

A Hierarchical Collective Agents Network for Real-time Sensor Fusion and Decision Support

Plamen V. Petrov¹, Qiuming Zhu², Jeffrey D. Hicks³, and Alexander D. Stoyen⁴

^{1, 3, 4} 21st Century Systems, Inc.

Omaha, Nebraska, USA 68154

plamen@21csi.com

² Department of Computer Science

University of Nebraska at Omaha, Omaha, Nebraska, USA 68182-0500

zhuq@unomaha.edu

Abstract

This research addresses a problem of how to make effective use of real-time information acquired from multiple sensor and heterogeneous data resources, and reasoning on the gathered information for situation assessment and impact assessment (SA/IA), thus to provide reliable decision support for time-critical operations. A hierarchical collective agents network (HCAN) is presented as a solution to this problem. The agents network supports multi-sensor registration, real-time sensor/platform cueing, level-2 and level-3 data/information fusion, and has an arm toward the level-4 fusion objectives. An agent component assembly and decision support system development environment, the 21 Century systems' AEDGETM software package, is used for the design and implementation of a HCAN system.

I. Introduction

The ability to integrate and correlate a vast amounts of disparate information from multiple sensor and heterogeneous data resources with varying degrees of uncertainty in real-time is an impediment issue for mission-critical decision support systems (DSS). For example, in crucial military operations command officers need real-time information and intelligences from sensors/data resources in a theater of reconnaissance and surveillance to build a whole picture of the battle-space (Petrov and Stoyen 2000). It is critical for the commanders to know and to understand the relationships among the data /information collected. While assessments are made, questions are asked, such as: what are the physical and functional constituency relations among the objects in a given geographic sector? Are there sequential or temporal dependencies of the objects and what will trigger them? What are the possible consequences of the action and re-actions? Decision

making based on these situation assessment and impact assessment (SA/IA) are particularly important for identifying and prioritizing "gaps" between the operation planning and the real-time interactions.

A field of technology that supports intelligence decision making is known as data fusion, that is, the automation of processes to combine diverse sets of raw data from different sources into a single set of meaningful information that is greater than the sum of its contributing parts (Chen, Zhu, and Chen 2001). In general, data fusion is defined as a process dealing with the association, correlation, and combination of data and information from single and multiple sources to achieve refined position and identity estimates. The results include a complete and timely assessment of situations and threats, and their significance. The process is characterized with continuous refinements of its estimates and assessments, and the evaluation of the need for additional sources, or modification of the process itself, to achieve improved decision reliability and efficiency. Steinberg (Steinberg, Bowman, and White 1999) later defined data fusion in a more concise statement as "the process of combining data to refine state estimates and predictions." The decision support component is partly kept in this definition as a coherent element of the estimates and predictions, though decision-making involves much more than just estimates and predictions.

In a mission-critical military theater/operation demanding decision support, timely and accurate data fusion is a force multiplier. The lower-level data fusion from single or multiple sensor resources has become relatively well understood, resulting in accurate positional tracks and identification of physical objects (Bogler, 1987). However, the processes for higher levels of data fusion, namely the level 2 – situation assessment, and level 3 – impact assessment (SA/IA), still requires the study and development of mathematically rigorous techniques and computational schemes. The kind of robust, integrated

fusion architectures for handling increasing diversity of input sources are especially important in contemporary decision support missions. For example, a well crafted computer system integrating knowledge acquisition tools and proper decision support models can assist military operation planners in their tactical decision-making situations in many different ways, particularly with respect to quickly identifying responses and counter-responses to enemy action or inaction, so as to provide a faster and comprehensive picture to the field units.

To support making situation assessment and impact assessment, a data fusion and decision support system is required to use a set of coherent patterns derived from the available data sets and then infer the implications (e.g., causal relations) from these data patterns to real world situations. The attribute coherence that is critical to the formation of meaningful knowledge patterns is often obscure in data sets obtained from various heterogeneous resources. In reality, the data collected are often incomplete, imprecise, and inconsistent due to various natural constraints and human faults. Decision makers naturally want to access large quantities of information expressed in diverse forms. However, as new sensor technology and various information sources have combined to create quantity and diversity, it has become increasingly difficult to provide decision makers with the right information at the right time and in the right quantity and format. The extensive body of work on Multi-Criteria Decision Aids (MCDA) (Vincke 1992) provides some concrete steps for solving decision problems. However, MCDA has limitations. Specifically, the technique is based on the assumptions that the relevant criteria for decision objectives are well defined and certain. Unfortunately, for cases where uncertain information exists the computation results a sum over an exponential number of terms (Stoyen et al. 1994).

Real-time decision support systems are constructed by integrating a number of diverse components from a variety of software modules. Some of these components are built as special-purpose applications, some are pre-existing off-the-shelf products, some are built in-house, and some are purchased from third party vendors. Much of the complexity found in DSS applications is a result of the way the different types of software components are pieced together. In order to ensure the quality of such a system, novel techniques are needed to integrate and evaluate the resulting connections of the various components. It is the most desirable to allow end users of the DSS to develop their own systems that meet their own needs by using user-friendly and efficient DSS development tools.

Software developers have come to a numerous ways of querying the local and centralized data resources to access and distill the large and diverse information for the purpose of providing effective decision support (Giri and Zhu 1998).

Meanwhile DSS are becoming more and more complex in terms of knowing which data resources to connect, how to keep track of the data dynamics, and assess the reliability of information from each resources. These tying links make the use of intelligent agent architecture necessary for allowing real-time responsibility and adaptive control of the DSS. It is very important that we understand the issues in designing, constructing, and implementing such systems to work with heterogeneous resources (Lee, Offutt and Alexnader 2000). While the classical method of going through the software development life-cycle still remains valid for programming decision support systems, a number of new issues are to be dealt with, such as the blending of diverse platforms in client-server protocol, agent-to-data resource connectivity, machine-learning for "intelligent" data mining engine, etc.

The field of data fusion and DSS development can benefit significantly by focusing the major concerns on employment of agent-based technologies (Maes 1994). Given the characteristics of most decision making situations, it seems that the most natural way to provide timely and critical support to decision making processes is to have a collection of distributed, autonomous problem solving intelligent agents working together on different aspects of a decision support endeavor. This research thus addresses the problems of how to make effective use of real-time information acquired from multiple and heterogeneous sensor and data resources, and reasoning on the gathered information for situation assessment and impact assessment through a hierarchical collective agents assembly organized in a network structure. The system is to provide reliable decision support for time-critical missions such as military operations. In the following, section II describes a structural paradigm of our hierarchical collective agents network (HCAN). Section III presents a software environment, the 21Century Systems' AEDGE™, for the development of the HCAN kind of data fusion and decision support systems. A design of HCAN using AEDGE™'s agent components architecture is described in section IV. An example of the HCAN application to defense operation is discussed in section V. Section VI contains conclusion remarks.

II. The HCAN Paradigm

As we have pointed out above, complex DSS are constructed by integrating a number of diverse software components. These components need to be intelligent, heterogeneous, distributed, autonomous, and concurrent. In addition to that they need to exhibit various types of real-time control and data coupling capabilities, and meet multiple sets of requirements for reliability and any-time/any-where availability (Sycara and Zeng 1996). Software agent architecture has been the favored, thus mostly

adopted, scheme for such DSS developments (Lejter and Dean 1996).

In performing the data fusion and decision support tasks, the agents assembly must be in charge of determining registered sensors in field of view of object in question, cuing applicable sensors to obtain additional information about objects, take data from various sources

and combine them into fused object information, and then provide real-time decision support for the operation. Figure 1 shows a basic modular structure of the agent-based data fusion and decision support system paradigm. We call it a hierarchical collective agent network (HCAN) for data fusion and decision support.

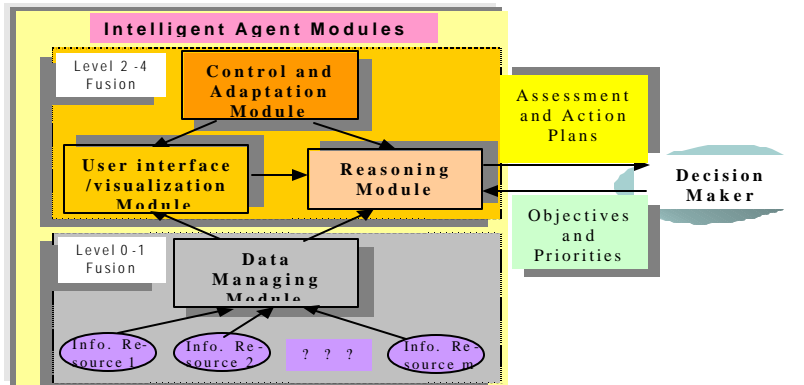


Figure 1 Modular diagram of the HCAN for data fusion and decision support

In this structure, four basic functional modules are included. They are (1) Data managing module, (2) Reasoning module, (3) Control and adaptation module, and (4) User interface/Visualization module. The data-managing module is mainly in charge of the level 0-1 fusion – the sensor fusion and data pre-processing tasks. It is at this modular level that connects to the various data resources are made. The module will automate the information retrieval and integration from heterogeneous resources in a user-friendly and hassle-free manner. The functions in this module will also provide effective means for extracting useful information from the sensor/data resources and perform filtering operations to assist the reasoning module in knowledge distilling. The reasoning module takes the

filtered data from the data-managing module, performs various kinds of correlation analyses and distills the data collections toward representations of decision supporting knowledge. The user interface/Visualization module facilitates the data fusion and decision support functions of the overall system. It is noted that a control and adaptation level is at the top of the reasoning and user-interface modules. That is, these modules are organized in a hierarchy in terms their functionality and inter-connections. Figure 2 shows a layered structure of the assembly. The collective efforts of these modules are combined together in the hierarchical assembly to provide necessary support for decision-making under critical and uncertain circumstances.

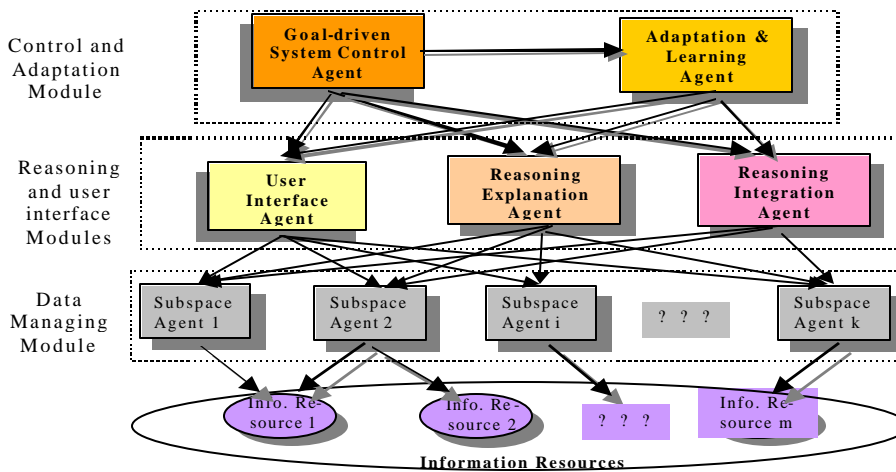


Figure 2 HCAN system structure and control diagram

The HCAN architecture has three distinct features as compared to previous approaches, especially the popular cooperative multi-agents architectures. These features are: (1) The agents in the HCAN are organized with layered supervision rather than layer-less objects; (2) Relations between agents at the same level of the HCAN are collective in nature; and (3) A top-down and bottom-up two-way iterative agent-activation process is employed to coordinate the agent actions.

Previous work in multi-agent systems concentrated on domain-independent frameworks, standard protocol definitions, some handling of uncertainty and utility, and extensive models of collaboration (Giampapa, Paoluc, and Sycara 2000). However, there lacks methods for solid decision-theoretic model of agents learning, adaptation, control and collaboration. desJardins (desJardins 1993) described a model of autonomous learning in probabilistic domains. The model incorporates techniques for using the agent's existing knowledge to guide and constrain the learning process and for representing, reasoning with, and learning probabilistic knowledge. Rodriguez and Poehlman (Rodriguez and Poehlman 1997) explored the use of multiple inference-driven agents cooperating over a network. It is interesting to note that information uncertainties are handled effectively by the agent technique. For example, Buller (Buller 1992) introduced an approach for fuzzy knowledge processing in which the fuzziness is assumed as a co-existence of populations of contradictory statements in a working space. Recently, Arai *et al* presented a reinforcement learning approach known as Profit-sharing that allows agents to learn effective behaviors with in dynamic and multi-agent environments (Arai, Sycara and Payne 2000). In an agent-based software developed by Giri and Zhu (Giri and Zhu 1998), the agent helps users to initiate reasoning queries upon the request of a conceptual query processor and to analyze intermediate results and form better queries consequently for decision explanatory.

Many popular multi-agent systems of today deploy agents in a uniform space of operating. The agents are supposed to respond to the same calls and cooperate at the same time toward the goals of operation. That kind of architecture is useful for some applications. However, it endues some difficulties in agent communications and task control. When applied in complex real-time DSS with intensive human and system interactions, the cooperative nature makes the system less robust because the disability of one agent would affect the successive operations of the entire agent assembly.

The collective nature of the agents in the HCAN paradigm overcomes some of these difficulties, for example, relieving the burden of data-exchanges between fellow agents by limiting agent communication to vertical layers

of the assembly only. The collective nature of agent relation in the hierarchical architecture simplifies the functional design of the agent interactions and enhances the security and efficiency of the information processing. The HCAN architecture also strikes a balance between the centralized control and distributed computation by allowing distributive agent operations within layers of the hierarchy and enforcing centralized control between the layers of the hierarchy, thus creating a federated agents integration structure.

In the HCAN, agents at the lower level (the data managing module) interface directly to individual sensor/information resources. These agents act in a distributive fashion to process conceptual queries, filter retrieved information using simple proposition logics, and extract useful information as instructed by upper-level (the reasoning or user interface modules) agents. The agents at the upper levels coordinate the activities of the agents at the lower levels using a centralized goal-driven control strategy. They issue conceptual queries, perform data integration and knowledge extraction, and make cross-reference of the information retrieved. The coordinate agents at these levels will apply certain data analysis models and employ reasoning-integration technique to fuse information reported by retrieval agents at the lower levels. Special human-system interfacing agents will provide continual support for interactions between user and the systems, and provide intelligent and dynamic information summarization, annotation, and presentation based on the user-originated inputs and queries.

Basically, the HCAN has the following functionalities.

- 1). A flexible software architecture for accommodating system augmentation and evolutions;
- 2). A powerful representation schema for accommodating heterogeneous forms of information;
- 3). A diverse interface for various input resources, output formats, and human interactions;
- 4). An ability of reasoning on incomplete and inconsistent information, and extracting useful knowledge from the data of heterogeneous resources;
- 5). An ability of incorporating real-time dynamics of the information resources into the system anytime during the operation, and promptly adjusting the reasoning mechanisms;
- 6). An ability of summarizing and refining knowledge extracted, and distinguishing mission and time critical knowledge from insignificant and redundant ones;
- 7). A capability of supplying meaningful and accurate explanations, both qualitatively and quantitatively, of the automated system actions; and
- 8). A capability of providing adequate control and scrutinizing of the system operations under the environmental constrains of the given situation.

There are many sources of uncertainty at different levels of the decision support computation in the HCAN. For example, even if a situation-assessor is aware of the presence of certain objects in the operation space, such as the type of contact, intention, reaction rational, etc., the exact dynamics of the object is still uncertain to the decision maker. The knowledge about the object dynamics is critical in constructing an optimal strategy of action. Various statistical methodologies and knowledge discovery techniques will be applied in the reasoning module. The level of uncertainty forces the reasoning agents to operate with different decision strategies.

The control and adaptation module is an important component of the agent assembly. It is responsible for supervising the activities of the agents at the reasoning level, and making analysis of the inferences made by the reasoning agents, and providing improvements to the reasoning models and parameters. The agents in this module would apply previously captured knowledge through experience or decoded from known decision rules to initialize the parameters of the reasoning model, and make dynamic updating of these parameters. For example, by activating the adaptation agents, the a priori probabilities and the conditional probabilities of the Bayesian decision model can be established, obtained, and refined conveniently. The presence of the adaptation module also allows the DSS to improve the performance of the knowledge acquisition functions over operation times.

The agents in the control module also constantly evaluate the available information about the operation environment and compute the probabilities of the assertions on each of the objectives. A set of states is defined to span the possible states of decision-making. Based upon the priorities selected, the state will change such that the actions recommended by the control agent are selected to optimize the desired outcome. The mathematical problem is subject to Stochastic Control Theory. The SCM determines the state of the encounter through the use of Discriminant Analysis (Nath, Jackson and Jones 1992) that uses multiple measurements of an object or situation to classify it in one of several categories. From this data, the reasoning agents will analyze the situation and recommend the optimal course of actions to achieve the goals. This optimization spans all possibilities and is computationally intensive.

III. AEDGE™ – A DSS Design Engine

The 21st Century Systems, Inc. has developed the AEDGE™ as an open DII-COE and CORBA compliant agent-based environment that enables the development of components-based agent systems. The system is implemented in Java™, with Java Database Connectivity™ for DB access, Java Swing, AWT, and Java3D for visual

interfaces, Java Media Framework and Java Speech API for audio/speech interface (Gosling and McGilton 1995). AEDGE™ defines Agents, Entities, Avatars and their interactions with each other and with external sources of information. This standardized architecture, as shown in Figure 3, allows additional components, such as service-specific DSS tools, to be efficiently built upon the core functionality. Common interfaces and data structures can be exported to interested parties who wish to extend the architecture with new components, agents, servers, or clients. When the core AEDGE™ components are bundled with customer-specific components in an integrated environment, a clean separation of those components, through APIs, is provided.

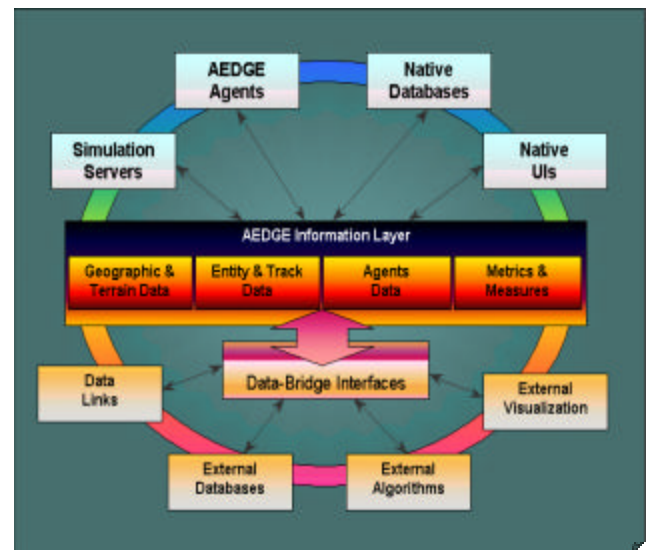


Figure 3. AEDGE™ system components architecture

The AEDGE™ is based on an extensible multi-component DSS architecture (EMDA, also referred to as the AEDGE™ Architecture). The architectures describe the data objects, interfaces, communication mechanisms, component interactions, and integration mechanisms for the AEDGE™ and its extensions (such as ABS- AEDGE™). In the AEDGE™ architecture, components communicate among each other via the Service Provider/Service Requester Protocol (SPSR). Service providers are components that implement an algorithm or need to share their data (data sources), as shown in figure 4. Service requesters are the components that need a function performed for them by some other component or need to import data from another component. Both service requesters and service providers implement remote interfaces, which enables such components to communicate over a TCP/IP network. The remote interface implementation is currently based on Java RMI (remote

method invocation, a type of simplified ORB service), though the Architecture is not dependent on this implementation.

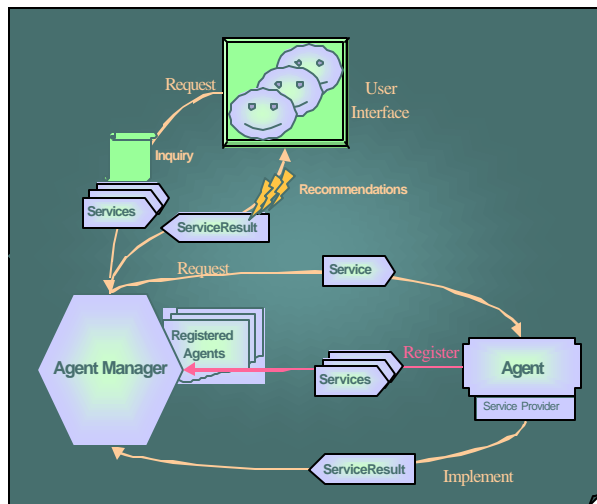


Figure 4 AEDGE SPSR protocol interactions

The SPSR protocol is based on three data objects: Service, Service-Result and Message. The Service object encapsulates the class, the type, the required quality of service (QoS) and the parameters of a service request. The Service-Result object provides a reference to the original service, a return code (success or failure), a return object (String, Recommendation, etc), and an actual received QoS. Messages provide a way of service providers to advertise the availability of new services and to notify subscribers of new data available. Figure 5 shows a diagram of the message passing process among the Web interactive agent components over AEDGE™ services. In this protocol, Service provider components register their location and the services they provide with a Component Registry, which is responsible for tracking and maintaining service provider information current. Service requesters lookup service provider information from the Component Registry and then establish a direct connection with the providers they wish to engage. A service request (either blocking or non-blocking) is sent from the service requestor to the service provider. The provider then replies immediately or at some future time with a Service-Result.

AEDGE™ provides multiple levels of customization. The subject-matter users are able to build scenarios and scripts or to automatically generate them using the AEDGE™-based Scenario Editor. Rules and triggers for agent behaviors can be created and modified by the advanced user. AEDGE™ also provides APIs for custom extensions of agents, data bridges, and the entity framework. The sophisticated user employs AEDGE™ as a

flexible development and testing environment for DSS components. The practical user will enjoy AEDGE™'s versatile data connectivity and its near-real-time execution

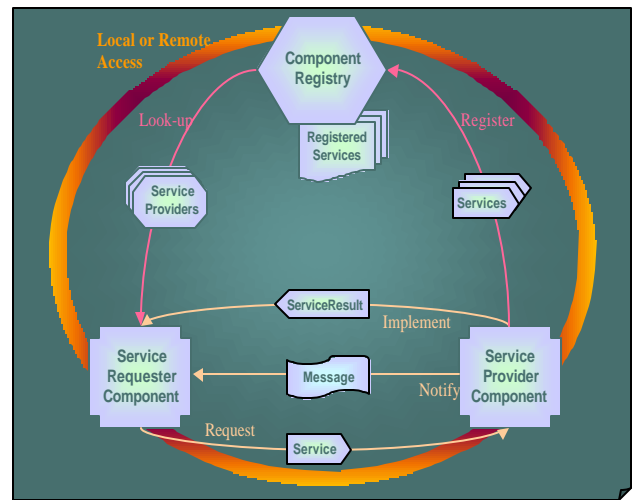


Figure 5. Message passing among Web Interactive Agent Components over AEDGE™ services

and monitoring of DSS functions. As a built-in bonus, AEDGE™ provides connections to a number of simulators and data formats, including HLA, DIS, DTED, DBDB2, XML, as well as support for multiple modes of distribution (CORBA, RMI, TCP/IP).

IV. A HCAN Design using AEDGE™

Using the AEDGE™, a HCAN-based data fusion and DSS can be designed with convenience by end-users with only limited amount of programming experiences. The following facilities of AEDGE™ are readily applied in developing the HCAN for data fusion and decision support applications.

1. Master Server. The Master Server is a Web component of AEDGE™ that facilitates connections and interactions among the rest of the AEDGE™ components. In a HCAN design, this component tracks the agent community and matches service providers with service requesters in sensor/data accesses. It also provides component registration and tracking services, interface matching services and component search, identification and connection services. The Master Server is also responsible for synchronizing simulated time (and real time) among multiple simulation servers and live links over communication networks. The interaction between distributed service providers and service requesters is accomplished through the use of Java's Remote Method Invocation (RMI) API. The communication mechanism is extended to support other distributed computing models (CORBA, TCP/IP sockets, XML-RPC) via a number of standard bridges and protocols.

2. Entity Representation Framework. The Entity Representation Framework is an integral part of AEDGE™, which provides the basic entities and events for live or simulated sensor/data sources. The generic object-oriented hierarchy of entities represents a wide range of pre-defined models, such as geographic data, vehicles, financial and other entities that appear in specific HCAN applications. More importantly, the framework includes a number of interfaces, which allow users to add new entities with new behaviors or with combinations of existing behaviors.

3. Agent Infrastructure and Components. Agent Components host one or more Intelligent Agents, which monitor the simulated/real world, react to changes in it, and interact among each other and with human users according to their specific agent behaviors. The Agent Infrastructure provides the basic inter-agent communication and synchronization mechanisms, as well as the interfaces for agents to use other data sources, such as simulation servers, live data links, databases, etc. The agent component receives information from the AEDGE™ core on the state of the world and sends back agent requests and feedback. Agent components are usually distributive according to their function and can connect and exchange data, recommendations and rationale via well defined protocols. A base hierarchy of agents is also provided, and it can be extended and customized for a particular user's need. This feature is especially beneficial to the HCAN design.

4. Database Connectivity. The HCAN makes use of the AEDGE™'s capability of storing and retrieving data to/from various databases. The Database Connectivity components provide generic and specific bridges to a number of proprietary and public databases. These are a natural extension of the AEDGE™ core Database Connectivity. Bridges to characteristics and performance data, such as weapons performance and effectiveness, and terrain databases are provided. Interfaces for new database bridges are also provided. This facility provides a basis for the learning and adaptation module of the HCAN system to function on updating and improvement of the decision supporting strategies. New Database Connectivity modules can be added by extending the provided database bridges and implementing the connectivity interfaces.

5. User Interfaces (UI). Visualization and interface components in the HCAN are responsible for interactions with the human users. They present data in a HCAN in a clear and intuitive manner, allowing for simple, yet powerful presentations of complex interdependencies in the simulated/real world. Visualization clients can interact with all components of the AEDGE-based DSS through bi-directional information flows. They receive information about simulated and live entities, their environment and interactions, as well as on agent evaluations and recommendations. The users' interactions with the UI

components provide feedback, which is then sent back to the HCAN core components. The user can choose to accept the agent recommendation and that interaction is fed back to other HCAN components to ensure that the accepted recommendation is now taking effect for the entities it concerns.

6. The Client Builder. The AEDGE™'s Client Builder provides an intuitive user interface to the operational picture. The Client user interface changes from application to application, depending on the user's situational awareness requirements. The Client is implemented on top of AEDGE™ and may consist of some of the sample components listed below.

- (1) Entity Listener. An entity listener connects to Simulation Servers and tracks entity updates. It also provides the Interactive Map with information to render the entities in HCAN. The Listener is based on AEDGE™ interfaces that enable the tracking and integration of entity data from multiple simulators and live links.
- (2) Agent Client. The Agent Client implements the behaviors of multiple intelligent agents in the HCAN. It is based on the AEDGE™ Agent Client and extends the existing agents with new, application-specific behaviors.
- (3) Interactive Map. A Map extends the Visualization Client interfaces to enable the rendering and interaction with simulation (as well as live) entities. The standard interfaces are extended to provide application-specific functionality, such as confidence intervals (error baskets), sensor properties, and multiple map views at different levels.

Agents in HCAN designed by using AEDGE™ are specialized components that generate recommendations either in response to a user inquiry or spontaneously, according to their function. As discussed in section II, agents in HCAN are organized in agent communities, unified under an Agent Manager component, which is responsible for invoking and synchronizing individual agents. The Agent Manager interacts with agents via the SPSR protocol, while users (through UIs) interact with the Agent Manager through more user-friendly Inquiry/Recommendation exchange protocol (IREP). The users can query the agent manager by sending context information (entities, geo-references, target information, etc.) and specific requests for recommendations. The query is internally translated to service requests and sent to the Agent Manager. The users are not limited to the IREP – they can use any query representation, such as SQL queries, as long as they can be internally converted to service requests. Upon receiving a user-level query, the Agent Manager selects and invokes the appropriate agents to perform the desired tasks. The Agent Manager has a

table of registered Agents and their capabilities. Thus, the Agent Manager is the one that partitions the problem, sends sub-tasks to the individual Agents, and later combines and de-conflicts the solutions. After an overall solution is reached, the Agent Manager forms a set of recommendations, which are returned to the User via a Service-Result object. In essence the IREP is a user-

friendly protocol build on top of the SPSR protocol. The interactions among agents and the Agent Manager are solely based on the SPSR protocol, as these are optimized for efficiency and not necessarily for user-friendliness. Figure 6 demonstrates four different modes of User/Agent-Manager/Agent interactions. The functionalities of these interactions are described below.

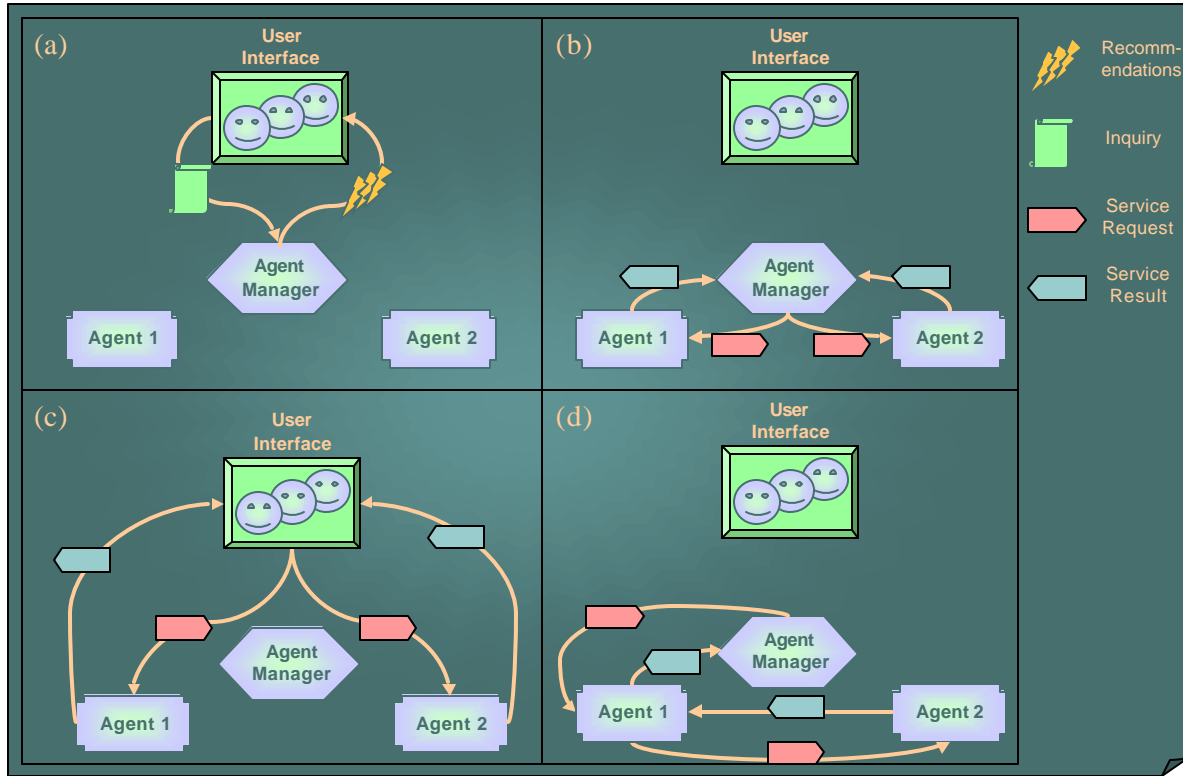


Figure 6. Different modes of agent interactions:
 (a) User to Agent Manager;
 (b) Agent Manager to individual agents;
 (c) User bypasses Agent Manager; and
 (d) Agent-Direct interaction.

The AEDGE™’s design of User/Agent-Manager/Agent interactions for HCAN includes the following.

a) User to Agent Manager Interactions. Essentially the user sends an inquiry to the Agent Manager, based on the user’s current needs and query representation language. The inquiry may consist of a task description (e.g. “Check Fuel 1022” or “Check Range 1022 from 1021”), and optionally a context update, such as platforms, targets, geo-references etc. The inquiry is internally serialized and translated into service requests, which are then sent to the Agent Manager via the SPSR protocol. After the Agent Manager performs the requested

tasks, it sends a reply in the form of a set of recommendations. Recommendations are core objects in AEDGE™’s framework that represent desired actions and commands. Recommendations may be produced by both Agent Managers and users and are interpreted by Entities to form tasks and orders. In this case Recommendations are generated by the Agent Manager and sent for approval to the User.

b) Agent Manager to Individual Agents. In this interaction the Agent Manager partitions the task to subtasks for the individual agents, registered under the Manager. Subtasks are then sent to the agents

via the SPSR protocol, encapsulated in Service objects. After the individual agents arrive at a solution they respond to the Agent Manager with Service-Result objects, which are interpreted by the Manager. The Agent manager performs synchronization and de-confliction of the individual agents' results to ensure that the user will receive a coherent set of recommendations (in case individual agents had provided conflicting information).

- c) User bypasses Agent Manager. The user can interface directly with the individual agents, using the SPSR protocol. If the user process can locate the Service Provider of an agent (via a Component Manager where that agent is registered), the user can send service requests directly to the agent and listen for the Service-Result object in the reply. This places the burden of locating and interfacing with the agent's service provider on the user, but it provides more flexibility and faster response.
- d) Agent-Direct interaction. Agents can communicate with each other indirectly (through the Agent Manager) or directly, via the SPSR protocol. The Requester agent looks up other agents' service providers from any component manager (including the Agent Manager) and can then send service requests to other individual agents. The Provider Agent handles the service request just like it would handle a request from the Agent Manager. The Requester agent needs to be able to handle the Service-Result returned by the Provider. Agent-direct interaction provides the flexibility of extending the agent community that belongs to an Agent Manager without having to modify the login of the Manager itself.

V. An Example of Application

As an example of the HCAN design using AEDGE™ for data fusion and DSS applications, we briefly describe our AWACS-AEDGE design (Hicks, Stoyen and Zhu 2001). The Airborne Warning and Control System (AWACS) Weapons Director (WD) team serves as a vital airborne

Command and Control (C2) node in the US Air Force, providing airborne surveillance and command, control, and communications functions for tactical and air defense forces. The AWACS-AEDGE was developed as an agent-based C2 team decision support platform for research and training. The platform is based on AEDGE™'s implementation of HCAN, a federation of collective intelligent agent-based functions that enable scenario construction and emulation of friendly and hostile entities. The behaviors and decision making capabilities of all hostile and friendly entities are guided by the distributed intelligent agents of HCAN. If a human decides to "log in" in a particular role, he/she may choose to view recommendations generated by the agent for that role. Even if the human operator chooses not to view recommendations, the agent recommendations are still logged by the computer. This enables human decision making to be directly compared to agent-based decision making. This capability will facilitate skill acquisition and decision making for events that are both typical and time consuming.

The intelligent-agent-based data fusion and decision support capabilities of the HCAN enable more detailed and innovative approaches to measurement and modeling of communication and decision making for AWACS operation. For example, the number and type of recommendations generated by the AWACS-AEDGE agents at any given time constitutes a new way of conceptualizing and representing cognitive workload. In addition, the AWACS-AEDGE platform can operate through speech – operators can speak to the system using predefined jargon, request tasks be performed or information provided/transferred, and the agents will respond verbally to the speech-driven requests, using voice generation technology. All agent communications to each other, as well as to humans, are logged in speech format and are available online. Figure 7 displays a screen shot of the AWACS-AEDGE decision support system in a typical military conflict resolving situation.

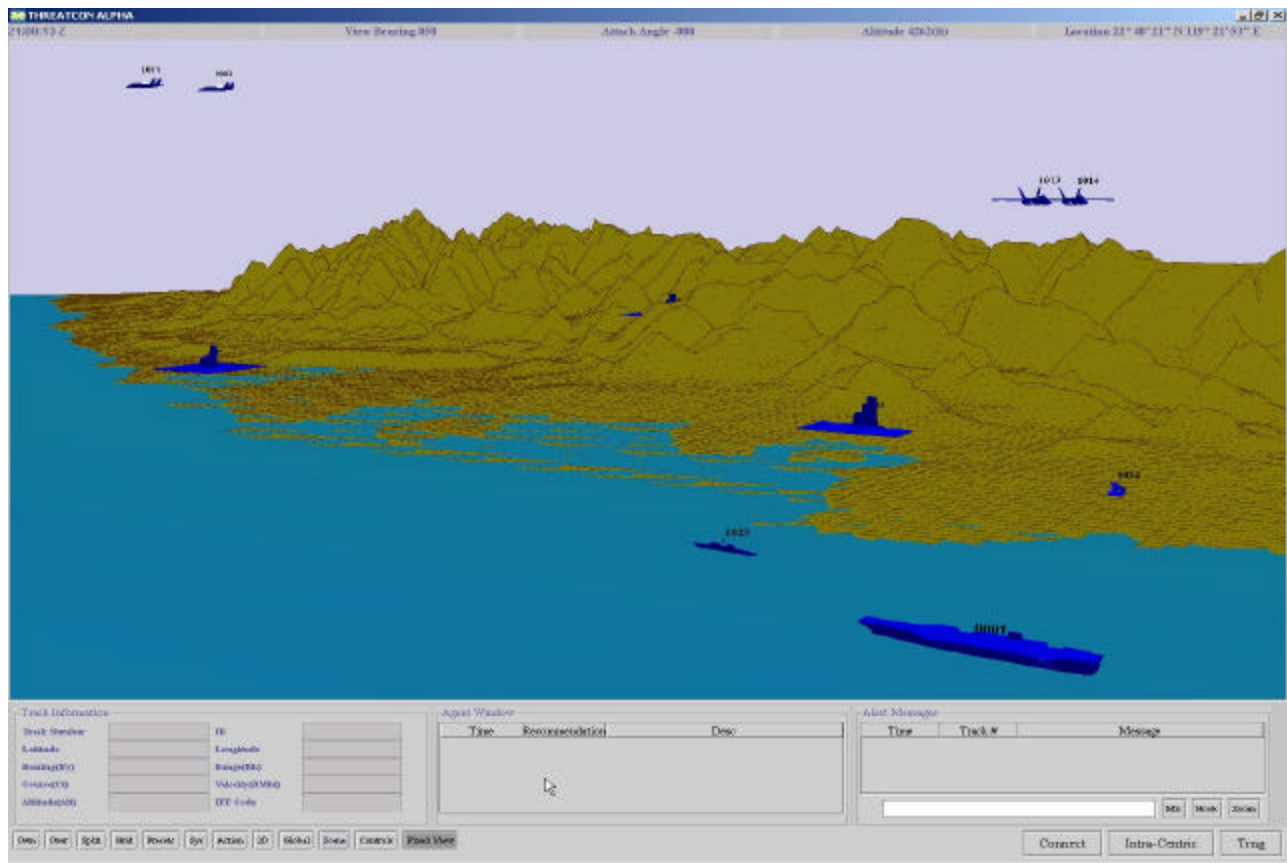


Figure 7. A screen shot of the AWACS-AEDGE data fusion and decision support system interface

The AWACS-AEDGE was conceived through cognitive and functional analysis of team member roles, responsibilities, and decision making process with the help of subject matter experts (SMEs), to optimize applicability of results to operational settings. Systematic descriptions of AWACS roles, responsibilities, requirements, interdependencies, tactics, strategies, and task demands were collected from SMEs, cognitive task analyses and focal-group interviews. These data were examined to identify decision events, which were generic to performance, regardless of mission scenario, and likely to bottleneck under high tempo situations. Based on this analysis agent behaviors, interactions and optimization processes were defined to match the operational tasks and environment.

VI. Conclusion

As real-time decision support becomes a feasible objective and toward reality, basic researches are still needed in a number of areas to build such systems by incorporating a series of intelligence-demanded tools and architectures.

The kind of robust, integrated information fusion techniques for handling increasing diversity of input sources is especially important in contemporary military missions. The HCAN provide the capability for mission planners to prioritize and configure the operation field pictures easily and quickly through the effective data analysis ability for the software-agent assisted human and information system interactions. As a well crafted information management and knowledge acquisition system, the HCAN can assist planners in many different ways, particularly with respect to quickly identifying responses and counter-responses to action or inaction, gaining a comprehensive insight to the action-counteraction dynamics, and determining the best allocation of tactical resources that will increase the assurance factors of the mission success. Via the use of probabilistic reasoning and goal-driven control functionalities, the HCAN will also provide ability for the information system to continue operation in the presence of hostile attacks and allow for anytime-anywhere accessibility even when a portion of the information resources is disabled. The 21st Century Systems, Inc.'s AEDGE™ provides a common framework, information

exchange mechanisms, and standard component libraries of agent algorithms. With the use of the AEDGE™ facilities, customer HCANs can be quickly built at the end-user site to provide efficient and effective data fusion and decision supports timely and reliably.

References

- Arai, T.; Sycara, K. and Payne, T. 2000. Experience-based reinforcement learning to acquire effective behavior in a multi-agent domain. *Proceedings of the 6th Pacific Rim International Conference on Artificial Intelligence*, pp. 125-135.
- Buller, A. 1992. Fuzzy inferencing as a competition between contradictory statements. *Proceedings 1st Singapore International Conference on Intelligent Systems*, pp. 103-106.
- Bogler, P. 1987. Shafer-Dempster Reasoning with Applications to Multisensory Target Identification Systems. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol.17, No.6, pp. 968-977.
- Chen, M.; Zhu, Q.; and Chen, Z. 2001. An Integrated Interactive Environment for Knowledge Discovery from Heterogeneous Data Resources. *Journal of Information & Software Technology*, Vol. 43, 487-496.
- desJarins, M. 1993. Representing and reasoning with probabilistic knowledge: a Bayesian approach. *Uncertainty in Artificial Intelligence*, pp. 227-234.
- Hicks, J.; Stoyen, A.; Zhu, Q. 2001. Intelligent Agent-based Software Architecture for Combat Performance under Overwhelming Information Inflow and Uncertainty. *Seventh IEEE International Conference on Engineering of Complex Computer Systems*, 200-210.
- Giampapa, J.; Paoluc, M.; Sycara, K. 2000. Agent Interoperation Across Multi Agent System Boundaries. *Proceedings of Agents 2000*, Barcelona, Spain.
- Giri, S.; and Zhu, Q. 1998. dbAgent: An Intelligent Web Agent for Database Mining. *International Conference on Computer and Informatics (CS&I'98)*, 466-470.
- Gosling, J.; McGilton, H. 1995. *The Java Language Environment - A White Paper*. Sun Microsystems, Inc.
- Lee, M.; Offutt, A.; and Alexander, R. 2000. Algorithmic Analysis of the Impacts of Changes to Object-oriented Software. *Proceedings of the Thirty Fourth International Conference on Technology of Object-Oriented Languages and Systems (TOOLS USA '00)*, 61-70, Santa Barbara CA.
- Lejter, M.; Dean, T. 1996. A Framework for the Development of Multiagent Architectures. *IEEE Expert*. 47-59.
- Maes, P. 1994. Agents that Reduce Work and Information Overload. *Communications of the ACM*. Vol 37, No 7, 30-40.
- Nath, R.; Jackson, W.; and Jones, T. 1992. A comparison of the Classical and the Linear Programming Approaches to the Classification Problem in Discriminant Analysis, *Journal of Statistical Computation and Simulation*, 41 (1), 73-93.
- Petrov, P.; Stoyen, A. 2000. An Intelligent-Agent Based Decision Support System for a Complex Command and Control Application. *Proceedings of the Sixth IEEE International Conference on Engineering of Complex Computer Systems*, Tokyo, Japan.
- Rodriguez, B. and Poehlman, W. 1997. A system for distributed inferencing. *Proceedings 1996 Rochester Forth Conference Open Systems*, p. 118.
- Steinberg, A.; Bowman, C.; and White, F. 1999. Revisions to the JDL Data Fusion Model. In *Proceedings of the SPIE Sensor Fusion: Architectures, Algorithms, and Applications III*, 430-441.
- Stoyen, A.; Laplante, A.; Harrison, R.; and Marlowe, T. 1994. Engineering of Complex Systems: A Case for Dual Use and Defense Technology Conversion. *IEEE Spectrum*, Vol. 31, No. 11, 32-39.
- Sycara, K. and Zeng, D. 1996. Multi-Agent Integration of Information Gathering and Decision Support. *Proceedings of the 12th European Conference on Artificial Intelligence*, 1996.
- Vincke, P. 1992. *Multicriteria Decision Aid*, John Wiley, New York.