Guidance, Rationale, and Interoperability Modalities for the Real-time Platform Reference Federation Object Model (RPR FOM)

Version 1.0

10 September 1999

Edited by:

Sean Reilly
Engineering, Test, and Evaluation Department
Naval Undersea Warfare Center
Newport, RI

Keith Briggs
iMT Incorporated & HLA Products
San Jose, CA

Acknowledgments

This document was created as a community effort by the Real-time Platform Reference Federation Object Model (RPR FOM) Standards Development Group (SDG). This SDG was chartered by the Simulation Interoperability Standards Organization's Standards Activity Committee on May 19, 1998. The RPR FOM standard would not have been possible without the hard work and dedicated efforts of the following individuals:

FOM Drafting Group

Editor: Graham Shanks
Associate Editor: Michael O'Connor

GRIM Drafting Group

Editor: Sean Reilly
Associate Editor: Keith Briggs

Standards Development Group Officers

Chairperson: Richard Schaffer
Recording Secretary: Douglas Wood
Technical Area Director:
David McKeeby *since February 1999*
Randy Saunders *through January 1999*

Assigned Reviewers

Jeff Fischer
Len Granowetter
Jim Hammond
John Hancock
Steve Monson
Steven Sheasby
Chris Turrell

Volunteer Reviewers

Randy Branham
Colby Cavin
Andrew Chew
Dan Coffin
Susan DelPriore
Margie Dey
Mike Donnelly
Jeff Fischer
H. J. Fontenot
Deb Fullford
Scott Hamilton
Rob Harrington
Peter Kassal
Debra Jones
Ted Mulder
Walter Pepere
David Pierce
Robert Puckett
Nils Reker
Lawrence Rieger
Roy Scrudder
John Skutnik
Larry Small
Steve Swenson
Joan Turner
Gary Warren
Stephanie White

and the members of the SISO-RPR e-mail reflector

SISO Support Staff

Allison Griffin

Community Ballot Group Members

Bill Andrews
Michael Bachmann
Michael Baldwin
Richard Bard
Keith Briggs
David W. Brown
Adin Burroughs
Craig D. Button
Robert Case
Julienne Picot Chappell
Bruce Clay
Glenn Conrad
James N. Cook
Demetrio Cortese
David Decamp
Em delaHostri
Steven R. Drake
Jeff Fischer
Milton L. Fulghum
William Garbacz
Len Granowetter
Glenn Gross
Jim Hammond
Marvin Hammond
Susan Harkrider
Rob Harrington
Ron Hofer
Kyle Isakson
Pamela M. Jacobs
Joanne M. Kuhns
Gary M. Lightner
Max Lorenzo
Randall R. Macala
Peter McCarthy
Bruce McDonald
Bruce McGregor
David McKeeby
Duncan Miller
Tom Mullins
Michael Myjak
David Nemeth
Michael O'Connor
David Porter
David Pratt
Tom Radgowski
Sean Reilly
Braulio Rivera
Susan Rodio
Peter Ryan
Joseph Sardella
Randy Saunders
Richard Schaffer
Achim Schreiber
Roy Scrudder
Richard Severinghaus
Graham Shanks
Sean Sharp
Jack Sheehan
Col. Mark Smith
Keith Snively
Susan Solick
Thomas Stanzione
Austin Stoudenmire
Kimberly Straight
Robert C. Turrell
Thad Vavrek
Ralph Whitney
Douglas Wood
John Woodyard
Philip Yanni
Robert Youmans
Lucien Zalcman
Philomena Zimmerman

When the SISO Standards Activity Committee approved this standard on September 12, 1999, it had the following membership:

SISO Standards Activity Committee

Philomena Zimmerman	**Chair**
Michael Myjak	**Vice Chair**
Jack Kramer	**Secretary**
Keith Briggs	
Rhonda Freeman	
Susan Harkrider	
Sam Knight	*CC Vice Chair*
David McKeeby	
Paul Nagele	
Randy Saunders	
Simone Youngblood	

When the SISO Executive Committee approved this standard on September 14, 1999, it had the following membership:

SISO Executive Committee

Col. Mark Smith	**Chair**
Chris Bouwens	**Vice Chair**
Steve Seidensticker	**Secretary**
Warren Katz	
Dell Lunceford	
Duncan Miller	*CC Chair*
James E. Shiflett	
Chuck Waters	
Philomena Zimmerman	*SAC Chair*

Table of Contents

1 Scope

A federation's object model defines the object classes, attributes and interactions which can be transmitted between any subscribed federates. This guidance document accompanies the Real-Time Platform Reference Federation Object Model (RPR FOM). It provides the usage rules for the RPR FOM, and the definitions, descriptions and rationale not otherwise specified within the standard FOM format.

As an example, this document, via reference to the IEEE Standard for Distributed Interactive Simulation (DIS) – Application Protocols [4,5], defines the responsibility of federates that fire a weapon and of federates that are targeted and hit by a weapon. While the FOM definition provides for the message definition, it does not directly define federate responsibilities in generating, transmitting, or responding to message content.

For simplicity, this document provides these definitions either directly or through references to other sources.

2 Applicable Documents

[1] "High-Level Architecture Rules", Version 1.3, U.S. Department of Defense, 5 February 1998.

[2] "High Level Architecture Interface Specification", Version 1.3, U.S. Department of Defense, 2 April 1998.

[3] "High-Level Architecture Object Model Template Specification", Version 1.3, U.S. Department of Defense, 5 February 1998.

[4] "IEEE Standard for Distributed Interactive Simulation - Application Protocols," IEEE Std 1278.1-1995.

[5] "IEEE Standard for Distributed Interactive Simulation - Application Protocols," IEEE Std 1278.1A-1998.

[6] Simulation Interoperability Standards Organization, "Reference FOM Study Group Final Report," Version 1.0, March 9, 1998.

[7] J. Towers, J. Hines, "Highly Dynamic Vehicles in a Real/Simulated Virtual Environment (HyDy), Equations of Motion of the DIS 2.0.3 Dead Reckoning Algorithms," Advanced Research Projects Agency, February 7, 1994.

[8] "Enumeration and Bit Encoded Values for Use with Protocols for Distributed Interactive Simulation Applications," developed by the Institute for Simulation and Training under contract to the US Army STRICOM, Contract # N61339-94-C-0080, February 22, 1999.

[9] Simulation Interoperability Standards Organization, “SISO Policies and Procedures,” Version 1.0, February 27, 1998.

[10] "Simulation Interoperability Standards Organization Standards Activity Committee: Standards Development Handbook for SISO Standards", DRAFT, SISO 1998.

[11] Simulation Interoperability Standards Organization, "Real-time Platform Reference Federation Object Model (RPR FOM)," Version 1.0.

[12] DoD Joint Chiefs of Staff Joint Publication 1-02, "DOD Dictionary of Military and Associated Terms" (www.dtic.mil/doctrine/jel/doddict/), Washington, DC

[13] The IEEE Standard Dictionary of Electrical and Electronics Terms, 6th Edition, 1997, Institute of Electrical and Electronics Engineers, NY, NY

[14] DoD Data Dictionary System (DDDS), December 97 release, Defense Information Systems Agency, Alexandria, VA

3 Introduction

3.1 Purpose of the RPR FOM

Prior to the development of the High Level Architecture (HLA) for Modeling and Simulation[1], the IEEE Standard for Distributed Interactive Simulation (DIS [4]) defined a standard set of protocols which permitted networked simulations to interact through the common communication of simulation data. These two mechanisms create significantly different environments for distributed simulation applications. DIS combines a simple data delivery architecture (broadcast UDP/IP datagrams) with strictly controlled message format standards (known as protocol data units (PDUs)) to create a system that maximizes interoperability between simulation partners. In contrast, the HLA defines a robust data delivery architecture but leaves the definition of content standards to individual simulator federations. The HLA Run Time Infrastructure (RTI) is a delivery mechanism designed to maximize network performance by allowing individual federates to filter data at many different levels in the delivery process. Data content standards are defined in an object oriented interchange format called the Object Model Template (OMT). The set of all attributes potentially available from a given federate is called the Simulation Object Model (SOM). The set of attributes that the federates agree to share during a particular execution (an HLA federation) is documented in the Federation Object Model (FOM). By allowing federates to specify data content standards on an execution-by-execution basis, the HLA allows rapid adaptation to changes in simulation requirements and objectives.

The Real-time Platform Reference Federation Object Model (RPR FOM, pronounced “reaper fom”) was designed to organize the attributes and interactions of DIS into a robust HLA object hierarchy. The priorities for developing this design are, in order:

1. Support transition of legacy DIS systems to the HLA.
2. Enhance a-priori interoperability among RPR FOM users.
3. Support newly developed federates with similar requirements.

Like DIS, the RPR FOM is designed to support real time simulations where the principal participants are discrete physical entities such as planes, ships, soldiers, and munitions. These simulations are considered "real-time" because each second of elapsed execution time is equivalent to one second of time in the virtual world. Real-time, platform simulations are often used to support man-in-the-loop or hardware-in-the-loop systems.

The RPR FOM is an instance of a Common Foundation Reference FOM (CF-RFOM) as defined by the SISO Reference FOM Study Group [6]. A CF-RFOM differs from a normal FOM because it refers to a notional rather than an actual collection of federates. The goal of a CF-RFOM is to

[1] The fundamental core of the High Level Architecture (HLA) is defined by three documents: the HLA Rules [1], the HLA Interface Specification [2], and the HLA Object Model Template (OMT) [3]. These documents are freely available through the DMSO HLA Web Site, http://hla.dmso.mil.

enhance *a-priori* interoperability by specifying content standards for commonly used attributes and interactions. Building upon the Reference FOM to meet the needs of a given execution creates the FOM for a particular federation. Because each federation's changes only extend this core functionality, simulations that do not require interoperability beyond the "starter" level of the CF-RFOM can participate, without software modification, in more specialized federations.

Version 1.0 of the RPR FOM is designed to provide an HLA conversion path for the full suite of DIS capabilities as defined in IEEE 1278.1-1995 [4]. A future Version 2.0 of the RPR FOM is planned that will add the functionality of the IEEE 1278.1A-1998 standard [5] and will support the IEEE 1516 HLA standard. Once this basic task of transitioning existing DIS functionality is completed, the standard development group plans to release a RPR FOM Version 3.0 that captures new real-time simulation data exchange solutions.

The HLA standards are evolving in parallel with the RPR FOM. Version 1.0 of the RPR FOM is designed to support Version 1.3 of the HLA [1][2][3]. It is anticipated that RPR FOM Versions 2.0 and 3.0 will be complient with the final approved versions of the IEEE 1516 HLA Standards.

Simulations in a RPR FOM federation are required to comply with all referenced elements of the High Level Architecture (HLA) including the HLA Rules, Object Model Template, and Run-Time Infrastructure (RTI) Specifications.

3.2 RPR FOM Development Process

The Real-time Platform Reference Federation Object Model (RPR FOM) Standards Development Group (SDG) created both the RPR FOM and this document as a community effort. Participation in the RPR FOM SDG was open to all interested parties. The development of the RPR FOM was conducted through semi-annual working group meetings, semi-annual SISO Simulation Interoperability Workshop meetings, periodic phone conferences, and via the RPR FOM e-mail reflector. Notifications concerning upcoming meetings and phone conferences were posted on the reflector well in advance of any scheduled activity. One could subscribe to the RPR FOM reflector via the SISO web page at http://www.sisostds.org. RPR FOM products currently under configuration management were also available through this site.

3.3 Interoperability and Compliance

The modal form "**shall**" is used in this document to indicate the mandatory practices that **shall** be implemented as the interoperability requirements for each object or interaction class. (This document will use bold typesetting for all occurrences of the word "**shall**" or "**shall not**" to make them easier to locate.) Compliance testing **shall** be limited to the classes implemented by each federate. The modals "should" or "may" are used to indicate recommended, but not required, practices. These optional capabilities are often driven by specific federate modeling requirements and are not required of all RPR FOM compliant simulations.

The RPR FOM seeks to provide an extendible framework that enhances interoperability by creating standard guidance for commonly used attributes and interactions. Users of this reference are not required to implement all of the classes contained herein. However, some classes may require the use of other classes to create a self-consistent interface to the federation. If an RPR FOM compliant simulation chooses to implement a RPR FOM feature it **shall** do so as specified in this document. For example, weapon firings **shall** be represented by the WeaponFire Interaction or by a compatible user extension.

The RPR FOM may be extended in several different ways, while still maintaining RPR FOM compliance. Changes that maintain compliance include adding new classes, attributes or parameters and extending enumerations. Existing RPR FOM applications may simply ignore the new elements. Unknown enumerators should be treated as the default value. Changes that prevent backward compatibility are non-compliant with this standard. These include modifying the structure of a complex data type, and changing the name of a class or attribute. Existing RPR FOM applications will expect the original elements to be present in the FOM.

Because the RPR FOM is a reference FOM, users are free to make non-compliant changes to FOM elements or practices to meet their own development needs. However, simulations based on these kinds of modified FOMs may not have *a-priori* interoperability with other systems based on the RPR FOM.

4 Overview of HLA Functionality

The RPR FOM has been designed to be compliant with all applicable HLA rules and standards as described in the following sections.

4.1 HLA Rules

RPR FOM compliant simulations **shall** comply with the HLA Rules [1].

4.2 HLA Object Model Template (OMT)

RPR FOM compliant simulations **shall** implement all required formats defined in the HLA Object Model Template [3].

4.3 HLA Services

The HLA Run-Time Infrastructure Specification [2] defines a set of services that permit distributed federates to participate in a common federation and transfer object and interaction data between the participant federates. Federates implementing the RPR FOM **shall** utilize the RTI to communicate with other federates participating in the federation.

Additional guidance on the use of HLA services is presented here:

4.3.1 Time Management

RPR FOM federate support for the Time Management services is optional and should be negotiated on an exercise by exercise basis. As a default and at a minimum, RPR FOM federates **shall** operate with time stepped, clock driven, independent time advance (see [1], rule 8.5). Verified operation of the RPR FOM in modes other than this time-flow mechanism is not guaranteed.

These clock driven simulations are considered "real-time" because each second of elapsed execution time is equivalent to one second of time in the virtual world. Time syncronization, if it is used at all, is performed outside of the simulation itself. For example, Network Time Protocol (NTP) is often used to syncronize “wall clock” times across a federation.

4.3.2 Ownership Management

Because of a lack of support in IEEE 1278.1, many DIS simulations did not implement an attribute transfer capability. Even those that were capable of some transfer of control often could not support the partial attribute transfer supported by the HLA. For this reason, the RPR FOM currently does not support attribute transfer as a default setting for interoperability.

4.3.3 Data Distribution Management (DDM)

Data distribution management services can be utilized to more specifically control the flow of object and interaction data. These services should primarily benefit large exercises where the total exercise traffic may require that the more advanced DDM data distribution services be implemented.

However, the RPR FOM currently makes no effort to standardize the use of DDM as the requirements for DDM may vary widely across the RPR FOM user domain. RPR FOM federate support for the DDM services is optional. Testing of the RPR FOM functionality for any specific set of defined routing spaces should be the responsibility of the participating federates.

5 General FOM Guidance and Rationale

5.1 Structural Changes from DIS

The RPR FOM maps DIS Protocol Data Units (PDUs) into appropriate HLA Object and Interaction Classes. In general, individual PDU fields are mapped into corresponding class attributes or parameters. An individual PDU may be mapped across one or more HLA Object or Interaction Classes. This change in structure is designed to take advantage of the HLA's data delivery features. The HLA's Run Time Infrastructure (RTI) uses field separation to limit network traffic by two mechanisms:

1. reducing the transmission of unchanged data, and
2. providing delivery only to federates which have expressed interest.

In an effort to separate the requirements of information content from those of delivery, the RTI hides the mechanisms for these enhancements behind a generic application programmer's interface (API).

Objects in the RPR FOM are organized into a four level class hierarchy. Classes are separated by logical distinctions between groups of attributes. An effort was made during the design to minimize the repeating of attributes between classes. This creates a hierarchy in which each class represents fundamental object characteristics (e.g. dead reckoning capability) instead of the behaviors of complete units (e.g. M1A1 tanks). This results in some attributes that may have no meaning for a particular subclass. Whenever this type of conflict occurs, it will be noted in the description of the attribute. For example, SubmersibleVessel(s) have AfterBurnerOn as an unrealistic attribute inherited from Platform. However, this form of organization was chosen to provide implicit guidance for Reference FOM specialization; new attributes and sub-classes are to be added to the RPR FOM based on attribute commonality. The SDG has chosen a similar, but shallower hierarchy for interactions (two levels). A mapping from the DIS PDU structures into the RPR FOM is provided in Section 8.

The new structure has a direct effect on the Boolean and enumeration values previously provided as DIS bit-structured fields. In the RPR FOM, all attribute and parameters previously represented as bit values have been expanded into independent fields.

The new structure also affects many of the fields used to express array size in DIS. Since RPR FOM arrays are transmitted as separate attributes, the number of elements can usually be derived directly from the array's length in bytes and the size of each element. In such cases, the RPR FOM generally excludes the array length as a separate attribute. Exceptions to this approach occur when the element size is variable or when the number of elements is commonly used for other purposes.

The DIS PDU structures were intentionally defined to limit the size of each packet to the minimum supported for UDP/IP broadcast. Because the HLA removes these UDP/IP restrictions, these limitations are not part of the RPR FOM. DIS gateway developers and the users of these gateway tools should be careful to characterize the performance of their DIS to RPR FOM translation in the case of large data updates.

5.2 Default Parameters

In many cases, federates may choose not to update attributes or send interaction parameters that have no meaning for that federate. The class and interaction definitions provided in Sections 6 and 7 of this document specify each attribute and parameter that may be treated in this manner. Subscribing federates **shall** assume default values for any attribute or parameter not provided by the publisher. Unless otherwise specified in the class or interaction definition, default values **shall** be treated in the following manner:

- All integer and floating point numeric attributes and parameters default to zero;
- All boolean attributes and parameters default to false;
- All enumerated attributes and parameters default to "other"; and
- All arrays and strings default to "empty".

Other attribute and parameter types do not have standard defaults. Modifications to the RPR FOM should attempt to use the above default values whenever practical.

5.3 Filter Support

Unlike DIS, the HLA supports multiple levels of data filtering:

- Application Level filtering refers to the ability of receiving applications to accept or reject individual packets of information based on their content. This level of filtering is accomplished at the application layer instead of by the RTI. Application Level filtering was the only form of filtering supported by DIS. Although this type of filtering is usually simple for information senders, it can require extensive processing on the receiver side when many simulations are involved.

- Declaration Management (DM) filtering refers to the ability of the RTI to deliver information based on each federate's expression of interest in (subscription to) object classes and interactions. Applications never receive information for objects classes or interactions to which they have not subscribed. This type of filtering adds processing to the senders, but reduces the application filtering performed by the receivers. In addition, this type of filtering can also decrease network bandwidth.

- Data Distribution Management (DDM) Filtering allows the RTI to route data based on information content. This differs from application level filtering where the DDM system focuses on information control during transmission rather than upon reception. DDM services provide an effective run-time capability to control the

> delivery of data within an exercise; however, the specific performance benefits may vary depending upon RTI and Federation design. In some cases, DDM usage may add processing overhead or data delivery latency.

In order to support common applications of DM filtering, several attributeless sub-classes are included in the RPR FOM Object Class Hierarchy. For example, in order to support DM filtering on the equivalent of the DIS Entity Type's "domain" field, the FOM includes seven attributeless subclasses of the Platform class. Subscribing to the Aircraft class, for instance, is the equivalent of passing entities with the domain of "Air" in a DIS Application Level filter.

To fully support this form of DM filtering, federates **shall** publish all objects at the leaf nodes of the RPR FOM. If a RPR FOM leaf node is subclassed, then the federate may publish objects at the newly created leaf nodes. A leaf node is defined as the lowest level available in the object class hierarchy table (no subclasses). In contrast, class subscription should be used at the highest level (farthest from the leaf nodes) that supports all of the attributes and DM filtering required by the receiving federate.

5.4 DIS Entity Identifiers

The publishing federate **shall** be responsible for generating a unique EntityIdentifier for each new object registered with an object class derived from the BaseEntity object class (see section 6).

DIS uses the Entity Identifier triplet of SiteID-HostID-EntityNumber to uniquely identify entities in a distributed simulation. Each application is responsible for establishing a unique SiteID-HostID pair and then to generate locally unique entity numbers. Additional identifiers are used to uniquely specify systems that are attached to an entity (e.g., radios and emitters). Since the HLA RTI provides an object naming mechanism to uniquely identify objects, the use of the DIS Entity Identifiers for object class attributes in the RPR FOM appears redundant and unnecessary. The fact that attribute and parameter references to federate objects use the RTI object name further supports this view. However, there are two purposes for the existence of these attributes. The first purpose is to support legacy applications that are migrating from DIS to HLA; the primary motivation for the RPR FOM design. The second purpose is the use of wild card addressing in simulation management interactions.

5.4.1 Entity Identifiers for DIS Legacy Applications

Many DIS legacy applications use the DIS Entity Identifier and system identifier internally for entity and system lookups. To ease the transition from DIS, the Entity Identifier and system identifier attributes were maintained in the associated RPR FOM object classes. It may be that over time, the lookup requirement for these identifier attributes becomes less prevalent, and ultimately they could be removed. In order for this to happen though, an alternative solution for simulation management addresses is required.

5.4.2 Simulation Management Addressing

Although the RTI provides many simulation management (SIMAN) functions, some DIS features are not supported directly by the API. The RPR FOM implements DIS compatible SIMAN services as HLA interaction classes (see Section 7.5). In DIS, the SIMAN services can be applied to a single entity, all entities on a host, all entities at a site, or all entities in an exercise. This feature is possible through the use of wildcard values for the DIS Entity Identifier components. No clear way to duplicate this wildcard-addressing scheme using the RTI object names is readily apparent. The addition of the "EntityIdentifier" attribute was required in appropriate object classes (see Section 6) to facilitate the use of wildcard addressing for these services in an HLA environment.

5.5 Dead Reckoning

The basic architecture of DIS specified the use of a dead reckoning mechanism for reducing communication processing (section 1.3.1.f of IEEE 1278.1 [4]). The RPR FOM has adopted this mechanism for the same purpose. For each registered object, the use of dead reckoning requires that a federate maintain a dead reckoning model in addition to its own internal model. The dead reckoning model **shall** follow one of the prescribed dead reckoning algorithms defined by IEEE 1278.1 1995 and enumerated in the RPR FOM. Dead Reckoning **shall** be applied to all objects that are derived from the BaseEntity object class.

A federate **shall** issue a Time-Space-Position Information (TSPI) update whenever the differences in position or orientation between its internal model and its dead reckoning model have exceeded established thresholds. The default thresholds for this TSPI update condition are defined by the IEEE 1278.1 1995 standard as DRA_ORIENT_THRSH_DFLT = 3 degrees and DRA_POS_THRSH_DFLT = 1 meter [4]. A TSPI update includes all of the applicable attributes from the following: WorldLocation, Orientation, VelocityVector, AccelerationVector, and AngularVelocityVector. The applicable attributes **shall** be those that are required by the specified DR algorithm. A TSPI update attribute may be omitted if its last updated value is within an established epsilon amount from its current value. The default values for the attribute epsilons are given in Table 5-1. If any of the TSPI update attributes are transmitted for reasons other than a TSPI update condition (e.g., a provideAttributeValueUpdate), then this condition **shall** be treated as if it were a TSPI update condition. In other words, if any of the TSPI attributes are updated, then all other applicable TSPI attributes with an epsilon difference are to be updated. A TSPI update **shall** be invoked through a single RTI updateAttributeValue call. At the time of this call, all of the TSPI attributes should have identical delivery categories; otherwise, it is not guaranteed that receiving federates will receive them atomically.

Table 5-1 Default Dead Reckonning Thresholds

Threshold	Value
Position	0.001 m
Orientation	0.00001 radians

Velocity	0.001 m/s
Acceleration	0.001 m/s/s
Angular Velocity	0.00001 radians/s

As in DIS, it is the receiving federate's responsibility to maintain a dead reckoning model for each external entity of interest. By applying the specified dead reckoning algorithm, the dead reckoning model provides a close approximation of the external entity's TSPI data. Reflected TSPI update attributes **shall** be used to correct the dead reckoning model so that future approximations are based on the most recent TSPI data.

5.6 Time Stamps

Dead reckoning and other simulation requirements supported by DIS required the transmittal of time stamp information. For example, this functionality can be used to account for network transport delays in exercises where federates are synchronized to a common external clock. In contrast, HLA's Time Management is concerned with the mechanisms for controlling the advancement of each federate to deliver information in a causally correct and ordered fashion. In the 1.3 version of the RTI, time stamp information is not passed between federates which are not using Time Management services. As a result, the RPR FOM encodes the time stamp within the RTI's user defined tag API parameter.

Federates **shall** send the time at which the data is valid in the user defined tag with every UpdateAttributeValues or SendInteractions call. The time **shall** be in the first 8 bytes (octets) of the user defined tag, using the DIS time stamp field format (see section 5.2.31 of IEEE 1278.1-1995) converted into hexadecimal ASCII character representation (0-9 and A-F), with leading zeros included. The ordering of the characters **shall** be in accordance with section 5.1.1 of IEEE 1278.1-1995, that is most significant octet first, with the most significant bits first (i.e. the character for bits 4-7 precedes the character for bits 0-3). This encoding is equivalent to the result of the "C" statement "sprintf(UserTag, "%08X", DIStimestamp)," where "DIStimestamp" is represented in native format. All federates **shall** transmit this field, even if they do not use it themselves, so that other federates can use its value.

Note: It is anticipated that future HLA specifications will incorporate an ability to directly transfer additional time stamp information, outside of the scope of the HLA Time Management functions, within the RTI's API parameters. Future versions of the RPR FOM may be modified to directly utilize this feature.

5.7 Basic Data Types and Endian Representation

The RPR FOM refers to basic data types such as float, double, short, long, and boolean. With the exception of the boolean type, the exact representations of these types are specified in the OMT standard. The current OMT standard definition of the boolean type is ill-defined. When the boolean type is used in the RPR FOM, it refers to an eight bit unsigned integer which has only two valid values: 1, which represents true, and 0, which represents false.

To insure interoperability among federates, federations must agree on byte ordering conventions. As in DIS, the big-endian network byte order convention **shall** be used in the RPR FOM (in [4], Section 5.1.1) Federations may choose to use the little-endian convention, but they may not have *a-priori* interoperability with other systems based on the RPR FOM.

5.8 Word Alignment

Some computer systems have alignment rules that must be taken into consideration when constructing complex types. The guidance for developing RPR FOM complex data types has been derived from the equivalent DIS guidance. Complex types **shall** be organized such that all base types (integers and floating point numbers) start on an offset which is a multiple of their own size. For example, the offset of a 32 bit float, within a complex type could be zero, 32, 64 or any other multiple of 32. Padding **shall** be added to the complex type if this internal alignment cannot be achieved through simple re-arrangement. All padding fields **shall** be set to zero.

The following example illustrates this guidance: Using C syntax, we show two versions of a complex data type below:

```
struct BadType {                          struct GoodType {
        char    aChar ;  /* 8 bits */             long    aLong ;  /* 32 bits */
        short   aShort ; /* 16 bits */            short   aShort ; /* 16 bits */
        long    aLong ; /* 32 bits */             char    aChar ;  /* 8 bits */
} ;                                       } ;
```

The "BadType" on the left is improperly aligned. The attribute "aShort" starts on an 8-bit boundary that is not a multiple of the size of a short (i.e. a multiple of 16). The attribute "aLong" starts on a 24-bit boundary, which is not a multiple of the size of a long (i.e. 32). The “GoodType” on the right is properly aligned. Even though the attribute “aChar” does not fill up the second 32 bit word, terminal padding is not required by these rules. Padding at the end of the data type is not required unless that form of alignment is needed for structures-within-structures or other forms of aggregation. For example, if the “GoodType” above were to be used as an array element, 8 bits of terminal padding would be required at the end to maintain proper alignment.

5.9 Delivery Category

The HLA supports two different delivery categories – reliable and best effort. Federation developers must specify in their FED file a delivery category to be used for each interaction class, and for each object class attribute. Either delivery category may be used with any of the elements of the RPR FOM, however, the following guidance is recommended:

On LANs, or other networks where reliability is not known to be a problem, best effort should be used for all interactions and attributes. This will usually allow optimal real-time performance of a federation execution. The RPR FOM convention of sending many attributes only upon change may necessitate the use of HLA reliable transport to maintain the level of reliability provided by the DIS "heartbeat" mechanism. If the HLA best effort category does not provide an acceptable level of reliability for a federation execution, the reliable category should be used for attributes that are unlikely to be updated periodically. For example, when entities are moving, WorldLocation is typically updated relatively frequently, so reliable delivery is typically not required for this attribute.

Atomic delivery of updates is not guaranteed when using different delivery categories for different attributes of an object class. That is, a set of attributes sent together may not be received together.

A subtlety to be aware of, is that atomic delivery of an attribute update is not guaranteed when using different delivery categories for different attributes of an object class. That is, a set of attributes sent together may not be received together.

5.10 RPR FOM Naming Conventions

The RPR FOM was developed using naming conventions from the HLA Object Model Data Dictionary (OMDD). For detailed information on the OMDD and access to the complete OMDD contents, visit the HLA Home Page at http://hla.dmso.mil. Use of OMDD names and definitions in the RPR FOM provides several benefits. OMDD names and definitions are based on existing data standards including the Defense Data Dictionary System (DDDS, [14]), Department of Defense (DoD) Joint Publications [12], IEEE dictionaries and handbooks [13], and other sources. As such, the names and definitions use common, well understood terms thereby enhancing the understandability of the RPR FOM. Because many of these names map to DoD data standards, integration with operational systems (which are required to use the same terms in their development) will be facilitated. Finally, integration of RPR FOM-based federates with other federates which have used OMDD terms will be easier because of the common terminology.

The goal of OMDD naming was to provide the most concise name that would accurately reflect the concept being named using commonly understood terminology. Use of terminology from Joint Publication 1-02, "DOD Dictionary of Military and Associated Terms" [12] was the highest priority in OMDD content development. Next in priority were data standards from the DDDS [14]. The DDDS has a rigorous set of naming conventions, defined in DoD 8320.1-M-1, that were used as guidelines for naming OMDD components. Among these conventions are that "prime words" (which formed the basis for OMDD object classes) and "data elements" (which formed the basis for OMDD attributes and parameters) must be named with noun phrases. Additionally, the use of acronyms is discouraged. Only well understood acronyms such as RF were used in OMDD names.

In the DDDS, "data element" names always end in one of eighteen "class words" which relay basic concepts such as code, weight, identifier, temperature, etc. Where this convention added to the meaning of an OMDD name, it was employed. However, when this convention added words to a name without adding meaning, it was not used. For example, the OMDD contains the term "EffectiveRadiatedPower." Using the DDDS naming convention would have added the class word "Rate" to this name, which adds no explanatory capability, and diverges from commonly used terminology. The DDDS conventions add class word modifiers, property modifiers, and an "entity" name to the class word to compose the complete "data element" name. As stated before, these "data element" names were used in the development of OMDD attribute and parameter names. The OMDD, however, does not associate an "entity" with all attributes and parameters. This was done to provide attributes and parameters that could be used in a variety of object classes, based on the need of the federation. Following the previous example, EffectiveRadiatedPower could serve as an attribute of Emitter, Radio, Radar, etc., depending on the federation needs.

To increase the readability and understanding of OMDD names, additional naming conventions were used: initial capitals are used in each word in an OMDD name; complex data type names end in "Struct"; enumerated data type named end in "Enum." These conventions reduce confusion when discussing the FOM.

5.11 Civilian Versus Military

It is anticipated that simulation exercises will contain both military and civilian entities, as well as require more than two-sided exercise play. The RPR FOM mappings specifically omit any distinction between civilian and military data at the class level. Any such distinction can be derived from the attribute or parameter values (e.g., EntityType and/or ForceIdentifier). Full support of this distinction may actually require extension of the enumerated types. It is the intent that any such extension would be reciprocated by changes to the DIS enumeration document.

This attribute-based distinction between military and civilian data requires Data Distribution Management to support interest management between the two. If declaration management support is required, exiting RPR FOM leaf classes can be extended with sub classes that make this distinction (e.g., MilitaryAircraft, CivilianAircraft). These distinctions should be reflected in the values of the existing inherited attributes so that interoperability is maintained with federates that rely on them.

6 RPR FOM Class Structure

The RPR FOM organizes attributes for objects instantiated in the HLA infrastructure. The class hierarchy is provided in Table 6-1. The remaining sections of this section provide details on the structure of each class.

Table 6-1 RPR FOM - Object Class Structure Table

Class 1	Class 2	Class 3	Class 4
BaseEntity	PhysicalEntity	Platform	Aircraft
			AmphibiousVehicle
			GroundVehicle
			Spacecraft
			SurfaceVessel
			SubmersibleVessel
			MultiDomainPlatform
		Lifeform	Human
			NonHuman
		Sensor	
		Radio	
		Munition	
		CulturalFeature	
		Expendables	
		Supplies	
	EnvironmentalEntity		
EmbeddedSystem	Designator		
	EmitterSystem		
	RadioReceiver		
	RadioTransmitter		
EmitterBeam	RadarBeam		
	JammerBeam		

In the class attribute tables that follow, required attributes will be indicated using "**bold**" typesetting. Attribute that may be conditionally required based on the value of other settings will be indicated using "*italic*" typesetting. Optional attribute will be indicated using "normal" typesetting.

6.1 BaseEntity Class

The BaseEntity class is designed to provide a basis for the individual entities that are the principal participants in RPR FOM federations. The core attributes shared by all entities include the entity's position and orientation in the virtual world, as well as velocity, acceleration, and angular velocity. These last three attributes allow reflecting applications to "dead-reckon" the entity – that is, to approximate its position and orientation during the period of time between state updates.

The dead reckoning algorithm attribute allows the simulating federate to dictate whether and how reflecting federates perform dead-reckoning. When all reflecting federates perform dead-reckoning in the same way, they are able to share a more consistent view of the state of the virtual world.

In order to provide for a consistent interpretation for all participants, all federates should apply a consistent version of the dead reckoning algorithms [7]. The coordinate system and dead reckoning models used **shall** follow the same form described in Sections 1.3.2, 4.5.2.1.2, 5.2.2, 5.2.17, 5.2.33, 5.2.34, Annex B of IEEE 1278.1-1995 [4].

By combining position/maneuver data with object classification information, the BaseEntity class provides the minimum set of attributes needed to visualize an object in the virtual world. The EntityType structure **shall** use DIS Entity Type enumerations to provide each object classification with a unique identifier. In addition to supporting discrete physical entities, this class also forms the basis for aggregations (like platoons and battle groups) and other classeswhich require basic position/maneuver data. An overview of the BaseEntity attributes is provided in Table 6-2.

All publishers of this class and its subclasses **shall** provide the EntityType, EntityIdentifier, DeadReckoningAlgorithm, Orientation, and Position attributes. The IsFrozen attribute **shall** be treated as an optional field. The remaining attributes may be required to provide input data to the particular DeadReckoningAlgorithm selected for this object. If not required by DeadReckoningAlgorithm, these "conditional" attributes **shall** be treated as optional fields.

Table 6-2 BaseEntity Attributes

Attribute Name	DIS PDU	DIS Field	Definition
AccelerationVector	Entity State	Dead Reckoning Parameters: Linear Acceleration	Acceleration, for the object, in a coordinate system (body or world) specified by the Dead Reckoning Algorithm.
AngularVelocityVector	Entity State	Dead Reckoning Parameters: Angular Velocity	Rate of change for the object's orientation in a coordinate system (body or world) specified by the Dead Reckoning Algorithm.

Attribute Name	DIS PDU	DIS Field	Definition
DeadReckoning Algorithm	Entity State	Dead Reckoning Parameters: Dead Reckoning Algorithm	Dead Reckoning Algorithm specified as an enumeration.
EntityType	Entity State	Entity Type	Kind, Country, Domain, Category, Subcategory, Specific, and Extra fields of the DIS Entity Type.
EntityIdentifier	Entity State	Entity ID	Identifies the site, application, and entity number of this object. It is used for group addressing in the SIMAN interactions.
IsFrozen	Entity State	Entity Appearance (bits)	True if object has been frozen by the simulation. Frozen entities should not be dead-reckoned. They should instead be displayed as fixed at the current location even if non-zero velocity, acceleration or rotation data received from the frozen entity.
Orientation	Entity State	Entity Orientation	Object orientation relative to the world coordinate system. Supports the computation of the Dead Reckoning Algorithm.
WorldLocation	Entity State	EntityLocation	Location for the object relative to the DIS world coordinate system. The shape of the earth **shall** be specified using WGS 84. The origin of the coordinate system **shall** be the centroid of the earth.
VelocityVector	Entity State	EntityLinearVelocity	Velocity, for the object, in a coordinate system (body or world) specified by the Dead Reckoning Algorithm.

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
AccelerationVector	all zeros	Section 5.3.3.1.k.3
AngularVelocityVector	all zeros	Section 5.3.3.1.k.4
DeadReckoning Algorithm	MANDITORY FIELD	Section 5.3.3.1.k.1
EntityType	MANDITORY FIELD	Section 5.3.3.1.e
EntityIdentifier	MANDITORY FIELD	Section 5.3.3.1.b
IsFrozen	False	Section 5.3.3.1.j
Orientation	MANDITORY FIELD	Section 5.3.3.1.i
WorldLocation	MANDITORY FIELD	Section 5.3.3.1.h
VelocityVector	all zeros	Section 5.3.3.1.g

As shown in this table, the attributes for the BaseEntity class were all derived from corresponding values in the DIS Entity State PDU. Although objects are generally not instantiated using the BaseEntity class directly, BaseEntity can act as a useful subscription level for applications where basic data is required for nearly all entities in the simulation (e.g. visualization tools).

The RPR FOM does not support one of the Dead Reckonning fields included in the DIS Entity State PDU. The DIS dead reckonning parameters include 120 bits to represent "user-defined" dead reckonning attributes not covered in the basic protocol. The RPR FOM mechanism for supporting this functionality is for individual federations to extend the BaseEntity class to support additional attributes that may be required.

6.1.1 PhysicalEntity Class

Objects that can be treated as discrete simulation participants are derived from the PhysicalEntity class. This class tailors the behavior of the BaseEntity to include both articulated parts and several object status attributes. The status attributes describe the current condition for those capabilities and states generally available to a large variety of physical entities. Articulated parts are attached components of the entity that may exhibit independent motion (such as landing gear or gun turrets). All of the attributes shown in Table 6-3 **shall** be treated as optional fields for publishers of this class and its subclasses.

Table 6-3 PhysicalEntity Attributes

Attribute Name	DIS PDU	DIS Field	Definition
AlternateEntityType	Entity State	Alternate Entity Type	Guise function. Allows both sides of an engagement to see their own team members as "friendly force" and their opponents as hostile. The force ID field is used to determine team membership.
ArticulatedParameters Array	Entity State	Articulation Parameters	The specification of articulation parameters for moveable parts and attached parts of an entity.
CamouflageType	Entity State	Entity Appearance (bits)	Describes the type of camouflage used on the entity.
DamageState	Entity State	Entity Appearance (bits)	Describes the damaged appearance of an entity.
EngineSmokeOn	Entity State	Entity Appearance (bits)	True if entity is creating engine smoke.
FirePowerDisabled	Entity State	Entity Appearance (bits)	True if an entity's fire power has been disabled.
FlamesPresent	Entity State	Entity Appearance (bits)	True if entity is aflame.
ForceIdentifier	Entity State	Force ID	Enumeration distinguishing the different teams or sides in an exercise.
HasAmmunition SupplyCap	Entity State	Capabilities (bits)	The Entity is able to supply some type of ammunition.
HasFuelSupplyCap	Entity State	Capabilities (bits)	The Entity is able to supply some type of fuel.
HasRecoveryCap	Entity State	Capabilities (bits)	The Entity is able to provide recovery (e.g. towing).
HasRepairCap	Entity State	Capabilities (bits)	The Entity is able to supply repair services.
Immobilized	Entity State	Entity Appearance (bits)	True if the entity has been immobilized (mobility kill).
IsConcealed	Entity State	Entity Appearance (bits)	True if entity is concealed.
Marking	Entity State	Entity Marking	Character set and the string of characters used to provide display identification for

Attribute Name	DIS PDU	DIS Field	Definition
			this entity.
PowerPlantOn	Entity State	Entity Appearance (bits)	True if entity's power plant is on.
SmokePlumePresent	Entity State	Entity Appearance (bits)	True if entity is creating a smoke plume.
TentDeployed	Entity State	Entity Appearance (bits)	True if entity's tent is deployed.
TrailingEffectsCode	Entity State	Entity Appearance (bits)	True if entity is creating a smoke trail.

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
AlternateEntityType	BaseEntity.EntityType	Section 5.3.3.1.f
ArticulatedParametersArray	No Articulated Parameters	Section 5.2.5, Section 5.3.3.1.n, Annex A
CamouflageType	UniformPaintScheme	Section 5.3.3.1.j
DamageState	NoDamage	Section 5.3.3.1.j
EngineSmokeOn	False	Section 5.3.3.1.j
FirePowerDisabled	False	Section 5.3.3.1.j
FlamesPresent	False	Section 5.3.3.1.j
ForceIdentifier	Other	Section 5.3.3.1.c
HasAmmunitionSupplyCap	False	Section 5.2.13, Section 5.3.3.1.m
HasFuelSupplyCap	False	Section 5.2.13, Section 5.3.3.1.m
HasRecoveryCap	False	Section 5.2.13, Section 5.3.3.1.m
HasRepairCap	False	Section 5.2.13, Section 5.3.3.1.m
Immobilized	False	Section 5.3.3.1.j
IsConcealed	False	Section 5.3.3.1.j
Marking	No Marking	Section 5.2.15, Section 5.3.3.1.l
PowerPlantOn	False	Section 5.3.3.1.j
SmokePlumePresent	False	Section 5.3.3.1.j
TentDeployed	False	Section 5.3.3.1.j
TrailingEffectsCode	None	Section 5.3.3.1.j

The PhysicalEntity class was designed to incorporate most of the DIS Entity State PDU attributes not associated with BaseEntity (see Section 5.3.3.1 and Annex A of IEEE 1278.1-1995 [4]). This combination of features provides PhysicalEntity with the minimum number of attributes needed to represent discrete entities.

6.1.1.1 Platform Class

The Platform class is a specialization of PhysicalEntity used to describe status information for vehicles, ships, and aircraft. All of its attributes are derived from the DIS EntityStatePDU (see Section 5.3.3.1 of IEEE 1278.1-1995 [4]). All of the attributes shown in Table 6-4 **shall** be treated optional fields for publishers of this class and its subclasses.

Table 6-4 Platform Attributes

Attribute Name	DIS PDU	DIS Field	Definition
AfterburnerOn	Entity State	EntityAppearance (bits)	True if entity's afterburner is on.
AntiCollisionLightsOn	Entity State	Entity Appearance (bits)	True if anti-collision lights are on.
BlackOutBrakeLightsOn	Entity State	Entity Appearance (bits)	True if blackout brake lights are on.
BlackOutLightsOn	Entity State	Entity Appearance (bits)	True if blackout lights are on.
BrakeLightsOn	Entity State	Entity Appearance (bits)	True if brake lights are on.
FormationLightsOn	Entity State	Entity Appearance (bits)	True if formation lights are on.
HatchState	Entity State	Entity Appearance (bits)	Describes the state of the primary hatch.
HeadLightsOn	Entity State	Entity Appearance (bits)	True if headlights are on.
InteriorLightsOn	Entity State	Entity Appearance (bits)	True if interior lights are on.
LandingLightsOn	Entity State	Entity Appearance (bits)	True if landing lights are on.
LauncherRaised	Entity State	EntityAppearance (bits)	True if entity's launcher is raised.
NavigationLightsOn	Entity State	Entity Appearance (bits)	True if navigation lights are on.
RampDeployed	Entity State	Entity Appearance (bits)	True if entity's ramp is deployed.
RunningLightsOn	Entity State	Entity Appearance (bits)	True if running lights are on.
SpotLightsOn	Entity State	Entity Appearance (bits)	True if spot lights are on.
TailLightsOn	Entity State	Entity Appearance (bits)	True if tail lights are on.

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
AfterburnerOn	False	Section 5.3.3.1.j
AntiCollisionLightsOn	False	Section 5.3.3.1.j
BlackOutBrakeLightsOn	False	Section 5.3.3.1.j
BlackOutLightsOn	False	Section 5.3.3.1.j
BrakeLightsOn	False	Section 5.3.3.1.j
FormationLightsOn	False	Section 5.3.3.1.j
HatchState	NotApplicable	Section 5.3.3.1.j
HeadLightsOn	False	Section 5.3.3.1.j
InteriorLightsOn	False	Section 5.3.3.1.j
LandingLightsOn	False	Section 5.3.3.1.j
LauncherRaised	False	Section 5.3.3.1 j
NavigationLightsOn	False	Section 5.3.3.1.j
RampDeployed	False	Section 5.3.3.1.j
RunningLightsOn	False	Section 5.3.3.1.j
SpotLightsOn	False	Section 5.3.3.1.j
TailLightsOn	False	Section 5.3.3.1.j

Most of these attributes for this class have been converted from the Entity Appearance in Section 4.3 of reference [8]. DIS only applied many of these only to a particular domain. To maximize interoperability, publishers of this data **shall** limit their use to those indicated by a "yes" in Table 6-5. This table makes this guidance easier to follow than incorporating the domain use into Table 6-4.

Table 6-5 Domain Appropriateness for Platform Attributes

Attribute Name	Aircraft	Amph. Vehicle	Ground Vehicle	Spacecraft	Surface Vessel	Submers. Vessel	Multi. Platform
AfterburnerOn	Yes						Yes

AntiCollisionLightsOn	Yes						Yes
BlackOutBrakeLightsOn		Yes	Yes				Yes
BlackOutLightsOn		Yes	Yes				Yes
BrakeLightsOn		Yes	Yes				Yes
FormationLightsOn	Yes						Yes
HatchState		Yes	Yes			Yes	Yes
HeadLightsOn		Yes	Yes				Yes
InteriorLightsOn	Yes	Yes	Yes		Yes		Yes
LandingLightsOn	Yes						Yes
LauncherRaised							Yes
NavigationLightsOn	Yes						Yes
RampDeployed							Yes
RunningLightsOn		Yes			Yes	Yes	Yes
SpotLightsOn	Yes	Yes	Yes		Yes		Yes
TailLightsOn		Yes	Yes				Yes

6.1.1.1.1 Aircraft Class

This class provides an attributeless sub-class of Platform used to support DM filtering. It is equivalent to the DIS Air domain in that it represents platform entities such as airplanes, balloons, etc. that operate mainly in the air, but that include some limited land operations. This class is publishable because it qualifies as a leaf node of the RPR FOM.

6.1.1.1.2 AmphibiousVehicle Class

This class provides an attributeless sub-class of Platform used to support DM filtering. . It is equivalent to a cross between DIS Land and DIS Surface domains. It represents platforms that can operate both on the land and the sea. This class is publishable because it qualifies as a leaf node of the RPR FOM.

6.1.1.1.3 GroundVehicle Class

This class provides an attributeless sub-class of Platform used to support DM filtering. . It is equivalent to the DIS Land domain in that it represents platforms that operate wholly on the surface of the earth. This class is publishable because it qualifies as a leaf node of the RPR FOM.

6.1.1.1.4 Spacecraft Class

This class provides an attributeless sub-class of Platform used to support DM filtering. . It is equivalent to the DIS Space domain in that it represents platforms that operate mainly in space. This class is publishable because it qualifies as a leaf node of the RPR FOM.

6.1.1.1.5 SurfaceVessel Class

This class provides an attributeless sub-class of Platform used to support DM filtering. . It is equivalent to the DIS Surface domain in that it represents platforms that operate wholly on the surface of the sea. This class is publishable because it qualifies as a leaf node of the RPR FOM.

6.1.1.1.6 SubmersibleVessel Class

This class provides an attributeless sub-class of Platform used to support DM filtering. . It is equivalent to the DIS Subsurface domain in that it represents platforms that operate either on the surface of the sea, or beneath it. This class is publishable because it qualifies as a leaf node of the RPR FOM.

6.1.1.1.7 MultiDomainPlatform Class

This class provides an attributeless sub-class of Platform used to support DM filtering. . It is equivalent to the DIS Other domain in that it represents platforms that operate in more than one domain (excluding those combinations explicitly defined as other subclasses Platform). This class is publishable because it qualifies as a leaf node of the RPR FOM.

6.1.1.2 Lifeform Class

The Lifeform class is a specialization of PhysicalEntity used to describe individual s. Like munitions, the DIS use for individual s was normally limited to special cases when they could not be addressed as part of a larger object such as an aggregation or a vehicle. All Lifeform attributes are derived from the DIS Entity State PDU. All attributes shown in Table 6-6 are optional for publishers of this class and its subclasses.

Table 6-6 Lifeform Attributes

Attribute Name	DIS PDU	DIS Field	Definition
FlashLightsOn	Entity State	Entity Appearance (bits)	True if flash lights are on.
PersonStanceCode	Entity State	Entity Appearance (bits)	Human behaviors (i.e., running, jumping, etc).
PrimaryWeaponState	Entity State	Capabilities (bits)	Describes the state of the Human's primary weapon.
SecondaryWeaponState	Entity State	Capabilities (bits)	Describes the state of the Human's secondary weapon.

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
FlashLightsOn	False	Section 5.3.3.1.j
PersonStanceCode	NotApplicable	Section 5.3.3.1.j
PrimaryWeaponState	NoWeapon	Section 5.2.13, Section 5.3.3.1.m

SecondaryWeaponState	NoWeapon	Section 5.2.13, Section 5.3.3.1.m

6.1.1.2.1 Human Class

This class provides an attributeless sub-class of the Lifeform used to support DM filtering. This class is publishable because it qualifies as a leaf node of the RPR FOM.

6.1.1.2.2 NonHuman Class

This class provides an attributeless sub-class of the Lifeform used to support DM filtering. This class is publishable because it qualifies as a leaf node of the RPR FOM.

6.1.1.3 Sensor Class

The Sensor class is a sub-class of PhysicalEntity used to describe the physical characteristics (location, appearance, etc.) of sensor installations such as radars. This class is publishable because it qualifies as a leaf node of the RPR FOM.

Table 6-7 Sensor Attributes

Attribute Name	DIS PDU	DIS Field	Definition
AntennaRaised	Entity State	Entity Appearance (bits)	True if the antenna has been raised.
BlackoutLightsOn	Entity State	Entity Appearance (bits)	True if blackout lights are on.
InteriorLightsOn	Entity State	Entity Appearance (bits)	True if interior lights are on.
LightsOn	Entity State	Entity Appearance (bits)	True if other lights are on.
MissionKill	Entity State	Entity Appearance (bits)	True if mission capability is disabled (e.g. dameaged antenna)

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
AntennaRaised	False	Section 5.3.3.1.j
BlackoutLightsOn	False	Section 5.3.3.1.j
InteriorLightsOn	False	Section 5.3.3.1.j
LightsOn	False	Section 5.3.3.1.j
MissionKill	False	Section 5.3.3.1.j

6.1.1.4 Radio Class

The Radio class is an attributeless sub-class of PhysicalEntity used to describe the physical characteristics (location, appearance, etc.) of radio installations. It currently has no attributes. This class is publishable because it qualifies as a leaf node of the RPR FOM.

6.1.1.5 Munition Class

The DIS protocols allowed for two types of munition classes. In general, small munitions were tracked at just the launch and impact points using the Fire PDU and Detonate PDU. Simulation developers also had the option of tracking weapons (torpedoes, missiles, etc.) throughout their transit by treating them as independent entities. This latter approach to munition representation was utilized if the representation of its travel between firing and detonation could affect the outcome of the simulation.

The Munition class is used to describe the attributes of munitions that act as independent entities. Capabilities equivalent to the DIS Fire PDU and Detontate PDU are now provided by the WeaponFire and MunitionDetonation interactions (see Section 7.2). All of the attributes shown in Table 6-8 **shall** be treated as optional fields for publishers of this class and its subclasses.

Table 6-8 Munition Attributes

Attribute Name	DIS PDU	DIS Field	Definition
LauncherFlashPresent	Entity State	Entity Appearance (bits)	True if launcher flash is present when munition is fired.

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
LauncherFlashPresent	False	Section 5.3.3.1.j

This class is publishable because it qualifies as a leaf node of the RPR FOM.

6.1.1.6 CulturalFeature Class

The CulturalFeature class is a sub-class of PhysicalEntity used to describe the physical characteristics of engineering and natural effects such as buildings, craters, bridges, and vehicle tracks. This class is publishable because it qualifies as a leaf node of the RPR FOM.

Table 6-9 CulturalFeature Attributes

Attribute Name	DIS PDU	DIS Field	Definition
ExternalLightsOn	Entity State	Entity Appearance (bits)	True if exterior lights are on.
InternalHeatSourceOn	Entity State	Entity Appearance (bits)	True if interior heat source is on (for infrared viewing).
InternalLightsOn	Entity State	Entity Appearance (bits)	True if interior lights are on.

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
ExternalLightsOn	False	Section 5.3.3.1.j
InternalHeatSourceOn	False	Section 5.3.3.1.j
InternalLightsOn	False	Section 5.3.3.1.j

6.1.1.7 Expendables Class

The Expendables class is a sub-class of PhysicalEntity used to describe the physical characteristics of countermeasures devices that are dispensed from another entity. Althought those devices may be active emitters or passive reflectors of energy, emmissions are handled separately. It currently has no attributes. This class is publishable because it qualifies as a leaf node of the RPR FOM.

6.1.1.8 Supplies Class

The Supplies class is a sub-class of PhysicalEntity used to describe the physical characteristics of supplies other than munitions, such as fuel, food and personnel. It currently has no attributes. This class is publishable because it qualifies as a leaf node of the RPR FOM.

6.1.2 EnvironmentalEntity

The EnvironmentalEntity class provides a compatible method for environmental representation with the DIS 1278.1 1995 [4] and does not provide the more advanced environmental features found in DIS 1278.1A 1998 [5].

The EnvironmentalEntity is a subclass of BaseEntity and as such inherits WorldLocation, Orientation, VelocityVector, and DRA attributes from BaseEntity. These attributes are utilized to locate the environmental entity representation. The inherited EntityType is used to identify the type of EnvironmentalEntity. This subclass provides an additional parameter, OpacityCode, that can be utilized to vary the density of a smoke EnvironmentalEntity. This class has no required attributes.

Table 6-10 EnvironmentalEntity

Attribute Name	DIS PDU	DIS Field	Definition
OpacityCode	Entity State PDU	Entity Appearance	Enumeration defining the density of a smoke object.

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
OpacityCode	Clear	Section 5.3.3.1.j

6.2 EmbeddedSystem

The EmbeddedSystem class provides a mecahism for associating system capabilities with a physical object. The assoication between an EmbeddedSystem and its host provides a means of aggregating many capabilities into a single platform without resorting to multiple inheritance. The attributes listed in Table 6-11 establish the host identity and describe the positional relationship between this entity and the host. This class has no optional parameters; the publisher **shall** provide all attributes specified in Table 6-11.

Table 6-11 EmbeddedSystem Attributes

Attribute Name	DIS PDU	DIS Field	Definition
EntityIdentifier	N/A	N/A	Identifies the site, application, and entity number of the host to which this object is attached.
HostObject Identifier	Electromagnetic Emissions or Designator or Transmitter or Receiver	Emitting Entity ID Designating Entity ID Entity ID Entity ID	Object ID of the host to which this object is attached.
RelativePosition	Electromagnetic Emissions or Transmitter	Location Relative Antenna Location	Location of the embedded system with respect to the host's coordinate system.

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
EntityIdentifier	MANDITORY FIELD	Section 5.3.7.1.b, Section 5.3.7.2.b, Section 5.3.8.1.b, Section 5.3.8.3.b
HostObject Identifier	MANDITORY FIELD	Section 5.3.7.1.b, Section 5.3.7.2.b, Section 5.3.8.1.b, Section 5.3.8.3.b
RelativePosition	MANDITORY FIELD	Section 5.3.7.1.e.4, Section 5.3.8.1

Embedded systems differ from articulated parts in that the non-visual elements of the embedded system (emissions/detections) are generally their primary simulation feature. The EmbeddedSystem class is used to specify the association information found in DIS PDUs such as the Electromagnetic Emissions, Designator, Radio Transmitter, and Radio Receiver. Attributes specific to the object's emission/detection role are provided by the sub-classes. Defining embedded systems as separate objects closely matches their use in DIS.

6.2.1 RadioTransmitter

This class provides electromagnetic properties of radio transmitting systems for the purpose of both simulated radio reception and electronic warfare. Four types of fields are incorporated in this class: state/identification, eletromagnetic characteristics, modulation, and cryptography. Each publisher of this class **shall** provide the state/identification fields RadioIndex, RadioSystemType, and TransmitterOperationalStatus. The publisher **shall** guarantee that a unique RadioIndex / HostObjectIdentifier combination is provided for each radio instance. The one remaining state/identification field, RadioInputSource, **shall** be treated as an optional field. The eletromagnetic characteristics data consists of the WorldLocation, AntennaPatternType, AntennaPatternData, EmissionFrequency, FrequencyBandwidth, and TransmittedPower attributes. All of eletromagnetic characteristics **shall** be provided by the publisher except AntennaPatternData, which is only required for non-OmniDirectional pattern types. All of the modulation, and cryptography attributes **shall** be treated as optional fields for publishers of this class and its subclasses.

Table 6-12 RadioTransmitter Attributes

Attribute Name	DIS PDU	DIS Field	Definition
AntennaPattern Data	Transmitter	Antenna Pattern Parameters	Specifies the radiation pattern from the antenna.
CryptographicMode	Transmitter	Crypto Key ID	Indicates baseband or diphase mode.
Cryptographic System	Transmitter	Crypto System	Identifies the cryptographic equipment used.
Encryption KeyIdentifier	Transmitter	Crypto Key ID	Key identifier number. The transmitter and receiver should be considered to be using the same key if these numbers match.
Frequency	Transmitter	Frequency	Center frequency of the radio transmissions.
Frequency Bandwidth	Transmitter	Transmit Frequency Bandwidth	Bandpass of the radio transmissions.
Frequency HopInUse	Transmitter	Modulation Type: Spread Spectrum: Time Hop	True if a frequency hop transmit algorithm is in use.
Modulation Parameters	Transmitter	Modulation Parameters	Set of modulation parameters whose interpretation is based on the Modulation System Type.
PsuedoNoise SpectrumInUse	Transmitter	Modulation Type: Spread Spectrum: Pseudo Noise	True if a psuedo-noise transmit algorithm is in use.
RadioIndex	Transmitter	Radio ID	Specifies the identification number for a each receiver on a given host. This ID **shall not** change during an exercise.
RadioInputSource	Transmitter	Input Source	Specifies which position (pilot, gunnery officer, etc.) or data port provided the original source of this information.
RadioSystemType	Transmitter	Radio Entity Type	Kind, Country, Domain, Category, Nomenclature Version, and Nomenclature of the DIS Radio Type. This ID **shall not** change during an exercise.
RFModulation SystemType	Transmitter	Modulation Type: System	Specifies the interpretation of the Modulation Parameters.
RFModulationType	Transmitter	Modulation Type: Major	Classification of the modulation type
TimeHopInUse	Transmitter	Modulation Type: Spread Spectrum: Time Hop	True if a time hop transmit algorithm is in use.
TransmittedPower	Transmitter	Power	Average transmitted power.

Attribute Name	DIS PDU	DIS Field	Definition
Transmitter OperationalStatus	Transmitter	Transmit State	On/Off state of the receiver as an enumeration.
WorldLocation	Transmitter	Antenna Location	Location of the antenna in world coordinates.

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
AntennaPatternData	Not required if using OmniDirectional source.	Section 5.3.8.1.i,
CryptographicMode	BasebandEncryption	Section 5.3.8.1.o
CryptographicSystem	Other	Section 5.3.8.1.n
EncryptionKeyIdentifier	None Assumed	Section 5.3.8.1.o
Frequency	MANDITORY FIELD	Section 5.3.8.1.j
FrequencyBandwidth	MANDITORY FIELD	Section 5.3.8.1.k
FrequencyHopInUse	False	Section 5.3.8.1.m
ModulationParameters	None Assumed.	Section 5.3.8.1.q
PsuedoNoiseSpectrumInUse	False	Section 5.3.8.1.m
RadioIndex	MANDITORY FIELD	Section 5.3.8.1.c
RadioInputSource	Other	Section 5.3.8.1.f
RadioSystemType	MANDITORY FIELD	Section 5.3.8.1.d
RFModulationSystemType	Other	Section 5.3.8.1.m
RFModulationType	Other	Section 5.3.8.1.m
TimeHopInUse	False	Section 5.3.8.1.m
TransmittedPower	MANDITORY FIELD	Section 5.3.8.1.l
TransmitterOperationalStatus	MANDITORY FIELD	Section 5.3.8.1.e
WorldLocation	MANDITORY FIELD	Section 5.3.8.1.g

This class is intended to describe only the radio gear itself and not the contents of any messages carried by this system. The contents of the messages transmitted by the radio are provided as discrete events by the RadioSignal interaction. The attributes for this class are derived from the Transmitter PDU as described in IEEE 1278.1-1995 [4] Section 5.3.8.1.

The RadioIndex field is used principly for conversion between HLA and DIS. See Section 7.3.1 for details on the mapping between the RPR FOM and DIS addressing schemes for radio numbers.

This class continues to follow a DIS RadioTransmitters positioning model that relies on two WorldLocation fields. The first version of position data is be derived from the WorldLocation of the host entity, a second version of WorldLocation is includes as part of the RadioTransmitter itself. The RadioTransmitter's WorldLocation is intended to be a low-resolution value that is not dead reckoned. It provides a basis by which RadioTransmitters can be geographically filtered out of a simulation without having to process the host's information. Once engaged, the host entity's WorldLocation, and the RelativePosition data from EmbeddedSystem are used to fine-tune the radio's position.

6.2.2 RadioReceiver

This class provides state information for a particular radio receiver in order to support radio network monitors, data loggers, and similar applications for use in debugging, supervision, and after-action review. The receiver's publisher **shall** always provide the RadioIndex and ReceiverOperationalStatus. The publisher **shall** guarantee that a unique RadioIndex / HostObjectIdentifier combination is provided for each radio instance. The remaining parameters **shall** also be provided whenever the radio is in a receiving state.

Table 6-13 RadioReceiver Attributes

Attribute Name	DIS PDU	DIS Field	Definition
RadioIndex	Receiver	Radio ID	Specifies the identification number for a each receiver on a given host. This ID **shall not** change during an exercise.
Receiver Operational Status	Receiver	Receiver State	On/Off state of the receiver as an enumeration.
ReceivedPower	Receiver	Received Power	RF power received after applying any propagation loss and antenna gain.
Received Transmitter Identifier	Receiver	Transmitter	RTI Id of the transmitter currently being received.

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
RadioIndex	Receiver	Section 5.3.8.3.c
ReceiverOperational Status	Receiver	Section 5.3.8.3.d
ReceivedPower	zero	Section 5.3.8.3.e
ReceivedTransmitter Identifier	empty	Section 5.3.8.3.g

This class is intended to describe only the radio gear itself and not the contents of any messages carried by this system. The contents of the messages transmitted by the radio are provided as discrete events by the RadioSignal interaction. The attributes for this class are derived from the Receiver PDU as described in IEEE 1278.1-1995 [4] Section 5.3.8.3.

Strictly speaking, the RadioIndex field is only needed for conversion between HLA and DIS. See Section 7.3.1 for details on the mapping between the RPR FOM and DIS addressing schemes for radio numbers.

6.2.3 Designator

The Designator class is used to describe the behaviors of targeting system illuminations such as those used in laser-guided weapon engagement. The Designator provides a representation at the target site (e.g. the laser spot on a target) instead of the source emission system (e.g. the laser targeting system itself). If the publisher does not supply the DesignatedObjectIdentifier object field, the default behavior **shall** treat the designator as if it was not located on an object.

In addition to supporting a location in the World Coordinate System, this class is also capable of dead reckoning the relative spot location. The use of the spot dead reckoning algorithms provided for within the IEEE 1278.1 specification is not widespread and its use is not recommended. The spot dead reckoning parameters provided are not believed to be broadly applicable as discontinuous spot translation is possible creating infinite accelerations. Spot dead reckoning is implemented exactly as described in DIS. It has been identified that this may not provide a complete solution; however, it preserves compatibility and consistency with DIS.

The DesignatedObjectIdentifier, DeadReckoningAlgorithm, RelativeSpotLocation, and SpotLinearAccelerationVector shall be treated as optional fields. All other fields in Table 6-14 are mandatory and **shall** be provided by the publisher.

Table 6-14 Designator Attributes

Attribute Name	DIS PDU	DIS Equivalent	Definition
CodeName	Designator	Code Name	Identifies the code name for the designator system.
Designated ObjectIdentifier	Designator	Designated Entity ID	RTI Object ID of the entity that is currently being designated.
DesignatorCode	Designator	Designator Code	This attribute identifies the designator code being used by the designating entity.
Designator OutputPower	Designator	Designator Power	Designator output power, in watts.
Designator SpotLocation	Designator	Designator Spot Location	Location of the Designator Spot in DIS World Coordinate System.
Designator Emission Wavelength	Designator	Designator Wavelength	DesignatorEmissionWavelength, in microns.
DeadReckoning Algorithm	Designator	DRAlgorithm	Algorithm used to dead reckon the position of the designator spot.
Relative SpotLocation	Designator	Designator Spot With Respect to Designated Entity	Designator spot with respect to the designated entity's coordinate system when the spot is on an entity.
SpotLinear Acceeration Vector	Designator	Entity Linear Acceleration	The linear acceleration used to dead reckon the position of the designator spot.

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
CodeName	MANDITORY FIELD	Section 5.3.7.2.c
DesignatedObjectIdentifier	empty	Section 5.3.7.2.d
DesignatorCode	MANDITORY FIELD	Section 5.3.7.2.e
DesignatorOutputPower	MANDITORY FIELD	Section 5.3.7.2.f
DesignatorSpotLocation	MANDITORY FIELD	Section 5.3.7.2.i
DesignatorEmission Wavelength	MANDITORY FIELD	Section 5.3.7.2.g
DeadReckoningAlgorithm	Static	Section 5.3.7.2.j.1
Relative SpotLocation	all zeros	Section 5.3.7.2.h

SpotLinearAcceleration Vector	all zeros	Section 5.3.7.2.j.2

The attributes for this class are derived from the Designator PDU as described in IEEE 1278.1-1995 [4] Section 5.3.7.2.

6.2.4 EmitterSystem

An EmitterSystem provides electromagnetic properties of radars, jammers, and other electronic warfare (EW) systems not covered elsewhere in the EmbeddedSystem hierarchy. This class has no optional parameters; the publisher **shall** provide all of the attributes specified in Table 6-15.

Table 6-15 EmitterSystem Attributes

Attribute Name	DIS PDU	DIS Field	Definition
EmitterType	Electromagnetic Emissions	Emitter System: Emitter Name	EmitterType specified as an enumeration.
Emitter FunctionCode	Electromagnetic Emissions	Emitter System: Function	Specifies the function for a particular emitter as an enumeration.
Emitter Number	Electromagnetic Emissions	Emitter ID Number	Specifies the identification number for a each emitter system on a given host. This ID **shall not** change during an exercise.
EventIdentifier	N/A	N/A	Used by the generating federate to associate EmitterSystem and EmitterBeam changes.

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
EmitterType	MANDITORY FIELD	Section 5.2.11, Section 5.3.7.1.e.3
Emitter FunctionCode	MANDITORY FIELD	Section 5.2.11, Section 5.3.7.1.e.3
Emitter Number	MANDITORY FIELD	Section 5.2.11, Section 5.3.7.1.e.3
EventIdentifier	MANDITORY FIELD	none

The attributes for this class are derived from the Electromagnetic Emission (EE) PDU as described in IEEE 1278.1-1995 [4] Section 5.3.7.1. Unlike the DIS structure, the RPR FOM divides the emission into two parts: an emitter system, and a series of emitter beams. The EmitterSystem class represents the properties of the electromagnetic system itself while the EmitterBeam class provides the beam data. (Although radio transmitters are sometimes used as detectable EW systems, the RadioTransmitter class is actually the proper EmbeddedSystem class for those objects.)

The EventIdentifier allows correlation of EmitterSystem and EmitterBeam data. Each change in the electromagnetic emission characteristics updates the EventIdentifier in both classes. The composite of these two emitter compoents is then re-assembled on the receiving side.

6.3 EmitterBeam

Emitter beams define the electromagnetic characteristics of the emission emanating from an emitter system (see Section 6.2.4). The attributes for this class are derived from the Electromagnetic Emission (EE) PDU as described in IEEE 1278.1-1995 [4] Section 5.3.7.1. The emitter beam attributes describe the fundamental parameter data of the emission. Emitter beams are associated with a specific instance of an emitter system. A reference to the emanating emitter system is required in order to determine the beam's full characteristics. The emitter system is required primarily for spatial correlation, but it may also be required for database lookups. The BeamAzimuthCenter, BeamAzimuthSweep, BeamElevationCenter, BeamElevationSweep, and SweepSynch are optional parameters and **shall** default to the value zero for beam functions where a scan volume does not apply (e.g., target tracking beam). All other fields in Table 6-16 are mandatory and **shall** be provided by the publisher.

Table 6-16 EmitterBeam Attributes

Attribute Name	DIS PDU	DIS Field	Definition
BeamAzimuth Center	Electromagnetic Emissions	Beam Azimuth Center	This attribute specifies the azimuth center angle of the beam's scan-volume relative to the emitter system. This attribute in conjunction with BeamElevationCenter, azimuth sweep, and BeamElevationSweep describe the scan volume covered by the emitter beam scan.
BeamAzimuth Sweep	Electromagnetic Emissions	Beam Azimuth Sweep	This attribute specifies the azimuth sweep of the beam's scan-volume relative to the azimuth center. This attribute in conjunction with BeamElevationCenter, azimuth sweep, and BeamElevationSweep describe the scan volume covered by the emitter beam scan.
BeamFunction Code	Electromagnetic Emissions	Beam Function	This enumerated attribute specifies the beam's function. It serves as a general data filter.
BeamIdentifier	Electromagnetic Emissions	Beam ID Number	This attribute specifies a unique database number assigned to differentiate between otherwise similar or identical emitter beams within an emitter system.
Beam Parameter Index	Electromagnetic Emissions	Beam Parameter Index	This attribute specifies a beam parameter index number that **shall** be used by receiving entities in conjunction with the emitter name attribute (EmitterSystem object class) to provide a pointer to the stored database parameters required to regenerate the beam.

Attribute Name	DIS PDU	DIS Field	Definition
BeamElevation Center	Electromagnetic Emissions	Beam Elevation Center	This attribute specifies the elevation center angle of the beam's scan-volume relative to the emitter system. This attribute in conjunction with BeamElevationCenter, azimuth sweep, and BeamElevationSweep describe the scan volume covered by the emitter beam scan.
BeamElevation Sweep	Electromagnetic Emissions	Beam Elevation Sweep	This attribute specifies the elevation sweep of the beam's scan-volume relative to the BeamElevationCenter. This attribute in conjunction with BeamElevationCenter, azimuth sweep, and elevation sweep describe the scan volume covered by the emitter beam scan.
EmitterSystem Identifier	Electromagnetic Emissions	N/A	This attribute specifies a reference to the emitter system object from which the beam is emanating.
Effective RadiatedPower	Electromagnetic Emissions	ERP	This attribute specifies the EffectiveRadiatedPower for the emission in dBm. For a radar or a noise jammer, this attribute **shall** indicate the peak of the transmitted power. Thus, it includes peak transmitter power, transmission line losses, and peak of the antenna gain.
EventIdentifier	N/A	N/A	Used by the generating federate to associate EmitterSystem and EmitterBeam changes.
Emission Frequency	Electromagnetic Emissions	Frequency	This attribute specifies the frequency of the emission in hertz. Frequency modulation for a particular emitter and mode **shall** be derived from database parameters stored in the receiving entity.
Frequency Range	Electromagnetic Emissions	Frequency Range	This attribute specifies the bandwidth of the frequencies corresponding to the Frequency attribute. Thus, if, for operational purposes, the Frequency is supposed to be a single number, then the Frequency Range **shall** be zero.
PulseRepetition Frequency	Electromagnetic Emissions	PRF	This attribute specifies the average PulseRepetitionFrequency of the emission in hertz. PulseRepetitionFrequency modulation for a particular emitter and mode **shall** be derived from database parameters stored in the receiving entity.
PulseWidth	Electromagnetic Emissions	Pulse Width	This attribute specifies the average pulse width of the emission in microseconds. Pulse modulation for a particular emitter and mode **shall** be derived from database parameters stored in the receiving entity.
SweepSynch	Electromagnetic Emissions	Beam Sweep SYNC	This attribute is provided to allow a receiver to synchronize its regenerated scan pattern to that of the emitter. This attribute when employed specifies the percentage of time a scan is through its patter from its origin. The pattern and origin data are derived from database parameters.

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
BeamAzimuth Center	zero	Section 5.2.22, Section 5.3.7.1.3.4.v
BeamAzimuth Sweep	zero	Section 5.2.22, Section 5.3.7.1.3.4.v
BeamFunctionCode	MANDITORY FIELD	Section 5.3.7.1.3.4.iv
BeamIdentifier	MANDITORY FIELD	Section 5.3.7.1.3.4.ii
BeamParameter Index	MANDITORY FIELD	Section 5.3.7.1.3.4.iii
BeamElevation Center	zero	Section 5.2.22, Section 5.3.7.1.3.4.v
BeamElevation Sweep	zero	Section 5.2.22, Section 5.3.7.1.3.4.v
EmitterSystem Identifier	MANDITORY FIELD	Section 5.3.7.1.b
EffectiveRadiated Power	MANDITORY FIELD	Section 5.2.22, Section 5.3.7.1.3.4.v
EventIdentifier	MANDITORY FIELD	none
EmissionFrequency	MANDITORY FIELD	Section 5.2.22, Section 5.3.7.1.3.4.v
Frequency Range	MANDITORY FIELD	Section 5.2.22, Section 5.3.7.1.3.4.v
PulseRepetition Frequency	MANDITORY FIELD	Section 5.2.22, Section 5.3.7.1.3.4.v
PulseWidth	MANDITORY FIELD	Section 5.2.22, Section 5.3.7.1.3.4.v
SweepSynch	zero	Section 5.2.22, Section 5.3.7.1.3.4.v

6.3.1 RadarBeam

The RadarBeam class represents all electromagnetic emitter beams, whose function are not represented in other classes (e.g., JammerBeam or RadioTransmitter). Primary examples would be search, acquisition, and tracking. To support the tracking function, the RadarBeam extends the EmitterBeam with attributes that indicate which simulation objects are being tracked by the beam. If supplied, the TrackObjectIdentifiers attribute **shall** indicate the identity of the targets being tracked. For a single-track emitter systems this field **shall** be used to identify the target the system is tracking. If the system is tracking a target cluster then all the targets in the cluster **shall** be identified in this attribute. The system **shall not** indicate a target(s) in this field if the system determines that the track has been physically offset from the target(s) by jamming. The attributes in this class are optional.

Table 6-17 RadarBeam Attributes

Attribute Name	DIS PDU	DIS Field	Definition
HighDensityTrack	Electromagnetic	High Density Track/Jam	This field is used to indicate whether or not all

	Emissions		targets can be considered within the scan pattern that the emitter can track.
TrackObject Identifiers	Electromagnetic Emissions	Track/Jam	This attribute identifies the targets in an emitter track or emitters a system is attempting to jam.

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
HighDensityTrack	False	Section 5.3.7.1.e.vii
TrackObjectIdentifiers	empty	Section 5.3.7.1.e.viii

6.3.2 JammerBeam

The JammerBeam class represents those electromagnetic emitter beams whose function is jamming other electromagnetic emitter beams. The JammerBeam extends the EmitterBeam with attributes that indicate which simulation objects (radar beams or other jam beams) are being jammed by the beam. If supplied, the JammedObjectIdentifiers attribute **shall** indicate the emitters the system is attempting to jam. The attributes in this class are optional.

Table 6-18 JammerBeam Attributes

Attribute Name	DIS PDU	DIS Field	Definition
HighDensityJam	Electromagnetic Emissions	High Density Track/Jam	This field is used to indicate whether or not all targets can be considered within the scan pattern that the jammer can jam.
JammingMode Sequence	Electromagnetic Emissions	Jamming Mode Sequence	Indicates the jamming mode technique or series of techniques being applied.
Jammed ObjectIdentifiers	Electromagnetic Emissions	Track/Jam	This attribute identifies the targets in an emitter track or emitters a system is attempting to jam.

Attribute Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
HighDensityJam	False	Section 5.3.7.1.e.4.vii
JammingMode Sequence	zero	Section 5.3.7.1.e.4.viii
Jammed ObjectIdentifiers	empty	Section 5.3.7.1.e.4.viii

7 RPR FOM Interaction Structure

Interactions provide a mechanism for objects to transmit discrete events to other simulation participants. In the RPR FOM, these interactions are used to handle simulation management tasks, collisions, munitions, and inter-object communications. In this document, the RPR FOM interactions will be described in terms of interaction "families". Although these families do not actually appear in the FOM structure, they do provide a useful mechanism for discussing common properties between interactions. The remaining sections of this section provide details on each family and class.

Table 7-1 RPR FOM Interaction Class Structure Table

Family	Class 1	Class 2
Entity Interaction	Collision	
Warfare	MunitionDetonation	
	WeaponFire	
Communications	RadioSignal	EncodedAudioRadioSignal
		RawBinaryRadioSignal
		DatabaseIndexRadioSignal
		ApplicationSpecificRadioSignal
Logistics	RepairComplete	
	RepairResponse	
	ResupplyCancel	
	ResupplyOffer	
	ResupplyReceived	
	ServiceRequest	
DIS Simulation Management	Acknowledge	
	ActionRequest	
	ActionResponse	
	Comment	
	CreateEntity	
	Data	
	DataQuery	
	EventReport	
	RemoveEntity	
	SetData	
	StartResume	
	StopFreeze	
HLA Simulation Management	AttributeChangeRequest	
	AttributeChangeResult	
	CreateObjectRequest	
	CreateObjectResult	
	RemoveObjectRequest	
	RemoveObjectResult	
	ActionRequestToObject	
	ActionResponseFromObject	

In the interaction parameters tables that follow, required attributes will be indicated using "**bold**" typesetting. Parameters that may be conditionally required based on the value of other settings will be indicated using "*italic*" typesetting. Optional parameters will be indicated using "normal" typesetting.

7.1 Entity Interaction Family

7.1.1 Collision Interaction

This interaction provides information on collisions between objects. It includes not only identification for the two objects involved, but also data required for damage assessment modeling. There should be a collision interaction issued for each object involved in a collision. If a simulation detects that one of its objects has struck another object, it should issue a collision interaction. If a simulation receives a collision interaction indicating that one of its objects has been struck, it should issue a response collision interaction, as long as it has not already issued one for the same collision event.

The form of this interaction closely follows the layout of the CollisionPDU specified in Section 5.3.3.2 of IEEE 1278.1-1995 [4]. Publishers of this class **shall** provide values for all parameters; there are no optional fields.

Table 7-2 Collision Parameters

Parameter Name	DIS PDU	DIS Field	Definition
Colliding ObjectIdentifier	Collision	Colliding Entity ID	The remote object to which the issuing object has collided.
CollisionLocation	Collision	Location	Relative location with respect to the remote object with which the issuing entity has collided.
CollisionType	Collision	Collision Type	Enumeration for collision type.
EventIdentifier	Collision	Event ID	ID assigned by issuing object to associate related collision events.
IssuingObjectIdentifier	Collision	Issuing Entity ID	The object that had detected the collision and issued the collision interaction.
IssuingObjectMass	Collision	Mass	Mass in kilograms of the issuing object
IssuingObjectVelocity Vector	Collision	Velocity	Velocity of the issuing object at the time the collision is detected.

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
CollidingObject Identifier	MANDITORY FIELD	Section 5.3.3.3.2.c
CollisionLocation	MANDITORY FIELD	Section 5.3.3.3.2.h
CollisionType	MANDITORY FIELD	Section 5.3.3.3.2.e
EventIdentifier	MANDITORY FIELD	Section 5.3.3.3.2.d
IssuingObject Identifier	MANDITORY FIELD	Section 5.3.3.3.2.b

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
IssuingObjectMass	MANDITORY FIELD	Section 5.3.3.3.2.g
IssuingObjectVelocity Vector	MANDITORY FIELD	Section 5.3.3.3.2.f

7.2 Warfare Family

The DIS protocols allowed for two types of munition classes. In general, small munitions were tracked at just the launch and impact points using the FirePDU and DetontatePDU. (Issuing rules for these PDUs are described in Section 4.5.3 of IEEE 1278.1-1995 [4].) Simulation developers also had the option of tracking larger weapon (torpedoes, missiles, etc.) through their transit by treating them as independent entities. The WeaponFire interaction alerts simulation participants of each weapon firing for either type of weapon. WeaponFire contains sufficient information so that the weapon may be tracked off-line without creating a corresponding Munition. The MunitionDetonation interaction alerts simulation participants when the weapon is detonated, and includes information used in battle damage assessment models.

7.2.1 WeaponFire Interaction

The Weapon Fire interaction alerts all simulation participants when a weapon is fired. The interaction **shall** be issued regardless of whether the munition will be tracked off-line, or simulated on-line using a corresponding Munition object. The form of this interaction closely follows the layout of the FirePDU described in Section 5.3.4.1 of IEEE 1278.1-1995 [4]. The “FireControlSolutionRange,” “FireMissionIndex,” “QuantityFired,” “TargetObjectIdentifier,” “MunitionObjectIdentifier” and “RateOfFire” are optional fields; publishers of this class **shall** provide values for all other parameters.

Table 7-3 WeaponFire Parameters

Parameter Name	DIS PDU	DIS Field	Definition
EventIdentifier	Fire	Event ID	ID generated by the firing entity to associate related fire and detonation interactions.
FireControlSolution Range	Fire	Range	Range in meters assumed by firing entity in computing the fire control solution. Zero if range is unknown or inapplicable.
FireMissionIndex	Fire	Fire Mission Index	A unique index to identify the fire mission (used to associated weapon fire interactions in a single fire mission).
FiringLocation	Fire	Location in World Coordinates	The location, in world coordinates, from which the munition was launched.
FiringObjectIdentifier	Fire	Firing Entity ID	Object ID issuing the Weapon Fire Interaction.
FuseType	Fire	BurstDescriptor: Fuse	The fuse as specified by an enumeration.

Parameter Name	DIS PDU	DIS Field	Definition
InitialVelocityVector	Fire	Velocity	Velocity of the fired munition at the point when externally visible effects become apparent (e.g. exhaust plume or muzzle flash).
MunitionObject Identifier	Fire	Munition ID	RTI Object ID of the fired munition, if an object is created. Used only for tracked munitions.
MunitionType	Fire	BurstDescriptor: Munition	Kind, Country, Domain, Category, Subcategory, Specific, and Extra fields of a DIS Entity Type.
QuantityFired	Fire	BurstDescriptor: Quantity	Represents the number of rounds fired in the burst when quantity > 1. Zero otherwise.
RateOfFire	Fire	Burst Descriptor: Rate	Rate of fire in rounds per minute when quantity > 1. One otherwise.
TargetObjectIdentifier	Fire	Target Entity ID	Object ID of the intended target.
WarheadType	Fire	BurstDescriptor: Warhead	The warhead as specified by a 16-bit enumeration(see Section 5 in EBV-DOC)

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
EventIdentifier	MANDITORY FIELD	Section 5.3.4.1.e
FireControlSolution Range	zero	Section 5.3.4.1.j
FireMissionIndex	zero	Section 5.3.4.1.f
FiringLocation	MANDITORY FIELD	Section 5.3.4.1.g
FiringObject Identifier	MANDITORY FIELD	Section 5.3.4.1.b
FuseType	MANDITORY FIELD	Section 5.2.7.c Section 5.3.4.1.h
InitialVelocityVector	MANDITORY FIELD	Section 5.3.4.1.i
MunitionObject Identifier	empty	Section 5.3.4.1.d
MunitionType	MANDITORY FIELD	Section 5.2.7.a Section 5.3.4.1.h
QuantityFired	zero	Section 5.2.7.d Section 5.3.4.1.h
RateOfFire	1	Section 5.2.7.d Section 5.3.4.1.h
TargetObjectIdentifier	empty	Section 5.3.4.1.c
WarheadType	MANDITORY FIELD	Section 5.2.7.b Section 5.3.4.1.h

7.2.2 MunitionDetonation Interaction

The MunitionDetonation interaction alerts all simulation participants that a weapon has detonated. The MunitionDetonation may have been preceded by a WeaponFire interaction indication when the munition was fired, or may stand on its own as in the case of a mine detonation. The form of this interaction closely follows the layout of the DetonationPDU used by DIS. The "ArticulatedPartData," "DetonationResultCode," "QuantityFired," "TargetObjectIdentifier," "MunitionObjectIdentifier," and "RateOfFire" parameters **shall** be always treated as optional fields. The publisher **shall** provide data for "FiringObjectIdentifier" and "FinalVelocityVector" for all munitions other than mines. All other parameters **shall** be unconditionally required.

Table 7-4 MunitionDetonation Parameters

Parameter Name	DIS PDU	DIS Field	Definition
ArticulatedPartData	Detonation	Articulation Parameters	Articulated Parts info is included when the firer determines an articulated part of the target entity has been effected by the detonation.
DetonationLocation	Detonation	Location in World Coordinates	The location, in world coordinates, at which the munition detonated.
DetonationResultCode	Detonation	Detonation Result	The type of detonation (Entity Impact, Ground Impact, Entity Proximate Detonation, etc.)
EventIdentifier	Detonation	Event ID	ID generated by the firing entity to associate related fire and detonation interactions.
FiringObjectIdentifier	Detonation	Firing Entity ID	Object ID issuing the Weapon Fire Interaction.
FinalVelocityVector	Detonation	Velocity	The velocity vector of the munition at the moment of the detonation.
FuseType	Detonation	BurstDescriptor: Fuse	The fuse **shall** be specified by an enumeration.
MunitionObject Identifier	Detonation	Munition ID	RTI Object ID of the fired munition, if an object is created. Used only for tracked munitions.
MunitionType	Detonation	BurstDescriptor: Munition	Kind, Country, Domain, Category, Subcategory, Specific, and Extra fields of a DIS Entity Type.
QuantityFired	Detonation	BurstDescriptor: Quantity	Represents the number of rounds fired in the burst when quantity > 1. One otherwise.
RateOfFire	Detonation	Burst Descriptor: Rate	Rate of fire in rounds per minute when quantity > 1. Zero otherwise.
RelativeDetonation Location	Detonation	Location in Entity Coordinates	The location, in coordinates relative to the target object, at which the munition detonated. Required if TargetObject Identifier is provided.
TargetObjectIdentifier	Detonation	Target Entity ID	Object ID of the intended target.
WarheadType	Detonation	BurstDescriptor: Warhead	The warhead **shall** be specified by a 16-bit enumeration(see Section 5 in EBV-DOC)

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
ArticulatedPartData	empty	Section 5.2.5 Section 5.3.4.2.l
DetonationLocation	MANDITORY FIELD	Section 5.3.4.2.g
DetonationResultCode	Other	Section 5.3.4.2.j
EventIdentifier	MANDITORY FIELD	Section 5.3.4.2.e
FiringObjectIdentifier	empty	Section 5.3.4.2.b
FinalVelocityVector	all zeros	Section 5.3.4.2.f
FuseType	MANDITORY FIELD	Section 5.2.7.c Section 5.3.4.2.h
MunitionObject Identifier	empty	Section 5.3.4.2.d
MunitionType	MANDITORY FIELD	Section 5.2.7.a Section 5.3.4.2.h
QuantityFired	zero	Section 5.2.7.d Section 5.3.4.2.h
RateOfFire	1	Section 5.2.7.d Section 5.3.4.2.h
RelativeDetonation Location	MANDITORY FIELD	Section 5.3.4.2.i
TargetObjectIdentifier	empty	Section 5.3.4.2.c
WarheadType	MANDITORY FIELD	Section 5.2.7.b Section 5.3.4.2.h

7.3 Communications Family

7.3.1 RadioSignal Interaction

The RadioSignal interaction provides a base class for the interactions that carry messages between radio systems. The radio system's carrier attributes are provided separately by the RadioTransmitter object class (Section 6.2.1). The layout for all of these interactions is derived from the Signal PDU described in Section 5.3.8.2 of IEEE 1278.1-1995 [4]. However, the RPR FOM changes the DIS structure by creating a separate interaction for each encoding class. All publishers of RadioSignal interaction subclasses **shall** provide a HostRadioIndex and RadioIndex. The HostRadioIndex and RadioIndex are used in combination to uniquely identify each radio in an exercise. This pair should be used to associate RadioSignal interactions with the appropriate RadioReceiver or RadioTransmitter object.

The radio addressing scheme used in the RPR FOM differs from that used in DIS. However, all of the information needed for a translation to DIS is still supplied. Under DIS, the Signal PDU included an Entity ID that specified the object carrying the radio and a Radio ID which distinguished multiple radios on the same entity. Since radios are objects in their own right in the RPR FOM (see Sections 6.2.1 and 6.2.2), the most direct means of addressing them under HLA is through their globally unique RTI Object ID. To reconstruct a DIS Entity ID / Radio ID combination, the following steps must be taken: The HostRadioIndex of the RadioSignal interaction is used to lookup the RadioTransmitter object that emanated this signal. The

HostObjectIdentifier of the RadioTransmitter (inherited from EmbeddedSystem) is then used to lookup the BaseEntity object that is carrying this radio. Not only does the BaseEntity provide the absolute location of the radio, but its EntityIdentifier is also equivalent to the Entity ID needed by the Signal PDU. Finally, the RadioTransmitter's RadioIndex attribute is used as an equivalent to the Radio ID field needed to complete the DIS radio addressing scheme.

7.3.1.1 EncodedAudioRadioSignal Interaction

This interaction is used to transmit encoded audio data to other simulation participants. Publishers of this class **shall** provide values for all parameters; there are no optional fields.

Table 7-5 EncodedAudioRadioSignal Parameters

Parameter Name	DIS PDU	DIS Field	Definition
HostRadioIndex	Signal	Entity ID and Radio ID	RTI Object ID of the embedded system host.
TransmitterSignalEncodingType	Signal	Encoding Scheme	Encoding class enumeration.
SignalSampleRate	Signal	Sample Rate	Samples per second for the audio signal
SampleCount	Signal	Samples	Number of samples in this transmission.
SignalDataLength	Signal	Data Length	Length of transmission in bits.
SignalData	Signal	Data	Information contents of this transmission.

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
HostRadioIndex	MANDITORY FIELD	Section 5.3.8.3.b Section 5.3.8.3.c
TransmitterSignal EncodingType	MANDITORY FIELD	Section 5.3.8.3.d
SignalSampleRate	MANDITORY FIELD	Section 5.3.8.3.f
SampleCount	MANDITORY FIELD	Section 5.3.8.3.h
SignalDataLength	MANDITORY FIELD	Section 5.3.8.3.g
SignalData	MANDITORY FIELD	Section 5.3.8.3.i

7.3.1.2 RawBinaryRadioSignal Interaction

This interaction is used to transmit raw binary data to other simulation participants. The values for TacticalDataLinkType and TDLMessageCount **shall** default to "other" and "zero" respectively. The publisher **shall** provide all other parameters.

Table 7-6 RawBinaryRadioSignal Parameters

Parameter Name	DIS PDU	DIS Field	Definition
HostRadioIndex	Signal	Entity ID and Radio ID	RTI Object ID of the embedded system host.
DataRate	Signal	Sample Rate	Bits per second for the binary signal
SignalDataLength	Signal	Data Length	Length of transmission in bits.
SignalData	Signal	Data	Information contents of this transmission.
TacticalDataLinkType	Signal	TDL Type	Tactical data link enumeration.

Parameter Name	DIS PDU	DIS Field	Definition
TDLMessageCount	Signal	Encoding Scheme	Number of tactical data link messages contained in this transmission.

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
HostRadioIndex	MANDITORY FIELD	Section 5.3.8.3.b Section 5.3.8.3.c
DataRate	MANDITORY FIELD	Section 5.3.8.3.f
SignalDataLength	MANDITORY FIELD	Section 5.3.8.3.g
SignalData	MANDITORY FIELD	Section 5.3.8.3.i
TacticalDataLinkType	other	Section 5.3.8.3.e
TDLMessageCount	zero	Section 5.3.8.3.d

7.3.1.3 DatabaseIndexRadioSignal Interaction

This class represents the transmittal of pre-recorded voice data or other messages that can be represented by using pre-defined database. The values for TacticalDataLinkType and TDLMessageCount **shall** default to “other” and “zero” respectively. The publisher **shall** provide all other parameters.

Table 7-7 DatabaseIndexRadioSignal Parameters

Parameter Name	DIS PDU	DIS Field	Definition
HostRadioIndex	Signal	Entity ID and Radio ID	RTI Object ID of the embedded system host.
DatabaseIndex	Signal	Data	Index into database of messages.
Duration	Signal	Data	Duration of transmitted signal.
StartOffset	Signal	Data	Time stamp indicating communications start time.
TacticalDataLinkType	Signal	TDL Type	Defines type of tactical data link.
TDLMessageCount	Signal	Encoding Scheme	Number of tactical data link messages contained in this transmission.

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
HostRadioIndex	MANDITORY FIELD	Section 5.3.8.3.b Section 5.3.8.3.c
DatabaseIndex	MANDITORY FIELD	Section 5.3.8.3.i
Duration	MANDITORY FIELD	Section 5.3.8.3.i
StartOffset	MANDITORY FIELD	Section 5.3.8.3.i
TacticalDataLinkType	other	Section 5.3.8.3.e
TDLMessageCount	zero	Section 5.3.8.3.d

7.3.1.4 ApplicationSpecificRadioSignal Interaction

This interaction is used for any case not satisfied by the other subclasses of the RadioSignal interaction. It is similar to RawBinaryRadioSignal but includes an extra UserProtocolID field that allows the application to translate the encoding scheme for each transmission. The values for

TacticalDataLinkType and TDLMessageCount **shall** default to "other" and "zero" respectively. The publisher **shall** provide all other parameters.

Table 7-8 ApplicationSpecificRadioSignal Parameters

Parameter Name	DIS PDU	DIS Field	Definition
HostRadioIndex	Signal	Entity ID and Radio ID	RTI Object ID of the embedded system host.
DataRate	Signal	Sample Rate	Bits per second for the binary signal
SignalDataLength	Signal	Data Length	Length of transmission in bits.
SignalData	Signal	Data	Information contents of this transmission.
TacticalDataLinkType	Signal	TDL Type	Tactical data link enumeration.
TDLMessageCount	Signal	Encoding Scheme	Number of tactical data link messages contained in this transmission.
UserProtocolID	Signal	Data	User protocol identification number.

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
HostRadioIndex	MANDITORY FIELD	Section 5.3.8.3.b Section 5.3.8.3.c
DataRate	MANDITORY FIELD	Section 5.3.8.3.f
SignalDataLength	MANDITORY FIELD	Section 5.3.8.3.g
SignalData	MANDITORY FIELD	Section 5.3.8.3.i
TacticalDataLinkType	other	Section 5.3.8.3.e
TDLMessageCount	zero	Section 5.3.8.3.d
UserProtocolID	MANDITORY FIELD	Section 5.3.8.3.i

7.4 Logistics Family

The logistics family is used to represent one battlespace object repairing or resupplying another. There are six interactions in the family. Four (ServiceRequest, ResupplyOffer, ResupplyReceived, and ResupplyCancel) are used to simulate resupply. Three (ServiceRequest, RepairResponse and RepairComplete) are used to simulate repair. The use of these interactions involves a detailed understanding of the state transitions and timing between events. The logistics family in the RPR FOM use the same state diagrams and timing defined in Clause 4.5.4 of IEEE 1278.1-1995 [4].

7.4.1 RepairComplete Interaction

The RepairComplete interaction is one of three interactions used to simulate one battlespace object repairing another. It **shall** be sent by the repairing object upon the completion of the repair, as described in IEEE 1278.1-1995 [4] Sections 4.5.4.3 and 4.5.4.9. Publishers of this class **shall** provide values for all parameters; there are no optional fields.

Table 7-9 RepairComplete Parameters

Parameter Name	DIS PDU	DIS Field	Definition

ReceivingObject	Repair Complete	Receiving Entity ID	Object requesting repairs.
RepairingObject	Repair Complete	Repairing Entity ID	Repairing object.
RepairType	Repair Complete	Repair	One of the enumerated repair types.

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
ReceivingObject	MANDITORY FIELD	Section 5.3.5.5.b
RepairingObject	MANDITORY FIELD	Section 5.3.5.5.c
RepairType	MANDITORY FIELD	Section 5.3.5.5.d

7.4.2 RepairResponse Interaction

The RepairResponse interaction is one of three interactions used to simulate one battlespace object repairing another. It **shall** be sent by the battlespace object receiving repairs on receipt of a RepairComplete interaction, as described in IEEE 1278.1-1995 [4] Sections 4.5.4.3 and 4.5.4.10. Publishers of this class **shall** provide values for all parameters; there are no optional fields.

Table 7-10 RepairResponse Parameters

Parameter Name	DIS PDU	DIS Field	Definition
ReceivingObject	Repair Response	Receiving Entity ID	Object requesting repairs.
RepairingObject	Repair Response	Repairing Entity ID	Repairing object.
RepairResultCode	Repair Response	Repair Result	One of the enumerated repair results.

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
ReceivingObject	MANDITORY FIELD	Section 5.3.5.6.b
RepairingObject	MANDITORY FIELD	Section 5.3.5.6.c
RepairResultCode	MANDITORY FIELD	Section 5.3.5.6.d

7.4.3 ResupplyCancel Interaction

The ResupplyCancel interaction is one of four interactions used to simulate one battlespace object resupplying another. It **shall** be sent by the resupplying or receiving battlespace object when conditions for resupply are no longer met, as described in IEEE 1278.1-1995 [4] Sections 4.5.4.2 and 4.5.4.7. Publishers of this class **shall** provide values for all parameters; there are no optional fields.

Table 7-11 ResupplyCancel Parameters

Parameter Name	DIS PDU	DIS Field	Definition
ReceivingObject	Resupply Cancel	Receiving Entity ID	Object that has requested resupply.
SupplyingObject	Resupply Cancel	Supplying Entity ID	Supplying object.

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
ReceivingObject	MANDITORY FIELD	Section 5.3.5.4.b

SupplyingObject	MANDITORY FIELD	Section 5.3.5.4.c

7.4.4 ResupplyOffer Interaction

The ResupplyOffer interaction is one of four interactions used to simulate one battlespace object resupplying another. It **shall** be sent by an identified supplying battlespace object in response to a ServiceRequest interaction, as described in IEEE 1278.1-1995 [4] Sections 4.5.4.2.2 and 4.5.4.5. Publishers of this class **shall** provide values for all parameters; there are no optional fields.

Table 7-12 ResupplyOffer Parameters

Parameter Name	DIS PDU	DIS Field	Definition
ReceivingObject	Resupply Offer	Receiving Entity ID	Object that has requested resupply.
SupplyingObject	Resupply Offer	Supplying Entity ID	Supplying object.
SuppliesData	Resupply Offer	Number of Supply Types Supplies	List of offered supplies.

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
ReceivingObject	MANDITORY FIELD	Section 5.3.5.2.b
SupplyingObject	MANDITORY FIELD	Section 5.3.5.2.c
SuppliesData	MANDITORY FIELD	Section 5.3.5.2.e

7.4.5 ResupplyReceived Interaction

The ResupplyReceived interaction is one of four interactions used to simulate one battlespace object resupplying another. It **shall** be sent by an identified receiving battlespace object to indicate the supplies actually transferred, as described in IEEE 1278.1-1995 [4] Sections 4.5.4.2.1 and 4.5.4.6. Publishers of this class **shall** provide values for all parameters; there are no optional fields.

Table 7-13 ResupplyReceived Parameters

Parameter Name	DIS PDU	DIS Field	Definition
ReceivingObject	Resupply Received	Receiving Entity ID	Object that has requested resupply.
SupplyingObject	Resupply Received	Supplying Entity ID	Supplying object.
SuppliesData	Resupply Received	Number of Supply Types Supplies	List of supplies taken by receiving object.

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
ReceivingObject	MANDITORY FIELD	Section 5.3.5.3.b
SupplyingObject	MANDITORY FIELD	Section 5.3.5.3.c
SuppliesData	MANDITORY FIELD	Section 5.3.5.3.e

7.4.6 ServiceRequest Interaction

The ServiceRequest interaction is one of four interactions used to simulate one battlespace object resupplying another. It is also one of the three interactions used to simulate one battlespace object repairing another. It **shall** be sent by a battlespace object requesting repair or resupply when appropriate conditions for repair or resupply exist, as described in IEEE 1278.1-1995 [4] Section 4.5.4.4. The SuppliesData parameter **shall not** be sent if the ServiceType parameter is not Resupply. All other fields **shall** be provided by the message originator.

Table 7-14 ServiceRequest Parameters

Parameter Name	DIS PDU	DIS Field	Definition
RequestingObject	Service Request	Requesting Entity ID	Object requesting service.
ServicingObject	Service Request	Servicing Entity ID	Object able to provide the requested service.
ServiceType	Service Request	Service Type Requested	Type of requested service.
SuppliesData	Service Request	Number of Supply Types Supplies	List of type and number of supplies requested, if ServiceType is Resupply. Otherwise, this parameter is not sent.

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
RequestingObject	MANDITORY FIELD	Section 5.3.5.1.b
ServicingObject	MANDITORY FIELD	Section 5.3.5.1.c
ServiceType	MANDITORY FIELD	Section 5.3.5.1.d
SuppliesData	unknown	Section 5.3.5.1.f

7.5 DIS Simulation Management Family

Although the RTI provides many simulation management (SIMAN) functions, some DIS features are not supported directly by the API. Two particularly difficult DIS areas to support using the native HLA Object Management data functions are the group addressing schemes supported by the EntityIdentifier triplet and the need for additional parameters in the function calls. This family provides a direct porting of the DIS SIMAN PDUs into a set of HLA interactions. State diagrams for these interactions are the same, in each case, as the corresponding diagrams in IEEE 1278.1-1995 [4] Section 4.5.5. In each case, the interaction name and parameter names were developed to closely match those of DIS. Although federates are not required to support this family of interactions, failure to do so may limit the system's ability to interact with federates derived from legacy DIS simulation management systems. A negative response to simulation management requests should be provided at a minimum to support this family.

DIS supported multiple levels of attribute visibility. Attributes described in the PDU structures were "public" values with full visibility to all simulations. However, through the Simulation Management (SIMAN) PDUs, another set of "private" attributes could also be manipulated.

These "private" attributes were generally considered to be those components of the entity that had no interoperability impact during an exercise, but that might be required for data collection or after-action review. In the interest of supporting the transition of legacy DIS simulations into the HLA, these visibility rules are maintained by the RPR FOM.

The interactions supported within the DIS Simulation Management Family are complementary to those supported within the HLA Simulation Management Family. It is recommended that RPR FOM federates support both capabilities to insure maximal interoperability with RPR FOM management facilities.

In addition to supporting addressing of a single entity, the DIS addressing scheme also supports wildcarding. The acceptable wildcard enumerations can be found in EBV-DOC [1].

7.5.1 Acknowledge Interaction

This interaction is sent in return to some type of SIMAN request (StartResume, StopFreeze, CreateEntity, RemoveEntity) made by the originating entity. It identifies which request the acknowledge is in response to, the actual type of request that was made, and the response to this request. All parameters in this interaction **shall** be considered as non-optional fields. The value of the RequestIdentifier comes from the originating request and allows the recipient of the acknowledgement to match it with an original request.

Table 7-15 Acknowledge Parameters

Parameter Name	DIS PDU	DIS Field	Definition
OriginatingEntity	Acknowledge	Originating Entity ID	The DIS Entity ID triplet of the object or application originating this response.
ReceivingEntity	Acknowledge	Receiving Entity ID	The DIS Entity ID triplet of the intended recipient of this response.
RequestIdentifier	Acknowledge	Request ID	Enumeration for the request
AcknowledgeFlag	Acknowledge	Acknowledge Flag	Enumeration for the acknowledgement
ResponseFlag	Acknowledge	Response Flag	Enumeration for the response

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
OriginatingEntity	MANDITORY FIELD	Section 5.2.29.b Section 5.3.6.5.a
ReceivingEntity	MANDITORY FIELD	Section 5.2.29.c Section 5.3.6.5.a
RequestIdentifier	MANDITORY FIELD	Section 5.3.6.5.d
AcknowledgeFlag	MANDITORY FIELD	Section 5.3.6.5.b
ResponseFlag	MANDITORY FIELD	Section 5.3.6.5.c

Although some of the functionality of the Acknowledge Interaction is simplified by the federate's ability to specify reliable transportation under HLA, a response message is still required to allow federates to reject simulation management requests.

7.5.2 ActionRequest Interaction

This interaction requests an entity to perform some type of action. An entity or application responds to this interaction with an ActionResponse interaction. The ActionRequest closely resembles the Action Request PDU with the primary differences being the lack of a PDU header, a number indicating how many fixed datums and a number indicating how many variable length datums are contained in the request. The "FixedDatums" and "VaribleDatumsSet" are used in conjunction with the action that is being requested, and vary with each type of action being requested. The "OriginatingEntity", "ReceivingEntity", "RequestIdentifier", and "ActionRequestCode" **shall** be considered required fields in this interaction.

Table 7-16 ActionRequest Parameters

Parameter Name	DIS PDU	DIS Field	Definition
OriginatingEntity	Action Request	Originating Entity ID	The DIS Entity ID triplet of the object or application originating this request.
ReceivingEntity	Action Request	Receiving Entity ID	The DIS Entity ID triplet of the intended recipient of this request. Wildcards allowed for group addressing.
RequestIdentifier	Action Request	Request ID	Enumeration that specifies the request
ActionRequestCode	Action Request	Action ID	Enumeration that specifies the specific action requested
FixedDatums	Action Request	Fixed Datum	A set of fixed length data items where each element specifies the type of item (enumeration) and its value (specific data type).
VariableDatumSet	Action Request	Variable Datum	A set of variable length data items where each element specifies the type of item (enumeration), its length, and its value (specific data type).

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
OriginatingEntity	MANDITORY FIELD	Section 5.2.29.b Section 5.3.6.6.a
ReceivingEntity	MANDITORY FIELD	Section 5.2.29.c Section 5.3.6.6.a
RequestIdentifier	MANDITORY FIELD	Section 5.3.6.6.b
ActionRequest Code	MANDITORY FIELD	Section 5.3.6.6.c
FixedDatums	none provided	Section 5.2.10 Section 5.3.6.6.d
VariableDatumSet	none provided	Section 5.2.10 Section 5.3.6.6.d

7.5.3 ActionResponse Interaction

This interaction returns a response to an earlier ActionRequest interaction. The "RequestIdentifier" originates from the ActionRequest interaction, and the "ResultStatus" contains the status of the requested action. The "FixedDatums" and the "VariableDatumSet" contain any relevant information that is being returned in response to the requested action. This interaction closely mimics the DIS ActionResponse PDU, and has the same differences as the ActionRequest interaction.. The "OriginatingEntity", "ReceivingEntity", "RequestIdentifier", and "RequestStatus" **shall** be considered required fields in this interaction. The "FixedDatums" and "VariableDatumSet" vary with the type of action that was requested.

Table 7-17 ActionResponse Parameters

Parameter Name	DIS PDU	DIS Field	Definition
OriginatingEnttity	Action Response	Originating Entity ID	The DIS Entity ID triplet of the object or application originating this response.
ReceivingEntity	Action Response	Receiving Entity ID	The DIS Entity ID triplet of the intended recipient of this response.
RequestIdentifier	Action Response	Request ID	Enumeration identifying the request
RequestStatus	Action Response	Request Status	Enumeration identifying the status of the response
FixedDatums	Action Response	Fixed Datums	A set of fixed length data items where each element specifies the type of item (enumeration) and its value (specific data type).
VariableDatumSet	Action Response	Variable Datums	A set of variable length data items where each element specifies the type of item (enumeration), its length, and its value (specific data type).

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
OriginatingEnttity	MANDITORY FIELD	Section 5.2.29.b Section 5.3.6.7.a
ReceivingEntity	MANDITORY FIELD	Section 5.2.29.c Section 5.3.6.7.a
RequestIdentifier	MANDITORY FIELD	Section 5.3.6.7.b
RequestStatus	MANDITORY FIELD	Section 5.3.6.7.c
FixedDatums	none provided	Section 5.2.10 Section 5.3.6.7.d
VariableDatumSet	none provided	Section 5.2.10 Section 5.3.6.7.d

7.5.4 Comment Interaction

The Comment Interaction is used to insert messages and information into a log stream and closely matches the structures used by the Data interaction. This information is usually unsolicited in nature. The information contained within the Interaction should be used for commenting purposes only.

Table 7-18 Comment Parameters

Parameter Name	DIS PDU	DIS Field	Definition
OriginatingEntity	Comment	Originating Entity ID	The DIS Entity ID triplet of the object or application originating this interaction.
ReceivingEntity	Comment	Receiving Entity ID	The DIS Entity ID triplet of the intended recipient of this interaction. Wildcards allowed for group addressing.
VariableDatumSet	Comment	Variable Datum	A set of variable length data items where each element specifies the type of item (enumeration), its length, and its value (specific data type).

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
OriginatingEntity	MANDITORY FIELD	Section 5.2.29.b Section 5.3.6.12.a
ReceivingEntity	MANDITORY FIELD	Section 5.2.29.c Section 5.3.6.12.a
VariableDatumSet	none provided	Section 5.2.32 Section 5.3.6.12.d

7.5.5 CreateEntity Interaction

This interaction is used to request the creation of a new entity and closely matches the Create Entity PDU. All parameters in this request **shall** be considered non-optional. The results of the CreateEntity are returned via an Acknowledgement interaction. The "EntityNumber" contained within the "ReceivingEntity" can contain two types of valid values. If the value is between 1 – 65533 this is the exact entity number requested to be created. If the value contains 65534, it is a placeholder value for the recipient to use the next entity number it has available. The numbers 0 and 65535 are invalid for EntityNumbers. The number 0 is reserved for applications, and 65535 is reserved to mean all entities.

Table 7-19 CreateEntity Parameters

Parameter Name	DIS PDU	DIS Field	Definition
OriginatingEntity	Create	Originating Entity ID	The DIS Entity ID triplet of the object or application originating this request.
ReceivingEntity	Create	Receiving Entity ID	The DIS Entity ID triplet of the intended recipient of this request. Wildcards allowed for group addressing.
RequestIdentifier	Create	Request ID	32 bit integer indicating the request ID

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
OriginatingEntity	MANDITORY FIELD	Section 5.2.29.b Section 5.3.6.1.a
ReceivingEntity	MANDITORY FIELD	Section 5.2.29.c Section 5.3.6.1.a
RequestIdentifier	MANDITORY FIELD	Section 5.3.6.1.b

7.5.6 Data Interaction

Data interactions are usually solicited responses to DataQuery and SetData interactions, and closely resemble DIS Data PDU. The “OriginatingEntity”, “ReceivingEntity”, and “RequestIdentifier” **shall** be considered non-optional fields and the contents of the “Fixed Datums” and “Variable Datum Sets” vary with the type of data being sent with the interaction. The value of the “RequestIdentifier” should come from the originating solicitation of the Data interaction, and at least one “FixedDatum” or “VariableDatumSet” should be present, but it is not required. An example of a DataInteraction that would contain no fixed or variable information would be a response to a set data that a simulator does not model any of the requested datums contained in the SetData. It is also possible for the receiver to get multiple responses to the same request since the reply may contain lots of variable length datums that in total is greater than the maximum size allowed for a single network packet.

Table 7-20 Data Parameters

Parameter Name	DIS PDU	DIS Field	Definition
OriginatingEntity	Data	Originating Entity ID	The DIS Entity ID triplet of the object or application originating this response.
ReceivingEntity	Data	Receiving Entity ID	The DIS Entity ID triplet of the intended recipient of this response.
RequestIdentifier	Data	Request ID	32 bit integer indicating the request ID
FixedDatums	Data	Fixed Datum	A set of fixed length data items where each element specifies the type of item (enumeration) and its value (specific data type).
VariableDatumSet	Data	Variable Datum	A set of variable length data items where each element specifies the type of item (enumeration), its length, and its value (specific data type).

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
OriginatingEntity	MANDITORY FIELD	Section 5.2.29.b Section 5.3.6.10.a
ReceivingEntity	MANDITORY FIELD	Section 5.2.29.c Section 5.3.6.10.a
RequestIdentifier	MANDITORY FIELD	Section 5.3.6.10.d
FixedDatums	none provided	Section 5.2.10 Section 5.3.6.10.c
VariableDatumSet	none provided	Section 5.2.10 Section 5.3.6.10.c

7.5.7 DataQuery Interaction

SIMAN DataQuerys allow for the solicitation of attributes, and takes the place of DIS Data Query PDU. The “TimeInterval” parameter allows the originator to request the receiving entity to periodically send the requested information at the specified interval. The field “TimeInterval”

shall be optional. If this value is zero or it is not provided, the recipient need only respond to the Data Query with a single Data interaction. The “OriginatingEntity”, “ReceivingEntity”, “RequestIdentifier” **shall** be considered non-optional. , The “FixedDatum” and “VariableDatums” indicate the attributes being queried and at least one of these parameters should be present but are considered optional. A query that contains no fixed or variable datums is not forbidden, but it is not a good practice, since it does nothing more than consume processing and network resources.

Table 7-21 DataQuery Parameters

Parameter Name	DIS PDU	DIS Field	Definition
OriginatingEntity	Data	Originating Entity ID	The DIS Entity ID triplet of the object or application originating this request.
ReceivingEntity	Data	Receiving Entity ID	The DIS Entity ID triplet of the intended recipient of this request. Wildcards allowed for group addressing.
RequestIdentifier	Data	Request ID	32 bit integer indicating the request ID
TimeInterval	Data	Time Interval	Timestamp indicating the amount of time that should elapse between continued responses to this request
FixedDatum Identifiers	Data	Fixed Datum	The set of datum identifiers that specify the requested fixed length datums.
VariableDatum Identifiers	Data	Variable Datum	The set of datum identifiers that specify the requested variable length datums.

Parameter Name	Default Value (if optional)	IEEE 1278.1a – 1995 Reference
OriginatingEntity	MANDITORY FIELD	Section 5.2.29.b Section 5.3.6.8.a
ReceivingEntity	MANDITORY FIELD	Section 5.2.29.c Section 5.3.6.8.a
RequestIdentifier	MANDITORY FIELD	Section 5.3.6.8.c
TimeInterval	zero	Section 5.3.6.8.b
FixedDatumIdentifiers	none provided	Section 5.2.10 Section 5.3.6.8.d
VariableDatumIdentifiers	none provided	Section 5.2.10 Section 5.3.6.8.d

7.5.8 EventReport Interaction

When a significant event occurs on a managed entity, the entity reports these incidents to the simulation manager through Event Reports. This interaction closely mimics the Event Report PDU utilized by DIS. The “OriginatingEntity”, “ReceivingEntity”, and “EventType” **shall** be considered non-optional fields. The contents of the “FixedDatums” and “VariableDatumSet” are contingent on the type of event being reported and vary with the EventType.

Table 7-22 EventReport Parameters

Parameter Name	DIS PDU	DIS Field	Definition
OriginatingEntity	Event Report	Originating Entity ID	The DIS Entity ID triplet of the object or application originating this report.
ReceivingEntity	Event Report	Receiving Entity ID	The DIS Entity ID triplet of the intended recipient of this report. Wildcards allowed for group addressing.
EventType	Event Report	Event Type	Enumeration indicating the type of event that caused the issuance of the Event Report Interaction
FixedDatums	Event Report	Fixed Datum	A set of fixed length data items where each element specifies the type of item (enumeration) and its value (specific data type).
VariableDatumSet	Event Report	Variable Datum	A set of variable length data items where each element specifies the type of item (enumeration), its length, and its value (specific data type).

Parameter Name	Default Value (if optional)	IEEE 1278.1a – 1995 Reference
OriginatingEntity	MANDITORY FIELD	Section 5.2.29.b Section 5.3.6.11.a
ReceivingEntity	MANDITORY FIELD	Section 5.2.29.c Section 5.3.6.11.a
EventType	MANDITORY FIELD	Section 5.3.6.11.b
FixedDatums	none provided	Section 5.2.10 Section 5.3.6.11.d
VariableDatumSet	none provided	Section 5.2.10 Section 5.3.6.11.d

7.5.9 RemoveEntity Interaction

This interaction is used to request the removal of an entity and closely matches the DIS Remove Entity PDU. All parameters in this request **shall** be considered non-optional. This interaction differs from the native HLA mechanism by the wide variety of responses possible in the Acknowledge interaction.

Table 7-23 RemoveEntity Parameters

Parameter Name	DIS PDU	DIS Field	Definition
OriginatingEntity	RemoveEntity	Originating Entity ID	The DIS Entity ID triplet of the object or application originating the request.
ReceivingEntity	RemoveEntity	Receiving Entity ID	The DIS Entity ID triplet of the intended recipient of this request. Wildcards allowed for group addressing.
RequestIdentifier	RemoveEntity	Request ID	32 bit integer indicating the request number

Parameter Name	Default Value (if optional)	IEEE 1278.1a – 1995

		Reference
OriginatingEntity	MANDITORY FIELD	Section 5.2.29.b Section 5.3.6.2.a
ReceivingEntity	MANDITORY FIELD	Section 5.2.29.c Section 5.3.6.2.a
RequestIdentifier	MANDITORY FIELD	Section 5.3.6.2.b

7.5.10 SetData Interaction

Requests to remotely set the value of an attribute are communicated with a SetData Interaction and it is often used to initialize new entities once they've been created. This interaction maps to the DIS Set Data PDU. "OriginatingEntity", "ReceivingEntity", and "RequestIdentifier" **shall** be considered non-optional. The "FixedDatum" and "VariableDatums" indicate the attributes being queried and at least one of these parameters should be present but are considered optional and vary from interaction to interaction.

Table 7-24 SetData Parameters

Parameter Name	DIS PDU	DIS Field	Definition
OriginatingEntity	SetData	Originating Entity ID	The DIS Entity ID triplet of the object or application originating the request.
ReceivingEntity	SetData	Receiving Entity ID	The DIS Entity ID triplet of the intended recipient of this request. Wildcards allowed for group addressing.
RequestIdentifier	SetData	Request ID	32 bit integer indicating the request ID
FixedDatums	SetData	Fixed Datum	A set of fixed length data items where each element specifies the type of item (enumeration) and its value (specific data type).
VariableDatumSet	SetData	Variable Datum	A set of variable length data items where each element specifies the type of item (enumeration), its length, and its value (specific data type).

Parameter Name	Default Value (if optional)	IEEE 1278.1a – 1995 Reference
OriginatingEntity	MANDITORY FIELD	Section 5.2.29.b Section 5.3.6.9.a
ReceivingEntity	MANDITORY FIELD	Section 5.2.29.c Section 5.3.6.9.a
RequestIdentifier	MANDITORY FIELD	Section 5.3.6.9.b
FixedDatums	none provided	Section 5.2.10 Section 5.3.6.9.c
VariableDatumSet	none provided	Section 5.2.10 Section 5.3.6.9.c

7.5.11 StartResume Interaction

This interaction informs federates that it should begin updating particular entities. It differs from the native HLA mechanisms for the behavior because it supports the DIS mechanisms for entity addressing and time of day. This interaction closely follows the format of the DIS Start/Resume PDU. All parameters in this interaction **shall** be considered non-optional.

Table 7-25 StartResume Parameters

Parameter Name	DIS PDU	DIS Field	Definition
OriginatingEntity	StartResume	Originating Entity ID	The DIS Entity ID triplet of the object or application originating the request.
ReceivingEntity	StartResume	Receiving Entity ID	The DIS Entity ID triplet of the intended recipient of this request. Wildcards allowed for group addressing.
RequestIdentifier	StartResume	Request ID	32 bit integer indicating the request number
RealWorldTime	StartResume	Real-World Time	GMT that the entity should be started/resumed
SimulationTime	StartResume	Simulation Time	Simulation time the entity should be started/resumed

Parameter Name	Default Value (if optional)	IEEE 1278.1a – 1995 Reference
OriginatingEntity	MANDITORY FIELD	Section 5.2.29.b Section 5.3.6.3.a
ReceivingEntity	MANDITORY FIELD	Section 5.2.29.c Section 5.3.6.3.a
RequestIdentifier	MANDITORY FIELD	Section 5.3.6.3.d
RealWorldTime	MANDITORY FIELD	Section 5.2.8 Section 5.3.6.3.b
SimulationTime	MANDITORY FIELD	Section 5.2.8 Section 5.3.6.3.c

The “RealWorldTime” parameter is the time that the start should take affect, and when it takes effect the “SimulationTime” indicates what the current simulation time will be. For example, if the real world time is set to 5:00PM GMT and the simulation time is 11:00 AM, when the wall clock on the simulator reaches 5:00PM, it should start its simulation, and use 11:00AM as its simulation time. But, just like any other standard, there is always an exception. While there was no specific rule in DIS to indicate this, common usage holds that simulations should start immediately if the RealWorld time is set to 0:00.

7.5.12 StopFreeze Interaction

This interaction informs federates that it should stop updating particular entities. It differs from the native HLA mechanisms for the behavior because it supports the DIS mechanisms for entity addressing and time of day. This interaction closely follows the format of the DIS Stop/Freeze PDU. All parameters in this interaction **shall** be considered non-optional.

Table 7-26 StopFreeze Parameters

Parameter Name	DIS PDU	DIS Field	Definition
OriginatingEntity	StopFreeze	Originating Entity ID	The DIS Entity ID triplet of the object or application originating the request.
ReceivingEntity	StopFreeze	Receiving Entity ID	The DIS Entity ID triplet of the intended recipient of this request. Wildcards allowed for group addressing.
RequestIdentifier	StopFreeze	Request ID	32 bit integer indicating the request number
RealWorldTime	StopFreeze	Real-World Time	GMT that the entity should be stopped/frozen.
Reason	StopFreeze	Reason	8 bit enumeration indicating the reason why the entity/simulation is frozen
RunInternal SimulationClock	StopFreeze	FrozenBehavior	True if the entities should continue to run their internal simulation clock when stopped/frozen.
UpdateAttributes	StopFreeze	FrozenBehavior	True if the entities should continue to update outgoing attributes while stopped/frozen.
ReflectValues	StopFreeze	FrozenBehavior	True if the entities should continue to reflect incoming attributes while stopped/frozen.

Parameter Name	Default Value (if optional)	IEEE 1278.1a – 1995 Reference
OriginatingEntity	MANDITORY FIELD	Section 5.2.29.b Section 5.3.6.4.a
ReceivingEntity	MANDITORY FIELD	Section 5.2.29.c Section 5.3.6.4.a
RequestIdentifier	MANDITORY FIELD	Section 5.3.6.4.e
RealWorldTime	MANDITORY FIELD	Section 5.2.8 Section 5.3.6.4.b
Reason	MANDITORY FIELD	Section 5.3.6.4.c
RunInternal SimulationClock	MANDITORY FIELD	Section 5.3.6.4.d
UpdateAttributes	MANDITORY FIELD	Section 5.3.6.4.d
ReflectValues	MANDITORY FIELD	Section 5.3.6.4.d

Like the StartResume, common usage here holds that simulations should stop immediately if the RealWorld time is set to 0:00.

7.6 HLA Simulation Management Family

For new HLA simulations, the DIS addressing schemes and redundant SIMAN interactions may not be appropriate in all cases. This set of interactions re-defines the DIS simulation management functions in a scheme that is closer to the underlying HLA architecture than those provided in Section 7.5. In each case, these interactions assume that the intelligence required to create, delete, or change objects resides at the remote site responsible for modeling that object. Graceful methods of refusal or redefinition of the requests are therefore required. These interactions support this functionality.

The interactions supported within the DIS Simulation Management Family are complementary to those supported within the HLA Simulation Management Family. It is recommended that RPR FOM federates support both capabilities to insure maximal interoperability with RPR FOM management facilities.

7.6.1 AttributeChangeRequest Interaction

This interaction, in conjunction with AttributeChangeResult, provides a mechanism for the remote manipulation of object attributes. Unlike the HLA's native attribute transfer mechanism, AttributeChangeRequest provides a means for the receiving object to evaluate and possibly modify the change request. It is intended to replace the functionality of the DIS SetData PDU with a more generic HLA alternative. Publishers of this class **shall** provide values for all parameters; there are no optional fields.

Table 7-27 AttributeChangeRequest Parameters

Parameter Name	DIS PDU	DIS Field	Definition
ObjectIdentifiers	SetData	Receiving Entity ID	Recipients as a list of Object ID's
AttributeValueSet	SetData	Fixed Datums	Set of attribute/value pairs to modify.

Parameter Name	Default Value (if optional)	IEEE 1278.1a – 1995 Reference
ObjectIdentifiers	MANDITORY FIELD	Section 5.2.29.c Section 5.3.6.9.a
AttributeValueSet	MANDITORY FIELD	Section 5.2.10 Section 5.3.6.9.c

7.6.2 AttributeChangeResult Interaction

This interaction is issued in response to an AttributeChangeRequest to indicate the success, failure, or redefinition of a remote attribute manipulation. Unlike the HLA's native attribute transfer mechanism, AttributeChangeResult provides a means for the receiving object to evaluate and possibly modify the change request. It is intended to replace the functionality of the DIS Data PDU with a more generic HLA alternative. Publishers of this class **shall** provide values for all parameters; there are no optional fields.

Table 7-28 AttributeChangeResult Parameters

Parameter Name	DIS PDU	DIS Field	Definition
ObjectIdentifier	Data	Originating Entity ID	Recipients as a list of Object ID's
AttributeChangeResult	Data	Fixed Datuns	Indicates ability to comply.
AttributeValueSet	Data	Fixed Datuns	Set of attribute/value pairs to modify.

Parameter Name	Default Value (if optional)	IEEE 1278.1a – 1995 Reference

ObjectIdentifier	MANDITORY FIELD	Section 5.2.29.b Section 5.3.6.10.a
AttributeChangeResult	MANDITORY FIELD	Section 5.2.10 Section 5.3.6.10.c
AttributeValueSet	MANDITORY FIELD	Section 5.2.10 Section 5.3.6.10.c

7.6.3 CreateObjectRequest Interaction

This interaction, in conjunction with CreateObjectResult, provides a mechanism for the remote initialization of new objects. Unlike the HLA's native attribute transfer mechanism, CreateObjectRequest provides a means for the simulation that creates the new object to evaluate and possibly modify the initial conditions. It is intended to replace the functionality of the DIS Create Entity PDU with a more generic HLA alternative. Publishers of this class **shall** provide values for all parameters; there are no optional fields.

Table 7-29 CreateObjectRequest Parameters

Parameter Name	DIS PDU	DIS Field	Definition
ObjectClass	Create Entity	N/A	Type of object to create
AttributeValueSet	Create Entity	N/A	Initial set of attribute/value pairs.
RequestIdentifier	Create Entity	Request ID	Sequence number identifier.

Parameter Name	Default Value (if optional)	IEEE 1278.1a – 1995 Reference
ObjectClass	MANDITORY FIELD	none
AttributeValueSet	MANDITORY FIELD	none
RequestIdentifier	MANDITORY FIELD	Section 5.3.6.1.b

7.6.4 CreateObjectResult Interaction

This interaction is issued in response to an CreateObjectRequest to indicate the success, failure, or redefinition of a remote initialization of new objects. Publishers of this class **shall** provide values for all parameters; there are no optional fields.

Table 7-30 CreateObjectRequest Parameters

Parameter Name	DIS PDU	DIS Field	Definition
CreateObjectResult	Acknowledge	Response Flag	Indicates ability to comply.
RequestIdentifier	Acknowledge	Request ID	Sequence number identifier.

Parameter Name	Default Value (if optional)	IEEE 1278.1a – 1995 Reference
CreateObjectResult	MANDITORY FIELD	Section 5.3.6.5.c
RequestIdentifier	MANDITORY FIELD	Section 5.3.6.5.d

7.6.5 RemoveObjectRequest Interaction

This interaction, in conjunction with RemoveObjectResult, provides a mechanism for the remote deletion of existing objects. It is intended to replace the functionality of the DIS Remove PDU with a more generic HLA alternative. Publishers of this class **shall** provide values for all parameters; there are no optional fields.

Table 7-31 RemoveObjectRequest Parameters

Parameter Name	DIS PDU	DIS Field	Definition
ObjectIdentifiers	Remove Entity	Receiving Entity ID	Objects to delete as a list of Object ID's
RequestIdentifier	Remove Entity	Request ID	Sequence number identifier.

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
ObjectIdentifiers	MANDITORY FIELD	Section 5.2.29.c Section 5.3.6.1.a
RequestIdentifier	MANDITORY FIELD	Section 5.3.6.2.b

7.6.6 RemoveObjectResult Interaction

This interaction is issued in response to a CreateObjectRequest to indicate the success or failure, of a remote deletion. Publishers of this class **shall** provide values for all parameters; there are no optional fields.

Table 7-32 RemoveObjectResult Parameters

Parameter Name	DIS PDU	DIS Field	Definition
RemoveObjectResult	Remove	N/A	Indicates ability to comply.
RequestIdentifier	Remove	Request ID	Sequence number identifier.

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
RemoveObjectResult	MANDITORY FIELD	none
RequestIdentifier	MANDITORY FIELD	Section 5.3.6.2.b

7.6.7 ActionRequestToObject Interaction

This interaction, in conjunction with ActionResponseFromObject, provides a mechanism for requesting objects to perform enumerated actions. It is intended to replace the functionality of the DIS ActionRequest PDU with a more generic HLA alternative. Publishers of this class **shall** provide values for all parameters; there are no optional fields.

Table 7-33 ActionRequestToObject Parameters

Parameter Name	DIS PDU	DIS Field	Definition
ObjectIdentifiers	Action Request	Receiving Entity ID	The list of objects that are the intended

			recipients of this interaction.
ActionRequestCode	Action Request	Action ID	The action that the recipient(s) are intended to perform.

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
ObjectIdentifiers	MANDITORY FIELD	Section 5.2.29.c Section 5.3.6.1.a
ActionRequest Code	MANDITORY FIELD	Section 5.3.6.6.c

7.6.8 ActionResponseFromObject Interaction

This interaction is issued in response to an ActionRequestToObject to indicate the success or failure, of an action request. Publishers of this class **shall** provide values for all parameters; there are no optional fields.

Table 7-34 ActionResponseFromObject Parameters

Parameter Name	DIS PDU	DIS Field	Definition
ActionResult	Action Response	Request Status	Indicates ability to comply.

Parameter Name	Default Value (if optional)	IEEE 1278.1a - 1995 Reference
ActionResult	MANDITORY FIELD	Section 5.3.6.7.c

8 Mapping from DIS Fields back to the RPR FOM

8.1 Entity Information / Interaction Family

8.1.1 Entity State PDU

DIS Field	FOM Class	FOM Attributes
PDU Header	N/A	
Entity ID	BaseEntity	EntityIdentifier
Force ID	PhysicalEntity	ForceIdentifier
Number of Articulation Parameters	PhysicalEntity	size of ArticulatedParametersArray
Entity Type	BaseEntity	EntityType
Alternative Entity Type	PhysicalEntity	AlternativeEntityType
Entity Linear Velocity	BaseEntity	VelocityVector
Entity Location	BaseEntity	WorldLocation
Entity Orientation	BaseEntity	Orientation
Entity Appearance	BaseEntity	IsFrozen
	EnvironmentalEntity	OpacityCode
	PhysicalEntity	DamageState, EngineSmokeOn, FlamesPresent, HatchState, Immobilized, PersonStanceCode, PowerPlantOn, RampDeployed, SmokePlumePresent, TentDeployed, TrailingEffectsCode, CamouflageType, FirePowerDisabled, IsConcealed
	Platform	AfterburnerOn, AntiCollisionLightsOn , BlackOutBrakeLightsOn, BlackOutLightsOn, BrakeLightsOn, FormationLightsOn, HatchState, HeadLightsOn, InteriorLightsOn, LandingLightsOn, LauncherRaised, NavigationLightsOn, RampDeployed, RunningLightsOn, SpotLightsOn, TailLightsOn
	Sensor	AntennaRaised, BlackoutLightsOn, InteriorLightsOn, LightsOn, MissionKill
	Munition	LauncherFlashPresent
	CulturalFeature	ExternalLightsOn, InternalHeatSourceOn, InternalLightsOn
Dead Reckoning Parameters: Dead Reckoning Algorithm	BaseEntity	DeadReckoningAlgorithm
Dead Reckoning Parameters: Entity Linear Acceleration	BaseEntity	AccelerationVector
Dead Reckoning Parameters: Entity Angular Velocity	BaseEntity	AngularVelocityVector

DIS Field	FOM Class	FOM Attributes
EntityMarking	PhysicalEntity	Marking
Capabilities	PhysicalEntity	HasAmmunitionSupplyCap, HasFuelSupplyCap, HasRecoveryCap, HasRepairCap
	Lifeform	PrimaryWeaponState, SecondaryWeaponState
Articulation Parameters	PhysicalEntity	ArticulatedParametersArray

8.1.2 Collision PDU

DIS Field	FOM Class	FOM Attributes
PDU Header	N/A	
Issuing Entity ID	CollisionInteraction	IssuingObjectIdentifier
Colliding Entity ID	CollisionInteraction	CollidingObjectIdentifier
Event ID	CollisionInteraction	EventIdentifier
Collision Type	CollisionInteraction	CollisionType
Velocity	CollisionInteraction	IssuingObjectVelocityVector
Mass	CollisionInteraction	IssuingObjectMass
Location	CollisionInteraction	CollisionLocation

8.2 Warfare Family

8.2.1 Fire PDU

DIS Field	FOM Class	FOM Attributes
PDU Header	N/A	
Firing Entity ID	WeaponFire	FiringObjectIdentifier
Target Entity ID	WeaponFire	TargetObjectIdentifier
Munition ID	WeaponFire	MunitionObjectIdentifier
Event ID	WeaponFire	EventIdentifier
Fire Mission Index	WeaponFire	FireMissionIndex
Location In World Coordinates	WeaponFire	FiringLocation
Burst Descriptor: Munition	WeaponFire	MunitionType
Burst Descriptor: Warhead	WeaponFire	WarheadType
Burst Descriptor: Fuse	WeaponFire	FuseType
Burst Descriptor: Quantity	WeaponFire	QuantityFired
Burst Descriptor: Rate	WeaponFire	RateOfFire
Velocity	WeaponFire	InitialVelocityVector
Range	WeaponFire	FireControlSolutionRange

8.2.2 Detonation PDU

DIS Field	FOM Class	FOM Attributes
PDU Header	N/A	
Firing Entity ID	MunitionDetonation	FiringObjectIdentifier

DIS Field	FOM Class	FOM Attributes
Target Entity ID	MunitionDetonation	TargetObjectIdentifier
Munition ID	MunitionDetonation	MunitionObjectIdentifier
Event ID	MunitionDetonation	EventIdentifier
Velocity	MunitionDetonation	FinalVelocityVector
Location in World Coordinates	MunitionDetonation	DetonationLocation
Burst Descriptor: Munition	MunitionDetonation	MunitionType
Burst Descriptor: Warhead	MunitionDetonation	WarheadType
Burst Descriptor: Fuse	MunitionDetonation	FuseType
Burst Descriptor: Quantity	MunitionDetonation	QuantityFired
Burst Descriptor: Rate	MunitionDetonation	RateOfFire
Location in Entity Coordinates	MunitionDetonation	RelativeDetonationLocation
Detonation Result	MunitionDetonation	DetonationResultCode
Number of Articulation Parameters	MunitionDetonation	size of ArticulatedParametersData
Articulation Parameters	MunitionDetonation	ArticulatedParametersData

8.3 Logistics Family

8.3.1 Service Request PDU

DIS Field	FOM Interaction	FOM Parameters
PDU Header	N/A	
Requesting Entity ID	ServiceRequest	RequestingObject
Servicing Entity ID	ServiceRequest	ServicingObject
Service Type Requested	ServiceRequest	ServiceType
Number of Supply Types	ServiceRequest	size of SuppliesData
Supplies	ServiceRequest	SuppliesData

8.3.2 Resupply Offer PDU

DIS Field	FOM Interaction	FOM Parameters
PDU Header	N/A	
Receiving Entity ID	ResupplyOffer	ReceivingObject
Supplying Entity ID	ResupplyOffer	SupplyingObject
Number of Supply Types	ResupplyOffer	size of SuppliesData
Supplies	ResupplyOffer	SuppliesData

8.3.3 Resupply Received PDU

DIS Field	FOM Interaction	FOM Parameters
PDU Header	N/A	
Receiving Entity ID	ResupplyReceived	ReceivingObject
Supplying Entity ID	ResupplyReceived	SupplyingObject
Number of Supply Types	ResupplyReceived	size of SuppliesData
Supplies	ResupplyReceived	SuppliesData

8.3.4 Resupply Cancel PDU

DIS Field	FOM Interaction	FOM Parameters
PDU Header	N/A	
Receiving Entity ID	ResupplyCancel	ReceivingObject
Supplying Entity ID	ResupplyCancel	SupplyingObject

8.3.5 Repair Complete PDU

DIS Field	FOM Interaction	FOM Parameters
PDU Header	N/A	
Receiving Entity ID	RepairComplete	ReceivingObject
Repairing Entity ID	RepairComplete	RepairingObject
Repair	RepairComplete	RepairType

8.3.6 Repair Response PDU

DIS Field	FOM Interaction	FOM Parameters
PDU Header	N/A	
Receiving Entity ID	RepairResponse	ReceivingObject
Repairing Entity ID	RepairResponse	RepairingObject
Repair Result	RepairResponse	RepairResultCode

8.4 Simulation Management Family

8.4.1 Create Entity PDU

DIS Field	Interaction	Parameter
PDU Header	N/A	N/A
Originating Entity ID	CreateEntity	OriginatingEntity
Receiving Entity ID	CreateEntity	ReceivingEntity
Request ID	CreateEntity	RequestIdentifier
	CreateObjectRequest	RequestIdentifier

8.4.2 Remove Entity PDU

DIS Field	Interaction	Parameter
PDU Header	N/A	N/A
Originating Entity ID	RemoveEntity	OriginatingEntity
Receiving Entity ID	RemoveEntity	ReceivingEntity
	RemoveObjectRequest	ObjectIdentifiers
Request ID	RemoveEntity	RequestIdentifier
	RemoveObjectRequest	RequestIdentifier

8.4.3 Start/Resume PDU

DIS Field	Interaction	Parameter
PDU Header	N/A	N/A
Originating Entity ID	StartResume	OriginatingEntity
Receiving Entity ID	StartResume	ReceivingEntity
Real-World Time	StartResume	RealWorldTime
Simulation Time	StartResume	SimulationTime
Request ID	StartResume	RequestIdentifier

8.4.4 Stop/Freeze PDU

DIS Field	Interaction	Parameter
PDU Header	N/A	N/A
Originating Entity ID	StopFreeze	OriginatingEntity
Receiving Entity ID	StopFreeze	ReceivingEntity
Real-World Time	StopFreeze	RealWorldTime
Reason	StopFreeze	Reason
Frozen Behavior	StopFreeze	RunInternalSimulationClock, UpdateAttributes, ReflectValues
Request ID	StopFreeze	RequestIdentifier

8.4.5 Acknowledge PDU

DIS Field	Interaction	Parameter
PDU Header	N/A	N/A
Originating Entity ID	Acknowledge	OriginatingEntity
Receiving Entity ID	Acknowledge	ReceivingEntity
Acknowledge Flag	Acknowledge	AcknowledgeFlag
Response Flag	Acknowledge	ResponseFlag
	CreateObjectRequest	CreateObjectResult
	RemoveObjectRequest	RemoveObjectResult
Request ID	Acknowledge	RequestIdentifier
	CreateObjectRequest	RequestIdentifier
	RemoveObjectRequest	RequestIdentifier

8.4.6 Action Request PDU

DIS Field	Interaction	Parameter
PDU Header	N/A	N/A
Originating Entity ID	ActionRequest	OriginatingEntity
Receiving Entity ID	ActionRequest	ReceivingEntity
	ActionRequestToObject	ObjectIdentifiers
Request ID	ActionRequest	RequestIdentifier

Action ID	ActionRequest	ActionRequestCode
	ActionRequestToObject	ActionRequestCode
Number Fixed Datums	ActionRequest	size of FixedDatums
Number Variable Datums	ActionRequest	size of VariableDatumSet
Fixed Datum	ActionRequest	FixedDatums
Variable Datums	ActionRequest	VariableDatumSet

8.4.7 Action Response PDU

DIS Field	Interaction	Parameter
PDU Header	N/A	N/A
Originating Entity ID	ActionResponse	OrignatingEntity
Receiving Entity ID	ActionResponse	ReceivingEntity
Request ID	ActionResponse	RequestIdentifier
Request Status	ActionResponse	RequestStatus
	ActionResponseFromObject	ActionResult
Number Fixed Datums	ActionResponse	size of FixedDatums
Number Variable Datums	ActionResponse	size of VariableDatumSet
Fixed Datums	ActionResponse	FixedDatums
Number Variable Datums	ActionResponse	VariableDatumSet

8.4.8 Data Query PDU

DIS Field	Interaction	Parameter
PDU Header	N/A	N/A
Originating Entity ID	DataQuery	OriginatingEntity
Receiving Entity ID	DataQuery	ReceivingEntity
Request ID	DataQuery	RequestIdentifier
Time Interval	DataQuery	TimeInterval
Number Fixed Datums	DataQuery	size of FixedDatums
Number Variable Datums	DataQuery	size of VariableDatumSet
Fixed Datum	DataQuery	FixedDatums
Variable Datum	DataQuery	VariableDatumSet

8.4.9 Set Data PDU

DIS Field	Interaction	Parameter
PDU Header	N/A	N/A
Originating Entity ID	SetData	OriginatingEntity
Receiving Entity ID	SetData	ReceivingEntity
	AttributeChangeRequest	ObjectIdentifiers
Request ID	SetData	RequestIdentifier
Number Fixed Datums	SetData	size of FixedDatums
	AttributeChangeRequest	size of AttributeValueSet
Number Variable Datums	SetData	size of VariableDatumSet
Fixed Datum	SetData	FixedDatums

	AttributeChangeRequest	AttributeValueSet
	CreateObjectRequest	AttributeValueSet
Variable Datums	SetData	VariableDatumSet

8.4.10 Data PDU

DIS Field	Interaction	Parameter
PDU Header	N/A	N/A
Originating Entity ID	Data	OriginatingEntity
	AttributeChangeResult	
Receiving Entity ID	Data	ReceivingEntity
Request ID	Data	RequestIdentifier
Number Fixed Datums	Data	size of FixedDatums
	AttributeChangeResult	AttributeValueSet
Number Variable Datums	Data	size of VariableDatumSet
FixedDatum	Data	FixedDatums
	AttributeChangeResult	AttributeValueSet, AttributeChangeResult
Variable Datums	Data	VariableDatumSet

8.4.11 Event Report PDU

DIS Field	Interaction	Parameter
PDU Header	N/A	N/A
Originating Entity ID	EventReport	OriginatingEntity
Receiving Entity ID	EventReport	ReceivingEntity
Event Type	EventReport	EventType
Number Fixed Datums	EventReport	size of FixedDatums
Number Variable Datums	EventReport	size of VariableDatumSet
Fixed Datum	EventReport	FixedDatums
Variable Datums	EventReport	VariableDatumSet

8.4.12 Comment PDU

DIS Field	Interaction	Parameter
PDU Header	N/A	N/A
Originating Entity ID	Comment	OriginatingEntity
Receiving Entity ID	Comment	ReceivingEntity
Number Fixed Datums	Comment	size of FixedDatums
Number Variable Datums	Comment	size of VariableDatumSet
Variable Datum	Comment	VariableDatumSet

8.5 Distributed Emission Regeneration Family

8.5.1 Electromagnetic Emissions PDU

DIS Field	FOM Class	FOM Attributes
PDU Header	N/A	
Emitting Entity ID	EmbeddedSystem	HostObjectIdentifier
Event ID	EmbeddedSystem	EventIdentifier
	EmitterBeam	EventIdentifier
State Update Indicator	EmbeddedSystem	computed data
Number of Systems	EmbeddedSystem	computed data
System Data Length	EmbeddedSystem	computed data
Number of Beams	EmitterBeam	computed data
Emitter System: Emitter name	EmitterSystem	EmitterType
Emitter System: Function	EmitterSystem	EmitterFunctionCode
Emitter System: Emitter ID number	EmitterSystem	EmitterIndex
Location (with respect to entity)	EmbeddedSystem	RelativeLocation
Beam Data Length	EmitterBeam	computed data
Beam ID Number	EmitterBeam	BeamIdentifier
Beam Parameter Index	EmitterBeam	BeamParameterIndex
Fundametal Parameter Data: Frequency	EmitterBeam	EmissionFrequency
Fundametal Parameter Data: Frequency Range	EmitterBeam	FrequencyRange
Fundametal Parameter Data: Effective Radiated Power	EmitterBeam	EffectiveRadiatedPower
Fundametal Parameter Data: Pulse Repetition Frequency	EmitterBeam	PulseRepetitionFrequency
Fundametal Parameter Data: Pulse Width	EmitterBeam	PulseWidth
Fundametal Parameter Data: Beam Azimuth Center	EmitterBeam	BeamAzimuthCenter
Fundametal Parameter Data: Beam Azimuth Sweep	EmitterBeam	BeamAzimuthSweep
Fundametal Parameter Data: Beam Elevation Center	EmitterBeam	BeamElevationCenter
Fundametal Parameter Data: Beam Elevation Sweep	EmitterBeam	BeamElevationSweep
Fundametal Parameter Data: Beam Sweep Sync	EmitterBeam	SweepSynch
Beam Function	EmitterBeam	BeamFunctionCode
Number of Targets in the Track/Jam Field	RadarBeam	size of TrackObjectIdentifiers
	JammerBeam	size of JammedObjectIdentifiers
High Density Track/Jam	RadarBeam	HighDensityTrack
	JammerBeam	HighDensityJam
Jamming Mode Sequence	JammerBeam	JammingModeSequence
Track/Jam: Site/Applic/Entity	RadarBeam	TrackObjectIdentifiers
	JammerBeam	JammmedObjectIdentifiers
Track/Jam: Emitter ID	EmitterBeam	EmitterIndex (of emitter targeted by jammer)
Track/Jam: Beam ID	JammerBeam	BeamIdentifier (of beam targeted by

DIS Field	FOM Class	FOM Attributes
		jammer)

8.5.2 Designator PDU

DIS Field	FOM Class	FOM Attributes
PDU Header	N/A	
Designating Entity ID	Designator	DesignatorIdentifier
Code Name	Designator	CodeName
Designated Entity ID	EmbeddedSystem	HostObjectIdentifier
Designator Code	Designator	Designator Code
Designator Power	Designator	Designator PowerDesignatorOutputPower
Designator Wavelength	Designator	DesignatorEmissionWavelength
Designator Spot with Respect to Designated Entity	Designator	RelativeSpotLocation
Designator Spot Location	Designator	DesignatorSpotLocation
Dead Reckoning Algorithm	Designator	DeadReckoningAlgorithm
Entity Linear Acceleration	Designator	SpotLinearAccelerationVector

8.6 Radio Communications

8.6.1 Transmitter PDU

DIS Field	FOM Class	FOM Attributes
PDU Header	N/A	
Entity ID	EmbeddedSystem	HostObjectIdentifier
Radio ID	EmbeddedSystem	RadioIndex
Radio Entity Type	RadioTransmitter	RadioSystemType
Transmit State	RadioTransmitter	TransmitterOperationalStatus
Input Source	RadioTransmitter	RadioInputSource
Antenna Location	RadioTransmitter	WorldLocation
Relative Antenna Location	EmbeddedSystem	RelativeLocation
Antenna Pattern Type	RadioTransmitter	AntennaPatternData
Antenna Pattern Length	RadioTransmitter	size of AntennaPatternData
Frequency	RadioTransmitter	Frequency
Transmit Frequency Bandwidth	RadioTransmitter	FrequencyBandwidth
Power	RadioTransmitter	TransmittedPower
Modulation Type: Spread spectrum	RadioTransmitter	TimeHopInUse, PseudoNoiseSpectrumInUse, FrequencyHopInUse
Modulation Type: Major	RadioTransmitter	RFModulationType
Modulation Type: Detail	RadioTransmitter	RFModulationType
Modulation Type: System	RadioTransmitter	RFModulationSystemType
CryptoSystem	RadioTransmitter	CryptoSystem
CryptoKeyID	RadioTransmitter	CryptographicMode, EncryptionKeyIdentifier
Length of Modulation Parameters	RadioTransmitter	sizeof ModulatonParameters

DIS Field	FOM Class	FOM Attributes
Modulation Parameter #1 ... #N	RadioTransmitter	ModulationParameters
Antenna Pattern Parameter #1 ... #N	RadioTransmitter	AntennaPatternData

8.6.2 Signal PDU

DIS Field	FOM Class	FOM Attributes
PDU Header	N/A	
Entity ID	EncodedAudioRadioSignal	HostRadioIndex
	RawBinaryRadioSignal	HostRadioIndex
	DatabaseIndexRadioSignal	HostRadioIndex
	ApplicationSpecificRadioSignal	HostRadioIndex
Radio ID	EncodedAudioRadioSignal	HostRadioIndex
	RawBinaryRadioSignal	HostRadioIndex
	DatabaseIndexRadioSignal	HostRadioIndex
	ApplicationSpecificRadioSignal	HostRadioIndex
Encoding Scheme	EncodedAudioRadioSignal	TransmitterSignalEncodingType
	RawBinaryRadioSignal	TDLMessageCount
	DatabaseIndexRadioSignal	TDLMessageCount
	ApplicationSpecificRadioSignal	TDLMessageCount
TDL Type	RawBinaryRadioSignal	TacticalDataLinkType
	DatabaseIndexRadioSignal	TacticalDataLinkType
	ApplicationSpecificRadioSignal	TacticalDataLinkType
Sample Rate	EncodedAudioRadioSignal	SignalSampleRate
	RawBinaryRadioSignal	DataRate
	ApplicationSpecificRadioSignal	DataRate
Data Length	EncodedAudioRadioSignal	SignalDataLength
	RawBinaryRadioSignal	SignalDataLength
	ApplicationSpecificRadioSignal	SignalDataLength
Samples	EncodedAudioRadioSignal	SampleCount
Data	EncodedAudioRadioSignal	SignalData
	RawBinaryRadioSignal	SignalData
	DatabaseIndexRadioSignal	DatabaseIndex, Duration, StartOffset
	ApplicationSpecificRadioSignal	SignalData, UserProtocolID

8.6.3 Receiver PDU

DIS Field	FOM Class	FOM Attributes
PDU Header	N/A	
Entity ID	EmbeddedSystem	HostObjectIdentifier
Radio ID	RadioReceiver	RadioIndex
Receiver State	RadioReceiver	ReceiverOperationalStatus
Received Power	RadioReceiver	ReceivedPower
Transmitter Entity ID	RadioReceiver	ReceivedTransmitterIdentifier
Transmitter Radio ID	RadioReceiver	ReceivedTransmitterIdentifier