

Agent-based Execution Monitoring: Results from the GCCS-ITEM Federation and Planned Activities with the GCCS-JWARS Federation

Ranjeev Mittu

Naval Research Laboratory
4555 Overlook Avenue
Washington, DC 20375
(202) 404-8716
Fax: (202) 767-1122
mittu@ait.nrl.navy.mil

Joshua Walters

ITT Industries, Inc
2560 Huntington Avenue
Alexandria, VA 22303
(703) 767-1879
Fax: (703) 767-1122
walters@ait.nrl.navy.mil

Keywords: Multi-Agent Systems, Intelligent Agents, Simulation, C4I, Plan Monitoring, Web Services, C2IEDM.

ABSTRACT: *For years, simulations have been used by analysis and planning staffs to develop and rehearse operation plans, analyze results, and develop doctrine. Typically, combat simulations are used most heavily during the planning stages of an operation, prior to battlefield action. However, simulations are increasingly being used during operations to perform course of action analyses (COAA) and develop real-time forecasts of future conditions on the battlefield. Recent efforts by the Defense Modeling and Simulation Office (DMSO) to improve the interoperability of C4I systems with simulations have provided a powerful means for rapid simulation initialization and analysis during exercises, and made simulations more useful and responsive as the exercises are executed. These DMSO efforts involve technology development to support the integration of operational C4I systems, such as those in the Global Command and Control System (GCCS), with simulations such as the Integrated Theatre Engagement Model (ITEM) and the Joint Warfare Simulation (JWARS).*

This paper will primarily describe the GCCS-ITEM-Intelligent Agent Federation project and its support for agent-based plan monitoring, discuss project results, and present its conclusions. We will then describe the FY04 Simulation-to-C4I Connectivity project, which is integrating the results from the GCCS-ITEM federation with the GCCS-JWARS federation; specifically, extending the monitoring agents from the GCCS-ITEM federation to support the use of JWARS as an embedded tool for the C4I operator. In this integrated project, interfaces between operational C4I systems, simulations, and intelligent agents are designed to exploit web-service technologies and the standardized information exchanges described by the NATO and Multilateral

Interoperability Programme Command and Control Information Exchange Data Model (C2IEDM).

1. Introduction

The Defense Modeling and Simulation Office (DMSO) sponsored the integration between the Global Command and Control System (GCCS), Integrated Theatre Engagement Model (ITEM) and intelligent agents to support execution monitoring. The idea behind this approach was to use ITEM, representing the simulation of the plan, in order to compare with the actual execution of the plan as represented in GCCS. The GCCS-ITEM federation communicated and exchanged track data through the High Level Architecture (HLA) Run-Time Infrastructure (RTI). The Critical Mission Data over RTI (CMDR) toolkit was used to bridge the RTI-based federation with an agent federation residing on the Control of Agent Based System (CoABS) Grid. Agents on the grid were responsible for detecting deviations between the execution picture as represented in GCCS and simulated tracks in ITEM to support Courses of Action Analysis (COAA). This proof of concept was successfully demonstrated to DMSO in February 2004.

Based on this initial success, the agents developed under that effort are now being integrated with the Joint Warfare Simulation (JWARS)-GCCS Federation. The GCCS-ITEM federation was integrated with intelligent agents through a 3-level tier (i.e., RTI, CMDR and CoABS grid). However, for the JWARS-GCCS federation, these agents will be adapted with more intelligent reasoning capabilities to support execution monitoring, and their integration with JWARS-GCCS will be accomplished through web-service technologies. Web service technologies are a maturing field, and they are being seriously examined as a viable option to support the concept of the Global Information Grid (GIG) [7], within which C4I systems and simulation will operate.

Section 2 will describe the GCCS-ITEM concept, including experimental results. Section 3 will then describe the proposed integration between JWARS, GCCS and agents through web service technologies. We will conclude in Section 4 with a discussion of the future technical challenges we hope to address with regard to agents, particularly their role in a web-services environment and the level of intelligence needed to support effective execution monitoring.

2. The GCCS-ITEM-AGENTS Federation

The GCCS/ITEM federation was integrated with intelligent agents via the CMDR “bridge” between the RTI and CoABS grid. The purpose was to conduct agent-based plan monitoring to support the analysis of both real and simulated information to detect deviations, and provide alerts to GCCS when those deviations were of significant consequence. The system architecture is shown in Figure 1. Further details of system components can be found in [2].

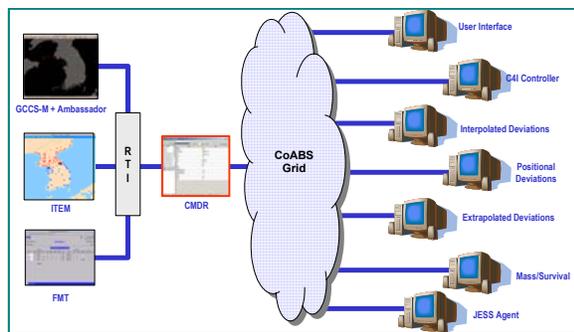


Figure 1: GCCS-ITEM-Agents federation System Architecture

The GCCS was used to publish “real world” track data while ITEM was used to publish simulated tracks. The CMDR subscribed to both the “real world” and the simulated track data published in the HLA federation, and provided the data as it became available via the CoABS Grid to the agents. As the agents analyze the data, alerts (or retraction of alerts) are generated according to pre-defined thresholds. Alert “retractions” are generated when movements result in a track moving back within a distance that is below the threshold distances.

We will now describe the sequence of steps that will demonstrate the user’s interaction with the various system components within the architecture.

Step 1: Develop a pair of scenarios in ITEM

Several pairs of combat scenarios were developed using ITEM. These scenarios were developed by a

Subject Matter Expert (SME) supporting U.S. Forces Korea (USFK). Three separate scenarios were used, the largest one consisting of 490 hostile and friendly ground units and 1054 hostile and friendly ships.

Step 2: Using a given scenario pair generated in step 1, modify the first of the pair such that some of the tracks deviate from those in the second scenario.

In our experiments, we needed to artificially create an execution of the plan and also a simulation of the plan. In our example, it was somewhat arbitrary which scenario we labeled as the “real-world plan” and which one we labeled the “simulated plan”. In either instance, ITEM was used to generate both.

One scenario from the pair was used in the simulation of the plan and the other was a variant of the plan where friendly and hostile land units and naval platforms behaviors differ from the plan. A thirteen hour segment was simulated and captured for each of the scenario files.

Step 3: Having created a representation of the real world plan as well as simulated plan, GCCS published the initial state of the battlefield, while ITEM subscribed to this initial state from GCCS. The purpose was to support a “correlation” step that would be done in the real world to get both systems “on the same page”

The GCCS HLA Ambassador was used to publish tracks from the TDBM representing the real-world plan execution onto the RTI. To support our experiments, the Ambassador was modified to be able to publish all tracks and their data to the RTI without the user having to rubber-band and select a group of tracks. This allowed the user to be somewhat “hands-off”, and guaranteed that the monitoring agents would receive all of the necessary tracks to monitor.

The archived scenario was set to stop at a timestamp shortly after all of the tracks appear on the screen. This constitutes the initial state of each track. Once published, ITEM (through a subscription mechanism), received those tracks. This synchronization step provided us with a one-to-one mapping between “real-world” and “simulated” tracks. ITEM was modified to “reflect” the same Local Track Number (LTN) - an internal “house-keeping” variable used by GCCS, as a primary key. This primary key was necessary for the agents to be able to match “real-world” and “simulated” tracks.

The use of the LTN was sufficient for our initial experimentation, however it was soon discovered that the LTN was not a good choice for primary key. The

LTN is a number assigned to a track when GCCS loads the information from the TDBM or acquires the track from real world input (ITEM does a similar assignment for its objects). This LTN is unique for the duration of the time that GCCS displays that track. It does not exhibit persistence in the TDBM database, and therefore is not consistent throughout multiple runs. It was soon concluded that a more permanent key would be needed in the future.

Another difficulty arises when the tracks are not correlated properly in ITEM. If the GCCS track is not correlated to an ITEM object, an ambiguous track will show up on the GCCS screen. When a track is ambiguous, it is not correlated with an object in ITEM. This means that the GCCS track and the ITEM object have different LTN's. As a result, the agents will not be able to match real and simulated track data, and monitoring process will fail.

Step 4: Next we published all ITEM objects onto the RTI (the objects contained the force composition information).

ITEM published the entire scenario to the RTI at this point in time, including track objects and health status objects. ITEM was enhanced to support the publishing of objects describing force combat worth or "health" in terms of "mass". For example, the relative value of each entity participating in the simulation is compared to the strength of an M1A1 tank (whose value was set to one). For example, a soldier with a Rocket Propelled Grenade (RPG) might have a mass of 0.1, signifying that ten human/RPG pairs would equal the combat strength of the M1A1. The objects published by ITEM contained unit strength (mass) information using these baseline values. The ability to analyze a force's combat worth prior to monitoring the actual execution of the plan may prove to be useful for initial plan refinement. In the system diagram, the *mass monitoring agent* was responsible for detecting deviations in combat worth based on thresholds defined in the original plan (located within ITEM).

The concept of Mass Monitoring is applied to Units; a parallel measurement is used to evaluate Platforms, and it consists of comparing probabilities of survival thresholds. Future work could involve the use of Unit Order of Battle information combined with Mass Monitoring as a means of monitoring force composition and battle readiness. The following constraints were used to trigger alerts:

$\frac{\text{(Current mass)}}{\text{(initial mass)}} < \text{Surrender threshold}$ $\text{(Probability of survival)} < \text{Survival threshold}$

Step 5: Once ITEM is finished and the agents have analyzed mass/survival of units and platforms, playback of GCCS is resumed.

Several types of track deviation agents were developed to monitor track movements. The GCCS Ambassador and ITEM published track information, representing the real world execution and simulated execution, respectively, to the RTI. These were fed through the RTI via CMDR to the agents registered on the CoABS grid. Several modifications were made to CMDR to support our experimentation, including the ability to translate data into XML as well as improvements to the record/playback feature.

Several types of track deviation agents were developed, including those for *extrapolated deviations* [3] and *interpolated deviations* [3]. These monitoring agents were built using the Java Expert System Shell (JESS), a rule-based inference engine [1]. Furthermore, two additional agents were developed, the *C4IController agent* and the *UserInterface agent*.

The *deviation-by-extrapolation* agent projects a position in the future when the course and speed is known and compares that projection against a simulated event. The *deviation-by-interpolation* agent does a linear interpolation between two temporally consecutive simulated events to estimate the current position. That estimation is then compared with the real-time event. The procedure for the distance calculation between those events is described in Figure 2. This distance serves as the decisive factor for triggering alerts when deviations occur above threshold (and retracting alerts when below threshold).

<p>The latitude and longitude, given in decimal degrees, are converted to radians and the earth radius (6378 kms) is added to the altitude. The angle α between 2 points, p_1 and p_2, is computed first:</p> $\alpha = \arccos((\sin a_1 * \sin a_2) + \cos(b_1 - b_2) * \cos a_1 * \cos a_2)$ <p>where a_1 is the latitude of p_1, a_2 is the latitude of p_2, b_1 is the longitude of p_1 and b_2 is the longitude of p_2. The distance is then computed using the cosine law:</p> $\sqrt{(r + c_1)^2 + (r + c_2)^2 - 2(r + c_1)(r + c_2) \cos \alpha}$ <p>where r is the earth radius and c_1 and c_2 the respective altitudes of p_1 and p_2.</p>

Figure 2: Distance Calculation

Through the use of the *UserInterface* agent, the user is able to spawn *mass monitoring* or *track deviation monitoring* agents. The user may enter whether units or platforms are to be monitored for each type of agent. Furthermore, the interface provides a

mechanism to specify threshold values, that, when exceeded, would warrant alerts (which are also captured in the display as well as sent to GCCS for display within its COP). Within our implementation, multiple track monitoring agents can be created to monitor the same types of deviations for different tracks. For example, several *deviation-by-extrapolation* agents can be created that monitor different tracks.

The monitoring agents, once created, register tracks of interest with the *C4IController agent*, which then routes track data to them as this information comes in from CMDR.

Step 6: *The agents generate alerts, which are then displayed on the GCCS COP for the warfighter (Figure 3).*

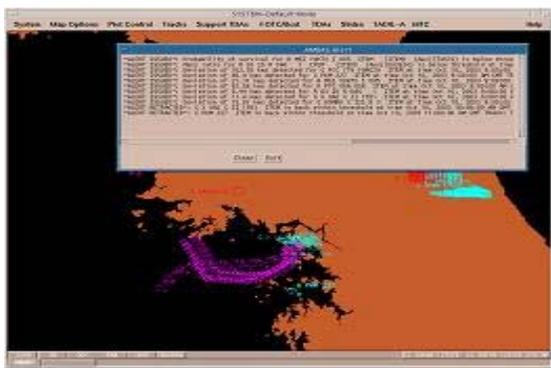


Figure 3: Alerts on the GCCS COP

2.1 Initial Results

Several scenarios were developed for experimentation. The largest scenario, representing a hostile incursion, consisted of 490 hostile and friendly ground units and 1054 hostile and friendly ships. The performance of the overall system was somewhat less than satisfactory. In the largest scenario, thirteen hours of simulated time was compressed to 2.5 hours of real time. While this is satisfactory in a laboratory environment, this is not sufficient to meet operational doctrine development and training needs. We theorized that, with newer versions of software and hardware (we used GCCS 3.X), that the playback time might be drastically reduced. Despite this, we were able to successfully experiment with the largest scenario without experiencing significant performance (throughput) degradation.

This experiment, by necessity, had to involve a scripted (simulated) operation of the operational GCCS system. It would be very interesting, as a

follow-on exercise, to observe the performance and utility of this project when GCCS is being operated in real time.

The older version of GCCS necessitated the inclusion of several other old component versions, namely the Solaris OS, the RTI, and available platforms. GCCS-M was the latest available version for this integration (which was still quite old – 3.x), however, a cost was paid. With technology moving forward as fast as it is, there is a delicate balance struck when exploring new concepts using old technology. For future work, a newer version of GCCS will be available, and current work can be transitioned forward.

3. C4I-to-Simulation Connectivity Program

In fiscal year 2004, DMSO is sponsoring the enhancement and integration of the agents developed under the GCCS-ITEM federation with the JWARS-GCCS federation (Figure 4). The exchange of data between each of the components is envisioned to occur through web service technologies. Web services technologies are rapidly maturing, and consequently, are becoming a viable option to provide the underlying backbone of the Global Information Grid (GIG) [7]. The C4I systems and simulations such as JWARS-GCCS are envisioned to be used within the GIG. Furthermore, the exchange of information in this architecture will be compliant with the Command and Control Information Exchange Data Model (C2IEDM) [8].

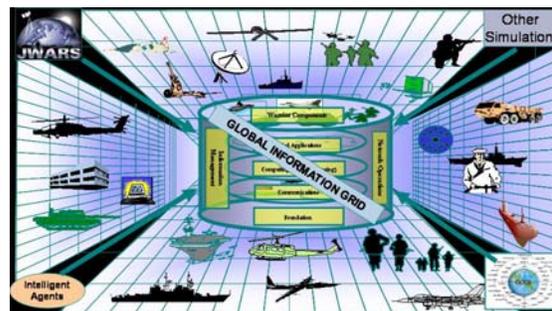


Figure 4: C4I-to-Simulation Interoperability in the GIG

Some of the initial technical challenges to be addressed include the ability to rapidly locate and federate with other components or systems in the GIG (the architecture must support the ability to rapidly form connections between systems and components), overcoming obstacles that impede information interoperability across legacy systems including both Joint and Allied and the ability of components and systems to automatically (or through a semi-automated fashion) locate and interact with

other components. These challenges are further exacerbated by the fact that the U.S. is entering a new era of warfare (e.g., asymmetric) in which responding to crisis action situations will be the norm and speed of execution will be critical for achieving successful military operations.

Several key technologies are being examined to overcome these challenges. Web service technologies are being leveraged to help overcome the challenges associated with rapidly finding and interacting with other systems and components within the GIG. Web services technologies are rapidly maturing, and consequently are becoming a viable option to exploit the underlying backbone of the GIG.

With regard to information interoperability, we are envisioning to use the Command and Control Information Exchange Data Model (C2IEDM) as the common vocabulary for exchange. The C2IEDM was developed under the auspices of the Multilateral Interoperability Programme (MIP) [8]. The MIP is comprised of volunteers from 27 nations, whose goal is to foster international interoperability between multi-national Command and Control (C2) systems.

Having an ability to semi-automatically locate and interact with services will be a key capability, as it will be inefficient to have users in the loop on every web service transaction. Furthermore, systems and components in the GIG lack the intelligence to form complex queries for information. We envision intelligent agents to support this functionality through their abilities to autonomously coordinate with each other to support system requirements for information.

The architecture being proposed to support the integration between JWARS, GCCS and intelligent agents is seen in Figure 5, in which the previously mentioned technologies will be applied to help address the integration and interoperability challenges envisioned in the GIG. The initialization data system (i.e., Army C4I Simulation Initialization System – ACSIS) will initialize both the C4I system (i.e., GCCS-M Track Database Manager) and Theater Battle Management Core System (TBMCS) as well as the simulation system (i.e., JWARS) with current Unit Order of Battle (UOB) such as organization and their relationships, including equipment and facilities. The tactical system (in our case will be an exercise replay through a C4I gateway) will deliver the actual data to the C4I system.

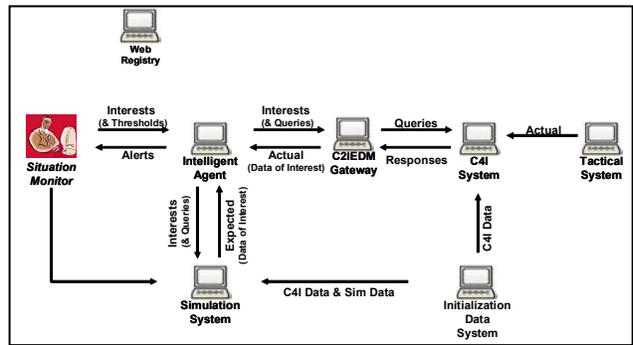


Figure 5: JWARS, GCCS, Intelligent Agent Federation via Web Services

The Situation Monitor is a graphical front end which will permit a JWARS user to specify tracks of interest that need to be monitored, and the corresponding threshold. This information will be communicated to intelligent agents that will make requests to the C4I system and Simulation to obtain the corresponding tracks for subsequent monitoring. These agents will compare both the real and simulated tracks in terms of the thresholds to generate alerts back to the situation monitor terminal. The alerts may warrant the exploration of alternate Courses of Action (CoA).

The interface between the various components will be accomplished through web service technologies [9]. These include the Universal Description and Discovery Interface (UDDI), Web Services Definition Language (WSDL) and Simple Object Access Protocol (SOAP).

UDDI is a framework that defines XML-based registries in which businesses can upload information about themselves and the services they offer. An XML-based registry contains names of organizations, services provided, and descriptions about service capabilities. XML registries based on the UDDI specification provide common areas through which systems can advertise themselves and their Web Services. WSDL is an XML vocabulary standard created just for Web Services. It allows developers to describe Web Services and their capabilities, in a standard manner. WSDL helps to expose the Web Services of various businesses for public access. SOAP is an XML vocabulary standard to enable programs on separate computers to interact across any network. SOAP is a simple markup language for describing messages between applications. SOAP provides a way for developers to integrate applications and business processes across the Web or an intranet, by providing the platform and

programming language independence needed to create the business integration of Web Services.

Each of the components will register their services with the UDDI registry (e.g., the WSDL specification). Each component that requires information from other components will perform a look-up in the UDDI registry and obtain the WSDL file, from where a determination can be made as to where the service resides and how to invoke it.

The C2IEDM gateway will map all information to the C2IEDM vocabulary. Systems and components will be required to map their native information content to the C2IEDM to support the common understanding of concepts. The C2IEDM is a generic model that can be extended as needed to suit evolving military requirements (serving as a “hub”; as such, it was originally named the “Generic Hub”, and evolved to Land C2IEDM and eventually C2IEDM to capture other areas including Air and Surface). The C2IEDM is comprised of a conceptual data model, logical data model and physical data model. The conceptual data model represents generalized concept, while the logical data model represents further details associated with the conceptual data model. The physical data model defines the physical data storage schema. The main purpose of the C2IEDM is to represent Information Exchange Requirements (IERS) between C2 systems.

The Intelligent Agents that were developed under the GCCS-ITEM federation will be enhanced to support additional monitoring capabilities. The enhancements will include the ability to monitor tracks that enter regions of interest as well as monitoring the changes made to the Air Tasking Orders (ATOs). Initially the agents will compare the changes made to the ATO within the TBMCS, and compare that to the changes made to the ATO in JWARS and alert the user when the plan associated with a specific air track of interest has been modified in some fashion. Later we plan to monitor the progression of the ATO and provide alerts based on user specified conditions or thresholds. The ATO’s will be represented in eXtensible Battle Management Language (XBML) format and will also be mapped to the C2IEDM.

4. Future Technical Challenges

We have concentrated strictly on the plan monitoring agents. However, there is significant research to be done in the area of agents that are able to decompose and understand plans. In our experimentation, we have placed the burden on the user to select tracks of

interest, for which thresholds need to be set, that should be monitored. Through plan understanding, we would like to be able to identify critical events and relationships, thereby permitting an intelligent agent to monitor the necessary plan in terms of those critical events. We are examining Natural Language Processing (NLP) techniques coupled with sublanguage ontologies to extract semantic relationships from free text documents such as Operational Orders (OPORDs). Yet, another promising area is the use of the Battle Management Language (BML), which provides some of the infrastructure necessary for intelligent agents to reason about OPORDS.

A promising area for future research is in agent teamwork [4,5,6]. To realize the power of distributed multi-agent systems, agents will need to cooperate in teams to accomplish their objectives on behalf of their users. For example, teams of distributed software agents with different goals may need to coordinate to decompose and relate multiple plans to determine critical points, which can be passed to a team of agents that are responsible for monitoring.

There are many challenges in realizing such an ambitious effort, such as the integration of large legacy systems through the application of new technology (web services) which, although maturing at a fast pace, is still evolving in order to reach the goal of becoming a standard for service-oriented transactions. We are faced with integrating systems that are fairly stable with new technology that is not quite stable. This, by itself, is a tremendous challenge!

Furthermore, we have the task of integrating fairly *well understood technologies* (e.g., intelligent agents) with a web-services computing paradigm. Although there has been considerable attention devoted to the field of multi-agent systems such as agent communication languages [12], standards, etc, there has not been significant research into how multi-agent systems will operate in a web-services world, primarily due to the fact that web service technologies are fairly new. A key issue for deploying multi-agent systems in a web-services environment includes the fact that agents require messaging for communication; it is not clear how the messaging will be handled between agents communicating in a web-services environment (and if it is possible, what overhead or other factors will affect the performance).

A second issue is mapping between ontologies within the GIG. It is envisioned that heterogeneous agents

will operate in the GIG, with different ontological representations. A key challenge that is certain to arise is the mapping between the ontology that describes what an agent understands, the C2IEDM and web-services, which can be further complicated in the future as semantic web services [10] are introduced into the GIG.

Lastly, a key issue that will need to be addressed is how web service technologies will interact with other technologies such as Peer-to-Peer (P2P) computing [11]. The big question here is will there be a single technology that provides the infrastructure for the GIG, or will there be several complementary technologies? If the latter is true, how to bridge the applications that rely on different technologies?

Our Simulation-to-C4I FY04 connectivity program will afford us the opportunity to begin to investigate a few of these issues, including integration of large scale legacy systems with new technology, multi-agent system operation within a web-services environment and the complex nature of mappings between agent ontologies, web services and the C2IEDM. The other areas will be investigated in later years as DISA charts out the vision for the GIG and as competing (complementary) technologies mature.

References

- [1] Friedman-Hill E. *Jess in Action, Rule-Based Systems in Java*. Manning, 2003.
- [2] Mittu, R., Furness, Z., Emami, R., "Agent-mediated Interface Between C4I and Simulation". Proceedings of 2003 Spring Interoperability Workshop (SIW). Orlando, FL, 2003.
- [3] Mittu, R., Walters, J., Abramson, M., "Improving Simulation Analysis through Interfaces to C4I systems and Simulations", In *Proceedings of the 2004 Spring Simulation Interoperability Workshop.*, Crystal City, VA.
- [4] Rao A. S. and Georgeff M. P. "BDI agents: from Theory to Practice". In *Proceedings of the First Intl. Conference on Multiagent Systems*, San Francisco, 1995.
- [5] Tambe, M. "Agent Architectures for Flexible, Practical Teamwork". In proceedings of the *National Conference on Artificial Intelligence*, 1997.
- [6] Tamma et al, "An Ontology for Automated Negotiation". In Proceedings of the International Workshop on Ontologies in Agent Systems. AAMAS 02 Conferece, Bologna, Italy.
- [7] <http://www.disa.mil/ns/gig.html>

- [8] <http://www.mip-site.org/>
- [7] Global Information Grid
[\[http://www.disa.mil/ns/gig.html\]](http://www.disa.mil/ns/gig.html)
- [8] C2IEDM and MIP
<http://www.mip-site.org/>
- [9] UDDI, WSDL, SOAP
<http://www.w3c.org>
- [10] Semantic Web Services
<http://swws.semanticweb.org>
- [11] Peer-to-Peer (P2P) Computing
<http://www.openp2p.com>
- [12] Foundation for Intelligent Physical Agents
<http://www.fipa.org>

Acknowledgements

The authors would like to thank the Defense Modeling and Simulation Office (DMSO) for providing the research funds for this effort.

Author Biographies

RANJEEV MITTU is the head of the Intelligent Decision Support Section at NRL. He has over fifteen years experience in designing and developing decision support systems for the naval community. He earned an MS degree in EE from The Johns Hopkins University in Baltimore, MD in 1995.

JOSHUA WALTERS is a Computer Engineer at ITT Industries, Inc. Mr. Walters has four years experience in web based technologies, computer security, and software engineering. He obtained his B.S. degree from Virginia Polytechnic and State University.