IMMACCS: An Experimental Multi-Agent C2 System

Jens G. Pohl
Anthony A. Wood (Col. USMC, Ret.)
Kym Jason Pohl
Collaborative Agent Design (CAD) Research Center, California Polytechnic State University,
One Grand Avenue (Bldg. 117T), San Luis Obispo, California 93407, USA
jpohl@calpoly.edu; awood@cdmtech.com; kpohl@cadrc.calpoly.edu

Abstract

The Integrated Marine Multi-Agent Command and Control System (IMMACCS) is a multi-agent, distributed software system, designed to provide a Common Tactical Picture (CTP) with integrated and meaningful decision-support facilities to authorized operators at any access node. IMMACCS has been implemented as a three-tier architecture that distinguishes between information, logic and presentation. It utilizes an object-serving communication facility with subscription and multi-casting capabilities that is based on the Common Object Request Broker Architecture (CORBA). With an emphasis on application, IMMACCS was designed and implemented in concert with its military users as an integral component of experiments conceived by the Marine Corps Warfighting Laboratory (US Marine Corps, Quantico, Virginia, USA) to test emerging concepts in military command and control. It was field tested as the command and control system of record during the Urban Warrior Advanced Warfighting Experiment conducted by the US Marine Corps in Monterey and Oakland, California, USA, March 12 to 18, 1999, and during a live fire Limited Objectives Experiment (LOE-6) held at Twentynine Palms, California, USA, in March, 2000.

Background

In July 1995, General Charles Krulak, the newly appointed Commandant of the Marine Corps, directed formation of the Marine Corps Warfighting Laboratory (MCWL). His action was based on a deep conviction that it was no longer sufficient to modify cold war practices and procedures, but that the era ahead demanded a new approach. It was his desire that the Sea Dragon program, a series of concept-based experiments, would provide the basis for examining new capabilities.

The Sea Dragon program was designed to be executed in phases. Hunter Warrior, the first phase, was planned to focus on the capabilities required for small units employing enhanced tactics and equipment to shape the battlefield through information and fires. Urban Warrior would follow Hunter Warrior as the second phase, and would focus on combat in cities. Capable Warrior, the third phase, drawing on the preceding four years of experimentation and integrating new concepts and technologies, would identify selected concepts and capabilities for introduction into the Marine Corps operating forces.

From the earliest internal debates the MCWL staff postulated that while various forms of cyber-warfare and even more ambiguous types of conflict were probable, armed conflict requiring commitment of trained military forces on the ground would remain decisive in forcing national will on potential enemies. Furthermore, it was argued that future warfare would have several other characteristics that collectively point to the need for a fresh approach to command and control. For example, warfare would be increasingly public, implying the need for quick and decisive results in complex conflicts. The outcomes of these conflicts would depend largely on the judgements of subordinate leaders, particularly the small unit leaders struggling simultaneously with the enemy, non-combatants, and rules of engagement. Additionally, potential foes eyeing the results of Desert Storm would employ asymmetric approaches to minimize the growing technological advantages in traditional conflicts.

Clearly future conflicts could involve widely divergent political objectives and scope. Furthermore, the location and nature of the conflicts could vary just as greatly, while the attitudes on all sides of multi-sided conflicts would likely differ and alter as the conflict progressed. What was needed was a command and control framework that could adapt to these wide variances and seamlessly integrate the air, ground, and logistic capabilities needed to support emerging concepts such as Operational Maneuver From The Sea (OMFTS). This developing framework of adaptive and integrated command and control capabilities became the major influence in determining the shape of the Integrated Marine Multi-Agent Command and Control System (IMMACCS).

While MCWL planners struggled to extrapolate trends and identify the likely characteristics of future battle grounds, they had far less difficulty in perceiving the enormous potential as well as serious threats inherent in the on-going information processing advances of the digital revolution. Nowhere was the promise and the threat posed by these technical advances in information gathering more hotly argued than in the discussions focused on centralized versus decentralized control.

In visualizing the new approach MCWL planners postulated the need for the subordinate leaders who actually did the fighting to exercise maximum initiative supported by greatly expanded access to information. However, they saw the potential for the significant advances in information gathering to create just the opposite situation; namely, to reinforce centralized control. The reasons for this concern were related to the role played by uncertainty. It was argued that centralized control would reduce uncertainty at the top, but the price to be paid was to constrain initiative and flexibility among the subordinates facing the enemy. Decentralized control and enhanced access to information would encourage initiative and flexibility among these subordinates, but at the price of increased uncertainty at the top. In the end MCWL opted for decentralized control coupled with the long-standing Marine notion that commanders must accustom themselves to a high degree of uncertainty as the norm.

While debate continued on how to deal with control, a parallel and equally passionate discussion simmered on the proper approach to command. This discussion focused on the commanders and the future decision environment in which they would operate. Here, MCWL planners felt that the art of command had been far less affected by changes in warfare or technological advances, than was the case with control. Instead there was a developing consensus among MCWL staff that the

'tempo' of a commander's (or any leader's) decision making capabilities could be significantly accelerated if:

1. A means could be found to improve and maintain individual decision skills.

2. The decision environment around the commander could be disciplined.

3. Useful decision-support were to be provided in the form of enhanced situation awareness at every level.

Fundamental to the discussion of command was the belief that while uncertainty would remain a permanent condition, the decision pace of skilled commanders would accelerate as they gained confidence in the currency and accuracy of the information available. To achieve that enhanced currency and accuracy, a way would have to be found for the supportive command and control system to filter and convert data into useful information and inference on entry into the system.

Another key system characteristic that emerged from this discussion on future command was the need for man-machine collaboration. It was generally agreed that while simulation and prediction can be useful in certain situations, these are necessarily linear capabilities. War is not linear, but presents a series of complex problems that defy simplistic approaches. Accepting this notion, MCWL planners felt that the chaos and chance that pervade conflict demand a command and control system that is collaborative, while maximizing human intuition, creativity and conceptualization. Thus, to the adaptive set of characteristics already mentioned was added a requirement that the emerging system concept provide tailored decision-support rather than reshaping combat problems to fit the mould of pre-determined solutions.

Finally, the nature of the Marine Corps as the principal expeditionary force in readiness within the US, demanded that the design of the new command and control system be focused on execution and be near real-time. Further, because no one could predict just what doctrine would be employed in future conflicts, the new command and control system would have to be capable of accommodating any doctrine. The framework of required capabilities was now complete. Building and testing IMMACCS, a proof-of-concept system embodying these capabilities was the next step [Pohl et al. 1999].

IMMACCS Design Principles

IMMACCS was conceived as a distributed, open architecture, command and control (C2) system to assist military commanders under battle-like (i.e., execution) conditions when dynamic information changes, complex relationships, and time pressures tend to stress the cognitive capabilities of decision makers and their staff. IMMACCS incorporates four notions that are fundamental to its decision-assistance capabilities.

Firstly, IMMACCS is an object-based system that processes information. Unlike conventional message-based systems that process data. In this context data are defined as numbers and words without relationships, while information is defined as data with relationships. For example, the meaning of the word 'rifle' derives from the associations that we make in our mind with our

experience and knowledge of rifles. This rich set of relationships converts data to information [Minsky 1982, Pohl 2000]. The key to the assistance capabilities of IMMACCS is that the system has some 'understanding' of the information that it is processing. In IMMACCS every entity in the screen display of the battlefield (e.g., road, building, truck, tank, enemy unit, civilian group, etc.) as well as intangible entities such as weather, attack, defense, and so on, are represented as individual objects with characteristics and relationships to each other. Therefore, the military commander and staff officer interact with a computer display that consists of hundreds of real world entities (objects) that all have some understanding of each other's nature, interests and objectives, and a great deal of understanding of their own characteristics and capabilities.

Secondly, IMMACCS is a collection of collaborative tools, not a library of predefined solutions. This approach is intended to overcome the deficiencies of legacy systems in which built-in solutions to predetermined problems often differ significantly from the complex operational situations encountered in the real world. IMMACCS is a collaborative decision-support system in which the operators interact with computer-based agents (i.e., decision making tools) to solve problems that cannot be precisely nor easily predetermined.

Thirdly, IMMACCS incorporates agents that are able to reason about the characteristics and the relationships of the many real world entities (objects), all of which embody meaning. In its field tests IMMACCS has included agents that: apply their domain specific knowledge to weapon selection and deconfliction; rules of engagement; and, anticipate logistical re-supply requirements. These may be referred to as static agents since they provide predefined services. IMMACCS also utilizes mentor agents that may be dynamically created to represent the interests of warfighters and warfighting machines. During the Urban Warrior and LOE-6 field tests mentor agents were configured to extend the capabilities of Marines at all levels by warning friendly units of enemy intrusions into their territory.

Fourthly, IMMACCS integrates planning, execution and training within one common C2 user environment. The computer-based agents and the IMMACCS users continuously collaborate as they interact with each other in rapidly changing battlefield situations. In this respect IMMACCS reflects the complexity of the real world where problem solutions must be continuously reviewed as conditions change, and it becomes increasingly difficult and inconvenient to separate planning, re-planning, execution, and training functions into artificially discrete activities.

System Components and Architecture

Although IMMACCS is designed as an integrated system it was developed as a team effort by several government and commercial organizations, each taking responsibility for one or more of the following principal components:

1. An Object Model (designed and developed by the Collaborative Agent Design (CAD) Research Center, Cal Poly, San Luis Obispo, California) that facilitates the internal representation of information, rather than data. In this respect data are defined as numbers and words, while information combines data with relationships that provide meaning and context. In particular, IMMACCS

supports the dynamic formation of associations among objects at both the user and agent levels.

2. An Agent Engine (designed and developed by the CAD Research Center) that automatically initiates an agent session in support of any desired 'view' of the battlespace.

3. A Shared Net communication facility (designed and developed by the Jet Propulsion Laboratory, Cal Tech, Pasadena, California) that manages the object-based interactions among the various components on a subscription basis. All IMMACCS components are clients of the Shared Net and indicate their information interests by registering a subscription profile. Whenever, information that is within the subscription of one or more clients (whether military commander or squad leader) becomes available the Shared Net automatically pushes this information into a cache memory area of the client, and if the information is of high priority the client will also receive an alert message. In addition, clients may also query for information to which they have not subscribed. Even individual agent sessions are clients to the Shared Net and can therefore take advantage of these communication capabilities.

4. A hardware independent Object Browser (designed and developed by the CAD Research Center) that facilitates user interaction within the object-based information context and the collaborative agent assistance capabilities of IMMACCS. Through the Object Browser the user may: set alert conditions (e.g., request warnings of enemy advances to within a user-specified radius of the current position of the operator); obtain agent reports and suggestions; request agent explanations; explore the location and capabilities of key resources (e.g., local police and fire stations, hospitals, and government buildings) on the object-based infrastructure display of the battlefield; and, enter information to automatically activate any other client(s) of the Shared Net.

5. A set of Translators (designed and developed by the SPAWAR Systems Center, San Diego, California) that are capable of mapping data received from external applications, such as the Joint Maritime Command Information System (JMCIS) and the Land Attack Warfare System (LAWS), to the object-based representation held within the IMMACCS Object Model.

6. A hardware independent, lightweight 2-D Viewer user interface (designed and developed by SRI International, Menlo Park, California) that connects the warfighter in the battlespace via wireless communication to IMMACCS. Since the Urban Warrior experiment the 2-D Viewer has been replaced by the Battlefield Visualization Tool (BVT) user-interface (developed by FGM, San Diego, California). In either case the user-interface hardware platform is provided with a differential GPS (Global Positioning System) device that transmits automatic position reports to IMMACCS. In this way IMMACCS is able to automatically track the current position of all beacon equipped friendly units, and

make this information available to agents as they spontaneously and opportunistically reason about events which might affect these units.

7. A Geographic Infrastructure Database (designed and developed by the Naval Research Laboratory at Stennis Space Center, Mississippi) that provides objectified battlespace infrastructure from National Imagery and Mapping Agency (NIMA) Vector Product Format (VPF) data.

These components are integrated (Fig.1) within the Integrated Cooperative Decision Model (ICDM) framework which the CAD Research Center has applied in several multi-agent decision-support systems over the past decade [Pohl 1994; Penmetcha et al. 1997, Pohl 1995, 1997, 1998].

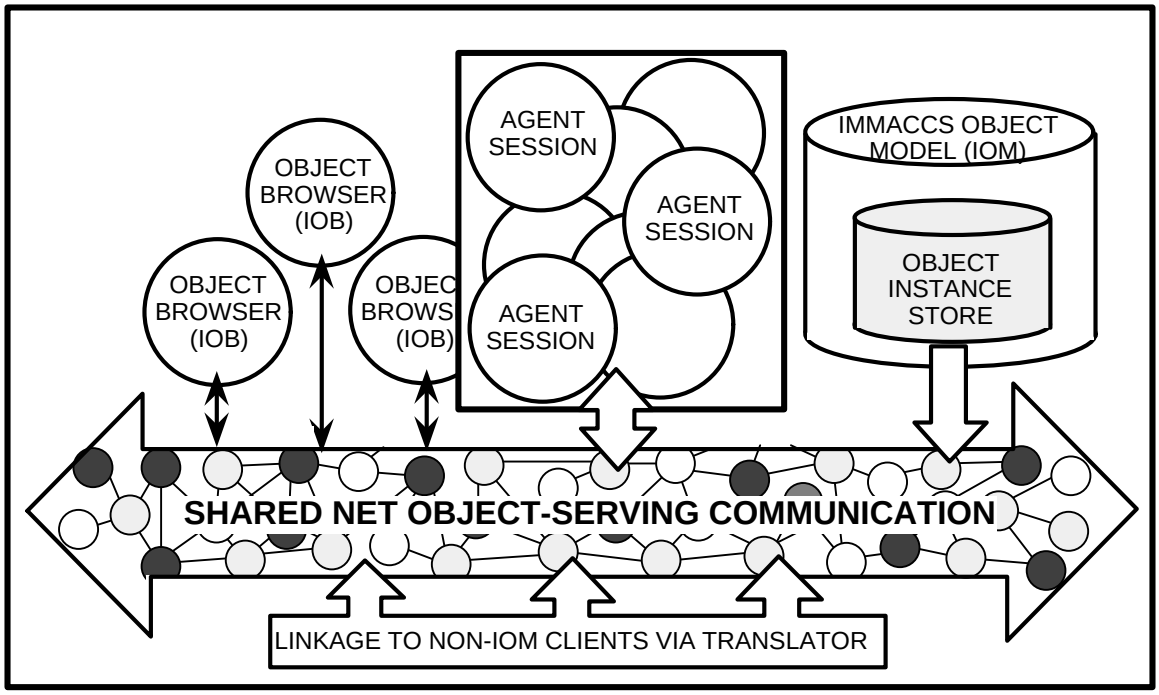

Fig. 1: Schematic representation of the IMMACCS components

The IMMACCS model is based on a three-tier architecture that makes clear distinctions between information, logic, and presentation. These tiers are represented by the three major IMMACCS system components; namely: the Shared Net (information tier); the Agent Engine (logic tier); and, the IMMACCS Object Browser (IOB), 2-D Viewer or BVT user-interface (presentation tier). Included in the information tier are two additional components. The first of these is the Translator providing bi-directional information translation between IMMACCS and external systems (e.g., JMCIS, LAWS, etc.). The second system is the Geographic Infrastructure Database responsible for providing geographic infrastructure information (e.g., buildings, roads, etc.) to the other IMMACCS components. Each of these tiers functions in an integrated fashion to form a comprehensive agent-based decision-support execution framework. This framework allows multiple human decision makers to solve complex problems in a collaborative fashion obtaining decision-support assistance from a collection of heterogeneous on-line agents.

The Shared Net Information Server

The Shared Net, developed by the Jet Propulsion Laboratory (JPL), functions as an object-serving communication facility. Clients subscribe to information and this information is automatically 'pushed' to the subscribers as soon as it is instantiated and posted in the Object Instance Store (Fig.1). Additionally, clients may send queries to the Shared Net and 'pull' information out of the Object Instance Store. In this respect the Shared Net operates very much in the fashion of a distributed object server based on the Common Object Request Broker Architecture (CORBA) specification [Mowbray and Zahavi 1995].

The information service capabilities of a distributed object broker obviate the need for clients to be knowledgeable of either the source or the form of the information. In other words, clients (including agents) communicate with the Shared Net and not directly with each other.

The Representation of Information

Fundamental to the decision-support capabilities of IMMACCS is the representation of information within the system as objects with behavioral characteristics and relationships with other objects [Myers et al. 1993]. It is important to note that the relationships among these objects are often far more important than the characteristics that describe the individual behavior of each object. While some of these associations are fairly static (e.g., a weapon is a kind of asset and a lethal weapon is a kind of weapon) many of the associations are governed by current conditions and are therefore highly dynamic. For example, as a platoon of soldiers moves through the battlefield it continuously establishes new associations (e.g., to windows in buildings from which snipers could fire on individual members of the platoon), changes existing associations (e.g., higher levels of risk as the platoon nears an active combat zone), and severs previous associations (e.g., as the platoon is forced to abandon its compromised command post).

Although distributed object servers by virtue of their name deal with objects, this in itself does not guarantee the kind of object-based representation described above. If the information is not represented at a high level upon its entry into the system, then the objects serve simply as shells (i.e., wrappers) for data. In IMMACCS, the Object Model serves as the information framework that preserves the objectified representation of information as it moves throughout the system, and the Shared Net incorporates an object-oriented database management system (OODBMS) for maintaining persistence.

The Agent Engine

The Agent Engine represents the logic-tier of the underlying three-tier architecture of IMMACCS. Existing as a client of the Shared Net the Agent Engine is capable of both obtaining and injecting information into the Shared Net. Architecturally, the Agent Engine consists of an agent server capable of supporting collections of agents. These collections, or agent sessions, exist as self-contained, self-managing, agent communities capable of interacting with the Shared Net to both acquire and contribute information. As a Shared Net client with interests in events and information, agent activity is triggered by changes in the environment represented by the IMMACCS Object Model (i.e., the battlespace). Regardless of whether agents are interacting

with the Shared Net or each other, interaction takes place in terms of objects. This again illustrates the degree to which an object representation is preserved as information as it is processed throughout IMMACCS.

Dividing agent analyses into heterogeneous collections of agents allows for a number of interesting configurations. These configurations determine the size, number, and individual scope of the agent sessions. While a wide variety of Agent Session configurations exist, IMMACCS has found considerable success in formulating these configurations based on two primary criteria.

The first criterion introduces the notion of a 'view'. A view is a conceptual perspective of reality. In other words, a view can be thought of as a single investigation into solving a problem whether it be based on fact or speculation. For example, a view may describe events and information relating to what is actually occurring in reality. Another view may describe an alternative or desired reality. As an example, IMMACCS uses a single view to represent the information and events actually occurring in the battlespace. In a similar manner, IMMACCS employs any number of additional views to represent hypothetical investigations to determine suitable strategies for dealing with potential events or circumstances. Regardless of use, however, there is a one-to-one correspondence between a conceptual view and an Agent Session (Fig.2). This means that independent of which version of reality a view represents, there exists a dedicated Agent Session providing users of that view with agent-based analysis and decision-support. Each agent of a particular Agent Session deals only with the view associated with its Agent Session. Organizing information analysis in this manner allows for an efficient and effective means of distinguishing activities and information relating to one view from activities pertaining to another. Unless prompted by user intervention, each set of information is completely separate from the other.

The second configuration criterion specifies the number and nature of the agents contained in an Agent Session at any particular point in time. IMMACCS employs two types of agents to populate an Agent Session. These agent types are Domain Agents and Mentor Agents [Pohl 1995]. Service-oriented Domain Agents embody expertise in various command and control domains (i.e., fires engagement, logistics, intelligence, fratricide, etc.). The collection of Domain Agents populating an Agent Session at any point in time determines the variety of domain specific perspectives and analytical depth available during analysis of the associated view. Under the configuration scheme utilized by IMMACCS, users can add or remove these domain perspectives in a dynamic fashion over time.

Mentor Agents, on-the-other-hand extend the notion of high-level information representation by essentially agentifying information through empowering information objects with the ability to act on their own behalf. This agentification of information into Mentor Agents can be initiated by both human users or other agents on an as-needed basis.

Within the IMMACCS model each of these agent contingents is dynamically configurable by both the user(s) in addition to the system itself. This approach to Agent Session configuration promotes the notion of offering assistance in the form of dynamically configurable tools rather than predefined solutions.

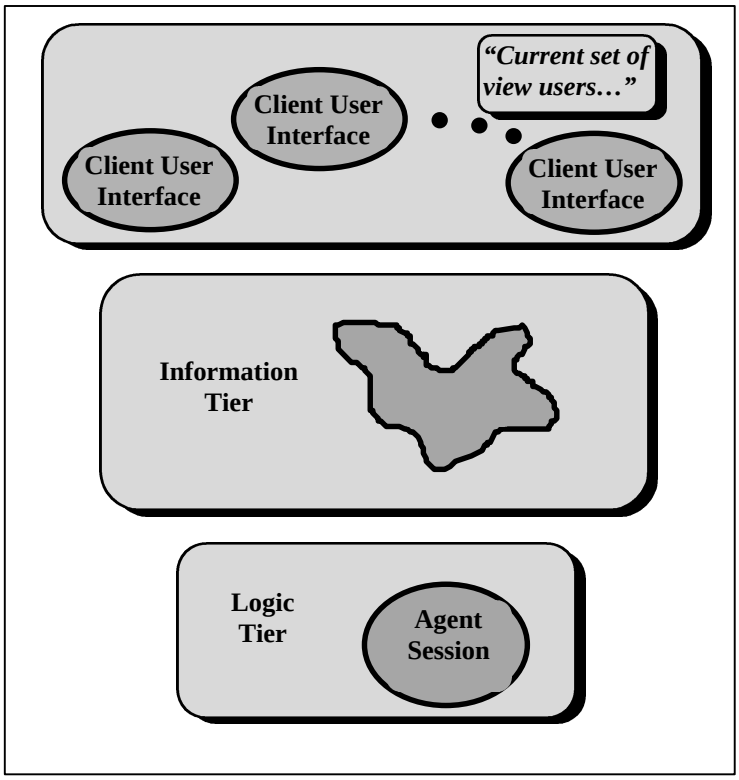

Fig.2: Multiple users can interact with a 'view' which in turn is analyzed by a single Agent Session.

Architecturally, an Agent Session consists of several components including the Semantic Network, Object Manager, Session Manager, Inference Engine, and Agent Manager (Fig.3). These components operate in an integrated fashion to maintain a current information connection between the agents residing in the Agent Session and the associated view described in the Shared Net.

The Semantic Network consists of a collection of two sets of application specific information objects. The first set is used for local collaboration among agents. Agents may use this local Semantic Network to propose recommendations or request various services from each other. This information is produced and modified by the agents and remains local to the Agent Session. The second set of information is essentially a mirror image of the view information stored in the Shared Net. In actuality, this information exists as a collection of object-based interfaces allowing access to information pertaining to a view stored in the Shared Net. Such interfaces are directly related to the IMMACCS information object model. In other words, these interfaces or proxies [Mowbray and Zahavi 1995], are represented in terms of the objects described in the IMMACCS Object Model (IOM). Through these interfaces, Shared Net clients have the ability to access and modify objects contained in the Shared Net as though they are local to the client's environment. All communication between the object interfaces and their remote object counterparts is encapsulated and managed by the Shared Net in a manner that is transparent to the clients.

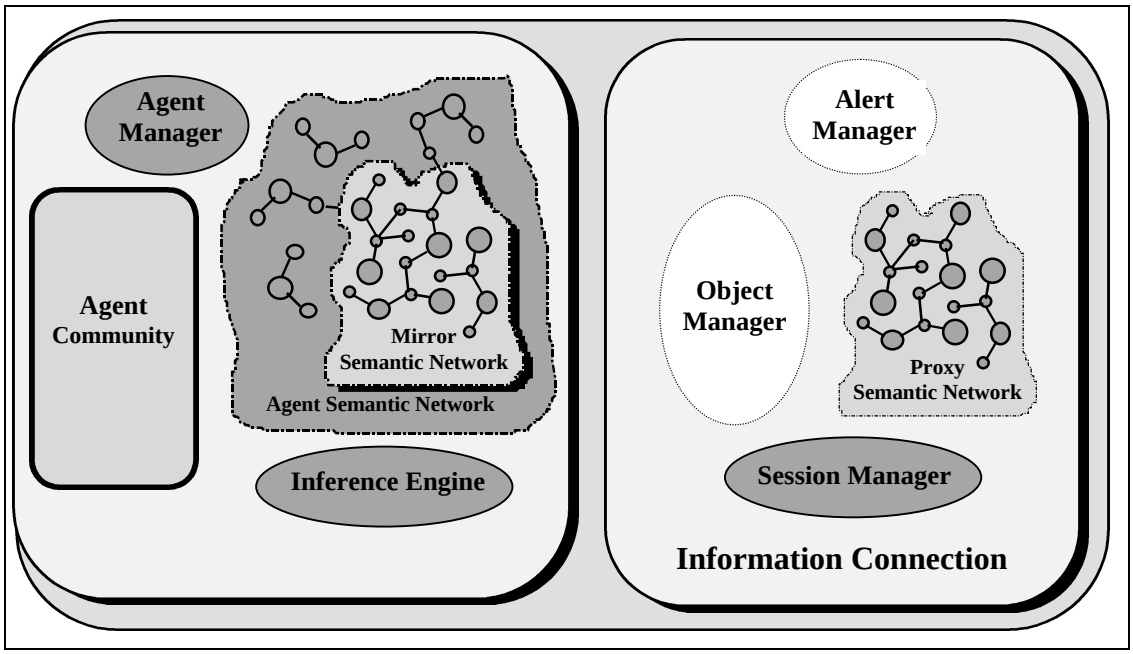

Fig.3: Agent Session Architecture

As the primary manager of the two sets of information described above, the Object Manager focuses the majority of its efforts on the management of the bi-directional propagation of information between Shared Net proxies and an equivalent representation understandable by the Inference Engine. The purpose of this manager is to maintain mappings between the Shared Net proxies and their corresponding Inference Engine counterparts. An additional responsibility of the Object Manager deals with the subscriptions, or interests held on behalf of the agent community. In other words, the Object Manager is responsible for maintaining the registration of a dynamically changing set of information interests held on behalf of the Agent Session agents. In addition, the Object Manager is responsible for processing notifications when these interests are subsequently satisfied. Such processing includes the propagation of information changes to the agent community which may in turn trigger agent activity. To perform these two interest-related tasks the Object Manager employs the services of the Alert Manager. The Alert Manager exists as an interface to the Shared Net subscription facility and is available to any Shared Net client wishing to maintain a set of information interests.

The Inference Engine provides the link between changes occurring in the Semantic Network and agent activation. For efficiency, IMMACCS employs the CLIPS expert system shell which is based on the RETE algorithm [NASA 1992]. Since agent activation can occur when a change in the Semantic Network is of interest to a particular agent, if follows that in such a case the Inference Engine having knowledge of specific agent interests in addition to changes occurring in the Semantic Network, is responsible for activating or scheduling the actions of the agents. This activation list forms the basis for the Agent Manager to determine which agent actions to execute on behalf of the currently scheduled agents.

The Agent Manager is responsible for the management of the agent community housed in an Agent Session. This management includes the instantiation and destruction of agents as they are

dynamically allocated and de-allocated to and from the agent community. In addition, the Agent Manager is responsible for managing the distribution of execution cycles allowing each agent to perform its actions. Disbursement of execution cycles occurs in a round-robin fashion allowing agent analysis to be evenly distributed among the activated agents.

As the overall manager of the Agent Session environment the Session Manager has two main responsibilities. The first focuses on the initialization of each of the other Agent Session components upon creation. When an Agent Session is created as a response to the establishment of a view, the Session Manager is the first component to be activated. Once initialized, the Session Manager activates the Object Manager and Inference Engine. Continuing its efforts, the Session Manager then activates the Agent Manager.

Upon start-up, the Agent Manager initializes itself by allocating an appropriate initial set of agents. Once allocated, these agents register various subscriptions with the Shared Net based on their current set of interests. Through various queries directed at the Shared Net, the agents also begin to familiarize themselves with the current events and conditions in the battlespace.

Current IMMACCS Agent Capabilities

During the Urban Warrior and LOE-6 experimental exercises the IMMACCS Agent Engine supported the following agent capabilities.

Sentinel Agents

Sentinel Agents are dynamic agents created by the Agent Engine as a user (e.g., military unit) in the battlespace logs into IMMACCS. Sentinel Agents monitor and generate alerts for enemy units coming within a specified radius (i.e., 300 meters during Urban Warrior) of its unit. The Sentinel Agent also monitors Calls for Fire (CFF) and alerts its unit of fire missions with targets closer than 300 meters.

In addition, Sentinel Agents handle the task of creating a unit's weapons assets. When logging into IMMACCS from the battlespace a unit typically supplies only its call sign, force code and location (i.e., actually, these are automatically received from the end-user terminal without the need for explicit user actions). The Sentinel Agent then maps that unit to encyclopedic data providing more robust information and relationships for the agents to reason about.

Fire Agent

The Fire Agent is a static agent that responds to Call for Fire (CFF) messages by determining the best weapon that is available, deliverable, and acceptable. The Fire Agent's weaponeering capabilities currently address range, time of flight, target type, urgency, circular error of probability (CEP), effective casualty radius (ECR), availability, and rules of engagement (ROE). The Fire Agent's deconfliction capabilities include the trajectory of munitions relative (i.e., within time and space) to the position of other friendly assets (e.g., people, equipment, and other munitions), civilian tracks, and infrastructure objects.

Rules of Engagement (ROE) Agent

The ROE Agent is a static agent that monitors Call for Fire (CFF) messages alerting to violations in rules of engagement (e.g., fire missions that target on or near buildings designated as off-limits). The ROE Agent augments the Fire Agent by alerting on available and deliverable weapons that violate the current ROE.

Engagement Agent

The Engagement Agent is a static agent that monitors Call for Fire (CFF) messages alerting to enemy units being directly or indirectly targeted.

Blue-on-Blue Agent

The Blue-on-Blue Agent is a static agent that monitors Call for Fire (CFF) messages alerting for fratricide conditions, such as when a friendly unit is being directly or indirectly targeted.

Logistics Agent

The Logistics Agent is a static agent that monitors the level of supplies of blue units. The Logistics Agent generates 'yellow' and 'red' alerts as levels of certain logistics supply items, such as fuel and water, fall below preset thresholds. Upon alert creation the location of potential re-supply points are highlighted on the map display.

Hazard Agent

The Hazard Agent is a static agent that monitors the battle space for indications of nuclear, biological and chemical (NBC) weapons. For example, upon receipt of atmospheric events the Hazard Agent generates an alert indicating the presence of an NBC condition and highlights an approximate area of coverage. Units within the coverage area are also automatically identified.

Intelligence (Intel) Agent

The Intel Agent is a static agent with some dynamic capabilities. Based on the request of the commander of the friendly forces (i.e., SPMAGTF(X)) during Urban Warrior and LOE-6 the Intel Agent was provided with two unassociated capabilities.

The static capability required the Intel Agent to monitor the battle space for instances of hostile Air Defense Systems (ADS). Detection of an ADS in passive mode generated an alert of the existence of this high value target, while detection of an ADS in active mode generated an alert of the existence and the automatic creation and submission of a Call for Fire (CFF).

The dynamic capability of the Intel Agent allowed the user to create Named Areas of Interest (NAI). The Intel Agent then monitored enemy track movements generating alerts when a track entered the NAI. The Intel Agent also monitored successive movements of the track alerting on its speed and bearing.

IMMACCS User-Interfaces

Representing the third and final tier of the three-tier architecture employed by IMMACCS the Client User Interface (CUI) exists as a culmination of instances of the 2D-Viewer or BVT, and the IMMACCS Object Browser. Collectively, these user-interface choices provide human users with a means of viewing and manipulating the information and analysis provided by the other two tiers of the IMMACCS decision-support system.

As clients of the Shared Net, CUI users have the ability to interact with each other and the agents. By either injecting or obtaining information from the Shared Net, CUI users working on the same view have the potential of exchanging strategic or other kinds of information in a collaborative manner. This type of information exchange occurs regardless of whether the relevant view represents the Common Operating Picture or exists as a localized strategy explored by a subset of users. All information and analysis remains localized within a particular view unless explicitly copied into another view through user interaction. In this manner, no informational or analytical collisions occur between conceptual views without the potential for user-based supervision and subsequent reconciliation.

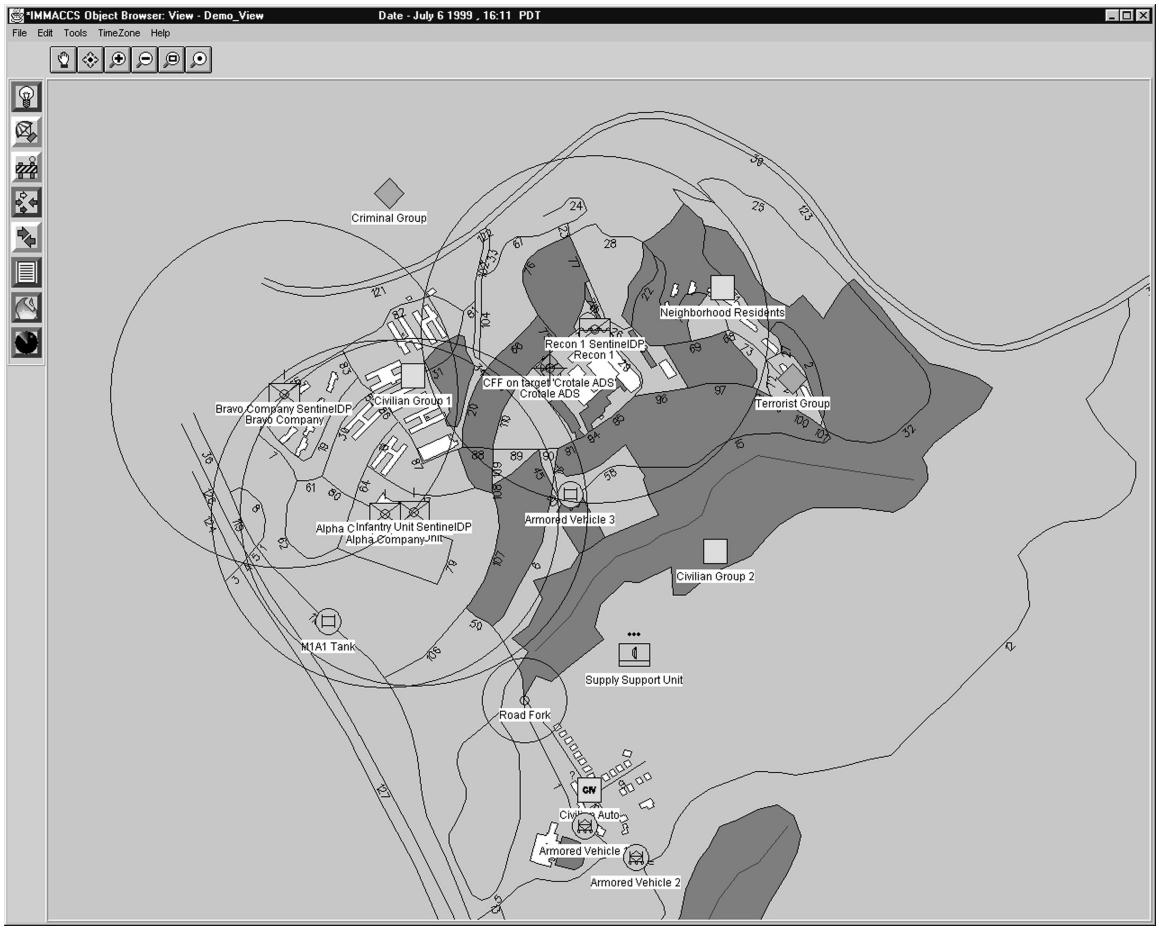

Fig.4: Typical Common Tactical Picture screen of the IMMACCS Object Browser user-interface showing a column of agent status windows on the left and the Oak Knoll (Oakland, California) battlefield site of the Urban Warrior experimental exercise in the central area of the screen.

In the typical Common Tactical Picture view provided by the IMMACCS Object Browser CUI, shown in Fig.4, agent status windows or icons are located on the left side of the screen. An agent alert is indicated by the appearance of a red border around the appropriate icon. Once this alert has been acknowledged by the user the color of the border changes to blue. The concurrent existence of multiple alert conditions in the same agent (i.e., some alerts have been acknowledged and others have not) is signified by a yellow border.

Conclusion

A collaborative agent-based command and control system, such as IMMACCS differs from conventional human-based command and control system in several significant respects. Firstly, the continuous and automatic monitoring of warfighting units by the various types of agents that operate spontaneously within the communication system potentially provides the warfighter with access to instantaneous advice and guidance. The agent to agent communication which facilitates this continuous access to information and intelligent analysis is not dependent on human to human interaction. In a conventional command and control system the communication channels are easily saturated by the continuous flow of human to human electronic and voice communications. Efforts to control this traffic inevitably require the imposition of communication restrictions that can easily prevent critical information from reaching the appropriate commander or warfighter. In addition, the human to human interaction encourages a build-up of support personnel in and around the theater. This build-up is costly in terms of transportation and logistics, increases the danger of casualties, and places an additional burden on the already overloaded communication facilities.

Secondly, the multi-agent system architecture decentralizes both the collection and analysis of information. Individual warfighting units serve equally well as collectors and generators of information, as they do as recipients of information. In this way a dispersed force of warfighters can represent an important sensor array, with the ability to add value by converting data into information and knowledge close to the source. This decentralization of the data analysis process is particularly valuable in terms of distributing the communication traffic and validating the results of the analysis at the collection source.

Thirdly, the seamless integration of planning, execution and training functions within the same command and control communication system allows the commander and the individual warfighter to continuously and instantaneously switch from one mode of operation to another. In fact, the parallel nature of the system allows specific planning, execution and training tasks to be undertaken concurrently. For example, the commander may wish to initiate a planning function through one set of agents while executing a specific operation in the theater, and at the same time simulate a particular ‘what if’ scenario in anticipation of a possible future situation.

Acknowledgements

The IMMACCS project is sponsored by the US Marine Corps Warfighting Laboratory, Quantico, Virginia, with design and development responsibilities assigned as follows: overall design concept, Agent Engine, Object Model, and Object Browser (Collaborative Agent Design (CAD) Research Center, Cal Poly, San Luis Obispo, California); Shared Net and Object

Instance Store (Jet Propulsion Laboratory, Cal Tech, Pasadena, California); objectified infrastructure (Navy Research Laboratory, Stennis Space Center, Mississippi); 2-D Viewer and Backup System (SRI International, Menlo Park, California); Battlefield Visualization Tool or BVT (FGM, San Diego, California); Translator(s) for external (i.e., legacy) applications and System Engineering Integration (SPAWAR Systems Center, San Diego, California).

References

[Minsky, 1982] M. Minsky. *Why People Think Computers Can't*. AI Magazine, 3(4), Fall, 1982.

[Mowbray and Zahavi 1995] T. Mowbray and R. Zahavi. *The Essential CORBA: Systems Integration Using Distributed Objects*. Wiley, New York, New York, 1995.

[Myers, *et al.* 1993] L. Myers, J. Pohl, J. Cotton, J. Snyder, K. Pohl, S. Chien, S. Aly and T. Rodriguez. *Object Representation and the ICADS-Kernel Design*. Technical Report (CADRU-08-93), CAD Research Center, Cal Poly, San Luis Obispo, CA 93407, January, 1993.

[NASA 1992] NASA. *CLIPS 6.0 Reference Manual*. Software Technologies Branch, Lyndon B. Johnson Space Center, Houston, Texas, 1992.

[Penmetcha, *et al*. 1997] K. Penmetcha, A. Chapman and A. Antelman. *CIAT: Collaborative Infrastructure Assessment Tool*. in Pohl J. (ed.) Advances in Collaborative Design and Decision-Support Systems, Focus Symposium: International Conference on Systems Research, Informatics and Cybernetics, Baden-Baden, Germany, Aug.18-22 (pp.83-90), 1997.

[Pohl 2000] J. Pohl. *Adapting to the Information Age*. InterSymp-2000, 12th International Conference on Systems Research, Informatics and Cybernetics, Baden-Baden, Germany, Jul.31 to Aug.4, 2000.

[Pohl, *et al*. 1999] J. Pohl, M. Porczak, K. Pohl, R. Leighton, H. Assal, A. Davis, L. Vempati and A. Wood. *IMMACCS: A Multi-Agent Decision-Support System.* Technical report (CADRU-12-99), CAD Research Center, Cal Poly, San Luis Obispo, CA 93407, August, 1999.

[Pohl, *et al*. 1997] J. Pohl, A. Chapman, K. Pohl, J. Primrose and A. Wozniak. *Decision-Support Systems: Notions, Prototypes, and In-Use Applications*. Technical Report (CADRU-11-97), CAD Research Center, Cal Poly, San Luis Obispo, CA 93407, January.

[Pohl, *et al*. 1994] J. Pohl, L. Myers and A. Chapman. *Thoughts on the Evolution of Computer-Assisted Design.* Technical Report (CADRU-09-94), CAD Research Center, Cal Poly, San Luis Obispo, CA 93407, September, 1994.

[Pohl 1998] K. Pohl K. *The Round-Table Model: A Web-Oriented Agent-Based Framework for decision-Support Applications*. in Pohl J. (ed.) Advances in Collaborative Decision-Support Systems for Design, Planning, and Execution, Focus Symposium: InterSymp-1998, International

Conference on Systems Research, Informatics and Cybernetics, Baden-Baden, Germany, Aug.17-21 (pp.47-59), 1998.

[Pohl 1995] K. Pohl. *KOALA: An Object-Agent Design System.* in Pohl J. (ed.) Advances in Cooperative Environmental Decision Systems; Focus Symposium: InterSymp-1995, International Conference on Systems Research, Informatics and Cybernetics, Baden-Baden, Germany, Aug.14-18 (pp.81-92), 1995.