20th ICCRTS
"C2 Agility: Lessons Learned from Research and Operations"

Linking C2-Simulation Interoperation Servers to Form Distributed Server Systems

Topics
Experimentation, Metrics, and Analysis
Modeling and Simulation
C2-Simulation Interoperability (primary)

Names of Authors
J. Mark Pullen
George Mason University C4I Center

Lionel Khimeche
Direction Générale pour l'Armement (DGA)
France

Xavier Cuneo
AIRBUS DS
France

Ulrich Schade and Thomas Remmersmann
Fraunhofer Institute for Communication, Information Processing and Ergonomics FKIE
Germany

Point of Contact
J. Mark Pullen
4400 University Drive, Fairfax, VA 22030
703-993-1538
mpullen@c4i.gmu.edu

ABSTRACT

Since 2003, a community focused on interoperation of command and control (C2) systems with simulation systems has developed a new area of technology, formerly known as Battle Management Language and more recently generalized under the name C2-Simulation Interoperability (C2SIM). Their vision is that a common basis for interoperation will lead to a future where military organizations can link their C2 and simulation systems, without special preparation, in support of coalition operations. C2SIM coalitions assemble complex systems of systems that can be seen as a strong basis for future agile C2 solutions to coalition interoperability. Such systems generally include a server that stores XML documents representing Orders and Reports to be exchanged and distributes them among the interoperating systems.

Recent advances in C2SIM system architecture involve distributed server systems that can provide for higher aggregated rates of message delivery and more efficient use of network capacity. This paper reports on the authors' work to combine multiple, heterogeneous C2SIM servers in flexible configurations of distributed server systems. The paper provides a summary of current battle management language (BML) server technology, focusing on why and how servers can be linked and the technical issues involved. Descriptions are provided for three different servers that operated as a distributed server system in 2014. The servers are compared and contrasted, including a description of the approach used to link them and technical issues encountered in the process. The paper concludes with a projection of the future of distributed server systems for C2SIM.

1. Introduction

Problems of military command and control (C2) are more difficult in the coalition environment, where each nation is likely to have different doctrine, equipment, and C2 information system (C2IS, also called C2 system). The difficulty is even greater where the national forces in the coalition are capable of incorporating simulations to increase the functionality of their C2IS. A new technology is emerging that enables C2-simulation interoperation (C2SIM), dealing with these problems to support interoperation of all the national C2IS and simulation systems.

This paper is about the role servers play in C2SIM interoperation among C2 systems and simulation systems and how this interoperation benefits from a distributed system of servers. The benefits will be illustrated by examples of such interactions exhibited by demonstration of C2 and simulation system interaction in the context of the research by NATO Modelling and Simulation Group 085 (MSG-085). In this introduction we first will discuss the goals of C2SIM and C2SIM interoperation. Then, we will comment on the role of C2SIM in the NATO MSG. After that we will relate the MSG-085 research to respective Simulation Interoperability Standards Organization (SISO) efforts. At the end of this introduction we will outline the rest of this paper.

1.1 Goals of C2SIM

Interoperation among C2 and simulation systems is a common and significant theme in the transformation of modern military forces. It is required to support the military enterprise in the execution of business activities and mission threads such as operational training, information sharing and decision support. This requirement implies the ability to seamlessly integrate C2 systems and simulation systems and to provide the means for a meaningful and unambiguous information exchange. C2SIM interoperation applies to systems of systems functioning toward a common goal on at least three different levels: (1) within services, (2) across services (i.e. joint) and (3) across nations in a multinational or coalition context.

Interoperation among C2 and simulation systems is required to support military activities such as: Force Readiness; Support to Operations; and Capabilities Development. Currently, interoperation of systems from different manufacturers and/or nations requires proprietary interfaces that cost time and money to be developed and to be maintained. Furthermore, in many cases, in addition to these vendor-specific interfaces, human intervention is required during military scenario definition, initialization and execution. The so-called “swivel-chair” interface entails feeding simulation operators with information that they must translate manually into instructions that the simulations can process. Replacing such operators with a standardized, automated interface can save considerable expense and at the same time result in a more robust and timely operations.

Developing standards that define common interfaces for the exchange of military information among C2 and simulation systems therefore can lead to significant cost-reduction and greatly facilitates systems integration. The benefits of standardizing C2SIM interoperation include: reduced cost and workload; reduced scenario preparation time; and increased realism and overall effectiveness. Furthermore, the advent of autonomous and unmanned vehicle systems (UVS) has led to requirements for increased interoperation among C2 systems and the emerging category of robotic forces. Increasing employment of unmanned systems creates the need to develop and validate new concepts of operation and therefore the need for experimentation capabilities. The requirements for communication between C2 systems and robotic systems are in many ways similar to those for communication between C2 systems and simulation systems.

In such a “systems of systems” environment, the control of one system by another requires an unambiguous, automated mechanism wherein C2 and modeling/simulation concepts can be linked in an effective and open manner. Furthermore, stakeholders have recognized the importance of establishing an internationally accepted standard that provides for a system-independent language and protocols. C2SIM has as its goal the ability to exchange C2 and situational awareness information in a seamless, transparent

form without building custom interfaces among the various C2 and simulation systems. To achieve interoperability, each system implements the vocabulary and syntax of the Military Scenario Definition Language (MSDL) and Coalition Battle Management Language (C-BML) standards [1, 2] as a generic interface that can work with any other system implementing the same standard. As a result, coalition forces will be able to prepare rapidly for new missions, learning to deal with the unique aspects of each national force while preparing those forces to work together toward their shared mission. The current architecture for C2SIM uses a server for exchange of C2 orders and reports, packaged as XML documents. Reference [3] describes this capability, up to and including a two-server distributed system. This paper reports on recent advances in server design that enable powerful distributed, heterogeneous server employing more than two servers, to increase the performance of C2SIM systems.

1.2 C2SIM and the NATO MSG

The mission of the NATO Science & Technology Organization (STO) is to help position the Nations' and NATO's science and technology (S&T) investments as a strategic enabler of the knowledge and technology advantage for the defense and security posture of NATO Nations and partner Nations, by conducting and promoting S&T activities that augment and leverage the capabilities and programs of the Alliance, of the NATO Nations and the partner Nations, in support of NATO's objectives, and contributing to NATO's ability to enable and influence security and defense related capability development and threat mitigation in NATO Nations and partner Nations, in accordance with NATO policies. The NATO MSG is part of the STO.

The NATO MSG charters technical activities conducted by groups from nations that are members of NATO or its associated Partners for Peace, to improve understanding and utility of technology involving modeling and simulation. Technical feasibility of coalition BML was demonstrated by NATO MSG-048 in a Technical Activity conducted 2006-2009. References [4-7] describe the C2SIM environment developed for NATO Technical Activity MSG-048, *Coalition Battle Management Language*. This activity included six national C2 systems, five national simulations, and two supporting software systems, a scale of interoperation not previously attempted. The follow-on Technical Activity, MSG-085, *Standardization for C2-Simulation Interoperation,* was chartered to demonstrate and facilitate the operational utility of MSDL and C-BML in military coalitions, and also to assess the operational relevance of MSDL and C-BML while enhancing the technical readiness level of their available implementations. [8, 9] describe developmental work for MSG-085, leading to an experimental operational environment where multiple national C2 and simulation systems can interoperate using MSDL and C-BML (see Figure 1).

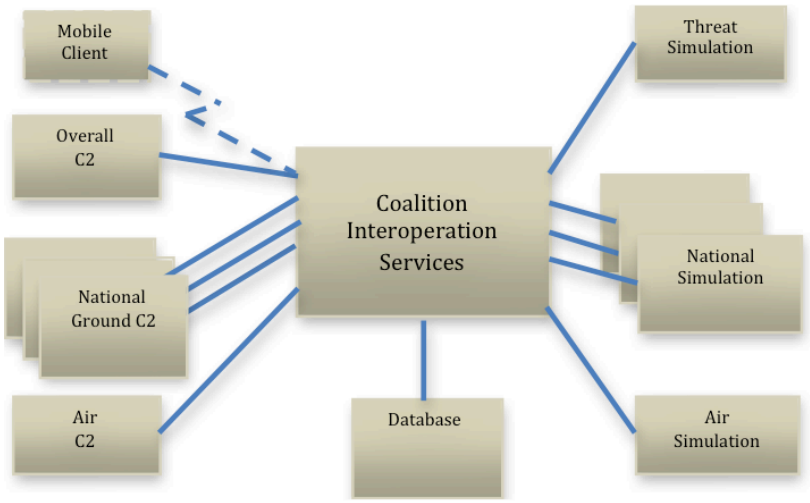

Figure 1: C2-Sim Coalition Services Architecture

1.3 C2SIM and SISO

SISO has a two-part standards effort supporting BML. The MSDL standard [2] was approved in 2008. It is intended to reduce scenario development time and cost by enabling creation of a separable simulation-independent military scenario format, focusing on real-world military scenario aspects, using the industry standard data model definition eXtensible Markup Language (XML) as input to initialize C2 and simulation systems. The C-BML standard [3] provides the tasking and reporting aspects of C2SIM. It was balloted in 2012 and approved as a SISO standard in 2014. The C-BML approach has generally followed the Lexical Grammar approach introduced by Schade and Hieb [10, 11]. Recent studies and implementations have addressed effective combination of MSDL and C-BML [12, 13]. Informing the standardization process have been multiple projects under various US DoD sponsors and an ongoing sequence of experimental BML configurations that were developed and demonstrated by the members of NATO MSG-048 and MSG-085. The experience gained in these activities has proved critical to shaping the MSDL and C-BML standards and implementing infrastructure, such as the servers described in this paper, and also in demonstrating the potential applicability of BML.

The remainder of this paper is organized as follows: Section 2 provides a technical overview of the basic C2SIM architecture; section 3 describes expansion of that architecture to one using distributed servers; section 4 describes the first public demonstration of a three-server implementation; this is followed by a concluding section.

2. C2SIM Architecture Details

A C2SIM Coalition is a system of systems. The basic architecture for C2SIM is quite simple: all C2 and simulation client systems connect to a C2SIM service, which copies the messages to its database and also replicates messages received from any one of them to the others. The general message architecture for Web service based C2SIM is shown in Figure 2.

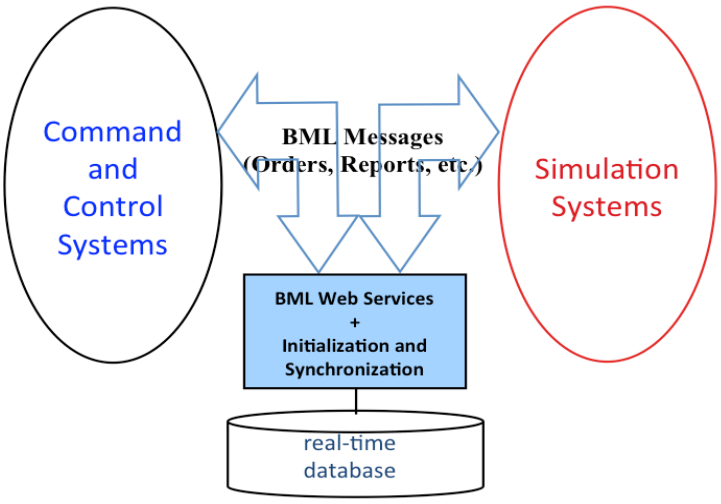

Figure 2: C2SIM Message Architecture

2.1 XML Schema and Message Flow

Clients provide inputs in XML format, using an agreed schema (for example, C-BML Light). It is optional for more than one such schema to be used; in this case all of them must have semantically equivalent data and the server must translate among the schema formats.

Inputs are made using the Representational State Transfer (REST) protocol, which provides for transmission of an Order or Report to be transferred. Like any Web service, each input message requires a separate connection of the underlying Transmission Control Protocol (TCP).

Each client also connects to the server's Simple Text-Oriented Messaging Protocol (STOMP) output, providing one or more Topics (for example, "C-BML Reports"). The STOMP connection is associated with a persistent TCP channel; all input messages belonging to the requested topic are replicated via that channel to the client.

2.2 Role of C2 and Simulation Clients

Clients both produce and consume XML messages: C2 systems produce Orders and consume Reports, while simulation systems consume Orders and produce Reports. Each client system must therefore be configured to assemble its required output in XML format and make a REST connection to the server for every event that requires a transaction. It also must open a STOMP connection to the server for any Topic that provides XML documents it needs as input. In addition, for effective operation, C2 systems must have a reliable way of identifying to the user whether they are operating in simulated or live mode, and simulation systems must have a reliable way for their operation to be started and stopped together with all other simulation systems in the coalition. Both types of client must be able to accept configuration information as specified in the MSDL standard.

2.3 Role of the Server

C2SIM servers in use today are of two types: *document* servers transfer the complete XML input document intact to their output channel(s), while *parsing* servers read the document and break it down into individual data elements, which in turn are used to assemble output messages. The document style of service is much less computationally demanding, so that higher performance is easier to obtain. However, document servers are not capable of translating among the C2SIM schemata. Translation may be required because C2SIM continues to evolve; this can result in cases where various clients that are candidates for interoperation do not share a common schema. This was the case in the MSG-085 Final Demonstration

[14]. Servers also may log copies of the message flow from their use; this supports after action review and scenario replay.

MSG-085 has used multiple servers: the Fraunhofer FKIE BML Server; the WISE-SBML Server, and the ELLIPSE server. The FKIE and ELLIPSE servers function in document mode only; the WISE-SBML server was used in parsing mode, enabling a mixture of schemata to be used by the clients and other servers. More details about these three servers are given below.

3. Multi-Server C2SIM Architecture

Figure 3 shows the two-server architecture that was used for the MSG-085 Final Demonstration [15], which took place at Fort Leavenworth, December 2013. The architecture shown in Figure 3 can be seen as the starting point for multi-server C2SIM architecture, with its detail of server interconnection shown in Figure 4. Figure 5 shows the next step, the three-server architecture that was demonstrated in December, 2014, and Figure 6 shows a completely general mesh configuration of distributed servers. In the following, we will discuss the evolution from the architecture shown in Figure 3 to the generalized architecture shown in Figure 6.

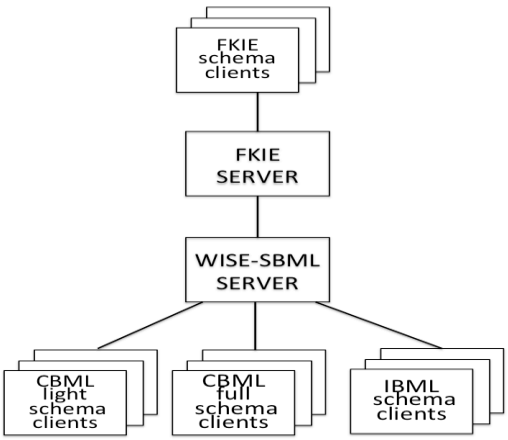

Figure 3. Two-Server Architecture

3.1 Connecting Two Servers

Figure 4 shows how a pair of Back-to-Back (B2B) clients interconnect any two servers which might be part of a multi-server configuration. Each B2B client consists of a STOMP client that receives input messages from a different server, coupled with a REST client that forwards those messages to its associated server. The B2B client also may perform filtering, either by selecting a STOMP Topic or by forwarding messages selectively. Two servers connected via B2B clients must use the same schema. However, as illustrated in Figure 4, one or both of the servers can also function in a translating role.

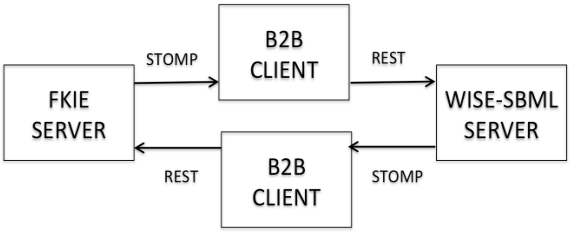

Figure 4. Two-Servers Interconnected With Back-to-back Clients.

3.2 Benefits of Distributing Servers

There are two primary benefits with respect to system performance to be obtained from distributed servers:

- Where clients are clustered in multiple geographic locations, significant reductions in network capacity requirements can be achieved by placing a server at each location, to support the clients located there. This arrangement avoids the need to pass a separate copy of the C2SIM Orders and Reports from a centralized server to each client in a distant cluster of clients.
- Where the message workload is beyond the capacity of a single server, the processing load for reproducing messages to be provided to multiple clients can be shared among distributed servers. This is most likely to happen with parsing servers since they have an inherently heavier computational load; however, it also could apply in the case of document-mode servers if the message rate is high or the number of clients is large.

When the C2SIM techniques discussed here are used operationally, distributing servers also provides the military benefit that server distribution avoid having a single point of vulnerability in a centralized server.

3.3 Looped Message Filtering – Star Configuration of Servers

Any number of servers can be interconnected using back-to-back clients as shown in Figure 4. However, there is a special consideration: messages would loop, circulating continuously among the servers, unless a mechanism is included to preclude looping. To solve this problem, we have added the parameter *forwarders* to the HTTP protocol parameters for the REST input to the server. This parameter contains a list of identifiers (separated by colons) for all servers that have forwarded the message, and is propagated in the output by headers used in STOMP subscription message forwarding. A server is responsible to drop any message that the forwarders list indicates it already has forwarded. (This process can be made somewhat more efficient by dropping the message in the B2B client before it is transmitted to the REST input of its associated server.) The server identifier could be any pattern unique to each server; we chose to use the server's IPV4 address within the virtual private network used by MSG-085 for experimentation and demonstration.

A problem with this solution is that it only works completely in a star configuration of servers connected by back-to-back clients. In a mesh network, where multiple paths exist among the servers, multiple copies may be forwarded (although each will be forwarded only one time from any given server). However, it is not hard to configure the multi-server system as a star; see Figure 5 for a general concept and Figure 7 for a specific example. The tree could have multiple levels and each server might have as many clients as it can support (including the B2B client). Efficiency of distribution in a multi-level tree could be further improved by designating local servers to serve clients for specific STOMP Topics, so that those servers do not have as input messages that are not of interest to their connected servers.

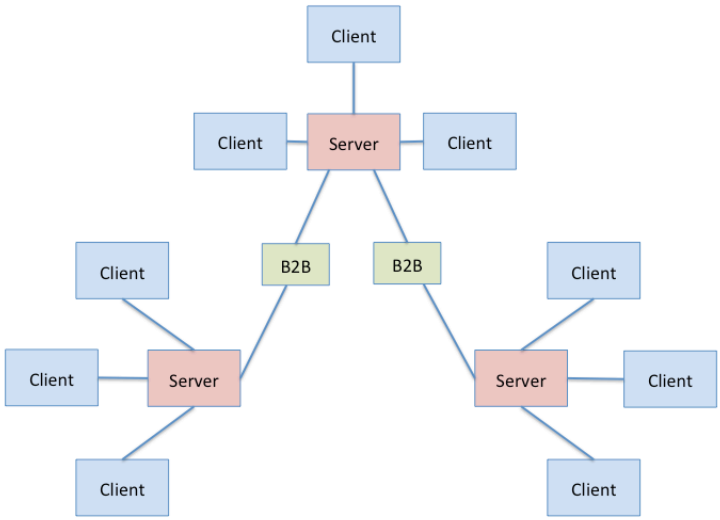

Figure 5. Distributed Server Simple Star Configuration

3.4 Looped Message Filtering – Mesh Configuration of Servers

This section aims to explain the underlying reasons for potential infinite loop and to provide analysis elements and recommendations to consider in the development of the next versions of the C-BML servers. The principle used for message distribution is to rely on the propagation of the message from each server to its logical direct neighbors, without predefined route set, as long as the message is "of interest" to further servers in the system, and never to have any server forward a message that it already has forwarded. There is no routing table that associates units (senders or recipients) to connection nodes (C2SIM servers); consequently there is no way to control or manage the distribution of the messages by an external agent. The following analysis assumes this general principle of message distribution (by direct propagation), although broader analysis might indicate other alternatives.

Figure 6 shows a representative mesh of servers. While the figure is a simple partial mesh, in principle any number of servers could be interconnected to any degree that was found useful. With interconnected servers, a loop is created when a same message goes through the same node (C2SIM server) more than once and this could cause the same message to be delivered repeatedly. In addition, multiple copies of the message may be delivered when there is more than one path from its source server to the receiving server. To prevent multiple deliveries from happening, records of the nodes crossed by the messages must be kept. There are two options to keep this information: either the message carries the IDs of the servers already crossed, or the servers keep the IDs of the messages already (re)distributed. With the first option, the loop is stopped when the server inspects the list of servers ID carried by the message and rejects the message if its own ID is in the list. With the second option, the server inspects the ID of the message and rejects it if the ID of the message is already in the list of the messages already transmitted.

Which option is better? With the first option, the information can be stored in meta-data of the message (like a header) that is added to the data for the transmission of the message. Message distribution protocols have been used to add such information; however, for reasons of modularity and flexibility, the solution could be made independent of transport protocol by modifying the standardized structure of the XML document itself. The C-BML schema supports a message header, which contains envelope information about message transmission (security classification, urgency, sending time, etc.). This header could be modified to add the "forwarders" information. This option requires the modification of the C-BML schema: the routing information is linked to the XML document and has the same life duration: it is created when he document is created; it is destroyed when the message is dropped.

The C-BML schema is intended to evolve under C2SIM. Consider this comparison: in the IPV4 packet header, there are many other elements: the version of the packet, the network congestion information, the "time to live" that indicates how many router hops a packet can be forwarded if not delivered. Comparable elements for C2SIM could provide reasons to update the message header in the next-generation C-BML schema.

With the second option, the server ID information can be stored in the servers; this option is interesting if we consider that the C-BML schema doesn't need to be modified; the implementation of the mechanism relies entirely on the servers themselves, with no adherence to a standard, only to the implementing technology. This is the only option that works with a mesh network, since the extra copies of a message come from alternate paths, not loops, so they will not appear in the forwarders list, whether it is carried as an HTML parameter or part of a C2SIM header. There is, however, one drawback: the C2SIM information is kept in the servers regardless the life duration of the message and, at some point, the servers would have to be "cleaned" to erase the growing history of routing the messages. The first option in which the server IDs are kept in the message headers is, as a result, the best one when having a tree of C-BML servers, and can become more general if the C-BML message schema is modified accordingly in the next generation.

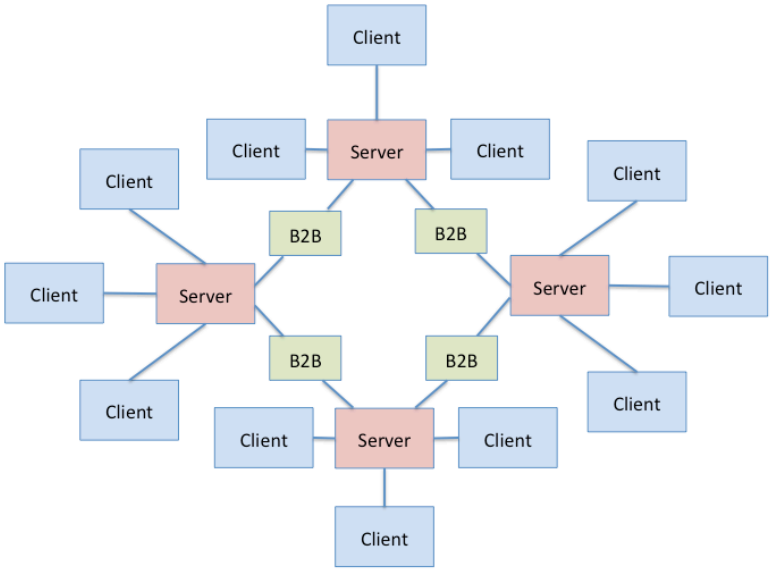

Figure 6. Distributed Server Simple Partial Mesh Configuration

4. Demonstration of Three-Server Distributed Operation

The first demonstration of multiple servers configured to have more than two servers was conducted in the NATO booth at the Interservice/Industry Training, Simulation and Education Conference (I/ITSEC) 2014, Orlando, Florida, 3-4 December 2014. Figure 7 shows the configuration used.

4.1 FKIE Server Background and Operation

The Fraunhofer FKIE BML Server is a light-weight server for exchanging BML over SOAP, REST, JMS and STOMP running on a JBoss Server with HornetQ. It supports multiple schemas in document mode, including IBML, SISO Full and Light C-BML phase 1, and the adjusted schema IBML++ used for Land Operations in the NATO MSG-085, but it does not convert among those. This server is advantageous to test small schema changes, since it is looking for keyword and not parsing according to a predefined schema. Though connected systems must agree on one schema before running a scenario.

For connection to other servers, a tool called FKIE Back-to-Back (B2B) Client connects to a STOMP Topic of another server and sends everything over REST into the FKIE BML Server. The FKIE B2B Client checks for each message if it was already received by the FKIE BML Server and drops it in that case.

4.2 WISE-SBML Server Background and Operation

The George Mason University C4I Center, under management of US Army PM OneSAF and in close cooperation with MITRE and QinetiQ personnel, developed a set of services that provide infrastructure to support implementation of MSDL/C-BML in MSG-085 C2 and simulation systems. The server implementation is available at http://c4i.gmu.edu/OpenBML as open source software. The scripted BML server provided an environment where new BML constructs can be implemented and tested rapidly, changes to the data model that underlies the database can be implemented and tested rapidly, and low cost to change the service facilitates prototyping. It did not, however, feature very high performance.

A new generation SBML server has been developed using Saab Corporation's Widely Integrated Systems Environment (WISE) for experimentation support. WISE supports a robust, high-performance information switching capability with a graphic setup editor that provides and improves upon the advantages associated with the scripted approach of SBMLserver. This capability enabled fundamental research at GMU, prototyping a new generation server that is expected ultimately to be transitioned by Saab to operational military use as described in [16]. WISE-SBML is a parsing server that supports translation among four variants of the BML schema and also supports MSDL [3].

4.3 ELLIPSE Server Background and Operation

The French Ministry of Defense aims to have a single system to support simulations interoperability. This system is called ELLIPSE, it can be deployed for the following operational use cases: Exercises for Force Readiness, Real operations with decision support simulations, studies or experimentations. More generally, any project requiring interoperability between simulation and its environment can take advantage of this system first with complexity reduction leading to time and resources savings and second with an improved efficiency brought by the multiple end user services.

The first version ELLIPSE V1 was delivered in 2014, it consisted in a technical framework for interoperability services only. The second version, ELLIPSE V2, will be released first quarter of 2015. This version embeds a simulation abstraction layer (simulation-simulation interoperability services) allowing connecting simulations regardless the technologies (HLA, HLA evolved, DIS…), a BML/MSDL server and its gateways (C2-simulation interoperability services) and a set of applications and tools (user services) providing functions such as exercise (or experiment) definition, planning, supervision, animation and after action review.

ELLIPSE interoperability is largely based on international standards or/and NATO agreements such as NETN FOM, MSDL, CBML, OTH-GOLD, and AdatP-3. The first version of ELLIPSE has been used already for the MSG-085 I/ITSEC 2014 reproducing parts of the MSG-085 final demonstration (connection of a land simulation and interconnection with FKIE and WISE-SBML servers) and for the MSG-106 final experiment that also was presented at I/ITSEC 2014 (scenario edition and federation supervision). ELLIPSE is also used in the French-German Electronic Cooperation (COMELEC) to interconnect multiple C2 and simulation systems.

4.4 Demonstration Configuration and Results

The demonstration repeated the scenario of the MSG-085 Final Demonstration [14]. This scenario involves a multi-national brigade, consisting of Danish, French, German, and US battalions, dealing with a broken ceasefire agreement in a fictitious country called Bogaland (which happens to have terrain identical to that of actual Sweden). C2 and simulation client systems were located physically as shown in Figure 7. The Danish and German forces were commanded from the C2LG GUI in Wachtberg, Germany

and represented in the SWORD simulation running in Orlando. The French forces were commanded from the French SICF C2 system running in Orlando and also were represented in the SWORD simulation. The US forces were commanded from the 9LandBMS system located in Orlando and were represented in the OneSAF simulation running in Orlando. Figure 8, a screenshot from the C2LG GUI, shows the military situation as the scenario began. Red rhombus icons represent opposing forces; blue rectangle icons represent friendly forces. The demonstration ran for about 20 minutes, during which time the blue forces engaged the red ones and blocked or defeated them.

The distributed servers were located and connected as shown in Figure 7. A simplified star configuration was used, designed such that any of the three servers could function as the hub. The HTTP and STOMP protocols were configured to use a *forwarders* list, as described in sections 3.3 and 3.4 above. The three-server configuration effectively reproduced the results of the MSG-085 Final Demonstration. Logs from the WISE-SBML server show the system of servers was operating at the peak capacity of the WISE-SBML server on the processor used, which is around 150 reports per second, when operated in parsing mode to support schema translation, as reported in [17].

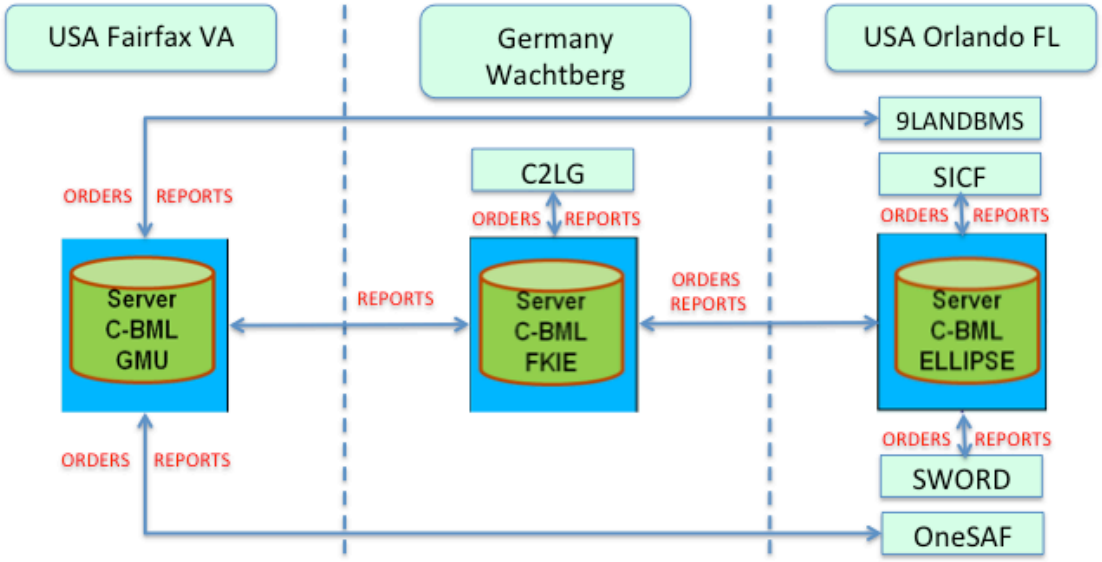

Figure 7. Three-Server Architecture as Demonstrated

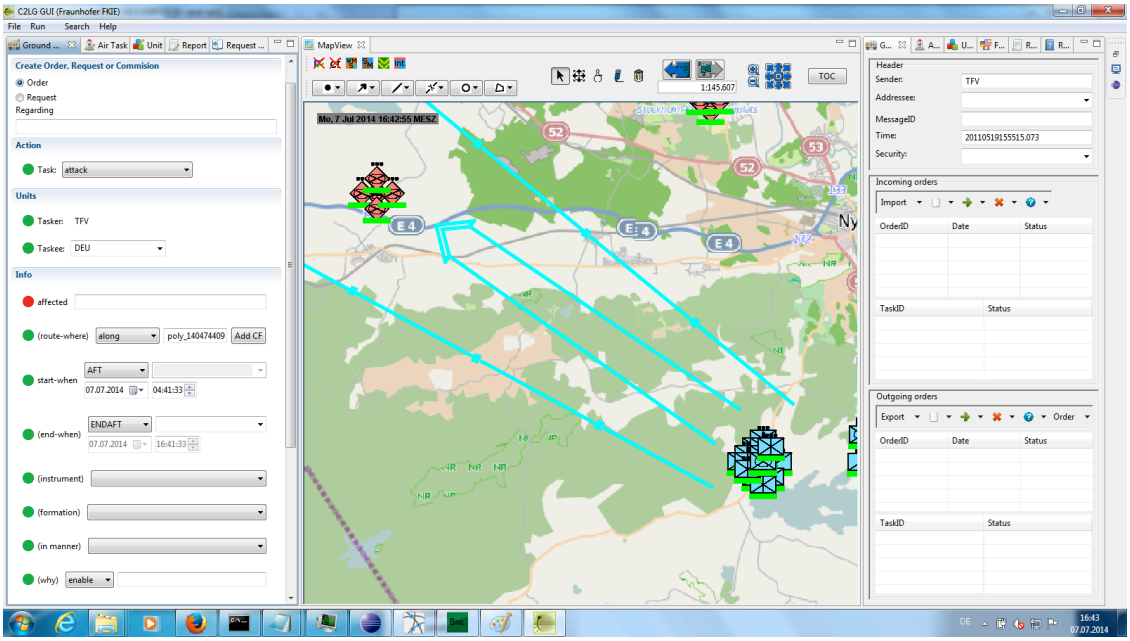

Figure 8. Military Situation at Beginning of Demonstration

5. Conclusions

C2SIM has great promise, but it has yet to see significant operational use. As applications of C2SIM increase in scope, the ability to distribute server workload can provide benefits both in reduced network loading and to support heavier message workloads. The techniques described in this paper enable server workload to be distributed, resulting in the ability to scale to large and geographically dispersed C2SIM coalitions. We recommend that the SISO C2SIM phase 2 development group consider adding for *forwarders* parameter to header envelope information as described in sections 3.3 and 3.4 above.

References

[1] Simulation Interoperability Standards Organization, *Standard for: Military Scenario Definition Language (MSDL)*, 2009

[2] Simulation Interoperability Standards Organization, *Standard for: Coalition Battle Management Language (C-BML)*, 2014

[3] Pullen, J., and L. Khimeche, "Advances in Systems and Technologies toward Interoperating Operational Military C2 and Simulation Systems," 19th International Command and Control Research and Technology Symposium, Alexandria, VA, 2014

[4] Galvin, K., W. Sudnikovich, P. deChamps, M. Hieb, J. Pullen, and L. Khimeche, "Delivering C2 to M&S Interoperability for NATO - Demonstrating Coalition Battle Management Language (C-BML) and the Way Ahead," IEEE Fall Simulation Interoperability Workshop, Orlando, FL, 2006

[5] Pullen, J., M. Hieb, S. Levine, A. Tolk, and C. Blais, "Joint Battle Management Language (JBML) – US Contribution to the C-BML PDG and NATO MSG-048 TA," IEEE European Simulation Interoperability Workshop, Genoa, Italy, 2007

[6] de Reus, N., R. de Krom, O. Mevassvik, A. Alstad, U. Schade, and M. Frey, "BML-enabling national C2 systems for coupling to Simulation,", IEEE Spring Simulation Interoperability Workshop, Newport, RI, 2008

[7] Levine, S., L. Topor, T. Troccola, and J. Pullen, "A Practical Example of the Integration of Simulations, Battle Command, and Modern Technology," IEEE European Simulation Interoperability Workshop, Istanbul, Turkey, 2009

[8] Pullen, J., D. Corner, P. Gustavsson, and M. Grönkvist, "Incorporating C2-Simulation Interoperability Services into an Operational C2 System," 18th International Command and Control Research and Technology Symposium, Alexandria, VA, 2013

[9] Pullen, J., D. Corner, R. Wittman, A. Brook, P. Gustavsson, U. Schade, and T. Remmersmann, "Multi-Schema and Multi-Server Advances for C2-Simulation Interoperation in MSG-085," NATO Modelling and Simulation Symposium 2013, Sydney, Australia

[10] Schade, U. and M. Hieb, "Formalizing Battle Management Language: A Grammar for Specifying Orders," 2006 Spring Simulation Interoperability Workshop, IEEE Spring Simulation Interoperability Workshop, Huntsville, AL, 2006

[11] Hieb, M. and U. Schade, "Formalizing Command Intent through Development of a Command and Control Grammar," 12th International Command and Control Research and Technology Symposium, Newport, RI, 2007

[12] Remmersmann, T., U. Schade, L. Khimeche, and B. Gautreau, "Lessons Recognized: How to Combine BML and MSDL," IEEE Spring Simulation Interoperability Workshop, Orlando, FL, 2012

[13] Pullen, J., D. Corner, and R. Wittman, "Next Steps in MSDL and C-BML Alignment for Convergence," IEEE Spring 2013 Simulation Interoperability Workshop, San Diego, CA, 2013

[14] Pullen, J., L. Khimeche, R. Wittman, B. Burland, J. Ruth, and J. Hyndøy, "Coalition C2-Simulation History and Status," NATO Modelling and Simulation Symposium 2014, Washington, DC

[15] Pullen, J., D. Corner, R. Wittman, A. Brook, P. Gustavsson, U. Schade, and T. Remmersmann, "Multi-Schema and Multi-Server Advances for C2-Simulation Interoperation in MSG-085," NATO Modelling and Simulation Symposium 2013, Sydney, Australia, 2013

[16] Pullen, J., D. Corner, P. Gustavsson, and M. Grönkvist, "Incorporating C2−Simulation Interoperability Services into an Operational C2 System," 18th International Command and Control Research and Technology Symposium, Alexandria, VA, 2013

[17] Pullen, J. and D. Corner, "Publish/Subscribe and Translation Performance of the WISE-SBML Server for MSDL and C-BML," IEEE Fall Simulation Interoperability Workshop, Orlando, FL, 2014