Customizable Speech Centers for Automated Entities within Simulation

Jason R. Potts, Todd Griffith, Ph.D., Kyle Roth, Jared Snyder
Discovery Machine Inc.
Williamsport, PA
jpotts@discoverymachine.com,
tgriffith@discoverymachine.com,
kroth@discoverymachine.com,
jsnyder@discoverymachine.com

ABSTRACT

As modeling and simulation continue to expand their roles in training environments, the need for intelligent, automated entities expands. By incorporating increasingly intelligent constructive entities, training exercises can increase in fidelity and complexity, without increasing the manpower costs associated with human operators. However, in order to fulfill the same roles that human operators play in training exercises, autonomous entities need to be able to interact with other entities (both autonomous and human-controlled) in a realistic and robust manner.

A critical aspect of this interaction involves entities being able to communicate with humans (and each other) in a way that closely parallels the types of communication which take place amongst humans playing the same roles. In this paper, we present work conducted under the Office of Naval Research which enables robust communication between autonomous entities and human users through natural means such as verbal conversation. Additionally, we present a framework which enables a subject matter expert (SME) to define and expand the communications employed by autonomous entities, without requiring the user to be aware of software engineering, programming, or artificial intelligence concepts underlying the implementation.

This framework, called the Customizable Speech Center (CSC), consists of a graphical, modular architecture for defining concepts about which entities can communicate. The CSC allows human users to easily expand the communication capabilities of autonomous entities, as well as integrate these capabilities into the larger behavior models used during training exercises. This incorporation of robust communication into the behavior models enables extremely dynamic constructive entities capable of adapting to situations in a more realistic and human-like manner. Not only does this present the opportunity to reduce instructor workload by allowing them to give voice commands to autonomous entities, but this also enables the training of users in the often complex aspects of team coordination and communication.

ABOUT THE AUTHORS

Jason R. Potts is the Vice President of Software Development for Discovery Machine Inc., overseeing the software development of their core technology offerings, as well as the application of their technology to customer problems across a variety of domains. Since graduating with honors from WPI with a Bachelor's of Science degree in Computer Science, Mr. Potts has been designing and developing software for Discovery Machine Inc. since 2001. This work has focused on development of tools facilitating knowledge capture, and representation of human expertise in a machine executable form. In addition to behavior modeling of simulated entities within constructive military training environments, these applications also include representation and automation of business process intelligence within commercial sectors such as manufacturing or pharmaceuticals.

Dr. Todd W. Griffith is the founder and CTO of Discovery Machine. He is responsible for acquiring funding and managing the research and development of Discovery Machine's core knowledge acquisition products, the Discovery Machine Knowledge Service Modeler™ and Knowledge Service Engine™. He has been working in the area of intelligent systems research for over twenty years and has published papers in the areas of cognitive science, human-computer interaction, and intelligent systems. He has presented regularly at conferences and meetings since 1996 and has received four patents in the area of knowledge systems. Prior to founding Discovery Machine, Dr. Griffith taught computer science and artificial intelligence at Georgia Institute of Technology and Bucknell

University. Dr. Griffith received his Ph.D. in computer science and artificial intelligence in 1999 from Georgia Tech with a focus on modeling the problem solving strategies of Subject Matter Experts (SMEs). A lack of available tools for knowledge acquisition, led to the founding of Discovery Machine and the commitment to build software that enables SMEs to encode their *own* problem solving strategies on the computer. In support of this effort, Dr. Griffith has been the principal investigator in research projects funded by DARPA, ONR, NWDC, NWSCDD, OSD, Army ERDC, NAVAIR, NASA, and NSF.

Kyle R. Roth is a Software Engineer employed by Discovery Machine, Inc. He graduated from Carthage College in 2009 earning his Bachelor degree in Computer Science (Honors) and Mathematics. While working at Discovery Machine, Kyle has created behavior models for entities across multiple domains and simulation environments, has created extensions of Discovery Machine's core technology, and has helped in integrating Discovery Machine's tools with different simulation environments.

Jared K. Snyder is a Software Engineer for Discovery Machine Inc., whose work focuses on behavior modeling and developing core technology offerings. After graduating with honors from Bucknell University with a Bachelor's of Science degree in Computer Science, Mr. Snyder has been designing and developing software for Discovery Machine Inc. since 2009. His work has focused on implementing graphical user interface tools in Discovery Machine Inc.'s core technology offerings and creating behavior models for simulated entities within constructive military training environments.

Customizable Speech Centers for Automated Entities within Simulation

Jason R. Potts, Todd W. Griffith, Ph.D., Kyle Roth, Jared Snyder
Discovery Machine Inc.
Williamsport, PA
jpotts@discoverymachine.com,
tgriffith@discoverymachine.com,
kroth@discoverymachine.com,
jsnyder@discoverymachine.com

INTRODUCTION

As virtual simulation environments begin to play a more central role in the training of warfighters, the need for automated entities expands. Autonomous entities enable large-scale exercises with limited resources, but create new issues which must be addressed. Specifically, while manpower costs can be reduced through automating the activities of simulated entities, new costs associated with the construction of those entities arise. The knowledge of subject matter experts (SMEs) in the training domain must be captured and encoded within the behaviors of the autonomous entities, such that they behave in a manner representative of the human actors they are replacing and assisting.

Traditional software engineering methods can prove costly, as behaviors captured from SMEs are often encoded in a syntactically complex programming language by software engineers. These engineers lack the direct experience of the SMEs required to fully understand the behaviors being represented, and the SMEs lack the programming expertise required to fully understand the encoded behaviors, thus creating a gap which must be bridged between the two groups in order to specify, debug, and maintain those autonomous behaviors. We have presented a methodology for addressing this disconnect between SMEs and software engineers, through use of a visual task decomposition language called Task-Method Knowledge (TMK) which enables subject matter experts to directly specify behaviors for autonomous entities within a variety of simulation systems, in a form which is human-readable and understandable by users who lack traditional computer programming skills (Potts, et al., 2010).

As simulations begin to address more nuanced aspects of training, interpersonal communication takes on an increased level of importance. Training scenarios often focus not only on physical operations such as movement across an environment, and using various pieces of equipment, but also in complex communication and coordination between entities. This paper presents an approach to incorporating extensive communication capabilities into TMK Behavior models, achieving the same accessibility to non-programmers while dramatically enhancing the realism and utility of autonomous entities within simulation by enabling them to take on larger roles within training events. This approach enables autonomous entities to engage in communications over a variety of mediums (radio, face-to-face, etc.) with each other, as well as with human users. In situations that traditionally employ voice communications, the solution presented here enables the easy integration of a variety of speech-recognition and synthesis solutions, allowing the automated entities to respond to verbal queues from human operators, as well as verbally convey information to those operators.

We present the results of work conducted under projects funded by ONR and NAVAIR-TSD, integrating complex communications into the behaviors for a wide variety of entities in the Joint Semi-Autonomous Forces (JSAF) simulation, as well as characters within Bohemia Interactive's Virtual Battlespace 2 (VBS2).

JSAF Behavior Modeling Example

Under projects funded by ONR and NAVAIR-TSD, a large set of behaviors were developed for various types of entities within the JSAF simulation environment. We developed these behaviors by constructing a library of Basic Level Actions (BLAs) which then enabled SMEs to compose complex mission behaviors for autonomous entities by combining these actions. For the JSAF simulation, this library consists of:

- 57 Subsurface BLAs
- 33 Rotary Wing Aircraft BLAs
- 74 Surface Vessel BLAs
- 5 Fixed Wing Aircraft BLAs

Each of these Basic Level Actions constitutes a modular building block which can be combined with other BLAs from the same domain to create novel missions which can then be tasked to individual entities.

As an example, the following BLAs were created for the P8 program (PMA290) through our contract with NAVAIR:

Table 1. JSAF Surface Ship BLAs

Activate CIWS	Land Avoidance	React to Vessel Within 500yds
All Stop	Launch Airplane	Search for Friendly Sub
Assignment Template	Launch Helicopter	Search for Friendly Surface Ship
Clear Baffles	Move on Aircraft Launching Route	Search for Hostile Sub
Collision Avoidance	Move Ship on Continuous Route	Search for Hostile Surface Ship
Deactivate CIWS	Move Ship on Heading	Search for Neutral Surface Ship
Ekelund Range Maneuver	Move Ship on Route	Task Entity
Fire Guns at Location	Move Ship to Location	Turn off all active sonar
Fire Guns at Target	Move to Aircraft Ready Position	Turn off all passive sonar
Fire Missile at Location	Move to User Selected Target	Turn off all Radar
Fire Missile at Target	Patrol on Bearing	Turn off ESM
Fire Torpedo on Bearing	Patrol Screen Box	Turn off Towed Array
Fire Torpedo on Target	React to Helicopter Recovery	Turn on All Active Sonar
Fish	React to Incoming Aircraft	Turn on all passive sonar
Follow Detected Target	React to Incoming Missile	Turn on all Radar
Follow Group Leader	React to Possible Sub	Turn on ESM
Follow User Specified Target	React to Subsurface Detection	Turn on Towed Array
Initialize Communication	React to Torpedo by Evading	Wait User Specified Milliseconds

Advantages

In general, the TMK based approach to behavior modeling offers the following advantages (Potts, et al., 2010):

- Rapid development of new behaviors for customized training scenarios
- Expert domain knowledge captured within the modeling tools to assist building more intelligent behaviors
- Transparency of behaviors for subject matter experts during and after development
- Transparency of behaviors at runtime, easing the debugging process and exposing decision making process of entities to operators

By incorporating a framework for supporting robust communication between automated entities and humans, we extend the applicability of these behavior models to include participation in complex training events centered on large-scale coordination amongst teams of entities. Through inclusion into the BLA based framework, we preserve the advantages inherent in our TMK approach, allowing SMEs to create and understand new behaviors easily, without programming knowledge.

BACKGROUND

Behavior Modeling Approach

Throughout the research effort described in this paper, we have employed a methodology and toolset designed specifically for capturing knowledge from subject matter experts (SMEs). The Discovery Machine® Methodology is a patented approach to the acquisition of knowledge from SMEs. The methodology works with the SME through five phases: expert domain scoping, conceptualize, formalize, operationalize, and test/deploy Each of these phases leverages the Discovery Machine Knowledge Service Modeler or Custom Console products to encourage and facilitate introspection and articulation by the expert. The visual nature of the modeling language enables a knowledge coordinator to probe more deeply into how and why decisions are made.

Basic Level Actions

As mentioned above, Basic Level Actions (BLAs) serve as building blocks for missions and reactions used within behaviors. The term “Basic Level” comes from Elenor Rosch (1976). In her research she found that humans tend to speak of concepts at a certain ontological level. A picture would be identified as a “dog” rather than a “Labrador retriever” or a “pet” or “mammal” or “animal”. This “basic level” was also shown to apply to actions. In telling a story one would relate actions such as: “Got up, brushed teeth, and got dressed” rather than “got up, walked 15 paces to the bathroom, unscrewed the cap on the toothpaste, etc.” Under projects funded by ONR and NAVAIR-TSD, a large set of behaviors were developed for various types of entities within the JSAF simulation environment. We developed these behaviors by constructing a library of Basic Level Actions (BLAs) which then enabled instructors to compose complex mission behaviors for autonomous entities by combining these actions. For the JSAF simulation, this library consists of BLAs in several categories: Sense, Move, Follow, Deploy, Attack, Search, Wait and Task.

CUSTOMIZABLE SPEECH CENTER

In this paper, we present an extension to the TMK and BLA based modeling paradigm, incorporating what we are calling a "Customizable Speech Center", or CSC. The CSC provides a mechanism by which the SME is able to customize the communication capabilities, transparently, while building up a new behavior for autonomous entities. Thus, in the same way that BLAs enable modularity and flexibility in missions that non-programmers can construct for simulations, the addition of the CSC framework injects modularity into the communications that autonomous entities can engage in during simulated training exercises.

Extension of Basic Level Actions

The CSC framework depends upon, and extends, the notion of Basic Level Actions. We introduce the notion of a "Communication Handler", represented as a data object attached to each BLA. During development of each individual BLA, a corresponding communication handler is also created, and performs the function of defining how the autonomous entity will communicate with other entities (both human and autonomous) while in the process of performing that BLA.

For instance, while performing a BLA designed to move an entity to a particular location, the BLA's communication handler may contain information about the entity's current destination, and be able to answer questions about it. While performing a different BLA, such as firing on a ground target, a different communications handler would be in effect, no longer containing information about a destination (which would have no meaning in its current context), but instead containing information about its current target, or the weapons it was deploying, etc.

By tying the entity's communication capabilities to the BLAs which are driving its manifested behavior, we are able to contextually separate various kinds of communications. Instead of forcing the behavior modeler to consider all possible communications which could take place within any possible scenario, they can focus on communications which are contextually specific to a particular type of action.

CSC Modeling Structure

The communication handler described above consists of two primary components: a TMK hierarchy representing the process required to handle incoming communications (see Figure 1), and a set of data representing information important to the types of communication which are relevant to the context in which the entity is performing the BLA (see Table 1). These components combine to form the process knowledge and data structure of a communications handler object.

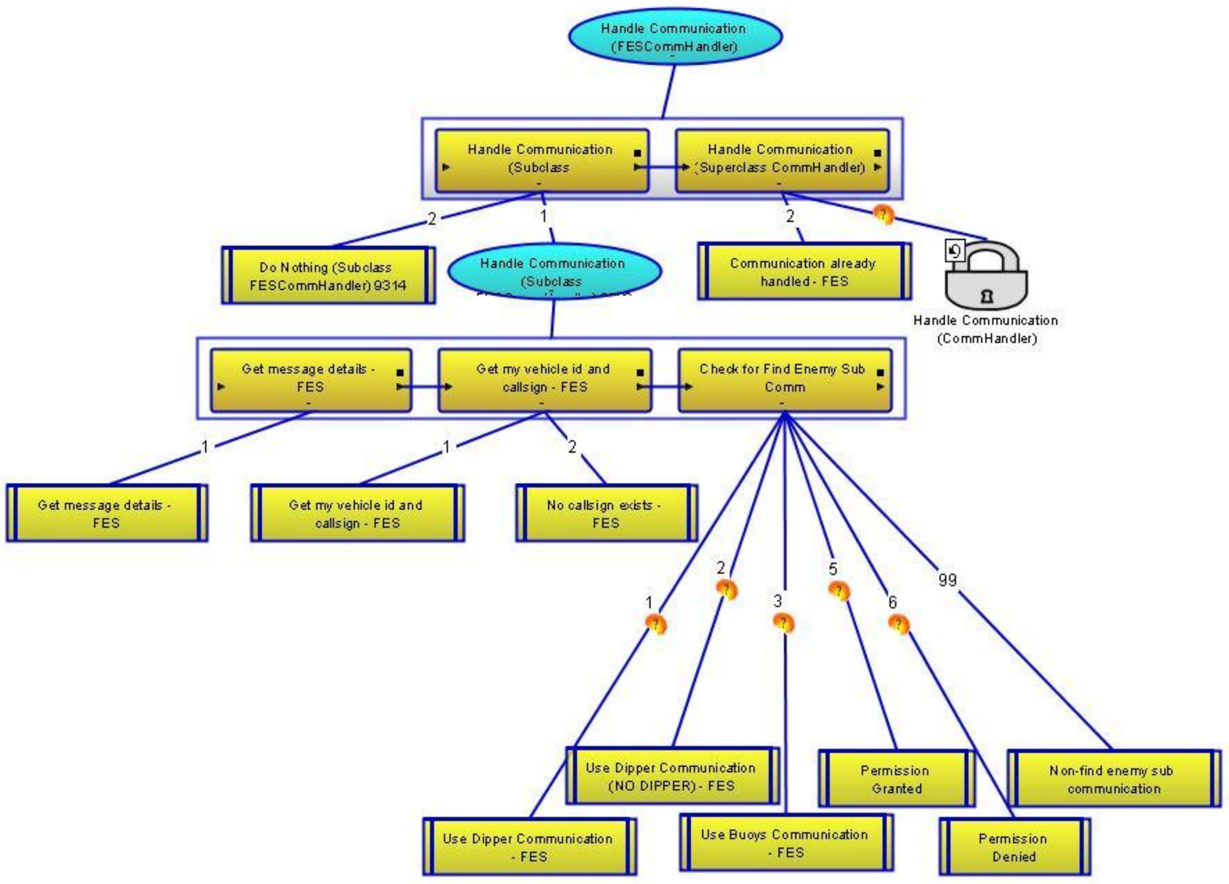
Handle Communication (FESCommHandler)
Handle Communication (Subclass
Handle Communication (Superclass CommHandler)
Do Nothing (Subclass FESCommHandler) 9314
Handle Communication (Subclass
Communication already handled - FES
Handle Communication (CommHandler)
Get message details - FES
Get my vehicle id and callsign - FES
Check for Find Enemy Sub Comm
Get message details - FES
Get my vehicle id and callsign - FES
No callsign exists - FES
Use Dipper Communication (NO DIPPER) - FES
Permission Granted
Non-find enemy sub communication
Use Dipper Communication - FES
Use Buoys Communication - FES
Permission Denied

Figure 1. Example TMK Structure of a Communication Handler for a Rotary Wing Aircraft

Figure 1. Example TMK Structure of a Communication Handler for a Rotary Wing Aircraft illustrates an example communications handler for a Rotary Wing Aircraft tasked with searching for an enemy subsurface vessel. This model is dominated by a broad branching structure, where each branch represents a response to a particular incoming communication. Additionally, the branch indicated by the padlock represents the TMK model contained within the superclass of this communications handler. This provides a visual representation of object-oriented polymorphic properties of the communication handler classes. During execution, if an incoming radio message is not handled by this specific communications handler, it will be passed to the hierarchy of the parent class, which will then attempt to handle it. This allows communications which are broadly applicable to a large set of tasks to be included in more abstract instantiations of the communications handler, while more specific instantiations can add in responses to narrowly applicable communications. These specific instantiations can override the broadly applicable responses for their specific situation, while enhancing understandability through encapsulating communications handling into separate classes.

Table 2. Example of Data Stored by a Communication Handler

Aircraft Has Dipper	Boolean
Permission to Deploy Buoys	Boolean
Time that Current Task Began	Timestamp
Has Handled Most Recent Communication	String
Radio Channels Entity is Listening To	Collection of Radio Channels
Name of Current Task	String
Most Recent Radio Message	Radio Message

Table 2 shows a set of data tied to the communications handler represented in Figure 1. This data is contained within the communication handler's class, and represents specific information about the currently assigned task which may be needed for effective communication. This data may contain information contained in an entity's mental model of the world, produced by its situational awareness processor, information relevant to the specific mission being performed, or information unique to this particular communications handler.

In the example, we see a particular piece of data that is unique to this specific communications handler, whether or not the aircraft executing this behavior has access to a dipping sonar. When the task using this communications handler begins, it fills in a value for this by querying the simulation about the entity. Then, during execution of this task, another entity (or a human user) can request that the RWA deploy its dipping sonar. In cases where such a piece of equipment is available, the branch of its communication handler's TMK hierarchy will call on the simulation to deploy it. In cases where the entity lacks the necessary equipment, the communications handler will respond appropriately over the radio that it cannot perform the requested action.

This framework enables the autonomous entities to engage in a wide variety of communications with human users as well as other autonomous entities, while reducing the set of decisions required to generate appropriate responses to only those directly applicable to tasks that the entity is currently engaged in. This has the benefit of improving understandability of the autonomous behaviors, while reducing complexity and thus reducing errors and maintenance costs.

Handling Unexpected Communications

The modeling framework described above provides a mechanism for handling well defined communications quickly and efficiently, using domain specific actions and terminology which can be described through knowledge capture with subject matter experts. However, communications often take the form of questions which humans generally respond to based on "common sense" rather than based on some specific training they received, or information provided through a mission briefing. For example, a human operator may want to ask a helicopter relay the number of remaining sonobuoys on board. Asking a human this would be easy enough, even if no specific protocol was agreed to ahead of time. Even if such an inquiry was not specifically defined in the pilot's mission briefing, he would still understand the question and be able to provide an answer on the fly.

While such an answer could be defined within the communications handlers for an autonomous entity, if we extend the idea to try and cover all possible things that an entity could be asked about, the communication handlers would quickly become extremely large and unwieldy, reducing both their operational efficiency, as well as their readability. Yet, while we do not want to bloat the communication handlers with responses to a huge set of questions answered by common sense, we also do not want the autonomous entities to completely fail to respond to questions that any human could answer easily.

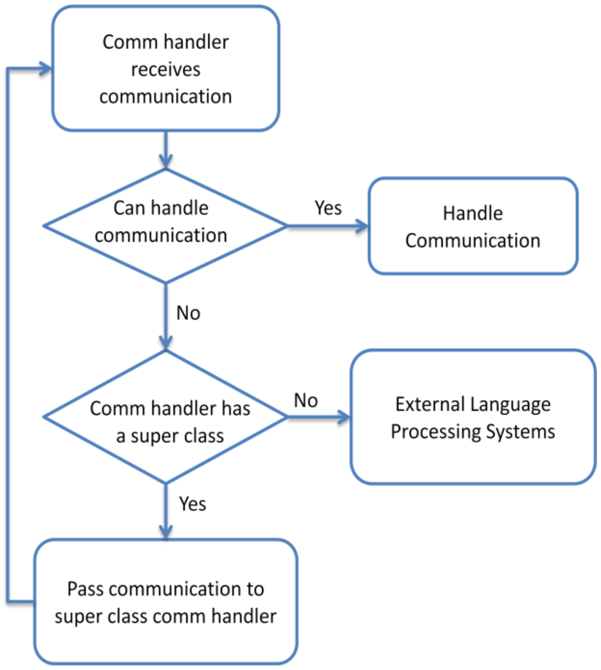
Comm handler receives communication
Can handle communication
Yes
Handle Communication
No
Comm handler has a super class
No
External Language Processing Systems
Yes
Pass communication to super class comm handler

Figure 2 - Handling Unexpected Communications

In order to address these two competing requirements, we have incorporated a multi-strategy approach to our handling of communications, illustrated in Figure 2.

When an incoming message is received by the autonomous entity, it first passes it through the above described communication handler that is associated with the currently executing BLA. If that particular communication handler is not prepared to handle the message, and a super-class exists for that handler, then the message is passed up the class hierarchy for processing. If the system fails to comprehend the message and no super-class exists for the communications handler, the system attempts to leverage a variety of external language processing systems to determine the semantic meaning of the message, as well as generate an appropriate response through action or further communication.

As a concrete example of this, we can imagine an entity performing a BLA called "Advance on To Location Using Cover" which causes the entity to move to a location by way of determining a path that leaves it least exposed to hostile fire. In this basic level action's communication handler, we may choose to respond to specific messages about this type of movement, perhaps, "Relay the nearest location of cover." In addition to messages like this which are specific to this particular BLA, this action can be classified as falling into the "Movement" category. As a result, its communications handler may extend a more general "Movement Communications Handler", which could respond to more broadly applicable questions like "Where are you currently moving to?" Finally, the Movement Communications Handler itself may extend the base Communications Handler class, which contains responses for core messages that all entities can handle, regardless of what particular BLA they are performing. Examples of these universally applicable messages might be, "What is your callsign?" or "What action are you currently performing?"

When a new message is received by the entity, it is handled by the most specific communications handler available, bubbling up the class hierarchy until recognized. This provides communications handler extensions to override handling of specific messages, if they need to alter the behavior or response generated by the super-class handler.

Finally, when no super-classes exist for the current communications handler, and a message is still left unhandled, the system can pass it out to external processing systems, including more complex natural language processing systems designed to extract semantic meaning from the raw message.

Systems Architecture

As part of ongoing efforts with ONR and NAVAIR, we have implemented a system designed to facilitate communication with autonomous entities within the JSAF simulation environment. The architecture used in this implementation example can be seen in Figure 3.

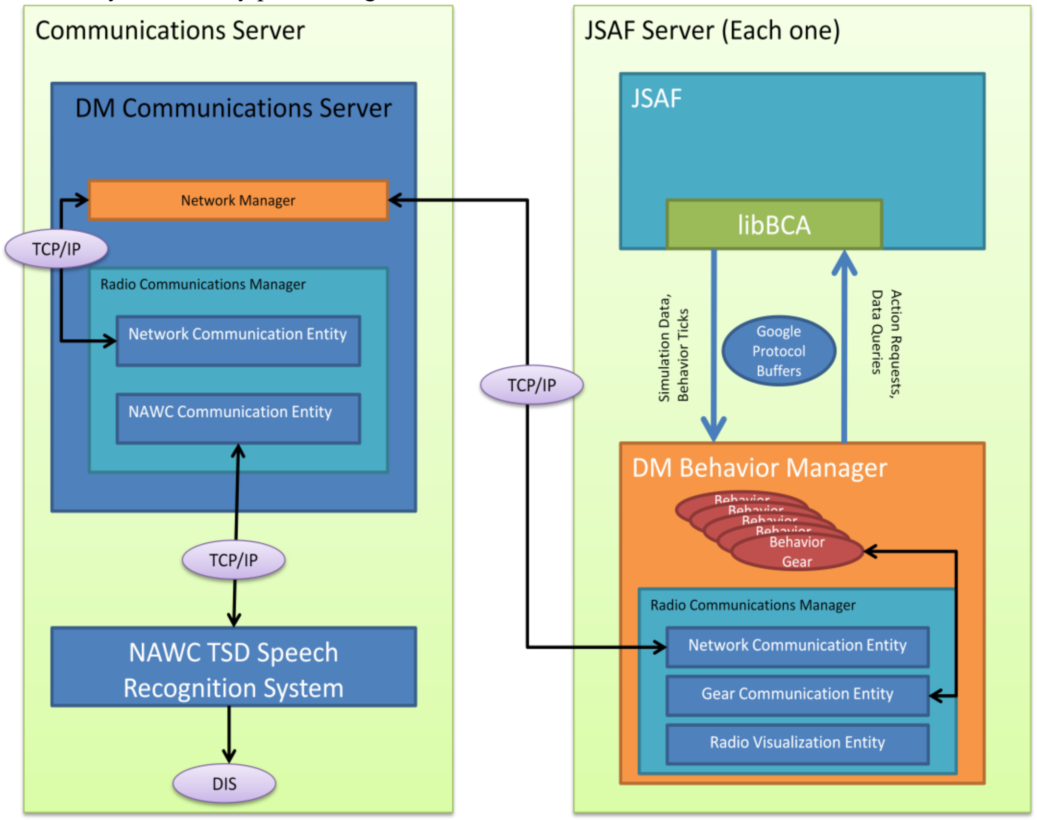
Communications Server
DM Communications Server
Network Manager
TCP/IP
Radio Communications Manager
Network Communication Entity
NAWC Communication Entity
TCP/IP
NAWC TSD Speech Recognition System
DIS
TCP/IP
JSAF Server (Each one)
JSAF
libBCA
Simulation Data, Behavior Ticks
Google Protocol Buffers
Action Requests, Data Queries
DM Behavior Manager
Behavior Gear
Radio Communications Manager
Network Communication Entity
Gear Communication Entity
Radio Visualization Entity

Figure 3 – Example Systems Architecture

This architecture is designed to be modular allowing for the implementation of specific communication entities which can be plugged into the Radio Network Manager at runtime, severing the specific implementation of a speech recognition solution from the behavior models themselves.

A single Communications Server runs on the network, communicating with the Behavior Manager running on each simulation via a TCP/IP socket. The server and behavior managers contain Radio Communication Managers which facilitate the registration of various types of "Communication Entities". Examples of these would be a "Network Communication Entity", designed to handle messages being transferred over a TCP/IP socket, such as those going to and from the Communications Server, or a "Gear Communication Entity", designed to handle messages being sent and received by individual behavior models within the simulation.

Each Radio Communication Manager can dynamically incorporate a set of these Communication Entities at runtime, enabling a variety of functionality depending upon the requirements of the system. An example of this is the Communication Entity designed to relay information to a speech recognition system. At runtime, the Communications Server constructs a communication entity whose implementation includes a mechanism for interacting with a separate speech recognition server (SRS). In the example presented here, the communication took place over a TCP/IP connection and passed information back and forth with the SRS. This including receipt of message from the SRS containing recognized commands, and transmission of messages to the SRS containing speech to be synthesized for consumption by human users.

This architectural approach allows for an extremely flexible solution, severing the implementation details of communication from the message contents and behavioral responses.

Modular Integration

The architecture described in this paper supports modular development of individual components. To date, we have integrated with three separate speech recognition solutions, each implementation requiring only that a new communication entity object be created, while the rest of the system remained untouched. Likewise, this framework supports communication between human users and simulated entities in a number of different simulation environments. Thus far, we have successfully implemented this approach in JSAF, Bohemia Interactive's Virtual Battlespace 2 (VBS2), and VT-Mak's VR-Forces simulation. By separating the implementation details of communications from the behavior models themselves, the solution presented here actually enables complex communications between entities deployed across a variety of simulated environments simultaneously. For example, a single Communications Server can communicate with entities existing within both JSAF and VBS2, as well as a speech recognition server, allowing a human user to interact over a virtual radio with simulated surface vessels deployed in JSAF as well as simulated ground troops deployed in VBS2, with those autonomous entities also communicating with each other as need be.

CONCLUSIONS

In this paper we have presented a framework for incorporating flexible communications between human users and simulated entities across a variety of simulations. This system enables complex messages to be sent between not only humans and their simulated counterparts, but also between the simulated entities themselves. This supports development of autonomous behavior models which function effectively, regardless of whether the individuals they are interacting with are played by humans or other autonomous agents.

This flexibility and independence of the behavior models allows for development of training scenarios where any individual role, or subset of roles, can be trained simply by replacing that entity with the human to be trained. These scenarios can reduce training costs by removing the requirement that large numbers of live human actors be available for training skills that require communication between individuals.

When paired with the TMK modeling based approach, we preserve human understandability of these behavior models, allowing subject matter experts with no experience in software engineering or programming to understand and directly participate in the behavior modeling process. Behaviors can be viewed in a graphical format, enabling subject matter experts to understand the actions taken by an autonomous entity, which helps build trust in the system. Finally, the graphical display of TMK hierarchies at runtime provide invaluable feedback to the behavior modeler, helping to identify and correct errors in the manifested behavior within the simulation.

By incorporating these communication handlers into individual Basic Level Actions, modularity and composability of behaviors is maintained. An end-user with no programming experience can create complex behaviors in a manner of minutes, automatically incorporating the ability to communicate in complex ways with both humans and autonomous entities. Large libraries of communication can be created, while only requiring any individual behavior model include those elements of communication relevant to its assigned tasks. This reduces the size of the behavior model, improving efficiency at runtime, and improving understandability.

ACKNOWLEDGEMENTS

We would like to thank the Office of Naval Research and NAVAIR-TSD for their continued support of the work described in this paper.

REFERENCES

Budanitsky, Hirst (2001), "Semantic Distance in WordNet", Univ. of Toronto, 2001 WordNet::Similarity (Perl module), http://wn-similarity.sourceforge.net/

Chandrasekaren, B. (1986) Generic Tasks in Knowledge-Based Reasoning: High-Level Building Blocks for Expert System Design, *IEEE Expert*. Fall, 1986.

Endsley, M.R. (1995) Toward a theory of situation awareness in dynamic systems. *Human Factors, 37(1)*, 32-64.

Garcia, C.J. & Griffith, T.W. (2005) A Composable Behavior Modeling System for Rapidly Constructing Human Behaviors, *Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC) 2005*.

Gray, W. D., John, B. E., & Atwood, M. E. (1993) Project Ernestine: Validating a GOMS analysis for predicting and explaining real-world task performance. *Human-Computer Interaction, 8*, pp. 237-309.

Griffith, T.W. (1999) A Computational Theory of Generative Modeling in Scientific Reasoning, PhD. Dissertation, College of Computing, Georgia Institute of Technology, July 1999.

Griffith, T.W. (2006) Domain Specific Knowledge Capture Interfaces for Behavior Modeling. *Proceedings of the Interservice/Industry Training, Simulation, and Education Conference.*

Potts, J.R., Griffith, T., Sharp, J.J., & Allison, D. (2010) Subject matter expert-driven behavior modeling within simulation. *Proceedings of the Interservice/Industry Training, Simulation, and Education Conference.*

Rosch, E., LLoyd, B. (1978) *Cognition and Categorization*, Erlbaum, Hillsdale, NJ.

Rosch, E., Mervis, C., Gray, D., Johnson, D., & Boyes-Braem (1976) Basic Objects in Natural Categories, Cognitive Psychology, 8, 1976, 382-439.

Sharp, J.J., & Potts, J.R. (2011) Improving Trainee Engagement Levels through Adaptive Entity Behaviors. *Proceedings of the Interservice/Industry Training, Simulation, and Education Conference.*

Discovery Machine, Inc. Patents Awarded:

US Patent 7,257,455 B1 – System and Method for Collecting and Representing Knowledge

US Patent 7,958,073 – Software and Methods for Task Method Hierarchies

US Patent 8,019,716 – Reflective Processing of Computer Hierarchies