Combating Network Load in High Entity Count Federations

Jennifer Lewis, Diana Vagiakos
Science Applications International Corporation
Orlando, FL
jennifer.e.lewis@saic.com, diana.m.vagiakos@saic.com

ABSTRACT

As the Army places greater priority on collaborative and joint simulation capabilities, the technical community must enable their simulation software and network infrastructure to handle the network load that comes with large, geographically-dispersed federations. The Army Capabilities Integration Center (ARCIC) is the technical lead for the Omni Fusion 2008 and 2009 experiments. Omni Fusion 2009 includes the technical baseline for the Joint Experiment Earth Wind and Fire (EWF) 2009, and is also the technical path to the Omni Fusion 2010/TALON STRIKE, a distributed coalition event used by the UK to train a brigade headquarters for deployment to Afghanistan. Each ARCIC-lead Omni Fusion experiment employed OneSAF as the primary entity driver to successfully model and record approximately 20,000 engaging entities across a Wide Area Network (WAN) using the High Level Architecture (HLA). The first experiment attempted to handle the high network load by improving network scalability and process robustness. The second experiment relied on a comprehensive Data Distribution Management (DDM) strategy, which lightened the load on the network but increased software complexity. The similarity of these experiments' simulation environments offers a unique opportunity to compare technical approaches for handling network load in complex federations. This paper aims to serve as a guide for others in the community who are supporting similar high-entity count federations, both for experimentation and for training applications. It gives detailed lessons learned from both approaches, such as helpful RTI and OneSAF configuration parameters, operating system tweaks, and common DDM pitfalls. As the DoD moves toward greater reliance on distributed training of joint forces prior to deployment, the lessons learned by the Omni Fusion experiments will have great value to the community as we train to fight and fight to win.

ABOUT THE AUTHORS

Jennifer Lewis, CMSP, is a simulation engineer supporting ARCIC's Battle Lab Collaborative Simulation Environment (BLCSE). She holds a Master of Science degree in Telecommunications and Networking and a Bachelor of Science degree in Computer Science from the University of Texas at Dallas. She has designed and implemented network protocols for the telecommunications and defense industries, and spent the past five years transitioning the BLCSE environment from DIS to HLA in support of distributed experimentation.

Diana Mercedes Vagiakos is a software engineer supporting ARCIC's Battle Lab Collaborative Simulation Environment. She holds a Bachelors of Science Degree in Engineering Physics from Embry Riddle Aeronautical University. She has been involved in systems engineering of military satellites and missiles system and as a software engineer for modeling and simulation for the past six years.

Combating Network Load in High Entity Count Federations

Jennifer Lewis, Diana Vagiakos
Science Applications International Corporation
Orlando, FL
jennifer.e.lewis@saic.com, diana.m.vagiakos@saic.com

INTRODUCTION

The Army Capabilities Integration Center (ARCIC) is the technical lead for the OmniFusion 2008 and 2009 experiments, which lay the technical foundation for TALON STRIKE, a distributed coalition event used by the UK to train a brigade headquarters for deployment to Afghanistan. The OmniFusion experiments had similar strategic goals, technical requirements, and federation architectures. For example, they both exercised the flow of intelligence to and from division level staff. They both employed OneSAF as the primary entity driver to model up to 30,000 engaging entities among 10 geographically-distributed sites on the Defense Research and Engineering Network (DREN). They both communicated with remote sites using the Modeling Architecture for Technology, Research, and Experimentation (MATREX) Federation Object Model (FOM) across the MATREX Runtime Infrastucture (RTI)'s implementation of the High Level Architecture (HLA). Figure 1 depicts the OmniFusion 2009's federation architecture, which is similar to the OmniFusion 2008 architecture and includes numerous entity-level simulations, tactical systems, data recorders and federation monitoring tools.

One of the major differences between the two experiments is the technical strategies used to handle the data load on the network. OmniFusion 2008 attempted to handle the high network load by improving network scalability and process robustness, while OmniFusion 2009 relied on a comprehensive Data Distribution Management (DDM) strategy. This paper details the issues encountered through both approaches and provides lessons learned for handling network load in other complex federations. While the lessons learned can be applied to varying areas of experiment support, this paper is intended for a technical audience with basic knowledge of HLA and networking concepts.

OMNIFUSION 2008

The OmniFusion 2008 scenario was originally planned to contain approximately 3,000 entities. Therefore, the integration team did not see the need to implement Data Distribution Management (DDM) to control the data load on the network. By StartEx, however, the scenario contained more than 20,000 engaging entities. Although the team considered using DDM as the entity count grew, the engineers were concerned about the risk of making major changes to the software so close to the beginning of record runs, especially given the team's limited experience with DDM implementation and strategy design. Knowing a poorly designed DDM strategy could make federation performance worse, the team decided to focus on correcting the various federation and network stability issues already presenting themselves during integration testing.

Fragmentation

A network can only transmit a certain amount of data at one time. Network-specific information, such as the data's destination and whether the sender requires an acknowledgement, must also be transmitted with the data. As the amount of network-specific information increases, the amount of application data that can be sent decreases. If an application attempts to send too much data at one time, the network will break the data into smaller data packets, called fragments. Fragmented data must then be reassembled on the receiving end. The work of breaking and reassembling data packets can cause a considerable strain on the network. In addition, if any one of the fragments is lost in transmission, the entire data packet is lost.

The integration team used Wireshark, a freeware network protocol analyzer, to determine that much of the network instability was caused by an inordinate

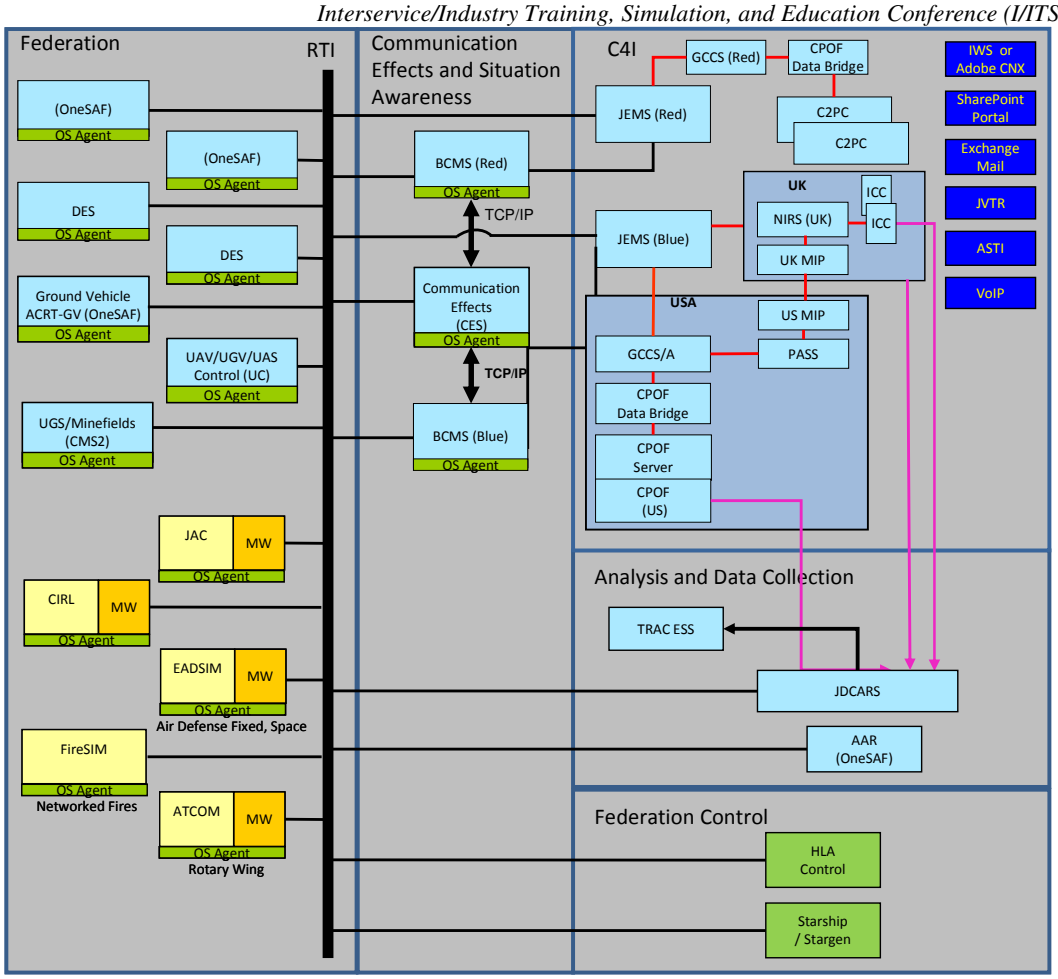

Figure 1. OmniFusion 2009 Federation Architecture

amount of fragmented data packets. The fragmentation was the result of additional network-specific data required by the DREN's tactical encryption routers. The team found that the default Maximum Transmission Unit (MTU) size, a value that controls how much data an application is allowed to put into a single network packet, was too large for the secure environment. The team reduced the MTU size to 1300 to allow room for encryption overhead within each data packet. The team made this change in each federate's operating system, the RTI Initialization Data (RID) file, and in all network routers, excluding the tactical encryption routers. After the change was made, Wireshark reported a dramatic drop in fragmentation, and federates were consistently able to join and resign the federation without incident.

Internet Protocol (IP) version 6

Network instability was also noted when IPv6 was enabled in a OneSAF federate's operating system. The problem stems from a simple misconfiguration of a OneSAF cluster. That is, the OneSAF installation guide specifically states to disable IPv6 since OneSAF does not support it. However, the problem was relatively difficult to track down, given the other network-related issues with which the team was dealing. The reason this setting causes network instability has to do with the way the Transport Control Protocol (TCP) validates incoming packets, a process called checksum. Checksums are calculated differently when TCP runs over IPv4 than when it runs over IPv6. Because OneSAF does not support IPv6, OneSAF could not validate any incoming TCP traffic over IPv6.

The problem is exacerbated by the fact that the sender, after not receiving an acknowledgement to its transmission, resends the data, which OneSAF again must process and interpret as corrupt. This cycle of false corruption put an undue burden on the network, especially when the data load was already significant.

Virus Scans

Another simple, yet frustrating and unexpected, source of network instability was virus scans. OmniFusion 2008 integration was extremely fast-paced, with federation testing running every minute of the work day. Facilities using closed storage could not leave their machines running after working hours, which meant that security-mandated virus scans could only be scheduled during the daily integration testing. The integration team, focused heavily on network configuration and scalability, did not consider such a simple explanation when federates suddenly could not join or resign the RTI successfully. The engineers were even more frustrated when the instability suddenly stopped, apparently for no reason. Inconsistent problems are by far the most difficult to solve, and it was only after several weeks of investigation that the actual source of this problem was identified.

Process Blocking

As the network load increased, some applications froze indefinitely, making no discernible progress after hours of runtime. The problem was eventually identified as an overburdened network connection between the federate's Local Runtime Component (LRC) and the RTI executive.

The solution was a change to the way the LRC interacts with the network connection. By default, the LRC returns control to the federate application once the data has been passed to the network's underlying transport protocol. However, because the network was too busy to accept the data, the LRC never returned control to the federate application. The integration team changed the *Federation.Networking.OrbPolicies.SyncScopePolicy* parameter in the federation RID file to allow the LRC to queue the data internally until the network could accept the data. The problem with this RID file setting is that if the network cannot accept the data for an extended period of time, the LRC could run out of memory in which to store the incoming federate data. If the LRC runs out of memory, the entire federate application will crash.

Reliable Transport Explained

Although modifying the RID file prevented federates from freezing, it did not alleviate the underlying problem. The network could not handle the federation's data load. More specifically, the network's TCP layer could not handle the federation's reliable data load.

TCP is designed to reliably send data across an unreliable network. Because data can be sent more reliably on a network that is not congested, the TCP layer controls the amount of data it sends using what is called a congestion window. The size of the congestion window dictates how much data the TCP layer can send out. The TCP layer must receive an acknowledgement of its transmission before any more data can be sent. If the TCP process senses that one of its messages has been lost or unduly delayed, the congestion window shrinks, and it shrinks quickly. One lost data packet typically cuts the window size in half. Delayed packets can cut the window size to 1. Therefore, lost and delayed packets can quickly cause huge, and sometimes unrecoverable, delays on the sending side. If the federate continues to attempt to send large amounts of reliable data when the congestion window is so small, the operating system's send buffers will eventually overflow, and what the application considered to be reliable data will be lost.

To add to the problem, as the network congestion clears, the TCP layer's congestion window opens very slowly. This, in addition to the non-stop flow of reliable data the federate application attempted to send, caused the frozen federates' network to never be able to recover. The problems became obvious by viewing the ever-increasing sizes of the operating system's incoming and outgoing message buffers, using the `netstat -ntp` command, which is available on most Windows and Unix-based operating systems. Because of these results, the technical team fine-tuned the TCP settings of the operating systems in the federation. The steps required to perform TCP tuning is operating system dependent, but many guides are available online. Unfortunately, even after TCP tuning, the message buffers could not handle the amount of reliable data in the federation.

Reliable Data over a Wide Area Network (WAN)

Another reason reliable communications are especially burdensome is that a separate data packet must be sent to each recipient, and that recipient must acknowledge its receipt. By contrast, unreliable data using the User Datagram Protocol (UDP) is sent only once; any application may receive it or not. Since the OmniFusion 2008 federation contained approximately 30 federate applications, every piece of reliable data was sent 30 times. When that much data is sent, the possibility of data loss is significant, especially across a WAN. To remedy this situation, the technical team configured the MATREX RTI to use a hierarchical interconnection

strategy. Each site connected to the WAN using a distributor node, which acts as a single TCP recipient for all of the applications at a particular site. Once the distributor node receives the TCP packet across the WAN, it can then resend the data out over its Local Area Network (LAN), where the possibility of loss or delay is theoretically much lower. This strategy significantly improves federation scalability by making more efficient use of WAN connections. After implementing hierarchical interconnection, each OmniFusion 2008 site hosting more than one federate ran a distributor node, which effectively decreased the amount of reliable data on the WAN by two thirds.

Unreliable Reliable Data

The distributor nodes made a noticeable difference in the amount of TCP traffic handled over the WAN. However, in times of congestion, the distributors' operating systems still suffered from overflowing TCP buffers. And just as the federate applications experienced, once the congestion began, the distributor could never recover and eventually had to be restarted. Restarting the distributor caused all of the data stored in its TCP buffers to be lost. The irony is that the most critical data items, the ones the federation agreed must be sent reliably, were the ones that were most likely to be lost. As integration continued, the technical team began to think of multicast as the reliable transport method.

The most noticeable problem caused by the lost TCP traffic was entities remaining in the federation after their modeling federate resigned. These orphaned entities remained because the removeObject callback was either lost when a distributor restarted or permanently delayed in a TCP message buffer. Several software workarounds were developed to handle missed critical callbacks. Unfortunately, there are no workarounds to address missed simulation data. Obviously, some data are more critical than others, which is why the FOM can be configured according to transportation type. However, an analytical experiment endeavors to be completely repeatable. Any lost simulation data lessens the validity of the overall experiment.

OMNIFUSION 2009

For OmniFusion 2009, the technical team knew from the beginning that there would be at least 20,000 engaging entities in the experiment. Therefore, the engineers placed high priority on DDM implementation and strategy as a way to limit data load on the network. In addition, the federation replaced the federate responsible for generating the common operating picture (COP). OmniFusion 2008 used a OneSAF Testbed (OTB)-based situational awareness (SA) server, which did not implement DDM. However, OmniFusion 2009 used the MATREX Battle Command Management Services (BCMS), which used DDM extensively for both reducing a federate's entity-level processing load and implementing federation services, such as delivering the COP to the divisional staff on the tactical network and performing comms adjudication. As a result, DDM became a requirement to achieve OmniFusion 2009 goals.

DDM Explained

DDM limits and controls the information exchanged by a simulation. It reduces data load on the network and processing load on federates by delivering updates only to those federates who require them. DDM differs from Declaration Management (DM) in that DM allows federates to receive particular object and interaction classes in which they are interested. DDM, however, filters based on the data associated with those objects and interactions. For example, a federate might only be interested in receiving updates from entities in a particular geographic region. DDM does this by calculating the intersections between sender-generated publication regions and receiver-generated subscription regions. The RTI only allows data to be exchanged by federates whose publication and subscription regions overlap. In Figure 2, the federate subscribed to S1 would receive updates for the federate publishing in P1. The federate subscribed to S2 would receive no updates.

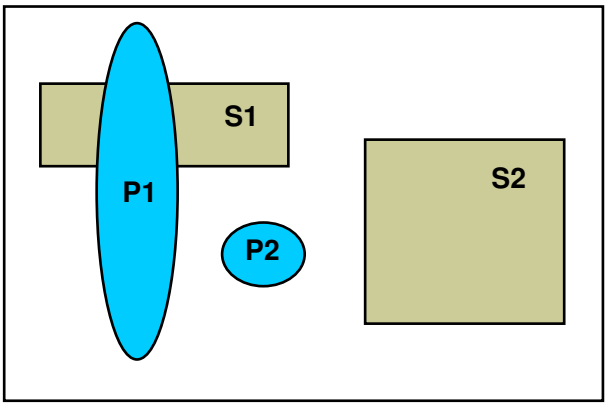

Figure 2. DDM Subscription and Publication Regions

Sender Side Filtering

An important detail to consider when deciding to use DDM is the way the RTI implements DDM. By default, the MATREX RTI performs the data filtering function within the receiver's LRC. This means that the RTI executive process still sends all federation data across the network. The receiving LRC then calculates the region intersections and filters out any data to which the federate application has not subscribed. While this approach lightens the federate's processing load, it does nothing to improve the situation on the network.

In order to lighten the data load on the network, the RTI needs to perform DDM's data filtering function within the sender's LRC. The MATREX RTI implements sender-side filtering if the federation's RID file *SpaceOptions* section is configured to include details of the routing spaces in use. These options allow the RTI to create partitions on which data can be directed only to those that subscribe to it. However, the use of these partitions can affect RTI efficiency in other areas, especially in the use of multicast channels. Therefore, only someone with a thorough understanding of all RID parameters should modify the *SpaceOptions* section.

The Flexibility of DDM

Although it is simple to envision DDM filtering based on geographic areas, DDM can filter based on any data. The filters are defined in the FOM as routing space dimensions. Depending on the FOM the federation is using, these dimensions can be quite abstract. For example, the MATREX FOM uses a dimension called RoutingNumber in many of its routing spaces. What is interesting about the RoutingNumber dimension is that it has no specific meaning assigned to it in the FOM. Therefore, it does not have to bear any relation to the data being filtered. If the federation would like to investigate which operating system experiences more TCP data loss, RoutingNumber could indicate what operating system the federate is being run on. If the federation would like to examine latency in receiving entity updates across the WAN during thunderstorms, RoutingNumber could indicate the current weather conditions at the federate's site. These values have nothing to do with the entity's affiliation or current location. However, they represent perfectly valid uses of DDM for more abstract purposes.

The Cost of DDM

Although DDM can provide necessary and clever features, there is a cost associated with using it. Obviously, calculating region intersections for thousands of updates per second places a burden on the RTI. In addition, the RTI uses attribute scope advisory callbacks to indicate when attributes come in and out of scope for the federate's subscription regions. When an entity enters the receiving federate's subscription regions, the receiving federate receives an *AttributeInScope* callback for each of the entity's attributes. When an entity is no longer of interest to a federate, it will receive an *AttributeOutOfScope* callback for each and every attribute. When the federation contains more than 20,000 entities, these callbacks consume a significant amount of bandwidth. And when the overall goal of DDM is to reduce the network's data load, these callbacks can seem self-defeating.

The callbacks can be disabled using the *Attribute Scope Advisory Switch* in the RID file or by calling *disableAttributeScopeAdvisory* in the federate software during runtime. However, without the callbacks, the federate has no way of knowing that the "out of scope" entity will never send another update. Without the callback, most of the OmniFusion 2009 federates continued to dead reckon this entity for 10 minutes – the predetermined timeout period for external objects. During those 10 minutes, the federate thinks it has a valid location and velocity for an entity when it does not. The "out of scope" entity could have reversed course or have been killed. The federate could attempt to engage or otherwise interact with an entity that is not there.

Whether to use the attribute scope advisories is an important DDM implementation decision that should be considered at both a federation level and a federate level. The overall goals of the experiment will help determine how to balance federate performance and data load against an accurate picture of the battlefield at any given instance.

One compromise the federation may consider is configuring the RTI, via the RID file parameter *FederationSection.Advisories.AttributeScopeAdvisories .TransportMechanism*, to send attribute scope advisory callbacks using UDP rather than TCP. This will help avoid TCP overuse while still attempting to deliver the needed callbacks using an unreliable protocol.

DDM Strategy

Once DDM implementation was complete, the technical team began examining the OmniFusion 2009 DDM strategy, focusing on geographic filtering to reduce

network load. The goal was to prevent federates from receiving data from entities that were well outside any of its entity's sensor ranges. The team attempted to divide the battlefield into appropriate grids, considering the entities' starting locations, probable engagement areas, and average sensor ranges. The engineers wanted to avoid dividing the battlefield in such a way that caused entities to continually cross over DDM region boundaries, which would cause additional overhead for the RTI and increase network load. In the end, the team determined that there was no guarantee as to what the operators would do during the experiment or where they would do it. Rather than implement a complicated geographic partitioning for no real gain, the team divided the battlefield into identical grids approximately three times the size of the average sensor range.

As it turned out, the decision to use a simple partitioning scheme was a good one, as only OneSAF used geographic filtering in the experiment. All other federates had some operational need to see the entire battlefield or at least large portions of it. For example, FireSimXXI must be able to conduct fire missions against targets outside of its entities' sensor ranges. Therefore, it needs to know about the existence of entities throughout the battlefield. As another example, the Advanced Tactical Combat Model (ATCOM) models aircraft that can fly at speeds of up to 200 knots. As its aircraft moves into a new DDM region, it can take up to two minutes before an external entity update from the new DDM region arrives from the network. By the time the update arrives, the ATCOM aircraft is six miles in front, thinking it has cleared the area behind it, only to find that an enemy suddenly appeared from nowhere.

However, even those federates not using geographic filtering were still required to use DDM to support BCMS's designs for COP generation and comms adjudication. The DDM strategy for both of these features relied on the FOM's more abstract routing space dimensions, RoutingNumber and Role, to control the flow of data to only those federates that should receive it. For example, BCMS used the RoutingNumber dimension to ensure the blue COP was received only by federates modeling blue entities. BCMS used the Role dimension to prevent communicated messages, such as salute and situation reports, from reaching their destination until the proper comms delay was applied. Although the primary purpose of these dimensions is to provide a service to the federation, they also helped reduce the data load on the network by limiting the number of federates receiving the messages.

Non-DDM techniques

Along with DDM, OneSAF took advantage of computers with dual network interface cards (NICs) by utilizing different cards to handle different types of network traffic. When a OneSAF is working in a cluster, separate machines work together to share the processing load of entity modeling. The separate machines must communicate with each other over the network. However, this intra-cluster communication is completely separate from the network traffic used to communicate with the rest of the federation via the RTI. Therefore, the integration team configured OneSAF to send intra-cluster communication on one NIC and RTI communication on a separate NIC. This simple configuration prevents overburdening one NIC with two distinct types of network data. To properly configure dual NICs, the integration team made changes to several RID file parameters, specifically *ProcessSection. Networking.FederateEndpoint*, *FederationSection. FederationExecutive.FederationExecutiveEndpoint* and *FederationSection.FederationInterconnect. HierarchicalEventChannelsOptions*. *InterconnectManagerEndpoint*.

In addition, the federation took steps to prevent federates from subscribing to data they did not need. Coming from a Distributed Interactive Simulation (DIS) environment, many sites were accustomed to receiving all of the data in the federation. Some sites received the COP and stored it, even if no operator used it. Others used god-view stations even if it was not required by the experiment goals. Although there was some cultural resistance to limiting the classes of data available at a particular site, this step helped to further reduce the amount of data on the network.

LESSONS LEARNED

As of this writing, OmniFusion 2009 record runs have not occurred. However, integration testing routinely runs vignettes containing more than 30,000 engaging entities, or approximately 50 percent more entities than were modeled during the OmniFusion 2008 record runs. While distributors continue to periodically exhibit symptoms of TCP overuse, it occurs much less frequently than it did during OmniFusion 2008.

The primary lesson learned through these experiments is that there is no magic bullet to lighten data load on the network. Without first addressing the more basic network issues, such as fragmentation, during OmniFusion 2008, the follow-on experiment could not

have been a success. And while DDM can provide an architectural way to limit the amount of data sent, a poorly designed DDM strategy can negatively impact data load by placing undue overhead on the RTI and significantly increasing the number of RTI callbacks crossing the network.

Transportation Type

The ultimate key to reducing network load is reducing the amount of reliable data. Unfortunately, the desire for highly repeatable analytical experiments means that TCP will most likely be overused in the future as well. Given this, ARCIC recommends federate applications develop their software under the assumption that RTI callbacks are delivered best effort. This means designing software that can handle missing critical callbacks. For example, the OmniFusion Federation Agreement now states that applications must remove an external object if it is not heard from for 10 minutes. This agreement shields the federation from the orphaned entities caused by lost removeObject callbacks.

In addition, ARCIC recommends tailoring the FOM transportation types to suit the needs of each experiment and limiting the number of reliable data elements to only those that are most critical. For example, OmniFusion experiments changed the transportation type of the Situation Report interaction from reliable to best effort in the FOM. The federation originally decided these interactions were critical to generating the full COP, especially given OmniFusion's goal of evaluating the flow of intelligence. However, because OmniFusion federates send regular entity updates, the impact of losing one Situation Report is limited. On the other hand, the effect of losing one Munition Detonation interaction, especially for a high powered munition, is unacceptable.

Understand the RTI

Thoughout the OmniFusion experiments, the integration team was in contact with the MATREX RTI developers. When what appeared to be insurmountable obstacles appeared, the RTI developers were often able to identify a better way to configure the federation RID file. At the very least, they helped the integration team understand what the existing parameters meant and how to comprehensively use the RTI logging functions and status tools to debug problem areas. Because the RTI is the heart of the HLA federation, and most RTIs have extensive RID file options, ARCIC recommends that the technical team become thoroughly familiar with the chosen RTI's implementation and configuration well in advance of the experiment.

Cultural Issues

DDM is difficult from a technical and cultural point of view. Many of the OmniFusion federate proponents were more comfortable with hardware solutions to limit network load, such as network segmentation using area of interest routers, because routers were used in previous DIS-based experiments. Therefore, these solutions are more familiar and easier to understand. It is recommended that federations introduce DDM concepts and capabilities at both a technical and managerial level before attempting to implement a new DDM strategy. This will reduce the cultural resistance to DDM and help explain how DDM can be used to provide services to the federation, in addition to limiting network load.

SUMMARY

Whether handling technical issues specific to a secure environment or implementing DDM concepts to simplify federation architecture, the technical community must enable their simulation software and network infrastructure to handle the network load that comes with large, geographically-dispersed federations. As the DoD moves toward greater reliance on distributed training of joint forces prior to deployment, the lessons learned by the OmniFusion experiments will have great value to the community as we train to fight and fight to win.

REFERENCES

Comer, Douglas E. (2006). *Internetworking with TCP/IP:Principles, Protocols, and Architecture*. 5th ed. Prentice Hall.

Snively, K. & Wilson, A. (2004). *Scalable Reliable Data Dissemination for Distributed Simulations using Hierarchical Interconnect.* Proceedings of the Simulation Interoperability Standards Organization Spring 2004 Simulation Interoperability Workshop.