1997

Building a Simulation Object Model of a Legacy Simulation / Simulation Interoperability Standards Organization (SISO) papers

Larimer, Larry R.; Buss, Arnold H.; Jackson, Leroy

Monterey, California: Naval Postgraduate School.

Building a Simulation Object Model of a Legacy Simulation

Larry R. Larimer
Graduate Student
Operations Research
Naval Postgraduate School
Monterey, CA

Arnold H. Buss
Adjunct Professor
Operations Research Department
Naval Postgraduate School
Monterey, CA

Leroy Jackson
Operations Research Analyst
United State Army
TRADOC Analysis Center—Monterey
Monterey, CA

KEYWORDS
SOM, HLA, ADS, OMDT, Janus, Legacy Simulation

ABSTRACT
The Department of Defense proclamation that all simulations comply with High Level Architecture (HLA) standards prompted the U.S. Army Training and Doctrine Command (TRADOC) Analysis Center (TRAC) to investigate the feasibility of including Janus in future HLA Federations. One of the Army's most extensively used models for training and analysis, Janus is an interactive, six-sided, closed, stochastic high-resolution simulation. Fielded in 1978, Janus was coded in FORTRAN and, with its numerous revisions and enhancements, represents a substantial investment for the U.S. Army. As a legacy model coded in a procedural language, there are considerable challenges for Janus to meet HLA requirements. For example, the notion of an object model definition was not envisioned during Janus development and is not intrinsic to its world view. In this paper we will describe the methodology we are using to develop a Simulation Object Model (SOM) of Janus independent of any existing federation of models. Our experience with this methodology will provide insight into the general problem of producing an HLA-compliant SOM for a legacy simulation.

1.0 INTRODUCTION

In this paper we discuss the background of the Janus simulation object model development process, the Janus combat simulation, and Janus as an analysis tool. We then describe the methodology used to create a simulation object model of Janus detailing the creation of the object class structure table, the attribute/parameter table, the interaction table, the SOM lexicon, the component structure table, the associations table, and the object model metadata. Finally, we highlight unresolved issues and preliminary results.

We discuss target acquisition in detail. This topic illustrates concerns which arise when identifying object model attributes and interactions. To use a simulation to for analysis, we define attributes and interactions beyond those required for distributed simulation interoperability.

2.0 BACKGROUND

In this section we discuss the project background and give detailed information about the Janus simulation.

2.1 Janus SOM Development Project

The United States Army has used computer simulations for years to train combat leaders and perform analysis. Simulation models have helped to determine optimal solutions to tactical, operational, strategic, procurement, and numerous other complex problems which otherwise would be difficult or impossible to solve by other means.

With recent advances in computer technology, the potential exists for the Army to significantly increase the efficiency and productivity of soldiers and leaders through the use of interactive virtual simulation training events and multi-unit/location training conducted through distributed simulations. Intense interest in the use of Distributed Interactive Simulations (DIS) for analysis and training have resulted in the development of a standardization program called the High Level Architecture (HLA).

The Defense Modeling and Simulation Office (DMSO) sponsored the development of the HLA to standardize the procedures for forming joint interoperating simulations. In the HLA, a federate refers to an individual simulation being considered for inclusion in a group simulation. The resultant group simulation is called a federation. The HLA facilitates simulation interoperability through the Simulation Object Model (SOM) and Federation Object Model (FOM). The idea is that each simulation is described using the SOM, and different simulations that are being considered for inclusion in a federation are reviewed for compatibility by comparing their individual SOMs. The SOM represents the information that a simulation can provide in a distributed simulation exercise. The result of this Federation development process is another object model called the FOM. The FOM represents a "contract" among the individual members of the federation that describes the public information that may be provided by members during the simulation execution. [2]

The Department of Defense has decreed that all DOD simulations will comply with HLA standardization requirements, or be replaced or excluded from distributed simulations. All military organizations that are proponents for individual simulations have been directed by DOD to review their legacy simulations over the next several months to determine if the simulations can and should be made HLA compliant

The Janus combat simulation has been used extensively and successfully by many military organizations. Janus represents a significant investment for the U.S. Army and we would like to provide a means for Janus to participate in future HLA federations. However, Janus, as a legacy model, provides some significant challenges in meeting the HLA requirements. Janus is coded in a procedural language with no well documented model definition. The purpose of our research is to determine if Janus can be described in an HLA compatible way in a SOM. If we are successful, then the road is paved for other legacy models to conform to HLA requirements and to participate in future HLA federations.

2.2 Janus Simulation

Janus is a high resolution, interactive, six-sided, closed, stochastic, ground combat simulation. Lawrence Livermore National Laboratories developed Janus to model nuclear effects, and the U.S. Army's Training and Doctrine Command (TRADOC) Analysis Center(TRAC) at White Sands Missile Range (TRAC-WSMR) is responsible for subsequent Janus development. TRAC-WSMR modified Janus extensively for Army high resolution combat model requirements. Since its fielding in 1978, Janus has been used extensively within the U.S. Army for both training and analysis. Janus is also used for analysis by RAND Corporation, the United States Marine Corps, and by the armed forces of the United Kingdom, Australia, France, and Germany.

Janus was coded in the procedural FORTRAN programming language and since 1978 has undergone numerous revisions (Janus 3.X/VMS Model Software Design Manual). Major changes in Janus occurred with version 4.0 which integrated terrain features (roads, water, buildings, vegetation, other man-made features) into the Janus environment. In addition to depicting these features on the Janus graphical terrain, algorithm adjustments in Janus code included consideration for these terrain features in Janus search and detection algorithms and probability of hit and probability of kill calculations. The current version of Janus is version 6.X which allows up to six different forces with varying enmity for coalition warfare. Previously, Janus allowed only two opposing forces.

Presently, the U.S. Army TRADOC Analysis Center at Monterey (TRAC-Monterey) has developed a version of Janus which is DIS compatible. This version of Janus linked to DIS (JLINK) allows the integrity of Janus to remain intact by enabling Janus to be DIS compatible through an external software package. This external software known as the World Modeler translates the Janus protocols to DIS protocols and vice versa as well as performs other functions for Janus that are required by DIS architecture. Some of these functions include dead reckoning, entity and terrain reconciliation, and where necessary, turn smoothing. Currently, JLINK can send and receive entity state, fire, detonation, radar emissions, and with specific DIS simulations obstacles, smoke, defilade status, prepared positions, minefields, and the ability to pause and resume an exercise during a run via DIS protocols. It has successfully interacted with four virtual and three constructive simulations which include both SIMNET and ModSAF. Further developments will enable JLINK to be more interoperable with a larger array of DIS compatible simulations with the eventual goal of releasing a DIS compatible version of Janus (Janus 8.X). [9]

A distributed version of Janus (version 8.0) is under development at TRAC-WSMR. The U.S. Army TRADOC Analysis Center at Monterey (TRAC-Monterey) has had significant success with a DIS Janus project called JLINK. JLINK consists of a slightly modified Janus simulation connected to a DIS network through a network gateway application known as

World Modeler. Through the World Modeller, Janus can interact with virtual and constructive simulations that are DIS compliant. In the JLINK form, Janus has been used for analysis of anti-armor weapons technology and is being used to allow National Guard units to train collectively as a battalion without leaving their home stations.

Janus models entities at the individual soldier and vehicle or aircraft level. Up to fifteen homogeneous entities can be aggregated for display and control. In Janus each member of an aggregation is fully represented and thus functionally independent. Janus can run in near real time if processing capabilities can support the level of interaction in the specific scenario. Janus will run slower than real time if too many entities are interacting. This is primarily due to the significant computing time required by the Janus target acquisition algorithm. Janus can also run faster than real time to expedite data generation. [4]

2.3 Janus as an Analytic Tool

Historically, Janus has been a highly successful analysis tool to research the effectiveness of new military systems and tactical doctrines. Two components are key for this success: a flexible database and a powerful post processor.

The robust representation of systems in the database allows the analyst to model new military systems and proposed modifications to existing systems. Systems are modeled as a combination of a platform, weapon systems, and sensors. The database includes nearly every ground vehicle combat system, dismounted crew-served weapon system, and Army rotary wing aircraft in the U.S. inventory and most of those used by threat nations. Systems that are not in the database can be easily created. Over 350 attributes are available in the database for the analyst to model platforms, weapon systems, sensors, projectiles, barriers, and weather. Table 1 provides the reader with a more detailed summary of the attributes in the Janus database available to model entities. The current version of Janus also represents limited types of fixed wing aircraft and precision guided munitions.

The Janus Post Processor details entity interactions that occur during the execution of each scenario. Output reports include an artillery fire report, indirect fire ammunition expenditure report, direct fire reports, detection tables, coroner's report, and killer/victim scoreboard. Additionally, Janus provides a supplementary tool, the Janus Analyst Workstation. This tool has an "instant replay" capability for viewing events graphically as they occurred during the scenario run. The Janus Analyst Workstation also provides statistical output that is synchronized with the scenario run time to assist the analyst. [4]

3.0 METHODOLOGY

It is important to understand that we approach modeling Janus as an HLA SOM from a theoretical perspective. Unlike the SOMs built for the proto-federations, we began this project with the Modeling and Simulation Resource Repository (MSRR) in mind rather than a specific target federation. We expected the finished SOM to be a more general representation of the simulation than some proto-federation SOMs. We approached the problem with the idea that our SOM would represent a conceptual mapping of Janus from its procedural state to an object representation of Janus, and with the intent that our work may be useful to others who may be interesting in bringing a legacy model into compliance with HLA.

We constructed of a detailed general object model for Janus as an intermediate step in the HLA SOM development. We found it more efficient to create this conceptual model of Janus and then extract the HLA SOM from it. The abstract classes in the object oriented representation of the conceptual model allowed us to identify classes to include in the SOM. The detail of the object oriented representation provided the attributes and interactions to complete the SOM tables.

Janus is not coded in an object oriented language. There are no declarations that allow us to easily identify the conceptual "objects" that are a part of Janus, nor is there a simple way to identify the attributes of these objects. Never the less, careful examination of the Janus commands, database, interactions, and to a lesser extent Janus' algorithms allowed us to produce a working set of objects, attributes and interactions. From this working set, we are now crafting the Janus SOM. We did not restrict our initial set of object attributes and interactions to those suggested by the HLA SOM. We included many terrain objects and private interactions in order to capture a more complete Janus object model.

3.1 The Object Class Structure Table

We produced the initial Object Class Structure Table using an organization chart format. This simple format provided a clearly defined class hierarchy and structure for later documentation in the HLA object model template. See Figure 1 for platform class hierarchy and Annex B for other class hierarchies.

The platform subtree of this class hierarchy is based primarily on the Janus database which lists each

platform the user might introduce into a scenario. Examples of these platforms include the M1A1 Abrams tank, the M2 Bradley Infantry Fighting Vehicle, and the individual rifleman. Using these platforms as the instantiable object classes in the class hierarchy, we categorized these classes, and worked upwards to produce a tentative hierarchy of abstract classes until we reached the base platform superclass.

This platform class hierarchy was refined to produce an alternative class hierarchy based on the army concept of battlefield operating systems. This illustrates the flexibility of the HLA simulation object model to provide more than one appropriate model of a simulation for military analysis and training. This alternative class hierarchy is not presented here.

We created a terrain class hierarchy to produce a more complete Janus object model even though this is not required under the HLA. In order to portray the terrain in Janus one must understand how the terrain and the combat platforms interact through the Janus algorithms. From the Janus graphical display, one can discern five basic terrain components: elevation, roads, buildings (towers, etc.), bodies of water, and vegetation. In Janus, the surface is partitioned into a gird with an elevation assigned to each grid cell. The number and size of cells is determined and then fixed within a particular database to support a scenario. The normal cell size for Janus is 100 meters by 100 meters. The elevation interacts with platform entities by restricting the speed that they are allowed to move and the Line of Sight (LOS) calculations in the Janus search and detection algorithm. Rapidly changing elevation reduces an entity's movement speed due to the slower movement rate required for steep terrain. Roads and bodies of water interact with platform entities in a similar manner.

The search and detection algorithm for Janus uses the elevation of each intervening cell between the searching entity and a potentially detected entity to determine LOS. The roads, buildings, vegetation, and water form a separate surface feature layer. The surface feature layer also interacts with the battlefield platform entities through the search and detection algorithm. After determining that LOS between two entities is not obstructed by terrain elevation, the algorithm adjusts the probability of detection appropriately based on the type of vegetation or building.

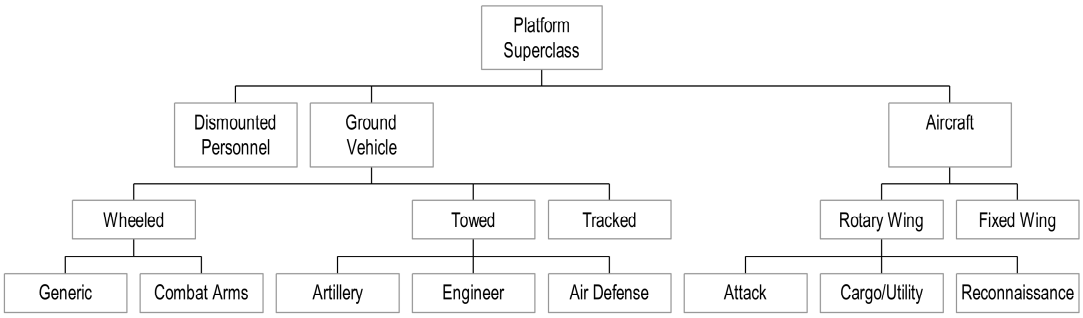

Figure 1. Janus Platform Class Structure Hierarchy

Atmospheric conditions are classified as objects in much the same manner as the terrain. Dust clouds, smoke, and fog all affect the detection algorithm in the same manner as vegetation and buildings. If the detection algorithm identifies one of these objects in the LOS between two objects, it degrades the probability of detection an appropriate amount based on the cloud thickness and type.

The barriers superclass was derived from the Janus simulation run initial parameters screen which allows the user to allocate barriers of six different types to each force in a scenario. While these barrier objects are depicted on the Janus graphical display symbolically like the combat platforms, barriers are not found in the Janus database.

Other object classes in the Janus object model are sensor objects, weapon system objects, and ordnance (ammunition) objects. These are components of the platform class and were derived from the Janus database.

3.2 Attribute/Parameter Table

The Attribute/Parameter Table is perhaps the most difficult of the OMT components to construct for a legacy model implemented in a procedural language. In the Janus architecture, every platform entity carries all the attributes available in the platform model. In object oriented terms, for platform entities, Janus really only has one object class--a superclass. Every object is an instantiation of this superclass with only certain

attributes filled with values. The values of these attributes define the type of object such as an A-10 fixed-wing aircraft or an individual soldier with a rifle.

Finding all these attributes requires detailed knowledge of the Janus database, commands, graphical display, and to a lesser degree, algorithms. For example, Janus platform entities can be suppressed. An entity that is suppressed has been engaged with direct or indirect fire and is unable to move or respond to the engagement for an amount of time specified by the user during initialization. This response is meant to model a situation where a soldier is receiving enemy fire of such intensity or precision that he is unable to move or return fire. However, there is no attribute in the database or command in the Janus command interface that alerts the user of this capability. One must know the Janus User's Manual in detail in order to identify this suppression state variable.

Another example is the location parameter. Each entity carries a parameter that tracks its current location (sometimes the last location prior to initiating latest movement). Again, this parameter is not in the database, commands available, or graphical display. However, each entity must store its current location to support search and detection calculations and determine engagement outcomes. Tables 1 through 3 depict the basic platform entity attributes we identified from the user's manual and graphic display, command interface, and the database.

User's Manual/Graphical Display
Status(Fully Operational/Casualty)
Suppressed
Entity Number
Smoke (VEES)
Passenger Status
Direction
Location
Destination
Speed

Table 1. Initial Platform Parameter List

Command Interface
Sprint
Hold Fire
Defilade
MOPP
Breach
Helicopter Pop-up

Table 2. Command Interface

Every platform entity in Janus, from the individual soldier to the M1 tank, is defined by values entered in the attributes listed in Tables 1 and 2 above and in Table 3 (see Annex B). However, in order to portray Janus in a SOM, it makes more sense to build an appropriate class hierarchy and include the attributes only at the appropriate level in the class hierarchy just as we would if we were redesigning an object oriented Janus. In other words, only attributes which correspond to the actual object will be listed for that class. The challenge is to identify the appropriate attributes for each level in the class hierarchy.

3.3 Interaction Table

There are three places in the Janus simulation to identify interactions between objects. The first source for interactions is the graphical interface. During a Janus simulation run, many interactions are portrayed graphically on the Janus battlefield screen. Engagements are the primary interactions depicted between platform entities. Between barriers and platform entities, interactions result in the destruction of the platform entity, or the halting of the platform entity's movement. Terrain interacts with platform entities by slowing the entity's movement as the elevation becomes steeper. This terrain/platform interaction is not depicted as explicitly as other interactions on the screen. One must follow an entity closely to see that its speed is slowing, or check the entity's speed using a status check option available in the Janus command interface.

The second source for interactions is the Janus command interface. Platform entities can be directed to mount onto another platform entity or to dismount from another platform entity. Also, the user can direct

a resupply platform to transload supplies to a combat platform. The third source for interactions is the Janus' algorithms. A review of Janus' algorithms reveals the interactions between the atmosphere class object and the terrain class object with the search and detection capability of a platform entity. Table 4 indicates the interactions in Janus with the method of identification annotated.

Platform vs. Platform
Engage with Direct Fire (Graphics Screen)
Mount on Platform (Command Interface)
Dismount from Platform (Command Interface)
Resupply Combat Platform (Command Interface)
Detect Platform (Graphics Screen/Algorithm)
Barrier vs. Platform
Destroy Platform (Graphics Screen)
Halt Platform Movement(Graphics Screen)
Atmosphere vs. Platform
Disrupt Search and Detect (Algorithm)
Terrain vs. Platform
Disrupt Movement (Graphics Screen)
Disrupt Search and Detect (Algorithm)
Barrier vs. Atmosphere
Create Smoke Cloud (Algorithm/Graphics Screen)
Platform
Engage with Indirect Fire (Command Interface)

Table 4. Interactions

Since the interactions are already identified, that part of the table is complete. The next step is listing the initiating and receiving classes. Some of these associations are easily resolved. For example in every case, a smoke pot object interacts with the atmosphere by creating a smoke cloud object. In other cases, the association is less well defined. The engage interaction between platform objects can be turned on or off by the Janus user. An M1 Abrams tank modeled by Janus must be specifically directed to engage other objects in the database by platform type (BMP, BTR-60, T-80 Tank). Therefore, the engage platform interaction reflected in the interaction table portrays engagement interactions that Janus is capable of initiating, sensing, or reacting to rather than what is allowed during any specific Janus scenario run. [5]

The attributes associated with Janus interactions can be identified in some cases by inspection (i.e. Alive/Dead Status for a platform that has been engaged). Other attributes are more difficult to identify and are found in the algorithms or in the database. A good example is the smoke cloud interaction with a platform object's target acquisition capability. A particular smoke cloud object provides a transmittance factor to the target acquisition algorithm. This transmittance factor is a parameter of the smoke cloud object expressed as a real number between 0 and 1. The thermal or optical signature of the potentially detected object is multiplied by the transmission factor resulting in a smaller signature value received by the observing object. The result of this interaction is a decrease in the probability that the observing object will detect the target object. [1,8]

3.4 SOM Lexicon

The OMT Lexicon required research into the definitions of object parameters and attributes. Most of this information was found in the Janus Database Manager's Manual, Janus User's Manual, or by inspection. Some cases such as the transmission factor parameter in the atmosphere class of objects required a detailed inspection of the target acquisition algorithm. We have not yet examined other SOM lexicons.

3.5 Component Structure Table

The component structure table followed directly from the object class structure table. Although not portrayed clearly in the Janus database, each platform has three sensor objects and zero to fifteen weapon system objects. If the platform contains a weapon system object, it necessarily must have appropriate ordnance objects. We considered ordnance objects a component of the platform rather than the weapon system because we felt that the type and quantity of ordnance is more closely associated with the platform than with the weapon system. Different platforms may carry the same weapon system but different types or quantities of ordnance objects. The Janus database allows each platform entity to carry any number of ordnance objects (limited only by user discretion).

3.6 Associations Table

We have identified two primary associations in Janus. They are the terrain/platform association and the weapon system/ordnance association.

All platform entities are associated with the terrain elevation object. As a platform moves across the terrain, the platform updates the elevation component of its location through this association. This association is important for the Janus object model, but not for the Janus SOM.

Ordnance objects are associated with weapon system objects. The weapon system fires or launches the ordnance. One could argue that this relationship is an interaction rather than an association. It is a momentary relationship as opposed to a lasting relationship. However, the relationship between ordnance and the weapon system is a usage relationship characterized by the HLA OMT Extensions Reference, Version 1.0, as an association rather than an interaction.

3.7 Object Model Metadata

The object model metadata table is straight-forward and requires only research into the back-ground of the Janus model.

4.0 ISSUES

Two issues have made the construction of the Janus SOM conceptually challenging. First, there was some question about the classification of an event which occurs when an object detects anther object. On the one hand, detection involves the interaction of two objects, and therefore could be classified as an HLA interaction. On the other hand, there is no explicit action taken by one object directed toward another. The searching object is simply scanning a field of search hoping to detect opposing force objects. Each time cycle the scanning object reviews a potential target list to determine if one or more potential targets are now observed. Although the scanning object does query the potential targets for their contrast and dimension data, this is transparent to the Janus user. Furthermore there is no potential to change the attributes of the potential target. These values are simply used in the target acquisition algorithm to determine if a detection occurs. Target acquisition/detection events are critical to any Janus scenario so this issue must be resolved. [1,8]

Secondly, atmospheric objects also impact on the detection event. It is thus possible for three or more objects to interact in Janus. A searching object may check a potential target that is shielded by one or more cloud objects. The cloud objects each attenuate the contrast attribute of the potential target as the searching object attempts to determine if it will make a detection. [1,8] This type of event is awkward to portray in the SOM.

5.0 PRELIMINARY RESULTS AND REMAINING WORK

In this section we describe the status of the HLA SOM tables and the remaining work to comply with the HLA federate rules.

5.1 SOM Tables

Although the SOM tables are 90% complete, we estimate that the Janus SOM is only 50% complete after 64 man-hours devoted strictly to production of the SOM (see table 5 for more detailed depiction of SOM status). Much of our time was spent reviewing the HLA References after producing the initial Janus object model to ensure the SOM complies with HLA requirements and meets with the Army's requirement for its use in analysis and training. For example, it was during this phase that we eliminated the terrain in the object class structure table. We excluded terrain from the SOM representation because each federate participating in a federation will have its own representation of the terrain so there is no need to publish/subscribe to this object class. We expect to spend another 60-70 man-hours on the project before a complete Janus SOM is produced.

Component	Completion Status
Object Class Structure Table	95%
Attribute/Parameter Table	70%
Interaction Table	70%
SOM Lexicon	80%
Component Table	90%
Association Table	80%
SOM Metadata	80%

Table 5. Current Status (6 January 1997)*

**Note that these completion percentages reflect the amount of time necessary to complete the table in terms of research to verify correct data and refinement of existing data in the tables as well as amount of the table completed.*

We are using the Aegis Research Object Model Development Tool (OMDT) to document the Janus SOM. This Windows95 based OMDT is a substantial improvement over our initial spreadsheet documentation. The OMDT links the OMT tables reducing errors and speeding model documentation. We found no inconsistencies in the completion of the OMT tables; however, we did encounter some minor difficulties with the command interface. We expect that these difficulties will be corrected in the final release of the software.

5.2 Work Remaining

Work remaining on the SOM will consist primarily of refining the tables produced thus far and identifying data that we have overlooked. Janus is a highly detailed simulation capable of representing hundreds of different types of entities and will require significant verification to insure that all facets and capabilities are captured in the SOM. Janus software developers at TRAC-WSMR will review in detail the content of the Janus SOM. JLINK developers at TRAC-MTRY will review the Janus SOM to assess its suitability to support a distributed Janus only federation.

In terms of the HLA rules for simulation federates, we have described the effort to comply with HLA Rule 6. The subsequent implementation work will satisfy the remaining four rules. (The six HLA federate rules are listed below.) Distributed Janus developers at TRAC-WSMR and TRAC-MTRY will review the Janus SOM to identify potential model related implementation difficulties not anticipated by the SOM developers.

6. Federates shall have an HLA Simulation Object Model (SOM), documented in accordance with the HLA Object Model Template (OMT).
7. Federates shall be able to update and/or reflect any attributes of objects in their SOM and send and/or receive SOM object interactions externally, as specified in their SOM.
8. Federates shall be able to transfer and/or accept ownership of attributes dynamically during a federation execution, as specified in their SOM.
9. Federates shall be able to vary the conditions (e.g., thresholds) under which they provide updates of attributes of objects, as specified in their SOM.
10. Federates shall be able to manage local time in a way which will allow them to coordinate data exchange with other members of a federation.

6.0 CONCLUSION

Our research indicates that Janus can meet HLA standards. Legacy model proponent organizations may benefit from detailed review of their simulations for possible compliance with HLA.

We recommend construction of a detailed general object model for legacy simulations implemented in procedural languages as an intermediate step in SOM development. We believe it is important to form a conceptual model of the legacy simulation in a general object oriented representation prior to mapping the simulation into an HLA SOM. We found it much easier to extract the HLA SOM from the conceptual model. We feel that the direct approach entails more risk that essential aspects of the legacy simulation will be overlooked. The object oriented representation of the conceptual model also documents the legacy simulation. We also found that by changing the structure of the class hierarchy, more than one useful conceptual model could be created to describe a legacy simulation.

We would not want to attempt to create an HLA compliant SOM without model development tools. The AEgis Research OMDT has greatly simplified SOM documentation and reduced the risk of errors. Other organizations with large complex legacy simulations would likely benefit from the ability to create conceptual models with commercial graphical object modeling tools and subsequently importing these models into an HLA object model development tool.

Significant work remains to complete and validate the Janus SOM. A detailed and complex model like Janus requires many man-hours to identify the components of the OMT tables, construct the tables, and finally refine and verify the data to arrive at an acceptable SOM which satisfies HLA Rule 6. The final phase of Janus SOM completion will include: continued refinement; capturing of objects, parameters, and interactions that were overlooked previously; and verification of the model.

7.0 REFERENCES

[1] *The Janus 3.X/VMS Model Software Programmer's Manual*, Contract No. DABT65-92-D-0002, November 1993.

[2] *HLA OMT Reference Version 1.0*, dated 15 Aug 96.

[3] *HLA OMT Extensions Reference Version 1.0*, dated 20 Aug 96.

[4] *The Janus Version 6.3 Software User's Manual*, U.S. Army Simulation, Training, and Instrumentation Command, Orlando, FL, 1995.

[5] *The Janus Version 6.0 Database Manager's Manual*, U.S. Army Simulation, Training, and Instrumentation Command, Orlando, FL, 1995.

[6] Larimer, Larry R., Master of Science Thesis, Operations Research, *Building a Simulation Object Model of a Legacy Simulation (Draft)*, Naval Postgraduate School.

[7] *OMDT User's Guide, Alpha 1.0*, Developed by Aegis Research and Sponsored by The Defense Modeling and Simulation Office. 1996.

[8] Parish, Randall M. and Alvin D. Kellner, *Target Acquisition in Janus Army*, U.S. Army TRADOC Analysis Command, White Sands Missile Range, NM, Oct 92

[9] "JLINK - A Distributed Interactive Janus" by Major Maria C. Pate and Major Glen G. Roussos. http://131.120.57.3/jlink/JLINKphlanxArticle.html

8.0 ABOUT THE AUTHORS

Dr. Arnold H. Buss is an Adjunct Professor of Operations Research at the Naval Postgraduate School. He received a B.A. in Psychology from Rutgers University, and M.S. in Systems and Industrial Engineering from the University of Arizona, and a Ph. D. in Operations Research from Cornell University. His research interests include simulation modeling and object-oriented software design. He is a member of INFORMS, MORS, and POMS.

Major Leroy A. Jackson is an army artillery officer with 19 years of enlisted and commissioned service. He graduated with a B.A. in Mathematics from Cameron University in 1990 and with an M.S. in Operations Research from the Naval Postgraduate School in 1995. He is currently assigned as an operations research analyst at the U.S. Army Training and Doctrine Command (TRADOC) Analysis Center (TRAC) Research Activities in Monterey, California and he continues his graduate studies at the Naval Postgraduate School.

Captain Larry R. Larimer is an army infantry officer with 14 years of enlisted and commissioned service. He graduated with a B.S. degree in Organizational Leadership from the United States Military Academy in 1986 and is currently pursuing a graduate degree in Operations Research at the Naval Postgraduate School. Captain Larimer's thesis research is the basis for this paper.

Annex A, Figure 2. Database Identified Platform Class Attributes

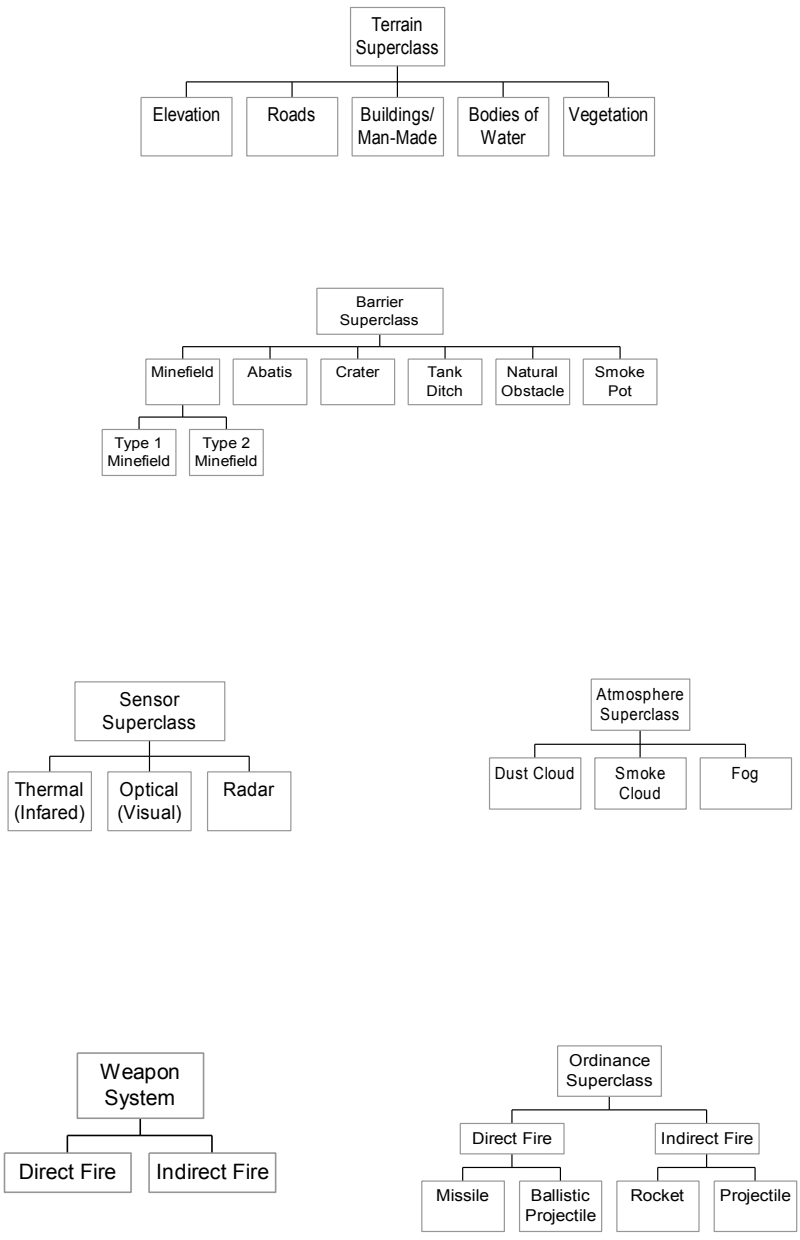

Figure 2. Database Identified Platform Class Attributes

Annex B Table 3, Database Identified Platform Class Attributes

Database
System Number
Name
Max Road Speed
Max Visibility
Weapon Range
Sensor Height
Crew Size
Element Space
Chem Transmit Factor
Graphic Symbol
Class Symbol
Host Capacity
Functionality
Laser Designator
Mine Disperse
Engineer Type
Firing Category
Flyer Category
Logistics Type
Movement Type
Radar Type
Smoke Dispersion
Surveillance Type
Swim Type
Weights/Volume
Normal Inclusive Weight
Normal Full Volume
Additional Weight
Additional Volume
Physical/Thermal Signat
Detection Data
Length
Width
Height
Sensor Type
Primary
Alternate
Defilade
Pop-up
BCIS Type
BCIS Function

Optical/Thermal Contrast
Optical Contrast
Thermal Contrast
Exposed
Defilade
Mine Vulnerability
Track Width
Belly Width
Total Magnetic Shadow
POL Data
Fuel Type
Tank Size
Consump, Stationary
Consump, Moving
Fuel Carrier
Smoke Grenade Data
Num Smk Gen Volleys
Weapons and Ordnance
Ord number
Relative
Absolute
Weapons/Ord Name
Basic Load
Upload Time
Weapon to use
Weapon Selection
Number
Name
Weapon 1
Range
Weapon 2
Firing Priority
Vulnerability to ARTY
Exposed
Protected

Table 3, Database Identified Platform Class Attributes