federate must subscribe to both class levels. Subscribers of the more general class will receive information from publishers of the more specific class level.

> Example: A national extension to the NETN FOM Modules subclasses existing NETN object classes and defines additional attributes. National models aware of this extension can publish and subscribe to the more specific level defined in the national FOM module extensions. Other existing federates not aware of the extension can still discover the object and receive updates but only on the level it subscribes to. In order for the national federates to discover and receive information from other federates they need to subscribe to the NETN class level as well as the national extension level. Be aware that the discovered object and attribute updates will be on the NETN level.

4.2 Platforms and Aggregate Units

Information about Platforms and Aggregate Units are exchanged in NETN Federations using the RPR-FOM v2.0 D17 FOM Module and extensions represented in the NETN Aggregate FOM Module v1.0.

The NETN Aggregate FOM Module v1.0 extends the representation available in the RPR-FOM with additional information enabling a higher level of interoperability between systems using these extensions. The NETN Aggregate FOM Module v1.0 is documented in chapter 9.

Mixing federates working on the basic RPR-FOM level with the more detailed NETN Aggregate FOM Module level is possible and allowed depending on the requirements of the federation in terms of interoperability. E.g. a federate only requiring RPR-FOM information can subscribe to information on that level while other federates exchange information on the NETN Aggregate FOM Module level.

4.2.1 Entity Identifier

In the federation agreements all known units and platforms shall have a unique identifier associated with the entity. In addition initial information about the state of the entity shall be documented. This includes

the following RPR-FOM and NETN Aggregate FOM Module information. Callsign (Who), Entity Type (What), Spatial (Where), Marking.

4.2.2 Entity Types

The RPR-FOM also requires an agreement with respect to platform and aggregate unit entity types. A key attribute of simulated entities and units are that they have entity type identifiers. In RPR-FOM this is a 7 digit code representing the Kind, Domain, Country, Category, Subcategory, Specifics and Extra information to define the type of a platform or unit. There exists standards that document common platform and unit types however for a specific federation agreements related to Entity Types must be documented.

All known entity types that will be represented in the federation shall be defined and listed in the federation agreements. Federates supporting a subset of the identified types shall list those types supported.

4.2.3 Symbols

In many federations there is a requirement to correlate the symbols used to represent platforms and units on 2D and/or 3D displays. Such agreements shall be documented in the federation agreements as mappings between Entity Types and Symbols. Other information required to map to symbols may be ForceIdentifier and unit size information.

4.3 Modeling Responsibilities

In a federation the responsibility of modeling and simulation of the synthetic environment is distributed. Each federate have intrinsic capabilities to represent certain aspects of entities, events and other phenomenon in the simulated environment. During federation design the roles and responsibilities of all federates are described and documented. The responsibility of modeling certain aspects can only be assigned to a federate with a capability that meets specified requirements. Initial modeling responsibilities and capabilities of dynamically transferring modeling responsibilities shall be documented in the federation agreements.

4.3.1 Services Modeling

The NETN Reference Federation Agreements includes FOM modules to federate to provide modeling and simulation as services to other federates. This is not a transfer of responsibility between federates but rather a service provided by one federate and consumed by another. The NETN Service-Consumer FOM Module v1.0 is a base module that allows federates to request, provide and consume services. In addition a NETN Logistics FOM module is provided specifically for logistics related services.

Services Modeling is the preferred method by which a federate transfers resources (Supply), transports (Convoy) and or provides maintenance (Repair) units and platforms simulated in another system.

The NETN Service-Consumer FOM Module is provided in chapter 7 and the NETN Logistics FOM Module is provided in chapter 8.

4.3.2 Dynamic Transfer of Modeling Responsibilities

No design pattern for the dynamic transfer of modeling responsibilities has been verified as part of the MSG-068 NETN work and no such module is included in this document. However there exist proposals that may be included in future versions of this document.

4.4 Radio Simulation Agreements

Radio simulation in NETN based federation shall use the RPR-FOM (DIS) Radio protocol/interactions.

5 RPR-FOM v2.0 D17 (r2) FOM Module

The NETN Reference FOM and FAD are dependent on the Standard Real-Time Platform Reference FOM. The version used is currently under development by the Simulation Interoperability Standards Organization (SISO) in a Product Development Group (PDG). The version selected is widely used by industry and a well know de-facto standard.

Recommendations:

- Use the NETN Logistics FOM Module services approach and not the RPR-FOM logistics interactions
- Use the extended NETN Aggregate FOM Module representation of platforms and Aggregate Entities

6 Link 16 FOM Module

The Link 16 FOM module is a derivate from the SISO-STD-002-2006: Standard for: Link16 Simulations (aka. Link16 BOM) and is dependent on the Standard Real-Time Platform Reference FOM.

The Link 16 BOM is available from SISO. It has been adapted to a FOM Module, dependent on the RPR-FOM v2.0.

7 NETN Service Consumer-Provider FOM Module

7.1 Introduction

This document describes a basic pattern for modeling request, negotiation and delivery of services. The interaction patterns required for different types of services may vary but the basic principles and interaction class definitions are outlined in this chapter.

In the RPR-FOM and DIS patterns exist to model logistics services. The intention is to allow these specific patterns to map onto the more general Service Consumer-Provider Pattern. This is described in more detailed in the Logistics FOM Module chapter.

The Service Consumer-Provider Pattern defines two types of entities.

- Service Consumer Entities: are entities requesting and consuming specific services offered and provided by other entities.
- Service Provider Entities: are entities able to offer and provide a specific service.

A provider/consumer entity can be of various kinds, e.g.:

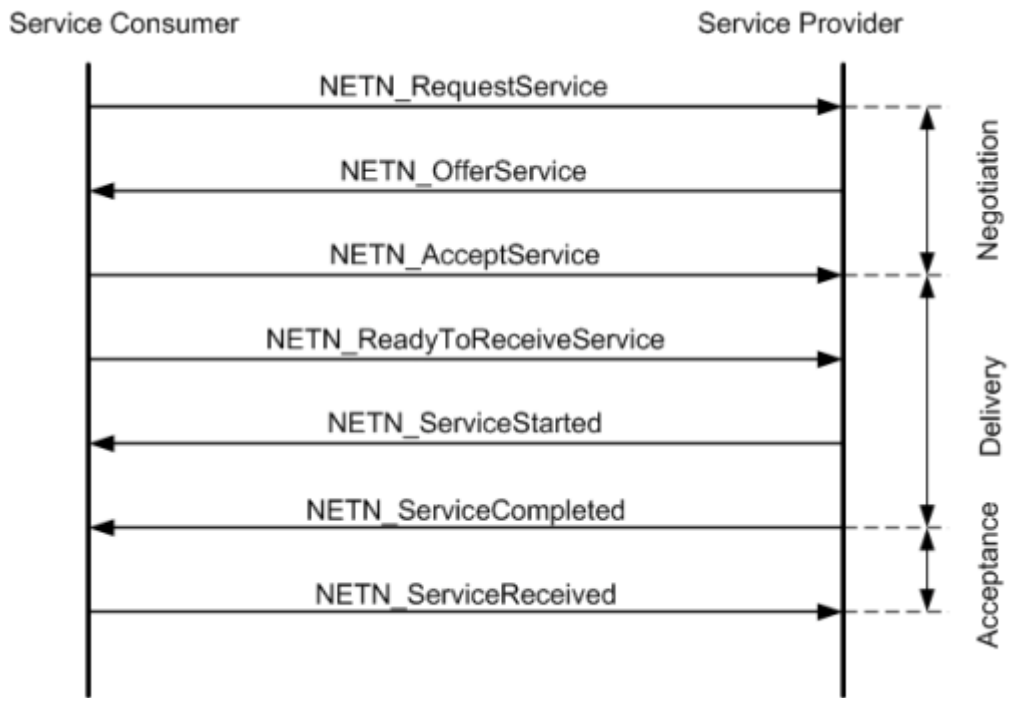
- An object instance in a federation execution
- A federate
- A controller of a function implemented in a federate application

The interactions between entities in this pattern will be published and sent using HLA services.

7.2 General Approach

The pattern is divided into three phases:

1. Service Negotiation: the service is requested, offers received and offers are either accepted or rejected.
2. Service Delivery: the consumer indicates that it the deliver process can start and the selected provider start delivering and continues until all service has been delivered.
3. Service Acceptance: the provider or consumer indicates the completion of the service delivery and waits for acknowledgment/acceptance from the other part.



The above interaction diagram shows the normal patterns for requesting services and receiving notification that the service transaction has completed.

Variations include:

- Service completion is determined by the consuming entity and sent as a NETN_ServiceReceived interaction before the corresponding **NETN_ServiceCompleted** interaction is sent.
- The service offer (**NETN_OfferService**) is NOT proceeded by a service request. This accommodates cases in which a service provider determines that a service is needed by one or more consumers
- and offers that service before being asked

Exceptions include:

- Early termination of the service by either the consumer or provider using the **NETN_CancelService**
- interaction
- Rejection of service offer by the service consumer entity using the **NETN_RejectOffer** interaction.

Federation agreements:

The condition for offering a service based on information in the request is an agreement for a specific federation. This agreement shall be documented in the federation specific agreements using the following reference template:

Scope/type of request Condition for	Offering
All types of requests exactly	Ability to provide the type of service must match
<scope>	<condition>

Extending the RequestSupply and adding parameters with description of the conditions for offering may be a future extension, e.g. in a RequestSupply the requested supplies (amount, type) needs to be

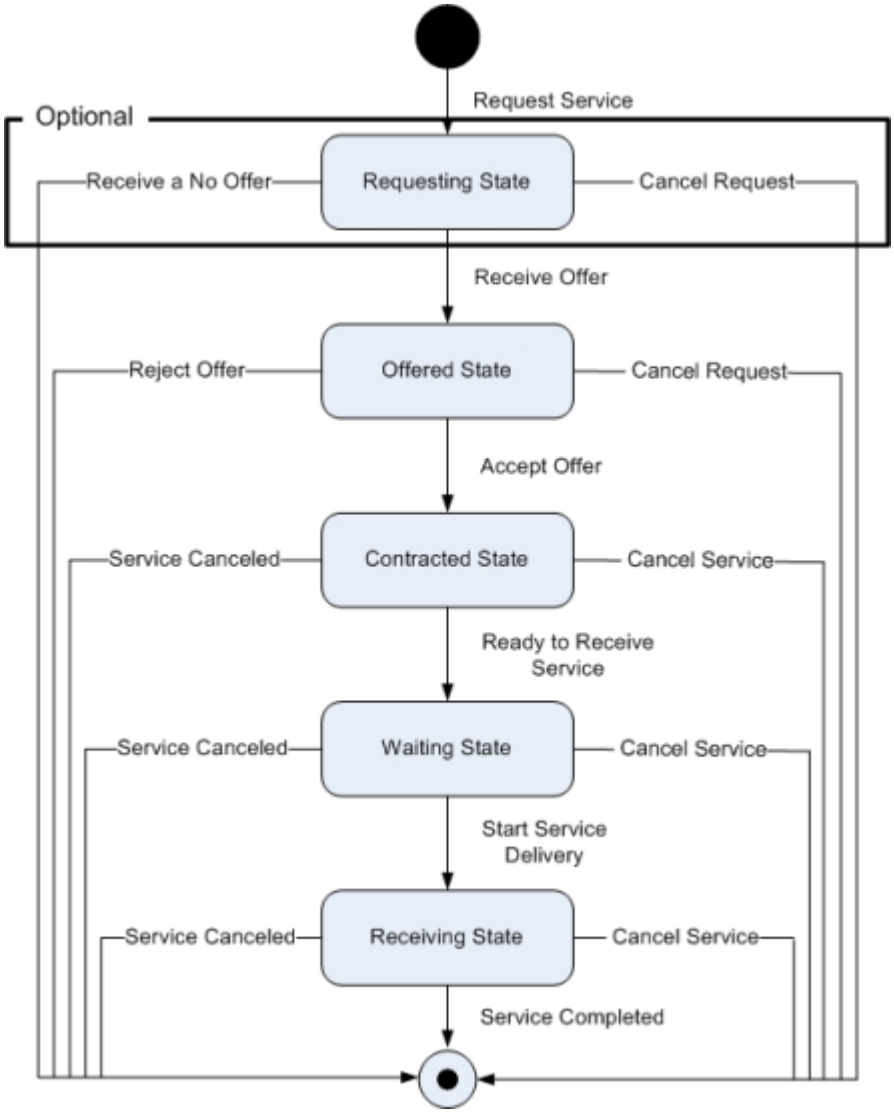
available by a producer in order to make an offer. On the other hand one might also consider providers making promises they know or don't know they cannot keep (like in real life).

7.2.1 Service Consumer

It is usually the service consumer that initiates a request for a specific service. A service consumer can be engaged in several concurrent service requests and deliveries. For each requested service the state of the service consumer can be described using a state-transition diagram (STD).

The Service Consumer entity may be in one of four states with respect to a requested service:

- **Requesting state**. A service consumer entity is in the Requesting State when it has requested a specific service from another entity
- **Offered state**. A service consumer entity is in the Offered State when an offer of the service delivery has been made by a service provider
- **Contracted state**. A service consumer entity is in the Contracted State when an offer has been accepted
- **Waiting State**. A service consumer is ready to receive the service and waiting for service delivery start
- **Receiving state**. A service consumer entity is in the Receiving State during service delivery

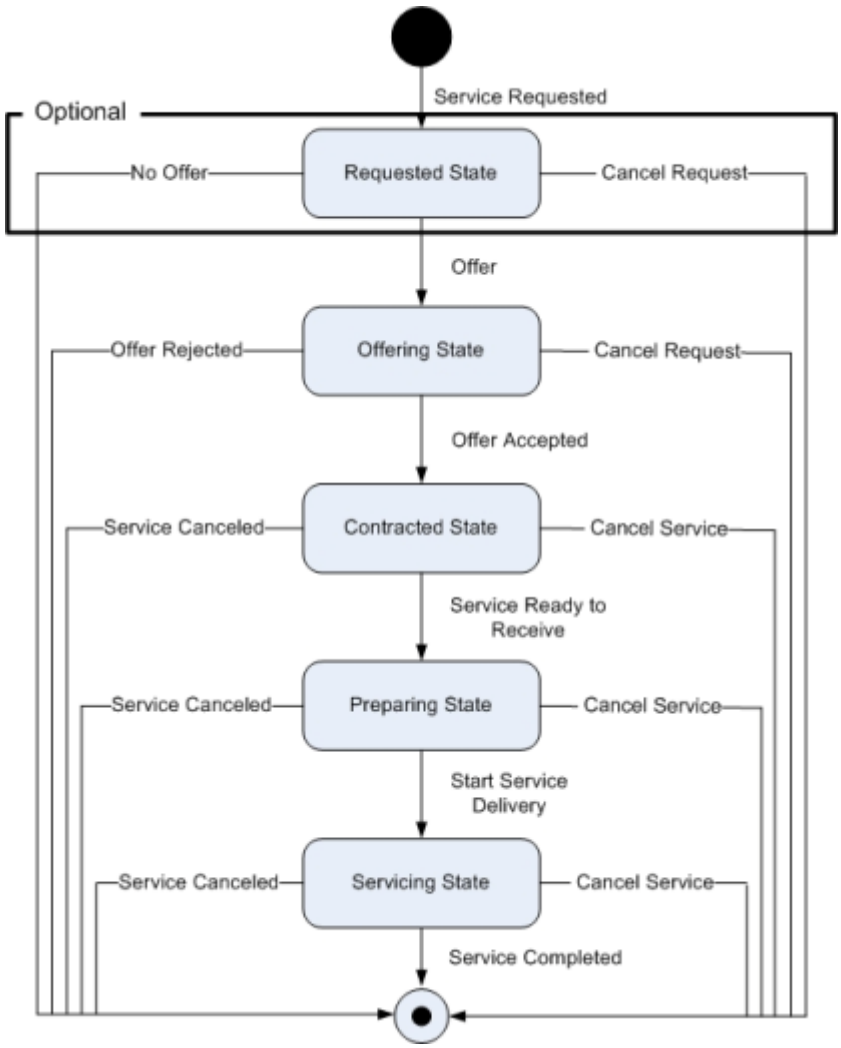


Transition	Condition and Actions
Request Service	When conditions for requesting a service are met, the consuming entity shall issue a NETN_RequestService interaction. The entity shall proceed from the Initial state to the Requesting state
Cancel Request	When conditions for requesting the services are no longer met a NETN_CancelService interaction is sent and the entity proceed from the Requesting state to the End state
Receive a no Offer	When a NETN_OfferService with a *NoOffer* indication is received, the entity shall proceed from the Requesting state to the End state
Receive Offer	When a NETN_OfferService with an *Offer* indication is received, the entity shall proceed from the Requesting state or Initial state to the Offered state
Reject Offer	When conditions for accepting a service offer are not met the service consuming entity shall issue a NETN_RejectOffer interaction and proceed to the End state
Accept Offer	Accept offer When conditions for accepting a service offer exists the service consuming entity shall issue a NETN_AcceptOffer interaction and proceed to the Contracted State
Ready to Receive Offer	When conditions for delivery of the service are met the service consuming entity shall issue a NETN_ReadyToReceiveOffer interaction and proceed to the Waiting State
Cancel Service	When conditions for receiving the services are no longer met a NETN_CancelService interaction is sent and the entity proceed to the End State
Service Canceled	When a NETN_CancelService interaction is received the service consuming entity shall proceed to the End State
Start Service Delivery	When an NETN_ServiceStarted interaction is received the service consuming entity shall proceed to the Receiving State
Service Completed	When a NETN_ServiceCompleted interaction is received or when the consuming entity determines that the conditions for service completed are met the NETN_ServiceReceived interaction shall be sent and the entity shall proceed to the End State

7.2.2 Service Provider

The Service Provider entity may be in one of four states with respect to a requested service:

- **Requested State**. A service producer entity is in the Requested State when it has received a request for a service from a service consumer entity
- **Offering State**. A service provider entity is in the Offering State when an offer in response to a requested service has been delivered to a service consumer
- **Contracted State**. A service provider entity is in the Contracted State when its offer has been accepted
- **Preparing State**. A service consumer has indicated it ready to receive the service, a service provider prepares the service delivery
- **Servicing state**. A service provider entity is in the Servicing state when a service is being delivered to a service consuming entity



Transition	Condition and Actions
Service Requested	When a NETN_RequestService is received the service providing entity shall proceed from the initial state to the Requested State
Cancel Request	When a NETN_CancelService interaction is received from a service consuming entity, the entity shall proceed from the Requested State to the End State
No Offer	When the conditions for delivering a requested service are not met, a NETN_OfferService with a *NoOffer* indication shall be sent and the service producing entity shall proceed to the End State
Offer	When the conditions for delivering a requested service are met, a NETN_OfferService including the offer shall be sent and the service producing entity shall proceed to the Offering State
Offer Rejected	When a NETN_RejectOffer is received the service producing entity shall proceed to the End State
Offer Accepted	When a NETN_AcceptOffer is received the service producing entity shall proceed to the Contracted State
Service Ready to Receive	When a NETN_ReadyToReceiveService is received the service producing entity shall proceed to the Preparing State
Cancel Service	When conditions for receiving the services are no longer met a NETN_CancelService interaction is sent and the entity proceed to the End state

Transition	Condition and Actions
Service Canceled	When conditions for providing the services are no longer met a NETN_CancelService interaction is sent and the entity proceed to the End state
Start Service Delivery	When the conditions for starting the service delivery is met, a NETN_ServiceStarted interaction is sent and the service providing entity proceeds to the Servicing state
Service Completed	When a NETN_ServiceReceived interaction is received or when the conditions for completed service delivery is met, a NETN_ServiceCompleted interaction is sent and the service producing entity proceeds to the End state

7.3 Interaction Classes

The Service Consumer-Provider Pattern defines a set of HLA interaction classes used to implement the three phases of the pattern. These interactions are provided as a FOM Module and can be extended to support specific service types.

Class 1	Class 2
NETN_Service	NETN_RequestService
	NETN_OfferService
	NETN_AcceptOffer
	NETN_RejectOffer
	NETN_CancelService
	NETN_ReadyToReceiveService
	NETN_ServiceStarted
	NETN_ServiceCompleted
	NETN_ServiceReceived

7.3.1 NETN_Service

The **NETN_Service** interaction class is the base class for all NETN Service Consumer-Provider Pattern interactions. It contains the basic required parameters (not optional) that are always sent.

Full Name: **HLAinteractionRoot.NETN_Service**

Parameter	Data type	Default value (if optional)	Definition
ServiceID	NETN_EventIdentifier	(Not Optional)	Unique identifier for a service
Consumer	NETN_Callsign	(Not Optional)	Requesting entity
Provider	NETN_Callsign	(Not Optional)	Providing or intended provider entity
ServiceType	NETN_ServiceTypeEnum8	(Not Optional)	Extension of RPR2 ServiceTypeEnum8

The NETN_ServiceTypeEnum8 enumerated data type is an extension of the **ServiceTypeEnum8** defined in the RPR-FOM v2.0.

7.3.2 NETN_RequestService

The request for a service is always initiated by a **NETN_RequestService**. Subclasses of this interaction for specific types of services may include parameters for detailing the requirements of this request. This may include information when, where and how the service is to be delivered.

Full Name: **HLAinteractionRoot.NETN_Service.NETN_RequestService**

Parameter	Data type	Default value (if optional)	Definition
RequestTimeOut	NETN_Integer64BE	0	Defined a deadline (date) for the provider response. Number of second since 01/01/1970. Default value zero (0) implies that the time out value has no meaning

7.3.3 NETN_OfferService

The **NETN_OfferService** is usually a response to a **NETN_RequestService** and contains information with respect to the providing entities ability to deliver the requested service. This ability is expressed as either an offer to provide the service or no offer. Subclasses of this interaction for specific types of offers should contain more detailed description of the offer. This may include information about when, where, how the service can be delivered.

Full Name: **HLAinteractionRoot.NETN_Service.NETN_OfferService**

Parameter	Data type	Default value (if optional)	Definition
IsOffering	HLAboolean	(Not Optional)	Defines if the requested service is offered (=true) or not (=false)
RequestTimeOut	NETN_Integer64BE	0	Defined a deadline (date) for the consumer response. Number of second since 01/01/1970. Default value zero (0) implies that the time out value has no meaning

7.3.4 NETN_AcceptOffer

The **NETN_AcceptOffer** is used to accept an offer made by a service providing entity as indicated in a **NETN_OfferService** interaction. By issuing a **NETN_AcceptOffer** interaction the service consuming entity enters a contract for service delivery with the service producing entity.

Full Name: **HLAinteractionRoot.NETN_Service.NETN_AcceptOffer**

The NETN_AcceptOffer interaction does not define any additional parameters but subclasses may include parameters with additional information.

7.3.5 NETN_RejectOffer

The **NETN_RejectOffer** is used to reject an offer made by a service providing entity as indicated in a **NETN_OfferService** interaction. By issuing a **NETN_RejectOffer** interaction the service consuming entity informs the providing entity that the offer has been rejected.

Full Name: **HLAinteractionRoot.NETN_Service.NETN_RejectOffer**

The **NETN_RejectOffer** interaction does not define any additional parameters but subclasses may include optional parameters for detailing the reasons for rejecting the service.

7.3.6 NETN_CancelService

The **NETN_CancelService** interaction is used by either a service consuming entity or a service providing entity to inform about early termination of the service delivery or in some cases termination of the service request before delivery has begun.

Full Name: **HLAinteractionRoot.NETN_Service.NETN_CancelService**

The **NETN_CancelService** interaction does not define any additional parameters but subclasses may include optional parameters for detailing the reasons for canceling the service.

7.3.7 NETN_ReadyToReceiveService

The **NETN_ReadyToReceiveService** interaction is issued by a service consuming entity to indicate that the start of service delivery can start. The time of service delivery start may be significantly later then indicating ready for service delivery.

Full Name: **HLAinteractionRoot.NETN_Service.NETN_ReadyToReceiveService**

The **NETN_ReadyToReceiveService** interaction does not define any additional parameters.

7.3.8 NETN_ServiceStarted

The **NETN_ServiceStarted** interaction is issued by a service providing entity to inform about the start of service delivery. The time of service delivery start may be significantly later then receiving a indication from the consumer that the service delivery can start.

Full Name: **HLAinteractionRoot.NETN_Service.NETN_ServiceStarted**

The **NETN_ ServiceStarted** interaction does not define any additional parameters.

7.3.9 NETN_ServiceCompleted

The **NETN_ServiceCompleted** interaction is used by a service providing entity to inform the service consuming entity that the service has been delivered.

Full Name: **HLAinteractionRoot.NETN_Service.NETN_ServiceCompleted**

The **NETN_ ServiceCompleted** interaction does not define any additional parameters.

7.3.10 NETN_ServiceReceived

The **NETN_ServiceReceived** interaction is used by a service consuming entity to inform the service providing entity that the service has been delivered.

Full Name: **HLAinteractionRoot.NETN_Service.NETN_ServiceReceived**

The **NETN_ ServiceReceived** interaction does not define any additional parameters.

7.4 Simple Datatypes

Name	Representation	Semantics
NETN_Integer64BE	HLAinteger64BE	
NETN_Integer32BE	HLAinteger32BE	
NETN_Integer16BE	HLAinteger16BE	

Name	Representation	Semantics
NETN_Float64BE	HLAfloat64BE	
NETN_Float32BE	HLAfloat32BE	

7.5 Arrays

Name	Representation	Encoding	Semantics
NETN_Callsign	HLAunicodeString	HLAvariableArray	

7.6 Fixed Records

7.6.1 NETN_EventIdentifier

Name	NETN_EventIdentifier	
Encoding	HLAfixedRecord	
Semantics	-	
Field Name	**Type**	**Semantics**
EventCount	NETN_Integer32BE	
IssuingObjectIdentifier	NETN_Callsign	

7.7 Enumerated Datatypes

7.7.1 NETN_ServiceTypeEnum8

Name	NETN_ServiceTypeEnum8	
Representation	HLAoctet	
Semantics	-	
Enumeration	**Value**	**Semantics**
Other	0	
Resupply	1	
Repair	2	
Storage	3	
Convoy	4	
CombatAdjudication	5	

8 NETN Logistics FOM Module v1.0

8.1 Scope

The NETN Logistics FOM Module is based on the NETN Service Consumer-Provider FOM Module with extensions to support specific logistics services as defined below. Detailed description on how these services map to the NETN Logistics interactions are included in this document.

The NETN Logistics FOM Module is also dependent on the RPR-FOM v2.0 due to the fact that several data types defined in the RPR-FOM are reused in the definition of parameters for logistics interactions.

In addition, a Transfer of Control pattern is introduced as an option for some logistics services. Extensions to existing RPR-FOM object classes are proposed.

The NETN Logistics FOM Module covers the following services:

- **Supply service** is provided by a facility, a unit or any battlefield entity with consumable materials supply capability
- **Storage service** is provided by a facility or means of transportation capable of storing consumable materials
- **Repair service** can be performed on equipment (i.e. non-consumables items such as platforms) by facilities or units capable of performing requested repairs. The repair service does not transfer the control of a damaged platform to the repairing facility
- **Transport service** provides a means of transport capable of storing and delivering non-consumable materials. Materials are embarked, transported and disembarked, with possible use of a Transfer of Control protocol
- **Embarkment service** provides a means of transport or a facility capable of storing non-consumable materials, with possible use of a Transfer of Control protocol
- **Disembarkment service** provides a means of transport or a facility capable of delivering non-consumable materials, with possible use of a Transfer of Control protocol

Examples of uses:

- Resupply of units (Consumer) by transportation means
- Supply of fixed wings in airports or during aerial refueling
- Supply of helicopter in assembly areas
- Transport of troops by train, ship and aircraft
- Repair of damaged platforms by a maintenance unit without changing platforms location
- Maintenance of damaged platforms previously deposited in a facility
- Embarkment and disembarkment of small or large units
- etc.

8.2 Definitions

Logistics supplies the troops with material and carries out maintenance.

Means of transport, transport resources and facilities for maintenance and storage are required for these tasks.

The term of **unit** will be used for individual platform entities as well as for aggregate entities.

8.2.1 Materials

Materials are differentiated between:

- Consumable materials
  - Ammunition
  - Mines
  - NBC Materials
  - Fuel (Diesel, Gas, Aviation fuel, etc.)
  - Water
  - Food
  - Medical materials
  - Spare parts
- Non-consumable materials
  - Vehicles
  - Aggregates
  - Reconnaissance and Artillery systems (Radar)
  - Missile

The NETN Reference Federation Agreements follows the RPR FOM convention by treating the above non-consumable materials as platform objects. Consumable materials, hereafter also referred to as supplies, differ from non-consumables in that the former can be transferred to a federation object, thereby "resupplying" that object with the appropriate consumable material. Consumable materials are further differentiated between piece goods and bulk goods (e.g. fuel, water, decontamination means). Material may therefore be requested as individual pieces (each), or in cubic meters for liquid bulk goods and kilograms for solid bulk goods.

The type of packaging (fuel in canisters, water in bottles, etc.) is not taken into account.

> *Note: Many types of materials are often grouped together. Examples of this, taken from different simulations, are:*

- *Many artillery models do not distinguish between propellants, fuses, warheads, etc.*
- *Diesel, gas, and aviation fuel are grouped together under "Fuel".*
- *Medical material is only roughly divided into different classes.*
- *The supply of food and water is not represented in detail.*

Only damaged platforms will be delivered to maintenance (repair). The required effort for the repair of damaged material is determined by the provider model. It is calculated, based on the degree of damage to the material.

8.2.2 Means of transport

Depending on the federation, means of transport are published as platforms or as equipment of aggregate units. If transportation means are used as provider in the NETN Service Consumer-Provider pattern, they have to be published and registered as platforms objects in the federation. This suggested approach does not require the publishing of the loading of the means.

For example, if a means of transport is to supply units, it then proceeds to a depot, or is itself a depot. Objects of other federates will then be supplied from the depot (supply facility).

8.2.3 Transport resources

The suggested transport resources are containers and flats. Depending on the federate agreements, resources can be published as platforms. If the transport resources are not exchanged between federates, they do not need to be published within the federation.

8.2.4 Facilities

Facilities are the central element through which material can be transferred. Facilities may be created during a simulation or may be a part of the infrastructure (railway station, storage tanks depot, port, etc.). A facility may be part of a unit (e.g. ship, etc.).

8.3 Transfer of Control

The main objective is to provide Transfer of Control mechanisms in support of NETN interactions between federates collaborating through logistics process.

8.3.1 Principles of Transfer of Control

Transfer of Control is optional depending on operational level of simulations:

- Some users will need to transport whole entities, including Aggregates units. It is often impossible to break down small units like platoons or sections.
- Some others will absolutely need to break down large units like brigades or battalions (often Aggregates), in order to make them transported and usable in more than one single position at the same time.

Therefore, the use of Transfer of Control as described in this Logistics FOM Module is not mandatory. Simulations will be able to cooperate with others, using logistics Convoy pattern without doing any Transfer of Control action.

In case of not using Transfer of Control, a transported unit remains **active** during transportation. So location of the transported unit remains at the embarkment location until disembarkment. Then, a jump of location at the disembarkment time occurs. That could cause an error because another unit could interact with the transported one, considering its embarkment position. Transportation can be made in one or several trips by one or several transporters.

The Transfer of Control protocol described here is **not an HLA service**.

From a Consumer point of view, Transfer of Control is limited because:

- The unit is transferred to the Provider, but the Consumer conserves its property. The Consumer is still in charge of updating the unit (position, operational status, consumptions, etc.) with obligation to declare an Active / Inactive status parameter (see below).
- The Consumer control on the transferred unit is limited to “passive” functions, e.g. supply or repair. Passive function implies no movement, combat or destruction.

There is no total only limited Transfer of Control of an entity. The control by the Provider applies only to some transferred elements (or sub-elements) of the unit.

Therefore, a transporter (here Provider) does not directly use the simulated object during transportation: entities or aggregates shall be represented by a structure that gives main characteristics of the embarked platform(s). This is the reason why a request for Convoy shall give a list of platforms (and their characteristics) to be transported. Transporter shall then manipulate this list during

transportation and on disembarkment a platform simulation shall build a new object that corresponds to the list.

8.3.2 Use of an Active / Inactive status parameter

Entities or aggregates shall have a status parameter that defines if they are “Active” or “Inactive” in the federation (the Logistics FOM Module extends all RPR-FOM Platform Object Classes with this parameter):

- “Active” is the default status of any element not transferred at initialization time. An “Active” element can react to the simulation environment.
- “Inactive” means that an element is not simulated. It does not react to the simulation environment, like a dead unit. Other federates following these agreements recognize its aggregate status as deactivated and should not try to interact. The HLA object of the “inactive” unit remains in the federation and models continue to subscribe to its HLA update.

8.3.3 Disaggregation of units during transportation

As transporters could generally not transport a large aggregate unit in one travel, the list of components of the aggregate to be transported (i.e.: list of the entities or sub-aggregates) should be managed by the consumer as non-divisible components.

The Consumer can provide a list of components (optional) to make the Transporter able to share this list in different means of transport, with obligation to transport all of it. If the list of components is not provided, the Transporter will consider the transportation of this whole unit, in order to accept or reject this service request.

From a Consumer point of view, to provide a list of components is recommended for units bigger than a platoon

Concerning **entities management**:

- **During Embarkment:** when a federate receives an EmbarkmentStatus, it must analyze the optional list of embarked objects. At this step, the consumer federate must set the entity (identified by a callsign) as inactive. The entity is no more taken into account in simulation.
- **During Disembarkment:** when a federate receives a DisembarkmentStatus, it must analyze the optional list of disembarked objects. If an object from the list is defined at entity level (see description of NETN_ArrayOfObjectDefinition), an entity object shall be retrieved with the given callsign. Federate updates the entity status as active, and its location at Disembarkment position.
- **During Transport**: combination of the two previous items.

Concerning **aggregates management**:

- **During Embarkment:** when a federate receives an EmbarkmentStatus, it must analyze the optional list of embarked objects and identify those corresponding to aggregates, using callsigns. Federate can then go through the embarked objects list (see description of NETN_ArrayOfObjectDefinition). This complete or partial list describes objects constituting the aggregate. At this step, federate has to update the aggregate representation by removing the identified objects from its internal list. When federate receives an EmbarkmentEnd interaction, and if all listed objects of the aggregate are removed, aggregate is set in an inactive state. Then, the aggregate could be removed from the federation.
- **During Disembarkment:** when a federate receives a DisembarkmentStatus, it must analyze the optional list of disembarked objects. If an object from the list is defined at aggregate level (see description of NETN_ArrayOfObjectDefinition), it could be necessary to create a new aggregate object, defined as Bridgehead. It must be identified by a new callsign on the federation.
- **During Transport:** combination of the two previous items.

8.3.4 Representation of "Inactive" units

When the status of a unit is set as « Inactive » the recommended representation in systems reflecting the unit is to not display an object (hidden).

8.3.5 Representation of "Active" units

When the status of a unit is set as « Active » the object shall be displayed in the systems reflecting the unit.

As the same aggregate entity can simultaneously exist in several locations, multiple representations have to be managed by the simulations. To ensure its membership to the same units in shared Order of Battle, simulations have to use similar callsign for different representations of a same unit.

For multiple naming, see section Transport by several transporters in several travels with ToC.

For multiple representations, see illustration in section Illustration.

A unit single representation during transportation is also possible:

- Using a non non-divisible unit (no available information on content, or to small unit to be divided),
- Using a transport in one travel by one single transporter (with or without Transfer of Control).

For a multiple represented unit; addition of its different potentials should not exceed 100% for a same item.

8.3.6 Example: Transport by one single transporter in one travel with ToC

In this case, the provider is able to transport a complete unit (from the consumer) using one single transporter (boat, aircraft, train) and doing one single trip.

The transported unit is deactivated at the embarkment time and reactivated at the disembarkment location at the right time.

If the transporter is destroyed, the consumer service is cancelled and the transported unit is never reactivated.

8.3.7 Example: Transport by several transporters in several travels with ToC

In this case, a transported unit transfers a list of platform entities for each transporter. This seems to be equivalent to a transfer of equipments for Supply pattern. For each embarkment event, a NETN aggregate object update is published by the transported unit.

If there are no more platform entities to transport, the transported unit is deactivated (aggregate status attribute to inactive).

If the transported unit is still active (because embarkment is not completed) when a disembarkment occurs, a temporary Bridgehead unit is activated at the disembarkment location. It uses a new HLA ID and the same callsign than the transported unit, but with an extension (-bh).

For each disembarkment event, embarked platform entities are transferred toward a Bridgehead unit.

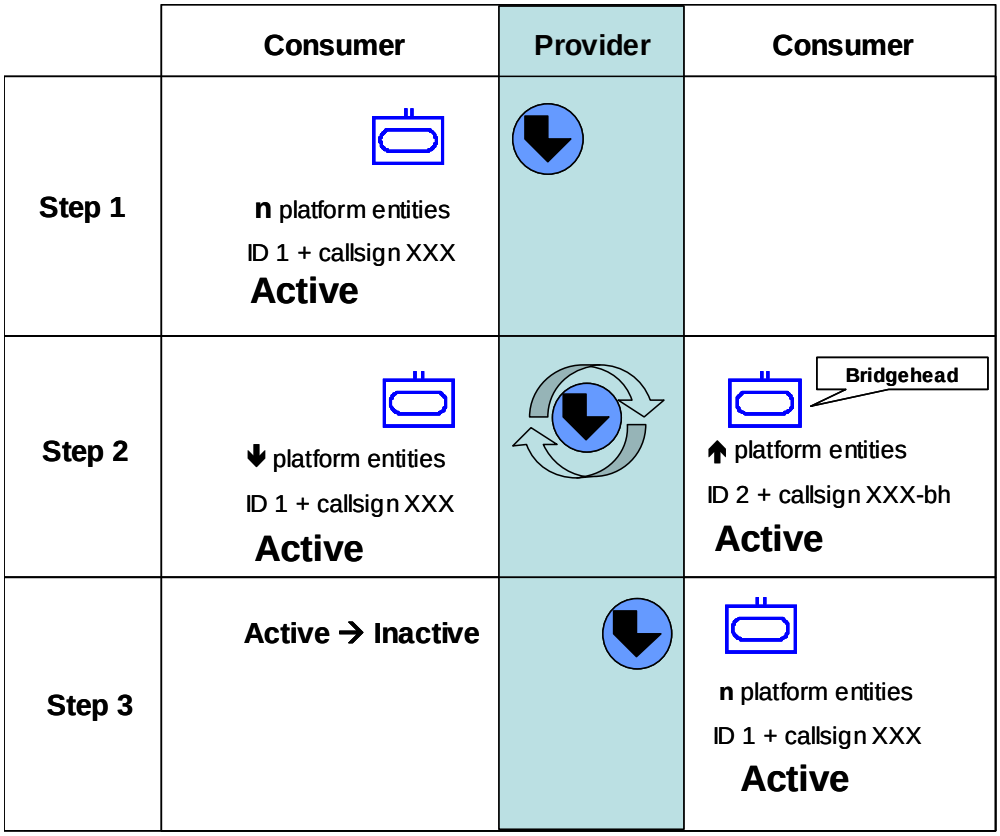
When the transported unit is no longer active (transport service is completed), the Bridgehead unit is suppressed and the original unit replaces it at the same location.

If a transporter is destroyed with transported units onboard, consumer service continues until service completion by other transporters.

Only one Bridgehead and one Disembarkment location per transported unit is supported.

8.3.8 Illustration

The following scheme illustrates a transport by several transporters (boats) in several travels, with Transfer of Control and multiple representation aspect.



8.4 Supply Pattern

Services for resupply of consumable materials include:

- **Supply service** provided by a facility, a unit or any battlefield entity with consumable materials supply capability,
- **Storage service** provided by a facility or means of transportation capable of storing consumable materials.

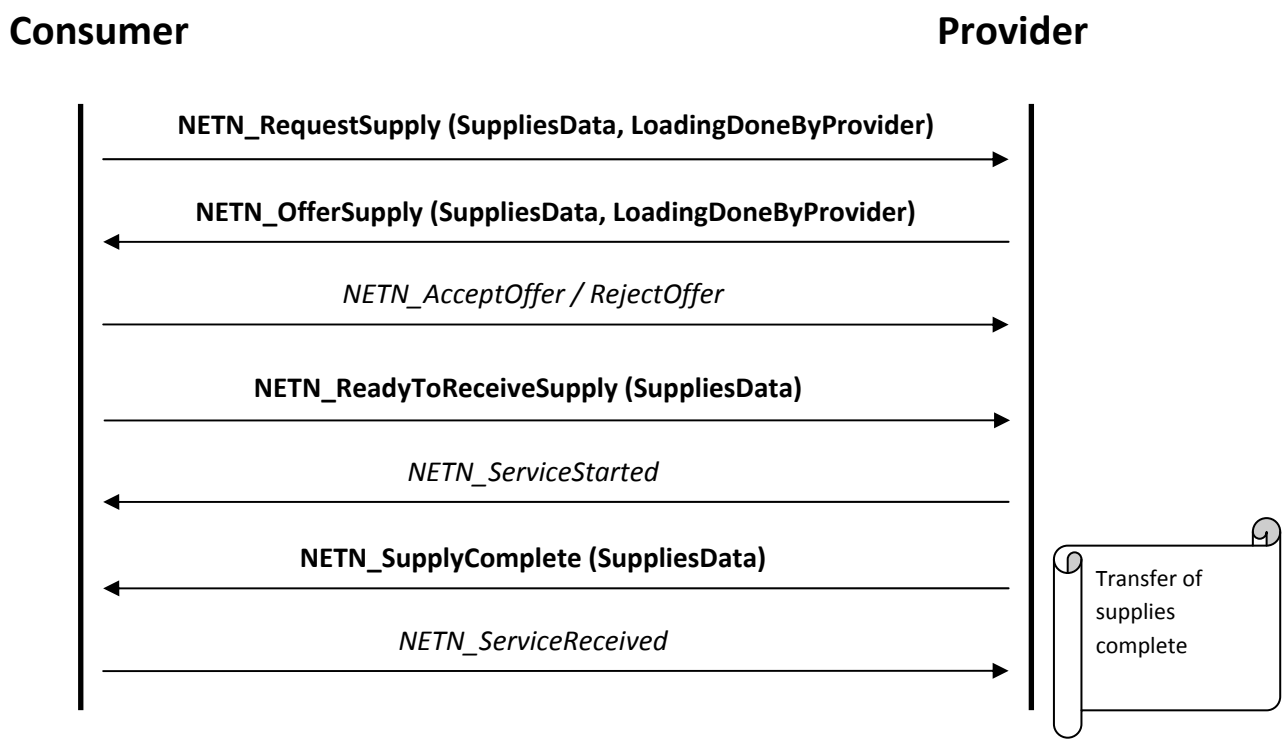
These two services are different in terms of flow of materials between service consumer and provider. In the supply service, materials are transferred from the service provider to the service consumer. In the storage service, the user of the storage facility necessarily has material which requires storage, thus the materials are transferred from the service consumer (e.g. a transport arriving at a depot) to the service provider (e.g. the storage facility) using the storage service.

Both services follow the basic NETN Service Consumer-Provider pattern to establish a service contract and a service delivery.

Materials will be transferred after the offer is accepted and the service is started. This pattern allows partial transfers. This means that only some of the materials described in the service contract are transferred. If the service is cancelled during service delivery, the provider must inform the consumer of the amount and type of material transferred.

If requested materials are only partially transferred in the NETN_ServiceStarted interaction, the consumer has to start another NETN_RequestSupply in order to obtain all desired supplies.

8.4.1 Supply Service



NETN_RequestSupply is used by a consumer to initiate a request for supply from a supply service provider. Amount and type of materials are included in the request. In this request, the consumer specifies a preference for whether the loading is done by the provider or by the consumer.

NETN_OfferSupply is used by a supply service provider to indicate which of the requested materials (amount and type) can be offered. In this offer the provider can agree with the consumer's loading request or counter-offer based on current loading capability.

NETN_ReadyToReceiveSupply is used by a service consumer to indicate that supply delivery may start. It also updates the requested amount of supplies based on the consumer's current supply requirements at the time the consumer is ready to receive supplies. Note that the updated supply amount(s) are subject to the constraint that the amount(s), by type, must be less than or equal to the amount(s), by type, of offered supplies.

NETN_SupplyComplete is used by the service provider to inform the consumer of the amount, by type, of supplies transferred. This allows for supply pattern interruptions due to operational necessity, death/destruction of either the consumer or provider during resupply, etc.

Transfer of materials in supply service is considered as complete when:

- The service consumer receives a NETN_SupplyComplete interaction **and**
- The service provider receives a NETN_ServiceReceived interaction.

If the time specified in the *RequestTimeOut* parameter of the NETN_RequestSupply passes without the Provider sending a NETN_OfferSupply, the Consumer **shall** send a NETN_CancelService. The Consumer may then again initiate a NETN_RequestSupply interaction.

The **LoadingDoneByProvider** parameter is used by the consumer to propose whether the loading is done by him or by equipment belonging to the facility; the provider can agree or disagree with the consumer’s proposal.

When ready to receive, the consumer can indicate a modified SuppliesData to include fewer/less supplies than offered by the provider. Note that NETN_ReadyToReceiveSupplies.SuppliesData must be less than or equal to the NETN_OfferSupply.SuppliesData amount.

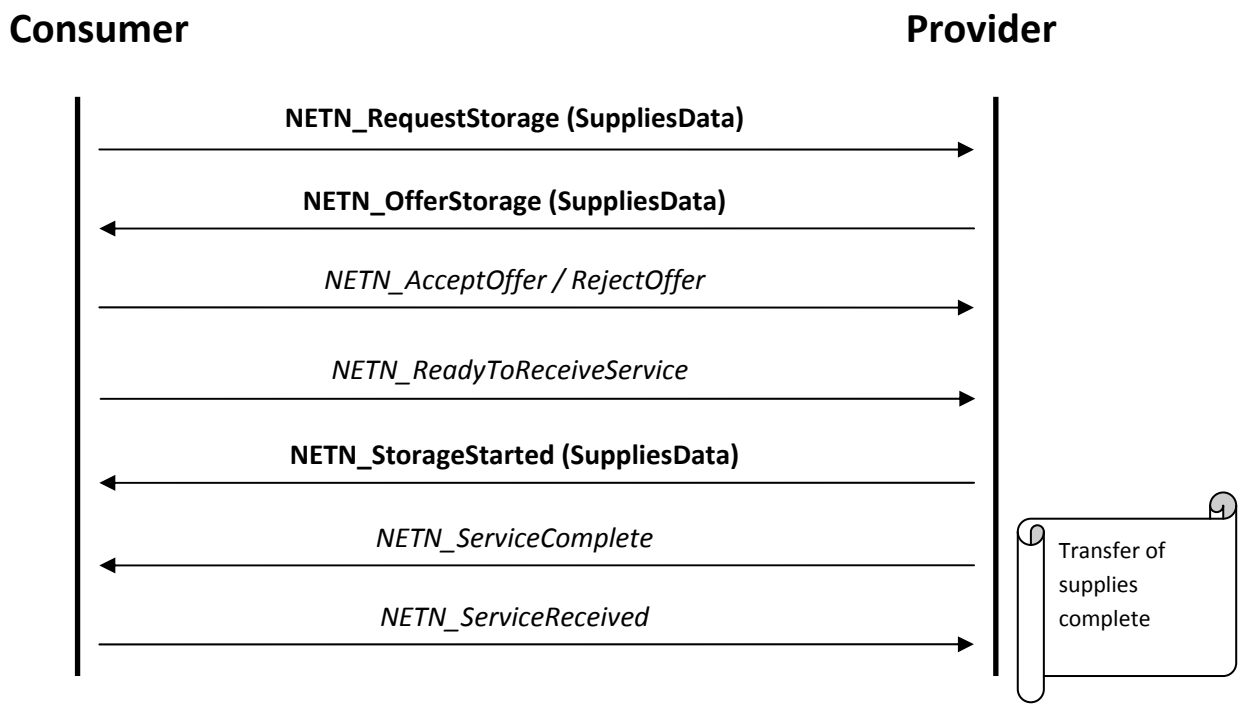
The provider can transfer only a part of the offered materials (partial transfer); the actual transferred supplies are identified in SuppliesData parameter of the NETN_SupplyComplete interaction.

The consuming entity shall issue a NETN_ServiceReceived in response to the NETN_SupplyComplete interaction. Transfer of supplies is considered as complete once the NETN_ServiceReceived is issued.

Early termination of the service request or delivery (as defined in the Service Consumer-Provider Pattern) is possible by either consumer or provider initiating a NETN_CancelService. If the NETN_CancelService occurs between NETN_ServiceStarted and NETN_SupplyComplete, the provider will inform the consumer of the amount of supplies transferred using NETN_SupplyComplete.SuppliesData.

Rejection of a service offer is allowed. In this case, no material will be transferred.

8.4.2 Storage Service



NETN_RequestStorage is used by a consumer to initiate a request for storage of supplies. Amount and type of materials are included in the request.

NETN_OfferStorage is used by a storage service provider to indicate which (amount and type) of the requested materials can be stored.

NETN_StorageStarted is used by a service provider to indicate that the (partial) storage of requested materials has started.

The consuming entity shall issue a NETN_ServiceReceived as response to the NETN_ServiceCompleted interaction. Transfer of supplies is considered as complete once the NETN_ServiceReceived is issued.

The consumer determines whether the loading is done by him or by equipments belonging to the facility.

The storage provider can limit the transfer of supplies to a subset of the offered supplies when issuing the NETN_StorageStarted interaction.

Early termination of the service request or delivery (as defined in the Service Consumer-Provider Pattern) is possible. On early termination, no materials will be transferred.

Rejection of a service offer is allowed. In this case, no materials will be transferred.

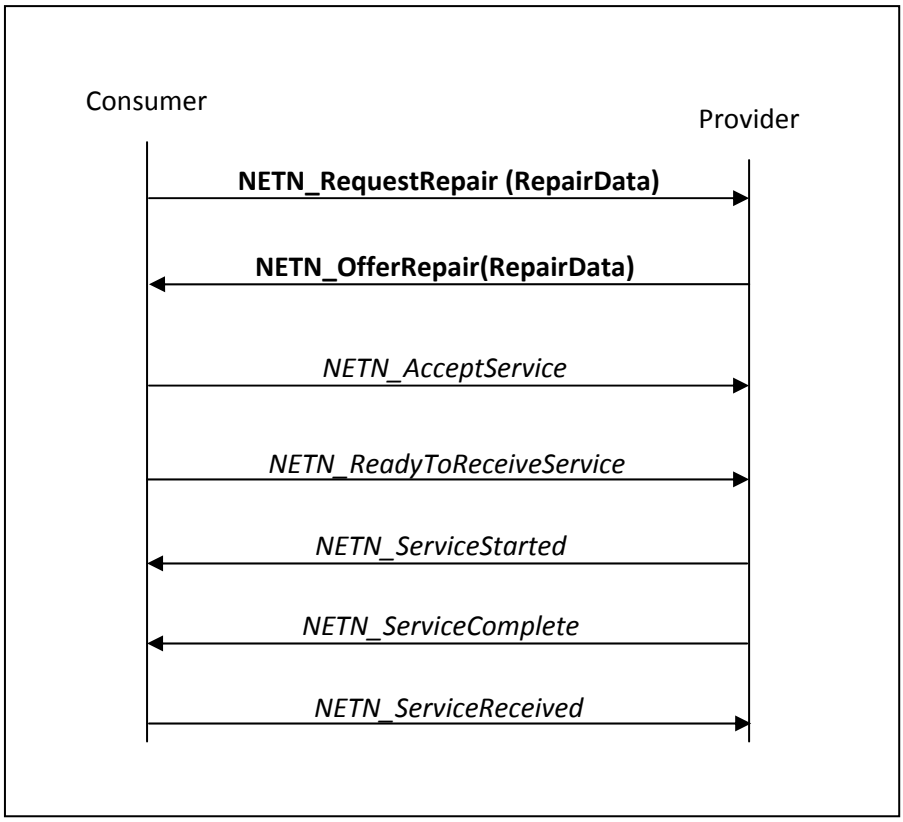
8.5 Maintenance Pattern

8.5.1 Repair Service

Repair service can be performed on equipments (i.e. non-consumables items such as platforms). Providers of this service are facilities capable of performing requested repairs.

The service is initiated by a **NETN_RequestRepair** interaction, sent by a federate modelling the owner of damaged equipments (for example damaged platforms). The service-provider offer the repair service by sending the **NETN_OfferRepair** interaction.The NETN Service Consumer-Provider interactions are used to complete the service.

Note: The maintenance pattern does not include a transfer of control of the damaged platform to the repairing facility. If such a transfer was required, the consumer should initiate a request for deposit (instead of request for repair) with an attached work order describing the required repairs.



The RepairData parameter is a list of equipments and type of repairs. List of offered repairs may be different from the list of requested repairs. If the HLA object (equipment to be repaired) has a damaged state, the list of requested repairs could be empty.

The provider federate models the efforts to repair a damaged platform.

If the consuming entity is an aggregate entity, its damaged equipment has to be listed in a platform list to get repaired.

8.5.2 Repair Types

The *RepairTypeEnum16* enumerated data type is defined in the RPR-FOM v2.0 and identifies a large set of repair types. DIS does not define the enumerated values as part of the core specification. Enumerated values are defined in a separate agreements document instead.

In the RPR-FOM however values are defined as a fixed part of some enumerated data types. In order not to violate the modular FOM merging rules, the NETN Logistics FOM module does not define any extensions to these data types as part of the FOM module. A separate table for adding values to the existing range of enumerations defined in the RPR-FOM is allowed instead.

This table shall be part of any federation specific agreements where extensions to an enumerated data type are required. It is also recommended but not required that any additional enumerated values added to this data type shall be submitted as Change Requests to the SISO RPR-FOM Product Development Group. All existing enumerators in RPR-FOMv2.0 and their values are reserved. Additional repair types are documented in the federation specific agreements.

8.6 Convoy Pattern

Convoy services are used in any cases covering management or transport of non-consumable materials such as platforms, units or battlefield entities.

Services for Convoy include:

- **Transport service** provided by a means of transportation capable of storing and delivering non-consumable materials.
- **Embarkment service** provided by a means of transportation capable of storing non-consumable materials.
- **Disembarkment service** provided by a means of transportation capable of delivering non-consumable materials.

Both Embarkment and Disembarkment services could also be extended to management of facilities; with the capability of delivering and storing non-consumable materials to/from other facilities, units or battlefield entities.

Convoy services include a "Transfer of Control" mechanism between a service consumer and a service provider over the unit managed by the means of transportation (see section Transfer of Control).

Convoy services differ in terms of the flow of units between service consumer and service provider:

- In Disembarkment service, units are transferred from a service provider to a service consumer.
- In Embarkment service, units are transferred from a service consumer to a service provider.
- In Transport service, both types of units transfer (generated by Embarkment and Disembarkment services) exist.

All convoy services follow the basic NETN Service Consumer-Provider pattern for establishing a service contract and a service delivery. The following interaction classes are extensions of the NETN Service Consumer-Provider interactions:

- **NETN_RequestConvoy** interaction is used by a consumer to initiate a request of convoy to a convoy service provider. Convoyed units and constraints are included in the request.
- **NETN_OfferConvoy** interaction is used by a convoy service provider to indicate which of the requested units can be managed. In this offer, a provider can change the constraints specified in the request and propose to manage only a part of asked units.
- **NETN_RejectOfferConvoy** interaction is used by a service consumer to signify his disagreement to the provider about the proposal. Consumer can indicate the reason of his reject.
- **NETN_CancelConvoy** interaction is used by a service consumer or a service provider to cancel the negotiated convoy service. The reason of the cancel can be indicated.

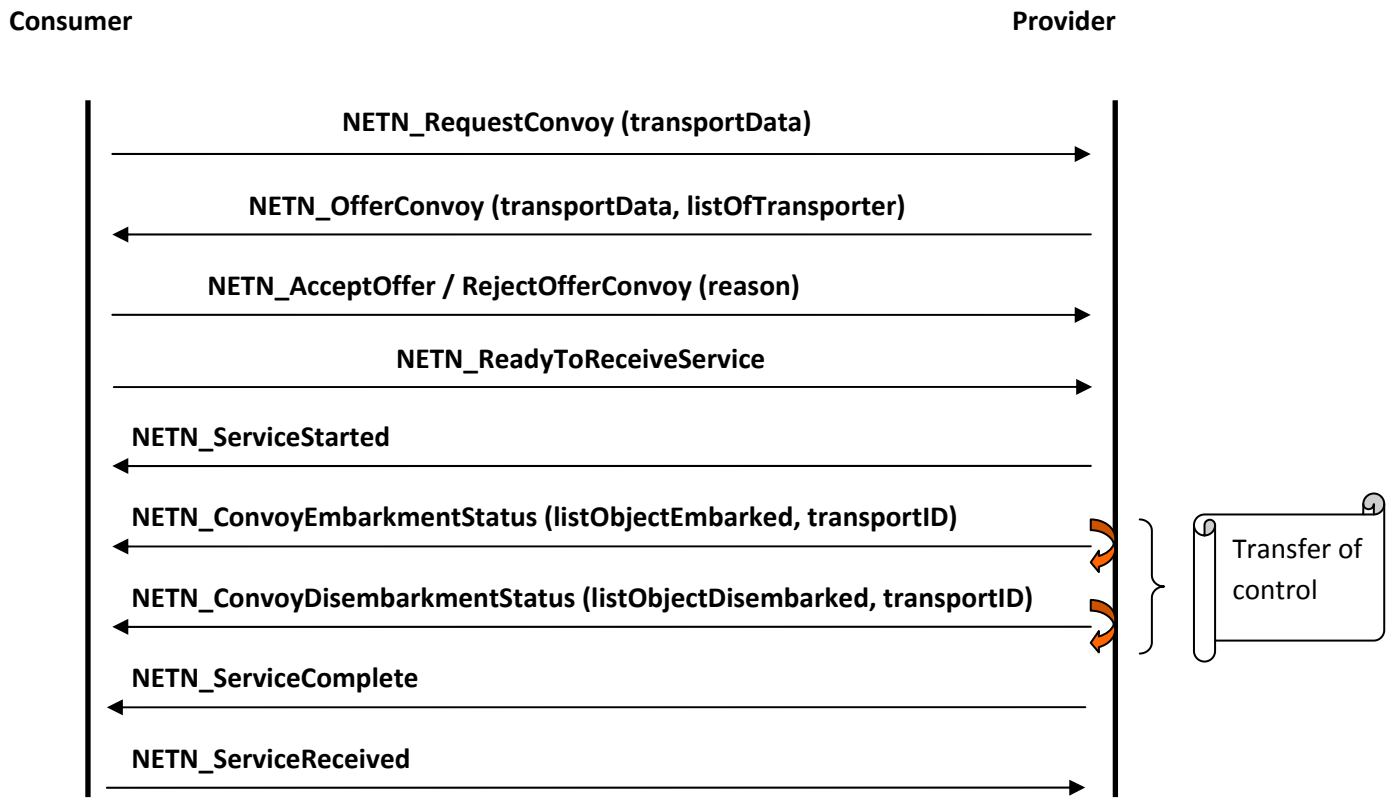
During Convoy services execution, a service provider can inform a service consumer about the service progress using the following interactions:

- **NETN_ConvoyEmbarkmentStatus** interaction is used by a service provider to indicate precisely when units are embarked, so when ToC is applied.
- **NETN_ConvoyDisembarkmentStatus** interaction is used by a service provider to indicate precisely when units are disembarked, so when ToC is restored.
- **NETN_ConvoyDestroyedEntities** interaction is used by a service provider to indicate the damage state of unit during the ToC.

The following NETN Service Consumer-Provider interactions are also used in Convoy pattern:

- NETN_AcceptOffer
- NETN_ReadyToReceiveService
- NETN_ServiceStarted
- NETN_ServiceComplete
- NETN_ServiceReceived

8.6.1 Transport service



A Consumer makes a request for transport with the following data:

- Time and location where units would like to embark and disembark,
- Characteristics of each simulated entity to manage,

A Provider offers a response to the consumer with the following parameters:

- Time and location where units could embark and disembark,
- List of simulated entities the Provider is able to manage and transport units planned to be used,

Offered services are accepted only when both service Consumer and Provider are agreeing about the conditions for delivery the service. Provider can change some elements of the request in this offer:

- Time and location where units must be embarked and disembarked, if the requirements of the request cannot be satisfied.
- Partial delivery offer by managing fewer units than requested.

To achieve Transport service, Consumer must be present on time at the meeting point in order to embark and publish the NETN_ReadyToReceiveService interaction.

During a Transport service execution, each transporter enters a loop where:

- It publishes a list of embarked units. The responsibility of units specified in this list is transferred to the Provider until disembarkment (see section Transfer of Control)
- It publishes a list of disembarked entities. The responsibility of entities specified in this list is restored to the Consumer when disembarked (see section Transfer of Control)

Both **NETN_ConvoyEmbarkmentStatus** and **NETN_ConvoyDisembarkmentStatus** interactions can be repeated as much as needed, if transportation needs to be realized in several iterations.

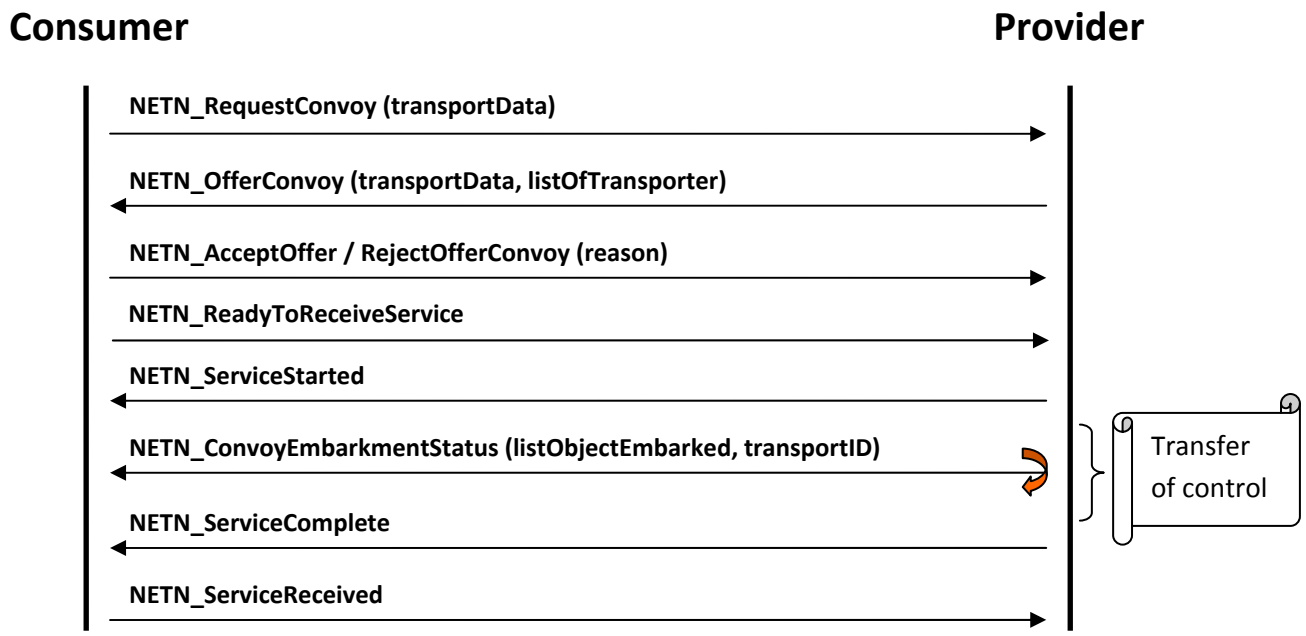
A Transport service is considered as closed:

- When service Provider receives a NETN_ServiceReceived interaction and service Consumer receives a NETN_ServiceCompleted interaction.
- Or when a NETN_CancelConvoy or a NETN_RejectOfferConvoy is used by the service Provider or Consumer.

If a Transport service is cancelled:

- During a delivery phase (between start and complete):
  - All units already embarked or partially embarked are kept by the service Provider. A new Request is needed by the service Provider to continue to embark or disembark units (restore).
  - All units already disembarked or partially disembarked are kept by the service Consumer. A new Request is needed by the service Provider to continue to disembark or re-embark units (restore).
- During a negotiation phase (before start): transaction between service Consumer and Provider is considered as closed without delivery of service.
- After a service delivery (after complete): a cancel does not make sense in this case (no effect).

8.6.2 Embarkment service



A Consumer makes a request for embarkment with the following parameters:

- Time and location where units would like to embark,
- Characteristics of each simulated entity to manage,

A Provider offers a response to the Consumer with the following parameters:

- Time and location where units could embark,
- List of simulated entities the Provider is able to manage and transport units planned to be used,

Offered services are accepted only when both service Consumer and Provider are agreeing about the conditions for delivery the service. Provider can change some elements of the request in this offer:

- Time and location where units must be embarked, if the requirements of the request cannot be satisfied.
- Partial delivery offer by managing fewer units than requested.

To achieve Embarkment service, Consumer must be present on time at the meeting point in order to embark and publish the NETN_ReadyToReceiveService interaction.

During an Embarkment service execution, each transporter enters a loop publishing a list of embarked units. The responsibility of units specified in this list is transferred to the Provider and never given back to the Consumer in this protocol (see section Transfer of Control).

A **NETN_ConvoyEmbarkmentStatus** interaction can be repeated as much as needed, if embarkment needs to be realized in several iterations.

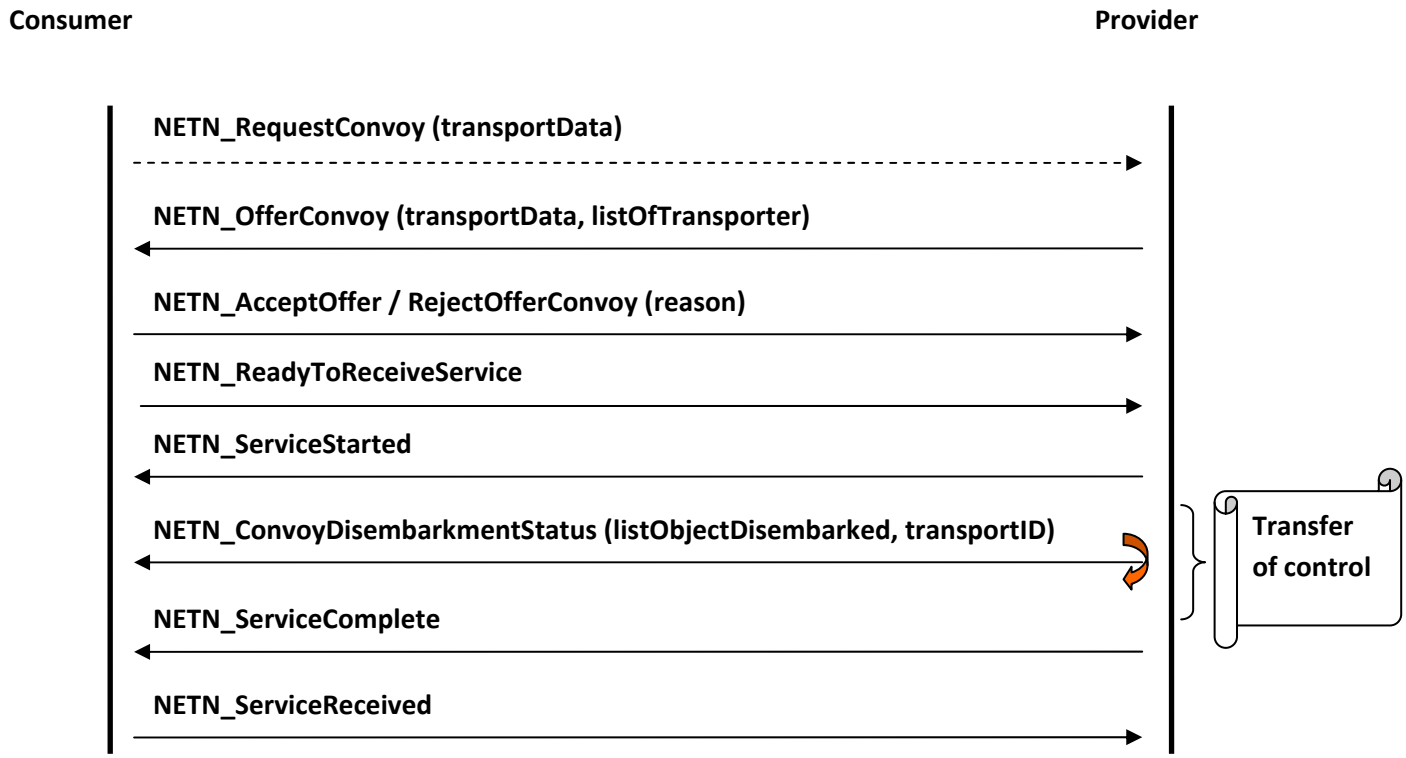
An Embarkment service is considered as closed:

- When service Provider receives a NETN_ServiceReceived interaction and service Consumer receives a NETN_ServiceCompleted interaction.
- Or when a NETN_CancelConvoy or a NETN_RejectOfferConvoy is used by the service Provider or Consumer.

If an Embarkment service is cancelled:

- During a delivery phase (between start and complete): all units already embarked or partially embarked are kept by the service Provider. The status of this units stay as Inactive. A new Request is needed by the service Provider to continue to embark or disembark units (restore).
- During a negotiation phase (before start): transaction between service Consumer and Provider is considered as closed without delivery of service.
- After a service delivery (after complete): a cancel does not make sense in this case (no effect).

8.6.3 Disembarkment service



A Consumer makes a request for disembarkment with the following parameters:

- Time and location where units would like to disembark,
- Characteristics of each simulated entity to manage,

A Provider offers a response to the Consumer with the following parameters:

- Time and location where units could disembark,
- List of simulated entities the Provider is able to manage and transport units planned to be used,

Offered services are accepted only when both service Consumer and Provider are agreeing about the conditions for delivery the service. Provider can change some elements of the request in this offer:

- Time and location where units must be disembarked, if the requirements of the request cannot be satisfied.
- Partial delivery offer by managing fewer units than requested

To achieve Disembarkment service, Consumer must publish the NETN_ReadyToReceiveService interaction.

During a Disembarkment service execution, each transporter enters a loop where it publishes a list of disembarked entities. The responsibility of entities specified in this list is restored to the Consumer when disembarked (see section Transfer of Control).

A **NETN_ConvoyDisembarkmentStatus** interaction can be repeated as much as needed, if embarkment needs to be realized in several iterations.

A Disembarkment service is considered as closed:

- When service Provider receives a NETN_ServiceReceived interaction and service Consumer receives a NETN_ServiceCompleted interaction.
- Or when a NETN_CancelConvoy or a NETN_RejectOfferConvoy is used by the service Provider or Consumer.

If a Disembarkment service is cancelled:

- During a delivery phase (between start and complete): all units already disembarked or partially disembarked are kept by the service Consumer. A new Request is needed by the service Provider to continue to disembark or re-embark units (restore).
- During a negotiation phase (before start): transaction between service Consumer and Provider is considered as closed without delivery of service.
- After a service delivery (after complete): a cancel does not make sense in this case (no effect).

Variations:

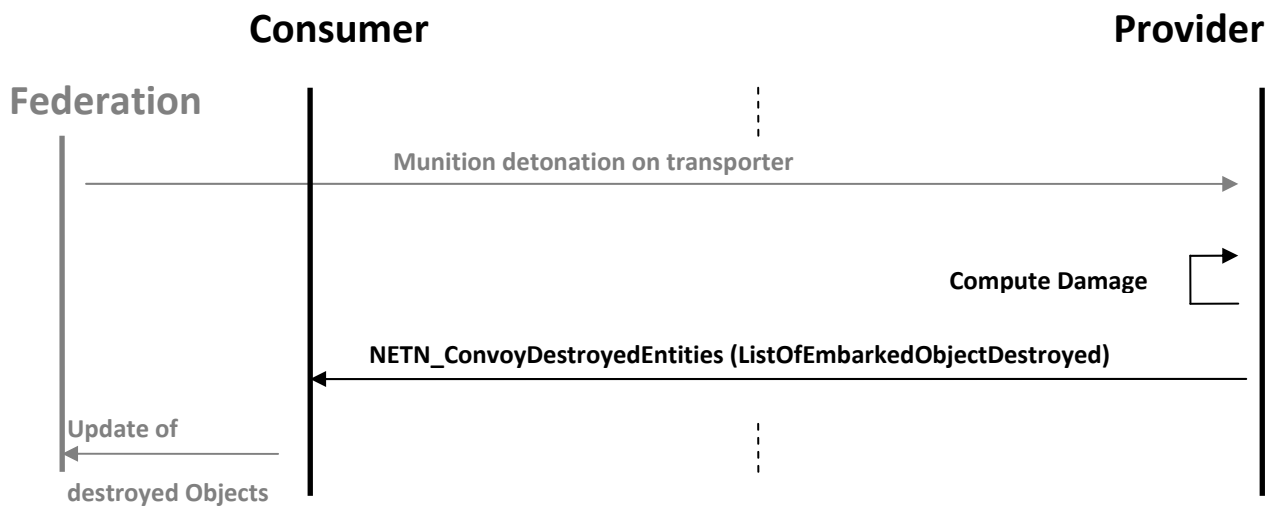
- Disembarkment protocol can start with a Consumer interaction as well as with a Provider one. If a Provider initiates a Disembarkment (planned operation), the protocol execution starts directly at the second step (NETN_OfferConvoy), without processing the query phase (NETN_RequestConvoy).

Exception: see section Scenario Preparation Phase.

In case of "scenario preparation phase", Disembarkment pattern could be affect. The case is detail at the end of this document.

8.6.4 Convoy Services and Attrition

During the execution of a Convoy service transporter may be engaged by other units or otherwise affected by the simulated environment. This attrition can affect the delivery of the service and requires additional interactions to notify service consumers about any lost/destroyed entities.



A Convoy service Provider can use a **NETN_ConvoyDestroyedEntities** interaction to define transported units destroyed during the service execution and inform the Consumer.

For example, if a transporter is destroyed with transported units on board, transported entities are also destroyed. Therefore, whenever a convoy service provider receives interactions (e.g.

MunitionDetonation) attrition on embarked entities is calculated. The Convoy service Provider then sends a list of destroyed objects to the service consumer. The Convoy service consumer can use this list to update its situation or to cancel transaction in progress.

The **NETN_ConvoyDestroyedEntities** interaction can take place at anytime between the start of the service (ServiceStarted interaction) and the end of the service (ServiceComplete interaction).

Impact on the Convoy service Pattern could be the following:

Ex: Vessel « 1 » and « 2 » must be transporters. Some units need to be transported in two rotations on each Vessel. We study the case where Vessel « 1 » is destroyed during its first rotation and Vessel « 2 » is destroyed during its second rotation:

Stage	Interactions
Negotiation phase and start of the service	**…**
First rotation:Vessel 1 and Vessel 2 embark units	**Provider send:** **NETN_ConvoyEmbarkmentStatus (list1, Vessel1)** **NETN_ConvoyEmbarkmentStatus (list2, Vessel2)**
During transport, service Provider received MunitionDetonation Vessel 1 is destroyed	**Provider send:** **NETN_ConvoyDestroyedEntities (list1)**
Vessel 2 disembark his units	**Provider send:** **NETN_ConvoyDisembarkmentStatus (list2, Vessel2)**
Second rotation: Vessel 2 embark units	**Provider send:** **NETN_ConvoyEmbarkmentStatus (list3, Vessel2)**
During transport, service Provider received MunitionDetonation Vessel 2 is destroyed	**Provider send:** **NETN_ConvoyDestroyedEntities (list3)**
End of service or Cancel	

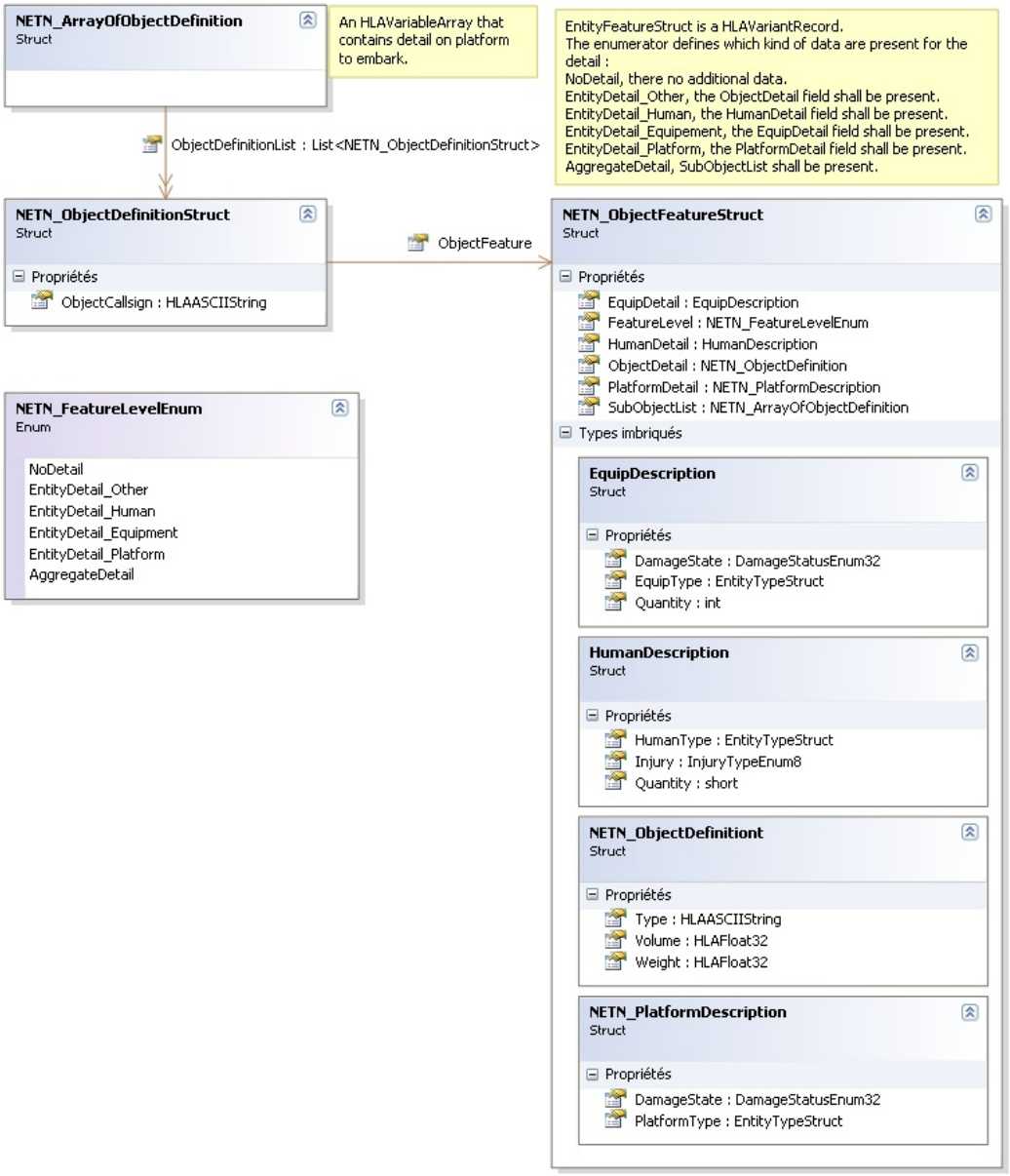
We notice that a **NETN_ConvoyEmbarkmentStatus** interaction does not necessarily fit a **NETN_ConvoyDisembarkmentStatus** interaction.

8.6.5 Convoy datatypes

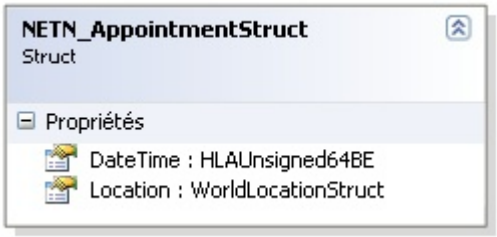
To be able to manage different levels of granularity between simulators, a definition of the object to convoy is encapsulated in a temporary structure.

To make a Convoy, federate shall build a list of elements and exchange it to negotiate the service.

The figure below shows the content of such a list and how elements for Convoy service are defined towards **NETN_ArrayOfObjectDefinition**.



Interactions use **NETN_AppointmentStruct** to define date, time and location of desired Convoy service. Both Consumer and Provider must agree on these data to accept the service.



8.6.6 Scenario preparation phase

A scenario preparation phase entails to share information (online or offline) between simulations. Therefore, a strategy is needed before running an experimentation to share data (scenario, units types (EntityType), callsign, etc.).

Concerning Disembarkment service, a scenario can start to run with already embarked units. The embarkment phase is supposed to have taken place during the scenario preparation phase, so simulations do not have to play it using interactions.

To allow simulations to play such cases, we propose the following solution:

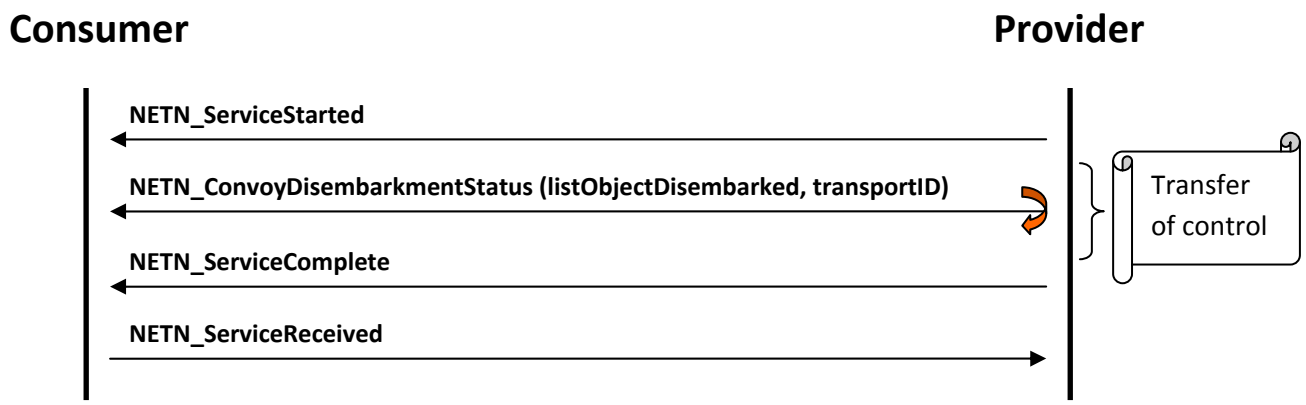
- For Embarkment, in preparation phase of a simulation, the EmbeddedUnitList allows to describe on board elements. This list contains items (like callsigns) provided by simulations that want to see them embarked at scenario start.
- The synchronization process for two simulations in a scenario preparation phase (offline) is the following:
    - Simulation A (Consumer) provides information (offline) to simulation B (Provider) on elements that must be on board at the beginning of scenario execution (Callsigns, scenario elements),
    - Simulation B initializes its scenario with these elements,
    - During scenario execution, Simulation B publishes information on embarked elements using « EmbeddedUnitList »,
    - During the first HLA objects discover, Simulation A analyzes « EmbeddedUnitList » attributes, looking for its own potentially embarked elements,
    - Simulation A updates its embarked elements (inactivation),
    - Simulations A & B use interactions as defined in Disembarkment protocol.
- This procedure allows limiting synchronization and data capture errors between simulations during preparation phase.

8.6.7 Convoy Pattern Exception

Two ways are possible in order to disembark units:

- A service Provider waits for a request service for disembarkment of embarked units (see standard Disembarkment protocol).
- A service Provider sends directly a NETN_ServiceStarted to the Consumer (see schema below).

This second way bypasses the regular negotiation phase (see standard Disembarkment protocol). It could be employed to disembark already embarked units at scenario beginning.



8.7 RPR-FOM Platform Objects Class Extensions

Platforms are a central element in the Logistics FOM through which units can be transported. In the Logistics FOM Module the RPR-FOM platform object classes are subclasses to add additional information used in the Logistics pattern described in this document.

We chose to add attributes to «RPR Platform object» children to define Current state of Embarked elements on transporter platforms. Indeed, interactions have a limited lifetime and do not allow to update a state at any time on the HLA federation.

EmbeddedUnitList provides the list of elements carried by a transporter. Listed items are composed of « Callsigns » of on-board units (aggregates or platforms). The list is updated by the transporter according to carried elements.

8.8 Facility object class

Facilities are a central element in the Logistics FOM through which material can be transferred.

Facilities could be created during a simulation or be a part of the infrastructure (railway station, storage tanks depot, port, etc.). A facility could also be part of a unit (ship, etc.).

The RPR-FOM v2.0 BaseEntity object class is extended with a subclass **NETN_Facility**, with the following attributes (inherited parameters are written in italics):

Full Name	HLAobjectRoot.BaseEntity.NETN_Facility

Attribute Name	Datatype	Default Value	Definition	Usage
EntityIdentifier	*EntityIdentifierStruct*	*Not Optional*	*Identifies the site, application, and entity number of this object instance. It is used for group addressing in the SIMAN interactions.*	*Required*
EntityType	*EntityTypeStruct*	*Not Optional*	*Kind, Country, Domain, Category, Subcategory, Specific, and Extra fields of the DIS Entity Type.*	*Required*
IsPartOf	*IsPartOfStruct*	*All zeros*	*Used to indicate that there is a spatial relationship between this entity and a host entity, i.e., one entity is "part of" another*	*Optional for NETN_Facility*
RelativeSpatial	SpatialStruct	*All zeros*	*Used to express the spatial relationship between the entity and a host entity. Used in addition to the normal spatial attribute which describes absolute location.*	*Optional for NETN_Facility*
Spatial	*SpatialStruct*	*Not Optional*	*Used to express the spatial relationship between the entity and the center of the Earth.*	*Required*
DamageState	DamageStatusEnum32	No Damage	Damage State of Facility	Required for NETN_Facility

Attribute Name	Datatype	Default Value	Definition	Usage
ForceIdentifier	ForceIdentifierEnum8	Other	Enumeration distinguishing the different teams or sides in an exercise.	Required for NETN_Facility
Callsign	NETN_Callsign	Not Optional	Callsign of the facility.	Required
UniqueID	NETN_UniqueID		Unique id of the facility instance.	
IsOperational	HLAboolean	HLAtrue	True if the facility is operational and can provide service	
StorageList	SupplyArray	Not Optional	List of Materials (Amount, Type) stored in the facility (no platforms). Material loaded on means or on resources of transport which are located in the facility are also included. Material belonging to an object of the object list is excluded.	Required
PlatformList	UniqueIdArrayStruct	Not Optional	List of non-active platforms/units in the facility. Includes those platforms transferred to the facility with a ServiceRequest (Type = Deposit) or located by the provider-model in the facility.	Required

The specific type of facility (Service Type) is defined as part of the EntityType attribute. The following enumeration scheme is to be used for the following service types.

Type of Facility	Kind.Dom.Cou.Cat.Sub.Spec.Ex (EntityType)
Storage (Material)	TBD
Maintenance	TBD
Medical	TBD
FARP	TBD
TBD Base	TBD
NBC	TBD
Camp	TBD

Enumerations to be definied in federation specific agreements

8.9 Interaction Classes

The NETN Logistics FOM module can be used to represent services for Supply, Storage, Repair, Transport, Embarkment and Disembarkment.

The basic pattern used by all NETN services is the Service Consumer-Provider Pattern which defines negotiation and delivery of services.

Logistics services use interactions from this Service Consumer-Provider Pattern and extend them (if needed) to support specific types of logistic services.

The following Logistics interactions class extensions to the Service Consumer-Provider pattern are used in the NETN Logistics FOM Module:

Class 1	Class 2	Class 3
NETN_Service	*NETN_RequestService*	**NETN_RequestSupply**
		NETN_RequestStorage
		NETN_RequestRepair
		NETN_RequestConvoy
	NETN_OfferService	**NETN_OfferSupply**
		NETN_OfferRepair
		NETN_OfferStorage
		NETN_OfferConvoy
	NETN_AcceptOffer	
	NETN_RejectOffer	**NETN_RejectOfferConvoy**
	NETN_ReadyToReceiveService	**NETN_ReadyToReceiveSupply**
	NETN_ServiceStarted	**NETN_StorageStarted**
	NETN_ServiceComplete	**NETN_SupplyComplete**
	NETN_ServiceReceived	
	NETN_CancelService	**NETN_CancelConvoy**
	NETN_ConvoyEmbarkmentStatus	
	NETN_ConvoyDisembarkmentStatus	
	NETN_ConvoyDestroyedEntities	

8.9.1 NETN_RequestConvoy

A request for Transport, Embarkment or Disembarkment of a platform is initiated by a **NETN_RequestConvoy** interaction with the following parameters (inherited parameters written in italics):

Full Name	HLAinteractionRoot.NETN_Service.NETN_RequestService.NETN_RequestConvoy		
Parameter Name	Datatype	Default Value (if optional)	Definition
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Not optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Convoy = 3*
RequestTimeOut	*HLAinteger64BE*	*Optional*	*Defined a deadline (date) for the provider response. Number of second since 01/01/1970*
TransportData	NETN_TransportStruct	Elements to convoy	Defined the type of service which will be done; transport, embarkment or disembarkment. The elements and

Parameter Name	Datatype	Default Value (if optional)	Definition
			constraint defined is function of the service to achieve

8.9.2 NETN_OfferConvoy

A **NETN_OfferConvoy** interaction shall be sent by the service providing federate in response to a **NETN_RequestConvoy** interaction (inherited parameters written in italics):

Full Name	HLAinteractionRoot.NETN_Service.NETN_OfferService.NETN_OfferConvoy		
Parameter Name	Datatype	Default Value (if optional)	Definition
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Not optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Convoy = 3*
IsOffering	*HLAboolean*	*Not Optional*	*Defines if the requested service is offered (=true) or not (=false)*
RequestTimeOut	*HLAinteger64BE*	*Optional*	*Defined a deadline (date) for the provider response. Number of second since 01/01/1970*
OfferType	NETN_OfferTypeEnum32	Not Optional	Provide high level information about the acceptance of the request (Partial, Full) without need to compare information from the initial request and the offer.
TransportData	NETN_TransportStruct	Elements to convoy offered if *isOffering = true* Could be contained less elements than the request	Defined the service which will be done for transport, embarkment or disembarkment.
ListOfTransporters	NETN_ArrayOfObjectDefinition	Optional (Default: empty list)	Informative - Platform list of transporters keep for the service (Callsign)

If needed, the TransportData parameter is allowed to be different from the corresponding information in the NETN_RequestConvoy depending on the constraint of service request. The list of elements offered by the service provider can therefore be either complete or partial (subset) of the elements requested. Constraints (time and location) can also be different from the request. If a Provider agrees with a TransportData request, it repeats the data provided by the Consumer without modification.

8.9.3 NETN_CancelConvoy

A **NETN_CancelConvoy** interaction shall be sent by the service providing or the service Consumer federate to cancel the service transaction (inherited parameters written in italics):

Full Name	HLAinteractionRoot.NETN_Service.NETN_CancelService.NETN_CancelConvoy		
Parameter Name	Datatype	Default Value (if optional)	Definition

Parameter Name	Datatype	Default Value (if optional)	Definition
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Not optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Convoy = 3*
Reason	HLAASCIIstring	Optional (Default: empty String)	Allows to inform about the reason of the cancel (free text)

8.9.4 NETN_ RejectOfferConvoy

A **NETN_RejectOfferConvoy** interaction shall be sent by a service Consumer federate to refuse a **NETN_OfferConvoy** interaction (inherited parameters written in italics):

Full Name	HLAinteractionRoot.NETN_Service.NETN_RejectOffer.NETN_RejectOfferConvoy

Parameter Name	Datatype	Default Value (if optional)	Definition
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Not optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Convoy = 3*
Reason	HLAASCIIstring	Optional (Default: empty String)	Allows to inform about the reason of the reject (free text)

8.9.5 NETN_ ConvoyEmbarkmentStatus

A **NETN_ConvoyEmbarkmentStatus** interaction shall be sent by a service Provider federate to inform a service Consumer of the embarkment state, after a **NETN_ServiceStarted** interaction (inherited parameters written in italics):

Full Name	HLAinteractionRoot.NETN_Service.NETN_ConvoyEmbarkmentStatus

Parameter Name	Datatype	Default Value (if optional)	Definition
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Not optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Convoy = 3*
ListOfObjectEmbarked	NETN_ArrayOfObject Definition	List of element embarked by the provider	Allows to follow the elements managed by the provider
TransportUnitIdentifier	NETN_Callsign	Identifier of transporter	Callsign of transporter

8.9.6 NETN_ ConvoyDisembarkmentStatus

A **NETN_ConvoyDisembarkmentStatus** interaction shall be sent by a service Provider federate to inform a service Consumer of the disembarkment state, after a **NETN_ServiceStarted** interaction (inherited parameters written in italics):

Full Name	HLAinteractionRoot.NETN_Service.NETN_ConvoyDisembarkmentStatus

Parameter Name	Datatype	Default Value (if optional)	Definition
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Not optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Convoy = 3*
ListOfObjectDisembarked	NETN_ArrayOfObject Definition	List of element disembarked by the provider	Allows to follow the elements managed by the provider
TransportUnitIdentifier	NETN_Callsign	Identifier of transporter	Callsign of transporter

8.9.7 NETN_ConvoyDestroyedEntities

A **NETN_ConvoyDestroyedEntities** interaction is used by a service Provider federate to give information to the Consumer on managed elements. This interaction is only used if the Provider simulates destruction of managed elements (inherited parameters written in italics):

Full Name	HLAinteractionRoot.NETN_Service.NETN_ConvoyDestroyedEntities

Parameter Name	Datatype	Default Value (if optional)	Definition
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Not optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Convoy = 3*
ListOfEmbarkedObje ctDestroyed	NETN_ArrayOfObjectDe finition	List of element destroyed during the convoy	Allows to follow the elements managed by the provider

8.9.8 NETN_RequestSupply

A request for supply is initiated by a **NETN_RequestSupply** interaction with the following parameters (inherited parameters written in italics):

Full Name	HLAinteractionRoot.NETN_Service.NETN_RequestService.NETN_RequestSupply

Parameter Name	Datatype	Default Value (if optional)	Definition
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Supply = 1*
RequestTimeOut	*HLAinteger64BE*	*Optional*	*Specifies an absolute deadline (date/time) for the provider response. Number of second since 01/01/1970*
SuppliesData	SupplyArray		List of type and quantity of supplies requested.
LoadingDoneByPro	HLABoolean	Optional	Determines whether the service

Parameter Name	Datatype	Default Value (if optional)	Definition
vider		(Default = true)	provider (LoadingDoneByProvider = true) or the service consumer (LoadingDoneByProvider = false) takes care of the (un)loading of the material

Note that if the time specified in the RequestTimeOut passes without the Provider sending an NETN_OfferSupply, then the Consumer will send a NETN_CancelService.

Note that a Consumer can ask Supply to multiple Providers by leaving the Provider Callsign empty.

8.9.9 NETN_OfferSupply

In response to a **NETN_RequestSupply** interaction, a federate simulating the service providing entity shall send a **NETN_OfferSupply** interaction, with following content (inherited parameters written in italics):

Full Name	HLAinteractionRoot.NETN_Service.NETN_OfferService.NETN_OfferSupply

Parameter Name	Datatype	Default Value (if optional)	Definition
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Not optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Supply = 1*
IsOffering	*HLAboolean*	*(Not Optional)*	*Defines if the requested service is offered (=true) or not (=false)*
RequestTimeOut	*HLAinteger64BE*	*Optional*	*Specifies an absolute deadline (date/time) for the provider response. Number of second since 01/01/1970*
SuppliesData	SupplyArray	All offered supplies if isOffering = true otherwise Undefined	List of type and quantity of supplies offered. May be different from the list of requested supplies.
LoadingDoneByProvider	HLABoolean	Optional (Default = true)	Determines whether the service provider (LoadingDoneByProvider = true) or the service consumer (LoadingDoneByProvider = false) takes care of the (un)loading of the material

The following agreements pertain to the NETN_OfferSupply interaction:

- The NETN_OfferSupply.SuppliesData must include an amount less than or equal to the requested amount.
- The provider will reserve the offered amount of the supplies when the NETN_OfferSupply is sent and not "un-reserve" them unless the consumer sends a NETN_RejectOffer or either federate sends a NETN_CancelService.
- If the resupply involves an aerial refuelling, and if the Provider sends a LoadingDoneByProvider = false value, the (provider) tanker aircraft must stay in its existing orbit until either the refuelling is complete or the consumer sends a NETN_RejectOffer or either federate sends a NETN_CancelService

8.9.10 NETN_AcceptOffer/NETN_RejectOffer

In response to a timely (i.e. one sent prior to the RequestTimeOut date/time) **NETN_OfferSupply** interaction, a federate simulating the service consumer entity shall respond in one of three ways:

- If the provider responds with the NETN_OfferService parameter IsOffering = "false, and if the request was not a multicast request, " the pattern terminates and the consumer must seek a new provider.
- If the values in SuppliesData and LoadingDoneByProvider are acceptable, the consumer will respond with a NETN_AcceptOffer.
- If the values in SuppliesData or the LoadingDoneByProvider are unacceptable, the consumer will respond with a NETN_RejectOffer.

8.9.11 NETN_ReadyToReceiveSupply

Subsequent to sending a NETN_AcceptOffer, the consumer will, when ready to receive transfer of supplies, initiate a **NETN_ReadyToReceiveSupply** interaction.

Full Name	HLAinteractionRoot.NETN_Service.NETN_ReadyToReceiveService.NETN_ReadyToReceiveSupply		
Parameter Name	**Datatype**	**Default Value (if optional)**	**Definition**
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Not optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Supply = 1*
IsOffering	*HLAboolean*	*(Not Optional)*	*Defines if the requested service is offered (=true) or not (=false)*
RequestTimeOut	*HLAinteger64BE*	*Optional*	*Specifies an absolute deadline (date/time) for the provider response. Number of second since 01/01/1970*
SuppliesData	SupplyArray		List of type and quantity of supplies desired. May be smaller or less than the list of requested supplies.

Note that the number(s) or amount(s) of supplies specified by the consumer in NETN_ReadyToReceiveSupply.SuppliesData must be less than or equal to the NETN_OfferSupply number(s) or amount(s).

8.9.12 NETN_SupplyComplete

Subsequent to sending a **NETN_ServiceStarted** interaction, and when a federate simulating the service providing entity has finished transferring the agreed upon material, the provider shall send a **NETN_SupplyComplete** interaction, with following content (inherited parameters written in italics):

Full Name	HLAinteractionRoot.NETN_Service.NETN_ServiceComplete.NETN_SupplyComplete		
Parameter Name	**Datatype**	**Default Value (if optional)**	**Definition**
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Not optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Supply = 1*
SuppliesData	SupplyArray	Not optional	List of type and quantity of supplies actually transferred.

One of two events may occur to cause the amount or quantity of supplies actually delivered to be less or fewer than the amount of supplies specified in the **NETN_ReadyToReceiveSupply.SuppliesData**:

1. The Provider object may die or be destroyed. If this occurs, the federate simulating the provider object will initiate the NETN_SupplyComplete interaction with the SuppliesData comprised of supplies transferred prior to the loss of the Provider object.

2. If the Consumer object dies or is destroyed, the federate simulating the consumer object will initiate a NETN_CancelService, thereby terminating the transfer of supplies. Note that either federate may also send a NETN_CancelService for reasons unrelated to the loss of a consumer or provider object. Regardless of the reason, if a NETN_CancelService is sent, the provider will initiate the NETN_SupplyComplete interaction with the SuppliesData comprised of supplies transferred prior to the NETN_CancelService.

8.9.13 NETN_RequestStorage

A request for storing supplies is initiated by a **NETN_RequestStorage** interaction with the following parameters (inherited parameters written in italics):

Full Name	HLAinteractionRoot.NETN_Service.NETN_RequestService.NETN_RequestStorage		
Parameter Name	Datatype	Default Value (if optional)	Definition
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Not optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Storage = 4*
RequestTimeOut	*HLAinteger64BE*	*Optional*	*Defined a deadline (date) for the provider response. Number of second since 01/01/1970*
SuppliesData	SupplyArray	Not optional	List of type and quantity of supplies requested to be stored.
LoadingDoneByProvider	HLABoolean	Optional (Default = true)	Determines whether the service provider (LoadingDoneByProvider = true) or the service consumer (LoadingDoneByProvider = false) takes care of the (un)loading of the material

8.9.14 NETN_OfferStorage

In response to a NETN_RequestStorage interaction, a service providing federate shall send a NETN_OfferStorage interaction, with following content (inherited parameters written in italics):

Full Name	HLAinteractionRoot.NETN_Service.NETN_OfferService.NETN_OfferStorage		
Parameter Name	Datatype	Default Value (if optional)	Definition
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Not optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Storage = 4*
IsOffering	*HLAboolean*	*(Not Optional)*	*Defines if the requested service is offered (=true) or not (=false)*
RequestTimeOut	*HLAinteger64BE*	*Optional*	*Defined a deadline (date) for the provider response. Number of second*

Parameter Name	Datatype	Default Value (if optional)	Definition
			since 01/01/1970
SuppliesData	SupplyArray	All supplies offered to be stored if isOffering = true otherwise Undefined	List of type and quantity of supplies offered to be stored. May be different from the list of supplies requested to be stored.
LoadingDoneByProvider	HLABoolean	Optional (Default = true)	Determines whether the service provider (LoadingDoneByProvider = true) or the service consumer (LoadingDoneByProvider = false) takes care of the (un)loading of the material

8.9.15 NETN_StorageStarted

In response to a **NETN_OfferReceived** interaction, a federate simulating the service providing entity shall send a **NETN_StorageStarted** interaction, with following content (inherited parameters written in italics):

Full Name	HLAinteractionRoot.NETN_Service.NETN_ServiceStarted.NETN_StorageStarted		
Parameter Name	Datatype	Default Value (if optional)	Definition
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Not optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Storage = 4*
SuppliesData	SupplyArray	Not optional	List of type and quantity of supplies to be transferred. May be different from the list of supplies offered to be stored.

8.9.16 NETN_RequestRepair

The request for repair of a platform is initiated by a **NETN_RequestRepair** interaction with the following parameters (inherited parameters written in italics):

Full Name	HLAinteractionRoot.NETN_Service.NETN_RequestService.NETN_RequestRepair		
Parameter Name	Datatype	Default Value (if optional)	Definition
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Not optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Repair = 2*
RequestTimeOut	*HLAinteger64BE*	*Optional*	*Defined a deadline (date) for the provider response. Number of second since 01/01/1970*
RepairData	RepairList	Not optional	List of all requested repairs.

8.9.17 NETN_OfferRepair

In response to a **NETN_RequestRepair** interaction, a federate simulating the service providing entity shall send a **NETN_OfferRepair** interaction, with following content (inherited parameters written in italics):

Full Name	HLAinteractionRoot.NETN_Service.NETN_OfferService.NETN_OfferRepair		
Parameter Name	Datatype	Default Value (if optional)	Definition
ServiceID	*NETN_EventIdentifier*	*Not optional*	*Unique identifier for a service*
Consumer	*NETN_Callsign*	*Not optional*	*Entity that has requested the service*
Provider	*NETN_Callsign*	*Not optional*	*Providing or intended provider entity*
ServiceType	*ServiceTypeEnum8*	*Not optional*	*Type of requested service. Repair = 2*
IsOffering	*HLAboolean*	*(Not Optional)*	*Defines if the requested service is offered (=true) or not (=false)*
RequestTimeOut	*HLAinteger64BE*	*Optional*	*Defined a deadline (date) for the provider response. Number of second since 01/01/1970*
RepairData	RepairList	List of all offered repairs if isOffering = true otherwise Undefined	List of the type of repairs offered. May be different from the list of requested repairs.

8.10 Fixed Record Datatypes

8.10.1 NETN_ObjectDefinitiontStruct

Name	NETN_ObjectDefinitiontStruct
Encoding	HLAfixedRecord
Definition	

Parameter Name	Datatype	Default Value	Definition
ObjectCallsign	NETN_Callsign	No	Callsign of the object (entity or aggregate) to embark
ObjectUniqueID	NETN_UniqueID	Optional	Optional ID usable to reference a published entity. This attributes must be specified if an HLA instance exist.
ObjectFeature	NETN_ObjectFeatureStruct	No	Detail of the object to embark

8.10.2 NETN_ObjectDescription

Name	NETN_ObjectDescription
Encoding	HLAfixedRecord
Definition	

Parameter Name	Datatype	Default Value	Definition
Type	HLAASCIIString	“Unknown”	
Weight	NETN_Float32BE	No	Weight of object to embark (in kg).
Volume	NETN_Float32BE	No	Volume of object to embark (in m3)

8.10.3 NETN_HumanDescription

Name	NETN_HumanDescription
Encoding	HLAfixedRecord
Definition	

Parameter Name	Datatype	Default Value	Definition
HumanType	EntityTypeStruct		The type of human defined by federate requesting the service
Quantity	NETN_Integer16BE		The number of person of the human

			type.
Injury	InjuryTypeEnum8		Degree of injury

8.10.4 NETN_EquipDescription

Name	NETN_EquipDescription
Encoding	HLAfixedRecord
Definition	

Parameter Name	Datatype	Default Value	Definition
EquipType	EntityTypeStruct		The type of equipment
Quantity	NETN_Integer32BE		The number of units of the Equip type.
DamageState	DamageStatusEnum32		Degree of damage

8.10.5 NETN_PlatformDescription

Name	NETN_PlatformDescription
Encoding	HLAfixedRecord
Definition	

Parameter Name	Datatype	Default Value	Definition
PlatformType	EntityTypeStruct		The type of equipment
DamageState	DamageStatusEnum32		Degree of damage

8.10.6 NETN_DataTStruct

Name	NETN_DataTStruct
Encoding	HLAfixedRecord
Definition	

Parameter Name	Datatype	Default Value	Definition
ObjectToManaged	NETN_ArrayOfObjectDefinition		
Appointment	NETN_AppointmentStruct		
FinalAppointment	NETN_AppointmentStruct		

8.10.7 NETN_DataEDStruct

Name	NETN_DataEDStruct
Encoding	HLAfixedRecord
Definition	

Parameter Name	Datatype	Default Value	Definition
ObjectToManaged	NETN_ArrayOfObjectDefinition		
Appointment	NETN_AppointmentStruct		

8.10.8 NETN_AppointmentStruct

Name	NETN_AppointmentStruct
Encoding	HLAfixedRecord
Definition	

Parameter Name	Datatype	Default Value	**Definition**

Parameter Name	Datatype	Default Value	Definition
DateTime	HLAinteger64BE	No	Number of second since 1 january1970
Location	WorldLocationStruct	No	Location

8.10.9 RepairStruct

Name	RepairStruct
Encoding	HLAfixedRecord
Definition	Repairs associated with a specific material

Parameter Name	Datatype	Default Value	Definition
MaterialID	NETN_UniqueID		HLA object id for the material.
Repairs	RepairTypeList		List of the types of repair associated with the material.

8.10.10 SupplyStruct

Name	*SupplyStruct*
Encoding	HLAfixedRecord
Definition	

Parameter Name	Datatype	Default Value	Definition
SupplyType	EntityTypeStruct		The type of supply
Quantity	NETN_Float32BE		The number of units of the supply type. The unit measure depends on the supply type and shall use the SI units of measurement used for such supplies.

8.11 Array Datatypes

Name	Datatype	Cardinality	Encoding	Definition
RepairList	RepairStruct	Dynamic	HLAvariableArray	List of repair descriptions (equipment and type of repairs).
RepairTypeList	RepairTypeEnum16	Dynamic	HLAvariableArray	List of repair types.
SupplyArray	SupplyStruct	Dynamic	HLAvariableArray	
NETN_ArrayOfObjectDefinition	NETN_ObjectDefinitionStruct	Dynamic	HLAvariableArray	

8.12 Enumerated Datatypes

8.12.1 ServiceTypeEnum8 datatype

Name	ServiceTypeEnum8
Representation	HLAoctet
Definition	-

Enumerate	Value	Definition
Other	0	
Resupply	1	
Repair	2	
Storage	3	
Convoy	4	

8.12.2 NETN_ConvoyTypeEnum32

Name	NETN_ConvoyTypeEnum32
Representation	HLAinteger32Be
Definition	-

Enumerate	value	Definition
Transport	0	For a transport service
Embarkment	1	For a Embarkment service
Disembarkment	2	For a Disembarkment service

8.12.3 NETN_OfferTypeEnum32

Name	NETN_OfferTypeEnum32
Representation	HLAinteger32BE
Definition	-

Enumerate	Value	Definition
RequestAcknowledgeWithRestriction	0	Partially Compliant in accordance of the request
RequestAcknowledgePositive	1	Full compliant
RequestAcknowledgeNegative	2	Not compliant

8.12.4 NETN_FeatureLevelEnum32

Name	NETN_FeatureLevelEnum32
Representation	HLAinteger32BE
Definition	-

Enumerate	value	Definition
NoDetail	0	
EntityDetail_Other	1	
EntityDetail_Human	2	
EntityDetail_Equipment	3	
EntityDetail_Platform	4	
AggregateDetail	5	

8.13 Variant Record Datatypes

8.13.1 NETN_TransportStruct

Name	NETN_TransportStruct
Discriminant Name	ConvoyType
Discriminant Type	NETN_ConvoyTypeEnum32
Encoding	HLAvariantRecord
Definition	

Discriminant Enum	Parameter	Datatype	Default Value	Definition
Transport	dataTransport	NETN_DataTStruct		Present only if ConvoyType = Transport
Embarkment	dataEmbarkment	NETN_DataEDStruct		Present only if ConvoyType = Embarkment
Disembarkment	dataDebarkment	NETN_DataEDStruct		Present only if ConvoyType = Disembarkment

8.13.2 NETN_ObjectFeatureStruct

Name	NETN_ObjectFeatureStruct
Discriminant Name	FeatureLevel
Discriminant Type	NETN_FeatureLevelEnum32
Encoding	HLAvariantRecord
Definition	

Discriminant Enum	Parameter	Datatype	Default Value	Definition
AggregateDetail	SubObjectList	NETN_ArrayOfObjectDefinition	No	Present only if FeatureDescriptionLevel equals “AggregateDetail”
EntityDetail_Other	ObjectDetail	NETN_ObjectDescription	No	Present only if FeatureDescriptionLevel equals “EntityDetail_Other”
EntityDetail_Human	HumanDetail	NETN_HumanDescription	No	Present only if FeatureDescriptionLevel equals “EntityDetail_Human”
EntityDetail_Equipment	EquipDetail	NETN_EquipDescription	No	Present only if FeatureDescriptionLevel equals “EntityDetail_Equipment”
EntityDetail_Platform	PlatformDetail	NETN_PlatformDescription	No	Present only if FeatureDescriptionLevel equals “EntityDetail_Platform”

9 NETN Aggregate Unit FOM Module v1.0

9.1 Introduction

The NETN Aggregated Unit Representation FOM Module documents extensions to the RPR-FOM v2.0 D17 FOM Module necessary for representation and use of aggregate units in the NETN. The chapter starts by documenting NETN Object Classes as extensions to RPR-FOM v2.0 classes and federation agreements regarding the use of the object classes.

It then discusses methods of adjudicating combat between objects owned by different federates and concludes by identifying necessary extensions to interactions.

9.2 NETN Aggregated Unit Object Classes

9.2.1 NETN_Aggregate

The NETN_Aggregate object class is a subclass of the RPR FOM class BaseEntity.AggregateEntity. The table below describes the attributes, data types, and semantics for the NETN_Aggregate class. Inherited attributes are shown in italics.

Attribute (Optional)	Datatype	Default Value (If optional)	Definition	Usage
EntityIdentifier	*EntityIdentifierStruct*	*(Not Optional)*	Identifies the site, application, and entity number of this object instance. It is used for group addressing *in the SIMAN interactions.*	Required
EntityType	*EntityTypeStruct*	*(Not Optional)*	*Kind, Country, Domain, Category, Subcategory, Specific, and Extra fields of the DIS Entity Type.*	Required
IsPartOf	*IsPartOfStruct*	*All zeros*	Used to indicate that there is a spatial relationship between this entity and a host entity, i.e., one entity is "part of" another	
RelativeSpatial	*SpatialStruct*	*All zeros*	Used to express the spatial relationship between the entity and a host entity. Used in addition to the normal spatial attribute which describes absolute location.	
Spatial	*SpatialStruct*	*(Not Optional)*	Used to express the spatial relationship between the entity and the center of the Earth.	Required
AggregateMarking	AggregateMarkingStruct	*All zeros*	*A unique marking or combination of characters used to distinguish the aggregate from other aggregates.*	
AggregateState	*AggregateStateEnum8*	*(Not Optional)*	*An indicator of the extent of association of objects form an operating group.*	Required (see FA1)
Dimensions	*DimensionStruct*	*(Not Optional)*	*The size of the area covered by the units in the aggregate.*	Required
EntityIdentifiers	*RTIObjectIdArrayStruct*	*(Not Optional)*	*The identification of entities that are contained within the aggregate.*	Required (see FA2)
ForceIdentifier	*ForceIdentifierEnum8*	*(Not Optional)*	*The identification of the force*	*Required*

Attribute (Optional)	Datatype	Default Value (If optional)	Definition	Usage
			that the aggregate belongs to.	
Formation	*FormationEnum32*	*(Not Optional)*	*The category of positional arrangement of the entities within the aggregate.*	*Required (see FA2)*
NumberOfSilentEntities	*short*	*(Not Optional)*	*The number of elements in the SilentEntities list*	*Required (see FA3)*
NumberOfVariableDatums	*unsigned long*	*0*	*The number of records in the Variable Datum structure*	*Not currently used*
SilentAggregates	*SilentAggregateStruct*	*(Not Optional)*	*The numbers and types, of silent aggregates contained in the aggregate. Silent aggregates are sub-aggregates that are in the aggregate, but that are not separately represented in the virtual world.*	*Required (see FA2)*
SilentEntities	*SilentEntityStruct*	*(Not Optional)*	*The numbers and types, of silent entities in the aggregate. Silent entities are entities that are in the aggregate, but that are not separately represented in the virtual world.*	*Required (see FA2)*
SubAggregateIdentifier	*RTIObjectIdArrayStruct*	*(Not Optional)*	*The identifications of aggregates represented in the virtual world that are contained in the aggregate.*	*Required (see FA2)*
VariableDatums	*VariableDatumStruct*		*Extra data that describes the aggregate.*	*Not currently used*
Activity	AggregateMissionEnum	*(Not Optional)*	The current activity of the aggregate. This may differ from the mission due to casualties, readiness, etc.	Required
Callsign	HLAunicodeString	(Not Optional)	The name of the object.	Required
CaptureStatus	CaptureStatusEnum8	0	The status of an aggregate with respect to its control or influence over its own activities.	
CombatValue	NETN_Float64BE	100	A summary value (in percent) of unit effectiveness based on level of training, leadership, moral, personnel and equipment operational status, etc.	
CoverStatus	CoverStatusStruct	0	Describes the unit's protection from the effects of weapons fire.	
Echelon	EchelonEnum8	(Not Optional)	The level of command of the aggregate	Required
ElectronicSignature	ElectronicSignatureStruct		Describes the aggregate's susceptibility to electronic detection both as a summary value and by identifying aggregate sensors together with	

Attribute (Optional)	Datatype	Default Value (If optional)	Definition	Usage
			their operational status	
EmbeddedUnitList	ArrayOfEmbeddedUnit		The list of objects carried by this aggregate.	
EntityList	EntityListVariableLength	(Not Optional)	Provides data on one or more entities comprising the aggregate. This includes the initial list of all entities and subsequent updates as entities on the list experience change. This attribute is optional outside the entity Interest Area (IA), but mandatory inside the IA.	Required within IAs (see FA 2)
Footprint	WorldLocationStructArray3		The region occupied by the aggregate. The region is defined as that bounded by line segments connecting the listed world locations	
HigherHeadquarters	NETN_UniqueID	(Not Optional)	A pointer to the aggregate's superior unit or headquarters. The highest level unit or headquarters on each side will publish its own UniqueID as its HigherHeadquarters value.	Required
HUMINTSignature	HUMINTSignatureStruct		Describes the unit's susceptibility to human intelligence (HUMINT), i.e. "information collected and provided by human sources."	
Mission	MissionStruct	(Not Optional)	The operational task the aggregate has been ordered to perform, the time the mission was assigned, and the estimated completion time.	Required
Mounted	NETN_Float64BE	(Not Optional)	The percentage of aggregate personnel traveling on or in their organic transport.	Required
SourceUnit	HLAunicodeString	0	Aggregate from which this aggregate was spawned.	
Status	ActiveStatusEnum8	1	An inactive object should not be shown on C4I systems and cannot move or interact with other objects.	
SupportUnit	SupportRelationshipStruct		Identifies unit(s) which support the aggregate logistically, or with a specified combat or combat support relationship, e.g. a Direct Support or General Support Artillery unit.	
Symbol	HLAunicodeString	(Not Optional)	The APP6A code for the aggregate.	Required
UniqueID	NETN_UniqueID	(Not Optional)	The unique identifier of the object.	Required

Attribute (Optional)	Datatype	Default Value (If optional)	Definition	Usage
UnitEquipment	ResourceStatusArray	(Not Optional)	This summarizes the health status of the equipment comprising the aggregate.	Required
UnitPersonnel	ResourceStatusArray	(Not Optional)	This summarizes the health status of personnel comprising the aggregate.	Required
UnitSupplies	SupplyStructArray1	(Not Optional)	The type and quantities of supplies available (on hand) to the unit.	Required
VisualSignature	VisualSignatureStruct100		Describes the unit's susceptibility to electro-optical detection.	
WeaponsControlOrder	WeaponsControlOrderEnum8		Describes current Weapon Control Order Free, Tight, or Hold.	

9.2.2 Federation Agreements

1. A federate not capable of or not intending to update an attribute should not include it in its declaration set.
2. A federate may request ownership of un-owned attributes after another federate declares its objects.
3. A federate shall not update an attribute unless its value changes.
4. If a federation intends on utilizing this FAD and accompanying FOM to adjudicate combat, all attributes shown as required must be updated.
5. NETN will use "Interest Areas" (IAs) to identify areas in which entity level data must be provided. Outside of the IAs it is acceptable for aggregate units to be "fully" aggregated, i.e. AggregateStateEnum8 = 1, though (until experimentation proves otherwise) it is not necessary that they be "fully" aggregated.
6. NETN will use the EntityList attribute within IAs rather than the attributes: EntityIdentifier, SubAggregateIdentifier, NumberOfSilentEntities, SilentEntities, and SilentAggregates
7. This attribute is intended as a pointer to an array of network objects of relevance to the aggregate but the network object itself is at this point undefined.
8. We will use both WeaponFire and MunitionDetonation for all except mines and IEDs.
9. The UniqueID shall be used in the Marking field as well as in the attributes and parameters which require it.

The enumerated DataTypes used in NETN_Aggregate are further defined as follows:

9.3 Enumerated Datatypes

9.3.1 ActiveStatusEnum8

Name	Value
Other	0
Active	1
Inactive	2

9.3.2 AggregateMissionEnum16

(from JC3IEDM action-event-category-code)

Name	Value
Abdication	1

Name	Value
Accident	2
AccidentAircraftGround	3
Accident_Mine	4
Accident_Traffic	5
Accident_Weapon	6
Accident_Workplace	7
Advancing	8
AerialEngagement	9
AerialShootDown	10
AirAssault	11
AirborneAssault	12
AircraftCrash	13
AircraftLanding	14
AircraftLaunchActivity	15
AircraftLoss	16
AirspaceViolation	17
AlertCancellation	18
Ambush	19
AmphibiousOperation	20
ArmsProduction	21
ArmsTrade	22
Arresting_Legal	23
ArrestingOrObstructing	24
Arson	25
ArtilleryFire	26
Assassination	27
Assembling	28
AssistingACriminal	29
AtmosphericPollution	30
Attack_Deliberate	31
Attack_Diversion	32
Attack_Electronic	33
Attack_Hasty	34
Attack_Main	35
Attack_NotOtherwiseSpecified	36
Attack_Supporting	37
AttemptedMurder	38
AttemptedRape	39
AttemptedRobbery	40
AttemptedSuicide	41
Avoiding	42
BellyLanding	43
Blocking	44
Bombing	45
Bombing_Accidental	46
Bombing_Deliberate	47

Name	Value
BoobyTrapDiscovery	48
BorderCrossing_Escorted	49
BorderCrossing_Forced	50
BorderCrossing_Illegal	51
BorderCrossing_Not-Planned	52
BorderCrossing_Planned	53
BorderCrossing_Surveilled	54
BorderIncursion	55
BorderRaid	56
Breaching	57
Build-Up	58
BurnedOutObject	59
Bypass	60
Canalise	61
Capture	62
CarrierLaunch	63
CarrierRecovery	64
CBRN-EVENT	65
CeremonyOrParade	66
CivilDemonstration_Illegal	67
CivilDemonstration_Legal	68
CivilDisobedience	69
CivilUnrest	70
CivilWar	71
Clearing_Air	72
Clearing_LandCombat	73
Clearing_Obstacle	74
Clearing_RadioNet	75
CodewordExecution	76
Collision_Mid-Air	77
Collision_Obstacle	78
CommunicationsActivation	79
CommunicationsDeactivation	80
CommunicationsDisruption	81
CommunicationsInterception	82
CommunicationsOutage	83
CommunicationsRestoration	84
ConductingConference	85
ConductingForwardPassageOfLines	86
ConductingMediaInterview	87
ConductingPreparatoryFire	88
ConductingRearwardPassageOfLines	89
ConductingRecreationalActivities	90
ConductingRoadService	91
ConductingSocialEvents	92
ConductingSportingEvents	93

Name	Value
Confiscation	94
ConsolidatingOfAPosition	95
Constructing	96
Containing	97
Cooperating	98
CounterAttack	99
CounterAttackByFire	100
Counter-BatteryFire	101
CoupDetat	102
Covering	103
CrimeAgainstHumanity	104
CriminalIncident	105
Crossing	106
Dazzle	107
Death_NaturalCauses	108
DeathOfChiefOfState	109
DeathOfSpiritualLeader	110
Deception	111
Deception_Electronic	112
Defeat	113
Defending	114
Deflecting	115
Delaying	116
Demolition	117
Demonstration	118
Denying	119
Deploying	120
Destroying	121
Disease	122
Disengaging	123
Disrupting	124
Distributing	125
Diversion	126
Drive-ByShooting	127
Drought	128
DrugConsumption_Illegal	129
DrugDistribution_Illegal	130
DrugManufacturing_Illegal	131
DrugOperation	132
DrugStorage	133
DrugTransportation	134
EarlyWarningAlert	135
Earthquake	136
ElectionAssociatedViolence	137
ElectronicEmission	138
ElectronicWarfare	139

Name	Value
EnemyContact	140
Engaging	141
Enveloping	142
Epidemic	143
EquipmentFailure	144
Escaping	145
Escorting	146
Evacuating	147
Execution	148
Exploitation	149
Explosion	150
Famine	151
Fire	152
Firefighting	153
Fix	154
Fix_Acoustic	155
Fix_Electromagnetic	156
Fix_Electro-Optical	157
Flood	158
FollowingAndAssuming	159
FollowingAndSupporting	160
ForcedLanding	161
FriendlyFire	162
GeneratingChemicalSmoke	163
Genocide	164
GovernmentalCollapse	165
Guarding	166
Gunnery_Air-To-Air	167
Harassing	168
Hiding	169
Hijacking_Boat	170
Hijacking_LandVehicle	171
Hijacking_NotOtherwiseSpecified	172
Hijacking_Plane	173
Hold_Defensive	174
Hold_Offensive	175
HostageTaking	176
HumanRightsViolation	177
Hunting	178
Identifying	179
Illumination	180
IndirectFire	181
IndiscriminateShooting	182
IndustrialEspionageIncident	183
Infiltration	184
Interception	185

Name	Value
Interdiction	186
Intimidation	187
Invasion	188
Isolation	189
IssuingMediaArticle	190
IssuingMediaDocumentary	191
IssuingPressRelease	192
Jamming	193
Kidnapping	194
LabourStrike	195
Leaguer	196
LetterBombExplosion	197
LetterBombIncident	198
LocalElection	199
Locating	200
Looting	201
Maintaining	202
Marking	203
MartialLawImplementation	204
MassingOfForces	205
MassiveDeportationOrBanishment	206
MedicalEvacuation	207
MilitaryMobilisation	208
Mine-Laying	209
MissingIndividual	210
MissionStaging	211
MortarFire	212
Moving	213
Murder	214
MutualAssistancePactAgreement	215
NationalElection	216
NationalHoliday	217
NationalStateOfEmergency	218
NaturalDisaster	219
NavalGunFire	220
NavalPlatformFlightOperations	221
NetworkSeizure	222
Neutralize_Chemical	223
Neutralize_Combat	224
Neutralize_Explosive	225
Obscure	226
Observing	227
Occupying	228
Oceans_SeasOrWaterPollution	229
OffensiveOrCounteroffensive	230
OrganisedCrime	231

Name	Value
OutbreakOfRacialOrTribalOrEthnicWarfare	232
Patrolling	233
PeaceConference	234
PeaceTreatyAgreement	235
Penetrating	236
Pestilence	237
PetroleumProductSpills	238
Picketing	239
Poisoning	240
PoliticalDemonstration	241
PoliticalExecution	242
POWReturn	243
PrisonerExchange	244
Procuring	245
Protection_Electronic	246
ProvidingAccommodation	247
ProvidingAgriculturalSupport	248
ProvidingBedding	249
ProvidingCamps	250
ProvidingConstructionServices	251
ProvidingDecontaminationServices	252
ProvidingEducationServices	253
ProvidingHealthcareServices	254
ProvidingHostNationSupport	255
ProvidingInfrastructure	256
ProvidingLaundryServices	257
ProvidingRepairServices	258
ProvidingSecurityServices	259
ProvidingShelter	260
ProvidingStorageServices	261
ProvidingTranshipmentServices	262
Proxy-Bombing	263
PsychologicalOperation	264
PublishingMediaArticle	265
PublishingMediaDocumentary	266
PublishingPressRelease	267
Pursuing	268
Rape	269
Reconnaissance	270
ReconnaissanceInForce	271
Reconstituting	272
Recovering	273
Recuperating	274
Redeployment	275
RefugeeMovement	276
Reinforcing	277

Name	Value
ReliefInPlace	278
ReligiousDemonstration	279
ReligiousViolence	280
ReligiousWarfare	281
Rendezvous	282
Reorganising	283
Repairing	284
Resting	285
Resupplying	286
Retain	287
Retire	288
Revolution	289
Riot	290
Robbery	291
RocketFire	292
Sabotage	293
Screening	294
SecessionOfPortionOfCountry	295
Securing	296
SecurityCompromise	297
SecurityViolation	298
Seizing	299
ServingAsABreakoutForce	300
ServingAsABridgeheadForce	301
ServingAsAFlankGuard	302
ServingAsAMainBody	303
ServingAsAnAdvanceGuard	304
ServingAsAnIn-PlaceForce	305
ServingAsARearGuard	306
ServingAsAReserve	307
SettingUp	308
Shooting	309
SniperAttack	310
SpaceAccident	311
Spying	312
StateOfWar	313
Strafing_Aerial	314
Strike	315
Suicide	316
Supporting	317
Suppressing	318
Surrender	319
Surveillance_Electronic	320
SuspensionOfHostilities	321
Terrorism	322
Threaten	323

Name	Value
Torture	324
Transporting	325
Traversing	326
TreatyViolation	327
Troublemaking_Agitating	328
Troublemaking_Bullying	329
Troublemaking_Harassing	330
Troublemaking_Hooliganism	331
Troublemaking_Inciting	332
Troublemaking_Intimidating	333
Turning	334
UnexplodedOrdnanceDiscovery	335
VandalismOrRapeOrLootOrRansackOrPlunderOrSack	336
Verifying	337
VesselSinking	338
VolcanicEruption	339
WarOrCrisisAlert	340
WarOrMilitaryConference	341
WarCrime	342
WeaponFiring	343
Withdrawal	344
WithdrawalUnderPressure	345
Witnessing	346
NotOtherwiseSpecified	347

9.3.3 CaptureStatus8

Name	Value
Other	0
Not-Captured	1
Captured	2
AttemptingSurrender	3
AttemptingEscape	4

9.3.4 ConcealmentEnum8

Name	Value
Invalid	0
InOpen	1
MountedInternally	2
MountedExternally	3
UnderNet	4
UnderGround	5
InsideStructure	6
FightingPositionCovered	7
FightingPositionUncovered	8

9.3.5 CoverEnum8

Name	Value
Other	0
HastyFightingPositions	1
IndividualFightingPositions	2
CrewServedWeaponsPositions	3
FightingPositionsWithOverheadCover	4
StrongPoints	5

9.3.6 DamageStatusEnhancedEnum32

Name	Value
NoDamage	0
SlightDamage	1
ModerateDamage	2
SignificantDamage	3
Destroyed	4

9.3.7 EchelonEnum8 (from JC3IEDM echelon-size-code)

Name	Value
Army	1
ArmyGroup	2
Battalion	3
BattalionGroup	4
BattleGroup	5
Brigade	6
BrigadeGroup	7
Company	8
CompanyGroup	9
Corps	10
Division	11
Fleet	12
Flight	13
Platoon	14
Regiment	15
Region	16
Section	17
Squad	18
SquadronAir	19
SquadronMaritime	20
TaskElementNavy	21
TaskForceNavy	22
TaskGroupNavy	23
TaskUnitNavy	24
TeamFireteamCrew	25
Wing	26
NotKnown	27
NotOtherwiseSpecified	28

9.3.8 EntityCategoryEnum8

Name	Value
Invalid	0
EquipmentEntity	1
PersonnelEntity	2
EmitterEntity	3
RadioEntity	4

9.3.9 SensorStateEnum8

Name	Value
Other	0
Off	1
OnButNotEmitting	2
OnAndEmitting	3

9.3.10 SupportRelationshipEnum8

Name	Value
Other	0
Logistics	1
DirectSupportArtillery	2
DirectSupportReinforcingArtillery	3
GeneralSupportArtillery	4
Engineering	5

9.3.11 UpdateTypeEnum8

Name	Value
Invalid	0
Create	1
Update	2
Addition	3
Delete	4

9.3.12 WeaponsControlOrderEnum8

Name	Value
Other	0
WeaponsFree	1
WeaponsTight	2
WeaponsHold	3

9.4 Complex Datatypes

9.4.1 CoverStatusStruct

Name	Type	Semantics
CoverStatusPercent	NETNPercentage	The percentage of protection enjoyed by an aggregate. A unit with 100 percent cover would be impervious to the effects of

Name	Type	Semantics
		weapons fire
CoverTypeEnum	CoverEnum8	An optional field describing the type of cover employed by the aggregate

9.4.2 ElectronicSignatureStruct

Name	Type	Semantics
ElectronicSignaturePercent	NETNPercentage	A summary percentage of an aggregates susceptibility to detection of its electronic emissions. Zero percent means that the aggregate has no electronic emissions.
SensorArray	SensorStructArray1	A list of sensors owned by the aggregate together with their respective operational status and range.

9.4.3 EntityListStruct

An array of elements of Datatype EntityStruct with cardinality 1+

9.4.4 EntityStruct

Name	Type	Semantics
Callsign	HLAunicodeString	The name of the object.
EntityCategory	EntityCategoryEnum32	Indicates whether the entity is equipment, person, emitter, etc.
EntityStatus	DamageStatusEnhancedEnum32	The damage state of the entity.
IsDistinctObject	OMT13boolean	OPTIONAL. A BaseEntity object has been created to represent this entity (true) or not (false).
IsUnavailable	OMT13boolean	OPTIONAL. This entity is in use by another object (true) or not (false).
Facing	Float15	OPTIONAL. Direction is measured in degrees clockwise from orientation of unit
Concealment	ConcealmentEnum32	OPTIONAL. Indicates whether the entity is concealed and, if so, how
OffsetLocation	RelativePositionStruct	The entity location given as an offset from the location of the aggregate unit in meters.

Note: Padding check needed after Callsign

9.4.5 EntityListVariableArrayStruct

Name	Type	Semantics
Update Type	UpdateTypeEnum32	Indicates whether this update creates the list or modifies its' values.
EntityList	EntityListStruct	Data about one or more entities comprising the list.

9.4.6 HUMINTSignatureStruct

Name	Type	Semantics
HUMINTSignaturePercent	NETNPercentage	A summary percentage of an aggregates susceptibility to detection by human intelligence collectors. Zero percent signature means an aggregate is impervious to HUMINT.

9.4.7 MissionStruct

Name	Type	Semantics
StartTime	DateTime	An optional field providing the mission start time

Name	Type	Semantics
EndTime	DateTime	An optional field providing the mission estimated end time
MissionEnum	AggregateMissionEnum16	The mission assigned to the aggregate

9.4.8 NetworkList

An array of elements of DataType HLAunicodeString with cardinality 0+. Note: Padding check needed after each element (except final element).

9.4.9 ResourceStatusNumber

Name	Type	Semantics
NumberHealthyOrIntact	NETN_Float64BE	The number of healthy or intact resources
NumberSlightlyDamaged	NETN_Float64BE	The number of slightly damaged resources
NumberModeratelyDamaged	NETN_Float64BE	The number of moderately damaged resources
NumberSignificantlyDamaged	NETN_Float64BE	The number of significantly damaged resources
NumberDestroyed	NETN_Float64BE	The number of destroyed or consumed resources
ResourceName	HLAunicodeString	The name of the resource

9.4.10 SensorStructArray1

An array of elements of DataType SensorStruct with cardinality 1+

9.4.11 SensorStruct

Name	Type	Semantics
SensorStateEnum	SensorStateEnum32	The operational status of the sensor
SensorDamageState	DamageStatusEnum32	The damage status of the sensor
SensorCoverage	Float2	The maximum range of the sensor
SensorID	HLAunicodeString	A sensor owned by the aggregate

9.4.12 SupportRelationshipStruct

Name	Type	Semantics
SupportConsumer	NETN_UniqueID	The unique ID of the consumer of the support.
SupportProvider	NETN_UniqueID	The unique ID of the provider of the support.
SupportType	SupportRelationshipEnum8	The type of support provided by the supporting unit.

Note: Padding check needed after SupportConsumer

9.4.13 VisualSignatureStruct

Name	Type	Semantics
DVOSignaturePercent	NETNPercentage	A summary percentage of an aggregates susceptibility to detection by direct view optics, i.e. the human eye, binoculars, or telescopes. A unit with zero percent signature would be concealed from DVO detection.
I2SignaturePercent	NETNPercentage	A summary percentage of an aggregates susceptibility to detection by Image Intensifying sensors. A unit with zero percent signature would be invisable to image intensifiers (I2).
ThermalSignaturePercent	NETNPercentage	A summary percentage of an aggregates susceptibility to detection by thermal sensors. A unit with zero percent signature would be invisable to thermal sensors.

9.4.14 WorldLocationStructArray2

An Array of element type WorldLocationStruct with cardinality 3+

9.4.15 WorldLocationStruct

Name	Type	Semantics
X	Double2	Distance in meters from the center of the earth projecting through 0 degrees latitude and 0 degrees longitude.
Y	Double2	Distance in meters from the center of the earth projecting through 90 degrees east longitude and 0 degrees latitude.
Z	Double2	Distance in meters from the center of the earth projecting through the geographic North Pole.

9.5 Entity Object Class Extensions

9.5.1 RPR-FOM Platform Object Class Extension

The following RPR-FOM platform object class extensions have been made:

Full Name
HLAobjectRoot.BaseEntity.PhysicalEntity.Platform.Aircraft.NETN_Aircraft
HLAobjectRoot.BaseEntity.PhysicalEntity.Platform.AmphibiousVehicle.NETN_AmphibiousVehicle
HLAobjectRoot.BaseEntity.PhysicalEntity.Platform.GroundVehicle.NETN_GroundVehicle
HLAobjectRoot.BaseEntity.PhysicalEntity.Platform.MultiDomainPlatform.NETN_MultiDomainPlatform
HLAobjectRoot.BaseEntity.PhysicalEntity.Platform.Spacecraft.NETN_Spacecraft
HLAobjectRoot.BaseEntity.PhysicalEntity.Platform.SubmersibleVessel.NETN_SubmersibleVessel
HLAobjectRoot.BaseEntity.PhysicalEntity.Platform.SurfaceVessel.NETN_SurfaceVessel

All of the above extensions include the following attributes:

Attribute	Data Type	Default Value (If Optional)	Semantics	Usage
Activity	AggregateMission Enum16	(Not Optional)	The activity of the object.	Required
Callsign	HLAunicodeString	(Not Optional)	The name of the object.	Required
Embedded UnitList	ArrayOfEmbedded Unit	0	List of unique IDs of on-board elements	
Status	AggregateStatusE num8	1	If an instance shall be taken into account by federates	

9.5.2 RPR-FOM Lifeform Object Class Extension

The following RPR-FOM lifeform object class extensions have been made:

Full Name
HLAobjectRoot.BaseEntity.PhysicalEntity.Lifeform.Human.NETN_Human
HLAobjectRoot.BaseEntity.PhysicalEntity.Lifeform.NonHuman.NETN_NonHuman

All of the above extensions include the following attributes:

Attribute	Data Type	Semantics
Callsign	HLAunicodeString	The name of the object.
Status	AggregateStatusEnum8	If an instance shall be taken into account by federates
Activity	AggregateMissionEnum16	The activity of the object.

9.6 Combat adjudication

This section discusses combat adjudication between objects owned by different federates. This design pattern has not been verified by MSG-068 but is included here as a reference. Any use of this design pattern should consider the fact it has not been thouroghly tested.

Use cases are defined as follows to decompose the problem:

"Case 1" consists of two federates which both publish and subscribe to Platform object A "platform object" is any of the BaseEntity.PhysicalEntity.Platform subclasses belonging to the NETN_Aggregate FOM, e.g. BaseEntity.PhysicalEntity.Platform.GroundVehicle.NETN_GroundVehicle sub-classes. This case is addressed by the existing RPR FOM WeaponFire and MunitionDetonation interactions.

"Case 2" consists of two federates which both publish and subscribe to the NETN_Aggregate object class. This case requires extensions as follows:

A. Case 2A involves ground-to-ground combat (between two Aggregate objects owned by different federates). The section below entitled "Combat Adjudication Service Federate (CASF)" proposes a means for avoiding the "fair fight" problems caused by modeling differences between the federates which may give an unfair advantage to objects owned by one federate over those owned by the other.
B. Case 2B involves an engagement between objects usually modeled as entities (even in an "aggregate" simulation) and an aggregate. If this is an inaccurate assumption, that is if a federate is unable to publish and subscribe to ships and aircraft as platform objects, other extensions to the BaseEntity.AggregateEntity class will probably be necessary. For example, a ship fires naval gunfire at a ground unit or an attack helicopter firing an AGM-114 Hellfire missile(s) at a tank in a mechanized task force comprised of tanks, infantry fighting vehicles (IFVs), and trucks. The (relatively) small number of munitions fired in such an engagement recommends use of the existing RPR FOM interactions with an extension as described in the section entitled NETN_MunitionDetonation below.

"Case 3" consists of two federates which publish and subscribe to dissimilar object types. That is, one federate publishes and subscribes to the Platform sub-class and the other publishes and subscribes to the NETN_Aggregate object class. In this case the objects owned by one federate may not acquire the object(s) owned by the other federate. Without acquisition, conflict is impossible even in circumstances where combat should occur. Engagements in this case will be resolved through use of the "Combat Adjudication Service Federate (CASF)."

The table below summarizes the proposed approach for the different use cases and engagement types.

Engagement Type		CASE 1	CASE 2	CASE 3
Air-to-Air	A2A	Resolve internally	2B: Resolve internally	CASF
Air-to-Ground	A2G	Resolve internally	2B: Resolve internally	CASF
Air-to-Naval	A2N	Resolve internally	2B: Resolve internally	CASF
Ground-to-Air	G2A	Resolve internally	2B: Resolve internally	CASF
Ground-to-Ground				
Indirect Fire Only	G2GI	Resolve internally	2B: Resolve internally	CASF
Direct Fire	G2GD	Resolve internally	Case 2A: Use CASF	CASF
Ground-to-Naval	G2N	Resolve internally	2B: Resolve internally	CASF
Naval-to-Air	N2A	Resolve internally	2B: Resolve internally	CASF
Naval-to-Ground	N2G	Resolve internally	2B: Resolve internally	CASF
Naval-to-Naval	N2N	Resolve internally	2B: Resolve internally	CASF

9.6.1 Combat Adjudication Service Federate (CASF)

This section introduces a means of adjudicating combat engagements between:

- NETN_Aggregate objects owned by different federates (CASE 2A)
- a NETN_Aggregate owned by one federate and one or more platform-level objects are owned by another federate (CASE 3)

A federate capable of adjudicating ground-to-ground combat, hereafter referred to as the Combat Adjudication Service Federate (CASF), must subscribe to the NETN_Aggregate and Platform object classes. The CASF must also utilize the NETN Service Consumer-Provider pattern as follows:

- In CASE 2A, if a NETN_Aggregate owned by one of the federates acquires a NETN_Aggregate owned by the other federate and wishes to engage it in combat, it will initiate a *NETN_RequestService* interaction to which the CASF must respond with a *NETN_OfferService* interaction.
- In CASE 3, CASF will initiate a *NETN_OfferService* interaction offering to adjudicate the combat between the two federates whenever two or more objects owned by different federates should engage in ground-to-ground combat according to its (CASF's) algorithms.

Assuming the negotiation process results with a *NETN_AcceptOffer* interaction (there will obviously be several assumptions and conditions surrounding this service; these will be addressed below), the CASF will adjudicate the combat according to its algorithms and inform the owning federates of adjudication results and ammunition consumed. ("inform the owning federates" is used because of the current prohibition against shared object ownership. Recommend abolishing that constraint and enabling transfer of health state and on-hand supply attributes to CASF in conjunction with initiating *NETN_AcceptOffer* interaction).

Assumptions:

1. CASF is able to utilize the NETN Service Consumer-Provider pattern.
2. CASF is able to subscribe to both platform level and aggregate level object classes and can adjudicate the combat between the object instances.
3. CASF knows enough about the two or more objects to be able to adjudicate combat between them.

What is the minimum set of required attributes necessary to satisfy this assumption?

   a. For all entities (platforms) comprising all units or individual platforms involved in the combat, the following attributes could be important: activity, location, mounted state, weapons control order, status with respect to health, equipment, ammunition, capture, cover, and concealment (from the different sensor types, i.e. the signature values in NETN_Aggregate).
   b. The objects engaged in direct fire combat maybe supported by other units, a supporting artillery unit for example, whose support and weapons expenditure should be accounted for by CASF.
4. Both federates accept the *NETN_OfferService* interaction. An exception, below, addresses what to do if one or more federates refuse the service offer.

Exceptions:

1. One of the federates refuses the NETN_OfferService interaction. Recommend an initial implementation of one of three courses of action. A federate which refuses the NETN_OfferService interaction must necessarily refuse combat by one of means:
    a. Retreat. This would seem particularly appropriate if an object is advancing, is notified that it is in danger of ground combat, and chooses to retreat along its route of advance so as to avoid combat.
    b. Surrender.
    c. Attempt to hide. Consider a scout or small reconnaissance patrol with intent on gaining information on an enemy's activities. The scout or patrol attempts to avoid detection by moving more slowly, concealing its-self, and adopting an appropriate weapons control order. If these attempts are communicated in the appropriate attributes, then it would seem appropriate to allow the opposing forces to operate in close proximity with each other without engaging in combat. It would also seem appropriate to periodically assess whether the reconnaissance element has been discovered and if so, then again initiate the NETN_OfferService interaction to start ground combat.
2. Both of the federates refuse the NETN_OfferService interaction. The same three options are available to both federates.

Variations:

- CASF is itself one of the federates which owns an object(s) that is part of the combat adjudication. CASF initiates the NETN_OfferService interaction as before and adjudicates the combat if accepted by the other federate. This variation is problematic only insofar as the conflict resolution is thought to be unfair. Recommend an initial implementation consistent with (the same as) CASF acting as a service for other federates and resolving issues if and when they arise. The alternative is dedicating a federate to conflict adjudication which would certainly be more expensive and might preclude using that federate for other federation functionality (if for some reason more than one instance of the federate is not supportable).

9.6.2 Area-effects Munitions

Area-effects munitions, whether fired by indirect-fire weapons systems or dropped by air, are typically modeled as discrete munitions and thus may be handled with the existing RPR FOM WeaponFire and MunitionDetonation interactions with the following caveats. Federation agreements are proposed to decrease the number of interactions sent as follows:

- For area-effects munitions fired in support of Aggregate vs. Aggregate combat, WeaponFire and MunitionDetonation interactions will NOT be used. Instead, the firing unit(s) will be included as supporting units as discussed in CASF section above.
- For artillery missions NOT fired in support of Ground-to-Ground Combat, the MunitionDetonation interaction will be used with the QuantityFired parameter sent to the number of rounds in the volley, e.g. for a battalion of 18 howitzers firing a battalion 10, we would expect ten interactions each with QuantityFired equal to eighteen.

For air-delivered "dumb bombs," the QuantityFired parameter will be sent to the number of simultaneously delivered bombs.

Note that in the cases above referencing use of the RPR FOM WeaponFire and MunitionDetonation interactions, we will use the convention that both interactions are necessary only if the munition will be

instantiated as a BaseEntity object, otherwise the firing federate need only initiate the MunitionDetonation interaction.

9.7 Interaction Extensions

9.7.1 NETN_CombatResult

The NETN_CombatResult is a subclass of NETN_Service and is initiated by CASF to communicate engagement results. CASF will usually send at least two interactions; one to each of the participating federates though for protracted engagements, CASF may send periodic interactions providing incremental engagement results. In all cases, the interaction consists of personnel, equipment, and supply status changes as a result of the engagement. Inherited attributes are shown in italics.

Parameters	Data Type	Default value (if optional)	Definition
ServiceID	*EventIdentifierStruct*	*(NotOptional)*	*Unique identifier for a service*
Consumer	*NETN_UniqueID*	*(NotOptional)*	*Entity that has requested the service*
Provider	*NETN_UniqueID*	*(NotOptional)*	*Providing or intended provider entity*
EngagementResults	EngagementResultsList Struct	0	The list of unique IDs damaged in the engagement together with their associated damage.
SuppliesConsumed	SupplyStructArray1	0	The supplies consumed during the engagement.

9.7.2 EngagementResultListStruct

An array of elements of Datatype EngagementResultStruct with cardinality 0+.

9.7.3 EngagementResultStruct

Name	Type	Semantics
UniqueID	NETN_UniqueID	The unique identifying alphanumeric code of the referenced object.
DamageState	DamageStatusEnhancedEnum32	The damage state of the referenced object.

9.7.4 SupplyStructArray1

An array of elements of Datatype SupplyStruct with cardinality 0+.

9.7.5 SupplyStruct

Name	Type	Semantics
SupplyType	EntityTypeStruct	The type of supply (as described in the Bit Encoded Values for Use with Protocols for Distributed Interactive Simulation Applications)
Quantity	Float4	The number of units of the supply type. The unit measure depends on the supply type and shall use the SI units of measurement used for such supplies.

9.7.6 NETN_MunitionDetonation

The NETN_MunitionDetonation, extends the existing RPR FOM interaction to enable Entity versus Aggregate combat engagements, that is, an Entity firing individual munitions against a unit comprised of equipment vulnerable to that munitions type. The extension is necessary to distinguish between the different equipment types in the aggregate which may be vulnerable to the munitions. For example, an attack helicopter armed with AGM-

114 Hellfire missiles attacking a mechanized task force would logically fire at tanks first, then infantry fighting vehicles, then trucks. Including the preferred target list in the munitions detonation allows communication of the tactical intent. If possible, the receiving federate should adjudicate the munitions effects against the first entity type on the list closest to the DetonationLocation of the munition.

Parameters	Data Type	Semantics
UniqueID	NETN_UniqueID	The unique identifying alphanumeric code of the aggregate for electronic transmissions.
PreferredTargetList	EntityTypeStructArray	A prioritized list of target types