Investigating Interoperability between JC3IEDM and HLA

Dirk C. Uys and Willem H. le Roux
Council for Scientific and Industrial Research
Meiring Naude Road
Pretoria, 0001
South Africa
+27 12 841 4758/4867
duys@csir.co.za, whleroux@csir.co.za

April 17, 2009

Abstract

The results of attempting to define a JC3IEDM (Joint Consultation, Command and Control Information Exchange Data Model) aware or compliant Federate Object Model for the High Level Architecture (HLA), are presented. The primary aim of the investigation was to firstly determine if it would be possible, and if so, design a partial prototype object model for contribution to the HLA community.

1 Introduction

HLA is a standard that facilitates communication between simulations or parts thereof [5]. It stipulates the method of interaction, the rules to adhere to and the data that can be exchanged (data model). There are different data models catering for specific needs. One such data model is the real-time platform reference federation object model (RPR-FOM) [8].

The Joint Consultation, Command and Control Information Exchange Data Model (JC3IEDM) is a data model developed by the Multilateral Interoperability Programme (MIP) to coherently describe all the entities of interest in a battle scenario and avoid ambiguities. [3]

Systems are not always modelled at the same level, for example the level of detail needed and data requirements of a simulator to train a pilot is different from those of a simulator used to plan the deployment of forces.

Sometimes it is desirable to combine simulations at different levels. Joint exercises are a good example of this - different forces can all collaborate in a single simulation in order to prepare for an event.

Exchanging data between simulations at different levels poses several problems. Lower level simulations may use HLA and an HLA data model like the RPR-FOM while Command and Control (C2) applications may use JC3IEDM. Using the same or a similar data model at different levels of simulation may enhance interoperability between C2 systems and other simulations.

In this report the possibility of translating JC3IEDM into a usable HLA FOM is investigated. We proceed by giving an overview of HLA and JC3IEDM. Thereafter the the possibility of creating a JC3IEDM compliant data model using HLA is discussed. Finally the findings are discussed and some suggestions are made.

2 High Level Architecture

The High Level Architecture (HLA) is a standard developed by the US Department of Defence to encourage interoperability, composability and reusability between modelling and simulation (M&S) applications [4]. There are two different versions of HLA available, 1.3 and IEEE1516-2000 [1]. Work is currently being done on an extension to the IEEE1516-2000 version called "HLA Evolved".

HLA provides a means for simulations to communicate between each other given a set of rules.

Communication can happen on a single system or over computer networks.

Entities that communicate using HLA are referred to as federates. These federates are grouped to form a federation. Each federate interacts with the RTI as indicated in figure 1. A single federate may be used to model a complete simulation, a collection of simulations or a small part of a larger simulation (see simulation granularity in [6]). This way HLA can be used to combine multiple simulations, to integrate a federate with existing simulations or to construct a distributed simulation.

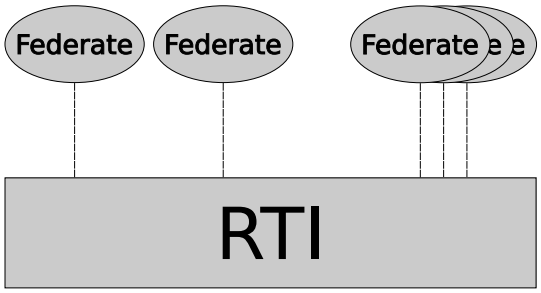


Figure 1: HLA Federation consisting of multiple federates

The three core parts of the HLA standard are the Object Model Templates(OMT), interface specification for the Run-Time Interface (RTI) and HLA rules. The OMT specifies how the object models should be defined, the RTI provides inter-process communications and the HLA rules specify the rules that should be adhered to when exchanging data.

The RTI component is the part that facilitates the actual communication between federates. HLA defines an interface that an RTI should expose in order to comply to the standard. There are several RTI implementations, for example MäK RTI [1], CERTI [2] and PitchRTI [3]. The choice of a specific RTI is normally a Federation wide decision since RTIs from different vendors are not necessarily interoperable.

[1] http://www.mak.com/products/rti.php
[2] http://www.cert.fr/certi/
[3] http://www.pitch.se/prti

2.1 Data Model

In order to be interoperable, federates must agree on the structure and scope of the data they will exchange. This is captured by the data model. The data model used by HLA is referred to as an object model. Federates that wish to inter-operate using HLA must use the same object model. HLA defines Object Model Templates (OMT) that specify the format of a Federation Object Model (FOM). The FOM is then used together with some other elements to form the Simulation Object Mode (SOM). A Management Object Model (MOM) is also defined, but will not be discussed here.

The OMT requires a FOM to specify two hierarchies, the object class structure and the interaction class structure. The classes in the object class structure are used to represent persistent data while the classes in the interaction class structure are used to represent temporal data. The object classes specifies a set of attributes while the interaction classes specifies a set of parameters. Object classes and interaction classes are very similar.

Attributes and parameters can consist of multiple data types. All the data types should be defined in the Data type tables as described in the OMT. Data types can contain other data types, but cannot inherit from another data type.

Although HLA defines a data model that corresponds to the concept of object orientation, it does not correspond to all the concepts of object orientation: only the object and interaction class structure can use inheritance and only the data types support composition. This should be kept in mind when defining an object model.

2.2 Data Exchange

A publish and subscribe method is used to exchange data between federates. A federate can choose to publish or subscribe to certain attributes or parameters of an object class or interaction class defined in the FOM. If a federate subscribes to a subset of attributes and an update to any of those attributes are published by another federate in the federation, the subscribing federate will be notified of the update.

When a federate is publishing certain attributes of an object defined in the FOM, the federate can determine through the RTI if there are any fed-

erations subscribed (listening) to the relevant attributes. If there are no subscribed federates, the publishing federate does not need to update the attribute values.

All communication between federates are done via the RTI. A RTI ambassador is used by the federate to send data to the federation and a federate ambassador is used by the RTI to send data to the federate.

The HLA standard does not impose any restrictions on any federates joining a federation (RTI implementations can impose licensing or security restrictions, but it's not defined within the HLA standard). A federate can receive any of the published data in a federation. A federate is also allowed to publish any of the data described by the FOM.

3 JC3IEDM

The Joint Command Control and Consultation Information Exchange Data Model (JC3IEDM) is being developed by the Multilateral Interoperability Programme (MIP). It is geared towards joint, interdepartmental and multi-national operations and aims to define a common well defined representation of all the data that needs to be exchanged in a multilateral C2 environment. All data must be uniformly represented, eg. an enemy unit must be represented in the same fashion as a friendly unit, to eliminate all possible ambiguities.

JC3IEDM is planned to be used as the default data model for all NATO operations by 2009 [7].

3.1 Data Model

At the top level, JC3IEDM specifies 19 entities (shown in figure 2). All of the entities are subclassed to provide more specific entities. These entities translate into tables in a relational database when the model is implemented.

At the core of the data model is the OBJECT-TYPE and the OBJECT-ITEM entities. The OBJECT-TYPE entity is used to represent information about a specific type (for instance a tank or a minefield). The OBJECT-ITEM entity is used to represent information about a specific instance of some type. A type can be assigned to an OBJECT-ITEM by means of the OBJECT-ITEM-TYPE bridge entity. This allows assigning multiple types to a single OBJECT-ITEM or reclassification of an OBJECT-ITEM.

OBJECT-TYPE and OBJECT-ITEM both have subtype hierarchies. These hierarchies are shown in figure 3. The first level of the hierarchies are parallel, but diverge thereafter.

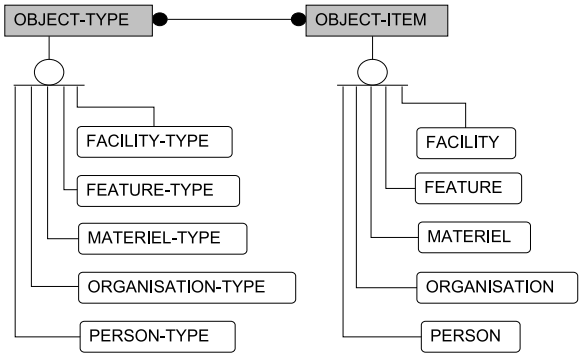


Figure 3: OBJECT-TYPE and OBJECT-ITEM subtype-tree. [3]

Since JC3IEDM is specified as a relational database, inheritance cannot be represented natively. In order to effectively achieve inheritance, derived entities are associated with their parent class by having the same key as the super class. The super class contains a field called category-code, that specifies the type of the object instance. This complicates the retrieval of data from a JC3IEDM database. To get all the information about a specific entity, multiple queries are needed to retrieve the information from the different tables. A simple *join* cannot be used since the value of the category-code cannot be used directly within a query to select the table to select data from.

3.2 Data Exchange (Mechanism)

There are three methods by which data can be exchanged between participating systems. The first method is by conventional interaction with a SQL database storing the information. This can be done directly by interacting with the SQL server, or indirectly by using a web service front end to the database.

Then there is the Message Exchange Mechanism (MEM). The MEM enables communication with JC3IEDM by message passing. Messages can be passed to the MEM using the simple mail transfer

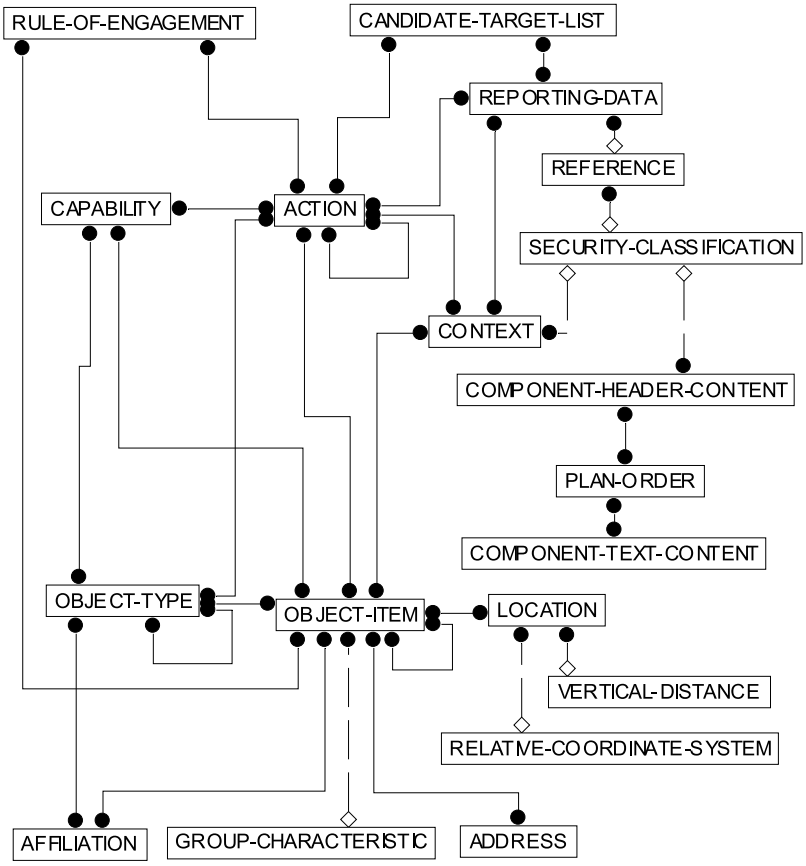


Figure 2: JC3IEDM Entities. [3]

protocol (SMTP). The MEM then interacts with the JC3IEDM database.

The last method is called the Database Exchange Mechanism (DEM). The DEM allows data exchange between different databases. DEMs connect directly to other DEMs using the Transfer Control Protocol (TCP).

4 JC3IEDM and HLA

When trying to inter-operate between applications using JC3IEDM and applications using an HLA FOM, there are several different ways to overlap an HLA FOM and JC3IEDM. The first option is to completely overlap an HLA FOM with JC3IEDM (figure 4a). This will require the creation of a FOM that can represent the whole of JC3IEDM. This may enable C2 simulations to be done using HLA.

Another option is to select only a part of JC3IEDM and create an HLA FOM to capture that part (figure 4b). The FOM can also be extended to describe additional data that is not within the overlapping region.

Yet another option is to use a bridge to translate between JC3IEDM and an HLA FOM (figure 4c). The HLA FOM used does not need to be aware of JC3IEDM as the translation between semantics is built into the bridge.

Before looking at the ways that an HLA FOM can be defined to capture the essence of JC3IEDM, it is worth having a look at possible patterns of interaction between various parts of possible systems.

4.1 Patterns of interaction

In parallel with the ways of overlapping the data models, there are a number of possible scenarios for combining systems or simulations using JC3IEDM with simulations using HLA. One possibility is to only use JC3IEDM (figure 5). Another possibility is to create a joint simulation using only HLA (figure 6). Yet another possibility is to combine applications using HLA with applications using JC3IEDM by using a bridge (figure 7).

Every scenario of combining systems and simulations have different requirements in terms of interoperability between the FOM used by HLA and JC3IEDM. Figure 5 shows a scenario where JC3IEDM is used. Although it is desirable to use a single representation of a single data model it may prove difficult to implement efficient kinetic simulations based on JC3IEDM only.

In the case where HLA is used for all IPC (figure 6), a FOM is needed that can satisfy the data interchange requirements of the C2 applications and simulations. Such a FOM may be an implementation of JC3IEDM in terms of the HLA OMT.

Figure 7 shows a scenario where C2 applications are using JC3IEDM and kinetic simulations are using HLA. This has the benefit of applications of different domains being used with familiar technologies in each domain. A bridge can then be used to translate from JC3IEDM to the data model used by the kinetic simulations. The kinetic simulations can use a FOM like the RPR-FOM, or a specialised FOM that corresponds more to JC3IEDM.

The next section presents the attempt to create a FOM that can be used in the place of JC3IEDM. This corresponds to figure 4a. This may be useful for the interaction patterns shown in figure 6 and figure 7.

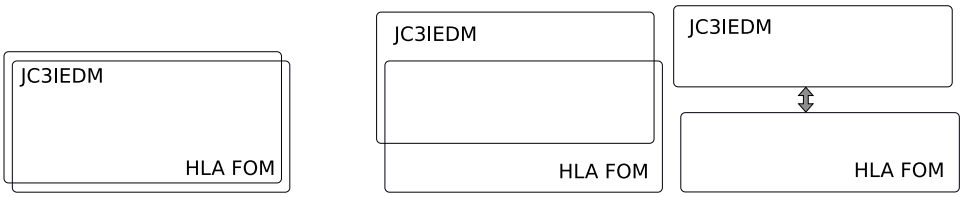
4.2 A JC3IEDM-aware HLA FOM

JC3IEDM is a comprehensive data model and investigating whether a complete mapping is possible is a tremendous task and not within the scope of this report. Therefore we will rather look at how certain types of data are represented in JC3IEDM and how that can be mapped to an HLA FOM. We start off by looking at spatial data.

4.2.1 Spatial data

Spatial data is of interest to any system that keeps track of entities' positions in two or three dimensions. It is an important aspect shared by C2 and kinetic systems and simulations. We will therefore look at how spatial attributes of an entity are represented using JC3IEDM and the possibilities of defining an HLA FOM to also represent these spatial attributes.

Figure 8 shows a subset of the database structure that can be used to specify a reported position of an OBJECT-ITEM. To represent spatial data in JC3IEDM the LOCATION entity is used. To associate one or more LOCATION entities to an OBJECT-ITEM, the OBJECT-ITEM-LOCATION bridge entity is used. Using the inher-



(a) 100% overlap (b) Partial overlap (c) No overlap

Figure 4: Using JC3IEDM and HLA data models together

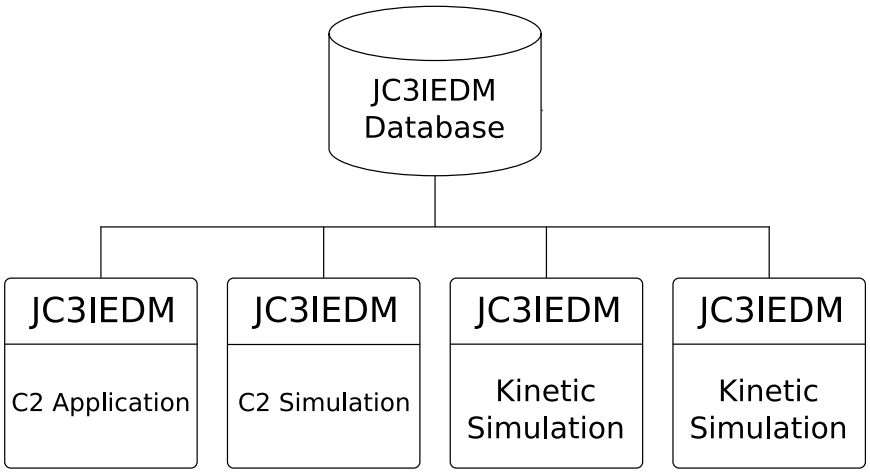


Figure 5: Using JC3IEDM to combine C2 applications and simulations with kinetic simulations

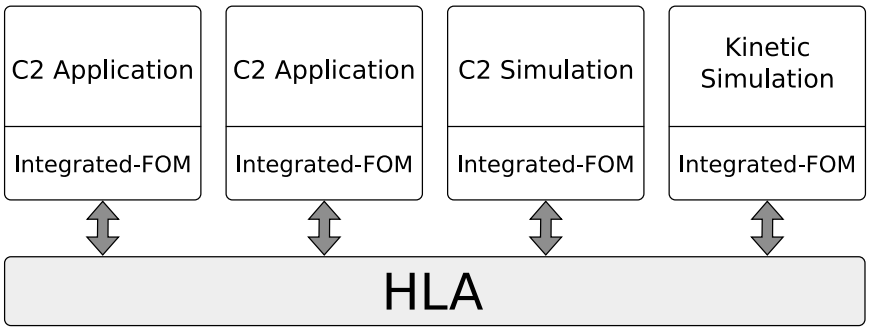


Figure 6: Using HLA to combine C2 applications and simulations with kinetic simulations

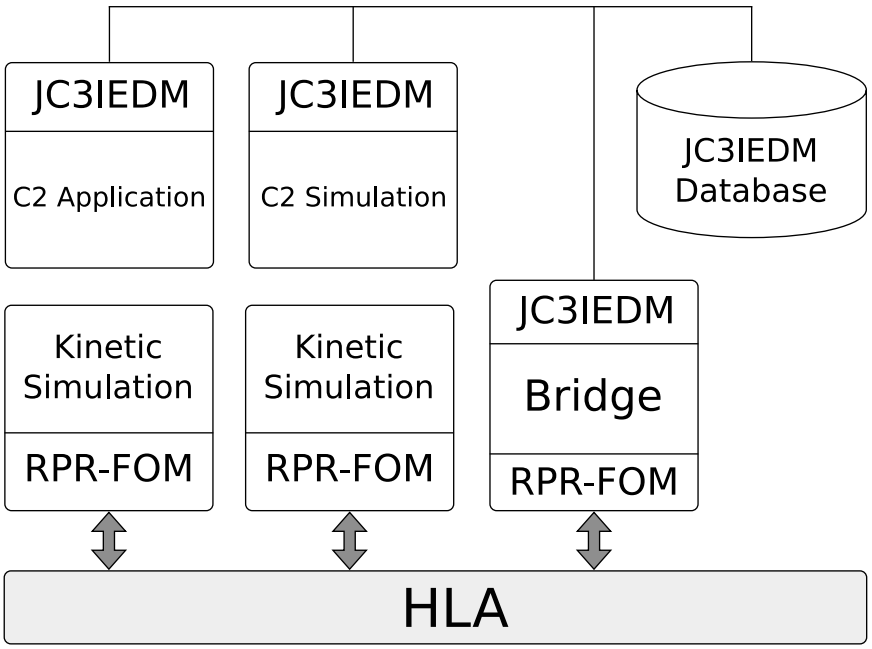


Figure 7: Combining JC3IEDM based applications with HLA based applications

itance scheme of JC3IEDM, the LOCATION can indicate an actual location represented in many different ways.

Figure 8 shows how to represent a location as a GEOGRAPHIC-POINT in JC3IEDM. GEOGRAPHIC-POINT is derived from ABSOLUTE-POINT which is a point in a geodetic system. A GEOGRAPHIC-POINT specifies the longitude and latitude. To indicate the height of an object, the VERTICAL-DISTANCE entity is used. The VERTICAL-DISTANCE entity specifies a relative height above some reference SURFACE. The GEOGRAPHIC-POINT uses the World Geodetic System 1984 (WSG84) as reference surface. The bridge entity OBJECT-ITEM-LOCATION also indicates where the information originated from by means of the REPORTING-DATA entity.

A naive way of implementing a structure like this in HLA is to directly map the subtype trees in the diagram to object hierarchies in HLA (see table 1). Every instance of an entity in JC3IEDM will correspond to an instance of an object in HLA.

While this may lead to a trivial mapping, various problems arise. Looking at an excerpt from the resulting attribute table (table 2, it can be seen that subscribing to LOCATION or ABSOLUTE-POINT in the HLA FOM will not provide any meaningful data. To get meaningful spatial data, one must subscribe to all the leaves of the LOCATION subtype tree (eg. CARTESIAN-POINT, GEOGRAPHIC-POINT, RELATIVE-POINT, etc). An update to the spatial information of an OBJECT-ITEM can then be caused by any of these subscriptions in HLA.

Another complication is that subscribing to an OBJECT-ITEM or any of its subtypes will not allow a federate to be notified of updates to the LOCATION relating to the specific OBJECT-ITEM since OBJECT-ITEM and LOCATION are separate objects in the FOM. One must also subscribe to the OBJECT-ITEM-LOCATION to know which LOCATION relates to which OBJECT-ITEM. And since this will mean that a federate is subscribed to the general spatial objects, it will be notified of all spatial updates within the federation. This may lead to significant performance overhead.

Another possible way to define the spatial structure using an HLA FOM is by using only the OBJECT-ITEM subtype tree for the object class structure of the FOM. These are generally the entities whose status are of interest to a simulation

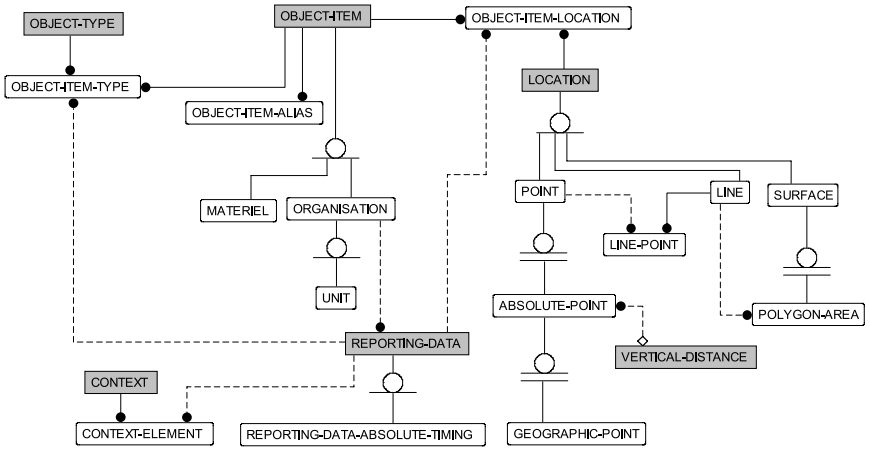


Figure 8: Reporting position in JC3IEDM. [2]

LOCATION	POINT	ABSOLUTE-POINT	GEOGRAPHIC-POINT
	LINE		
	SURFACE	POLYGON-AREA	
OBJECT-ITEM	MATERIAL		
	ORGANISATION	UNIT	
OBJECT-ITEM-LOCATION			

Table 1: Object class structure table (Direct mapping)

Object	Attribute	Data type
LOCATION	id	unique identifier
	category-code	enumeration
POINT	id	unique identifier
	category-code	enumeration
ABSOLUTE-POINT	id	unique identifier
	category-code	enumeration
	vertical-distance-id	unique identifier
GEOGRAPHIC-POINT	id	unique identifier
	latitude-coordinate	real number
	longitude-coordinate	real number
	latitude-precision-code	enumeration
	longitude-precision-code	enumeration
...		
OBJECT-ITEM	id	unique identifier
	category-code	enumeration
	name-text	text
...		
OBJECT-ITEM-LOCATION	object-item-id	unique identifier
	location-id	unique identifier
	index	integer
	vertical-accuracy-dimension	real number
	horizontal-accuracy-dimension	real number
	bearing-angle	real number
	bearing-accuracy-angle	real number
	bearing-precision-code	enumeration
	inclination-angle	real number
	inclination-accuracy-angle	real number
	inclination-precision-code	enumeration
	speed-rate	real number
	speed-accuracy-rate	real number
	speed-precision-code	enumeration
	meaning-code	enumeration
	relative-speed-code	enumeration
	reporting-data-id	unique identifier

Table 2: Attribute table (Direct mapping)

(like the classification, location, etc of an armoured vehicle). An example object class structure table is shown in table 3.

The rest of the entities are defined as data types that can be present in the attributes of the objects (see table 4). To handle the inheritance, a new data-type is defined for every entity in the inheritance tree. Each child entity's data type contains all the data defined by its parent entity. Notice the similarity between the LocationStruct in table 4 and the SpatialStruct defined by the RPR-FOM version 2.

The bridge entity OBJECT-ITEM-LOCATION are needed by JC3IEDM to associate a LOCATION with an OBJECT-ITEM. This becomes unnecessary when defining an HLA FOM. The LOCATION can be directly associated with an OBJECT-ITEM by defining a location attribute for OBJECT-ITEM (see table 5).

A complication with this approach emerges when having to store multiple positions (or types, etc.) for a single OBJECT-ITEM. Each position can be a different type of position (relative position or geographic location). Different type of positions will have different sizes. Storing this in a FOM creates difficulties since there is no intuitive way to define arrays with elements of variable size in an HLA FOM. You can define the size of every item in the location array to be the size of the biggest possible location type, but this can be very big and may change with the addition of a new subtype.

4.2.2 Capabilities

Figure 9 shows an excerpt of the structure used to represent capabilities in JC3IEDM. There are 10 entities derived from the base CAPABILITY entity (not all shown in figure 9). They are STORAGE-CAPABILITY, ENGINEERING-CAPABILITY, HANDLING-CAPABILITY, MAINTENANCE-CAPABILITY, MOBILITY-CAPABILITY, OPERATIONAL-CAPABILITY, SUPPORT-CAPABILITY, SURVEILLANCE-CAPABILITY, FIRE-CAPABILITY and TRANSMISSION-CAPABILITY.

Capabilities are associated to OBJECT-ITEMs by the OBJECT-ITEM-CAPABILITY entity and to OBJECT-TYPEs by the OBJECT-TYPE-CAPABILITY-NORM entity. This makes provision for the many-to-many relationship that can exist between CAPABILITIES and OBJECT-ITEMs or OBJECT-TYPEs.

Some of the capabilities, like STORAGE-CAPABILITY, ENGINEERING-CAPABILITY, FIRE-CAPABILITY and TRANSMISSION-CAPABILITY, have associations with other entities. The STORAGE-CAPABILITY for instance references the OBJECT-TYPE that can be stored.

The ACTION-REQUIRED-CAPABILITY entity allows specifying dependencies of ACTIONs on certain capabilities. This indicates the capabilities that are needed to complete the action.

In order to capture this data within an HLA FOM the object class structure defined as shown in table 6.

The CAPABILITY entity presents a problem. If it is included in the object class structure, the same problem is encountered as with the LOCATION structure as part of the object class structure as mentioned in the previous section. If, on the other hand, CAPABILITY is included as an attribute of OBJECT-ITEM and OBJECT-TYPE, a lot of redundancy may be introduced as multiple OBJECT-ITEMS or OBJECT-TYPES may have the same capabilities. Table 7 and table 8 show the Attribute table and the data type table for the second approach.

The association between CAPABILITY and ACTION would rather be made on the action side, since capabilities are a requirement for an action and it's more common to know what are required rather that what actions a capability can enable.

4.3 Differences

A few fundamental differences between HLA and JC3IEDM account for many of the problems when trying to create a mapping for JC3IEDM in HLA terms.

JC3IEDM is defined as a relational database management system (RDMS). This has the implication that data is obtained by querying a database. It is possible to get specific data on ad hoc basis. Data is kept in a few central repositories and interested parties do not need to keep a local state of the data. JC3IEDM also maintains a history of the information.

HLA works differently. It is based on a publish and subscribe mechanism. The publish and sub-

OBJECT-ITEM	MATERIAL	
	FACILITY	
	FEATURE	
	ORGANISATION	UNIT
	PERSON	

Table 3: Object class structure table

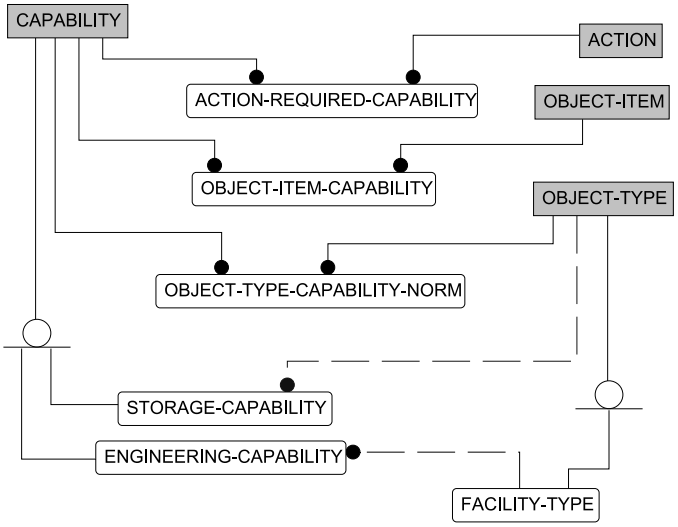


Figure 9: CAPABILITY adapted from JC3IEDM. [3]

Complex Data type	Field Name	Data type	Cardinality
LocationStruct	location-type	enumeration	1
	object-item-location	ObjectItemLocationStruct	1
	cone-volume	ConeVolumeStruct	0..1
	sphere-volume	SphereVolumeStruct	0..1
	surface-volume	SurfaceVolumeStruct	0..1
	geographic-point	GeographicPointStruct	0..1
	Cartesian-point	CartesianPointStruct	0..1
	relative-point	RelativePointStruct	0..1
	line	LineStruct	0..1
	corridor-area	CorridorAreaStruct	0..1
	polygon-area	PolygonAreaStruct	0..1
	fan-area	FanAreaStruct	0..1
	track-area	TrackAreaStruct	0..1
	orbit-area	OrbitAreaStruct	0..1
	ellipse	EllipseStruct	0..1
GeographicPointStruct	latitude-coordinate	real number	1
	longitude-coordinate	real number	1
	latitude-precision-code	enumeration	1
	longitude-precision-code	enumeration	1
	vertical-distance	VerticalDistanceStruct	1
VerticalDistanceStruct	reference-code	enumeration	1
	dimension	real number	1
	precision-code	enumeration	1
	datum-text	enumeration	1
ObjectItemLocationStruct	vertical-accuracy-dimension	real number	0..1
	horizontal-accuracy-dimension	real number	0..1
	bearing-angle	real number	0..1
	bearing-accuracy-angle	real number	0..1
	bearing-precision-code	enumeration	0..1
	inclination-angle	real number	0..1
	inclination-accuracy-angle	real number	0..1
	inclination-precision-code	enumeration	0..1
	speed-rate	real number	0..1
	speed-accuracy-rate	real number	0..1
	speed-precision-code	enumeration	0..1
	meaning-code	enumeration	0..1
	relative-speed-code	enumeration	0..1
	reporting-data	ReportingDataStruct	0..1

Table 4: Data type Table

Object	Attribute	Data type
OBJECT-ITEM	id	unique identifier
	location	LocationStruct

Table 5: Attribute Table

OBJECT-ITEM	MATERIAL	
	FACILITY	
	FEATURE	
	ORGANISATION	UNIT
	PERSON	
OBJECT-TYPE	FACILITY-TYPE	
	MATERIAL-TYPE	CONSUMABLE-MATERIAL-TYPE
		EQUIPMENT-TYPE

Table 6: Object class structure table

Object	Attribute	Data type
OBJECT-ITEM	id	unique identifier
	capability	Capability

Table 7: Attribute Table

Complex Data type	Field Name	Data type	Cardinality
Capability	day-night-code	enumeration	1
	unit-of-measure-code	enumeration	1
	engineering-capability	EngineeringCapability	0..1
	storage-capability	StorageCapability	0..1
EngineeringCapability	category-code	enumeration	1
	descriptor-code	enumeration	0..1
	facility-height-dimension	real number	1
	facility-length-dimension	real number	1
	facility-width-dimension	real number	1
	facility-type-code	enumeration	1
	facility-type-id	unique identifier	1
StorageCapability	cargo-category-code	enumeration	1
	descriptor-code	enumeration	0..1
	condition-code	enumeration	0..1
	object-type-code	enumeration	1
	object-type-id	unique identifier	1

Table 8: Data type Table

scribe method is tailored for notifying subscribing parties when changes of interest occur. With HLA, data tends to be distributed amongst the different federates and each federate must locally maintain the part of the current state it's interested in.

Although an HLA FOM resembles an object oriented data model, there are some issues preventing a traditional object-relational mapping. Traditional object composition is not supported by an HLA FOM. An object may contain a reference to another object, but if the state of the referenced object changes, it is not considered as a change of state for the referencing object. This is in contrast with traditional object orientation where the state of an object is determined by the state of its fields and where the fields themselves may be other objects.

JC3IEDM is designed to be extensible. Adding information to be associated with an OBJECT-ITEM is very easy. An entity needs to be defined for the additional information. This information can then be associated with an OBJECT-ITEM using a bridging entity.

Doing this in HLA is not that easy. The addition of information will require the addition of an attribute or the modification of a data type. The addition of an attribute may cause incompatibility with the older version of the FOM.

4.4 Conclusion and Future work

Attempting to create an HLA FOM to capture JC3IEDM brings several challenges to the surface. While these challenges may not be impossible to overcome they do highlight the fundamental difference between how a typical system using HLA and one using JC3IEDM will operate.

It can not be shown to be impossible to create an HLA FOM to cover all of JC3IEDM, but in the opinion of the authors it is more desirable to create an HLA FOM to capture a specialised part of JC3IEDM.

It would be interesting to investigate the possible role HLA may play together with the JC3IEDM DEM and the MEM. Currently a popular way of implementing the MEM and the DEM is by using the Simple Mail Transfer Protocol (SMTP) and the Transfer Control Protocol (TCP) to exchange data respectively. Using HLA for this data exchange may prove to be useful.

References

[1] *IEEE 1516, High Level Architecture (HLA)*, Mar. 2001. www.ieee.org.

[2] *JC3IEDM Annex O*, 2007. www.mip-site.org.

[3] *JC3IEDM Draft*, 2007. www.mip-site.org.

[4] D. Cutts, A. Bowers, and K. Brandt. Advancing a joint federation object model. In *Proceedings of the 2007 Spring Simulation Interoperability Workshop*, 2007.

[5] F. Kuhl, R. Weatherly, and J. Dahmann. *Creating Computer Simulation Systems: An Introduction to the High Level Architecture*. Prentice Hall, 1999.

[6] W. H. le Roux. Implementing a low cost distributed architecture for real-time behavioural modelling and simulation. In *Proceedings of the 2006 European Simulation Interoperability Workshop (EURO-SIW 2006)*, pages 81–95, Stockholm, June 2006.

[7] U. Schade and M. R. Hieb. Development of formal grammars to support coalition command and control: A battle management language for orders, requests and reports. In *11th International Command and Control Research and Technology Symposium*, 2006.

[8] SISO. *RPR FOM SISO-STD-001.1-1999*, 1999. www.sisostds.org.