The Role of Speech in a Distributed Simulation: The STOW-97 CommandTalk System[*]

Alan J. Goldschen, Lisa D. Harper, E. Richard Anthony
The MITRE Corporation
1820 Dolley Madison Blvd.
McLean, VA 22102
{alang, lisah, ranthony}@mitre.org

1. Abstract

We describe the role of CommandTalk during the 1997 Synthetic Theater of War (STOW-97) interoperability exercise. We present issues relating to the use of the speech recognition and synthesis components, and to the observed user error-feedback during this exercise. The purpose of CommandTalk for STOW-97 was to provide a spoken language interface to the ModSAF simulation. CommandTalk allows exercise participants to interact with synthetic entities using the same commands appropriate for directing human forces. This ability enables military personnel from the Air Force, Army, Marine Corps, and Navy to interact directly with the simulation without first learning specific skills for operating the simulation system. Thus, CommandTalk allows exercise participants to concentrate on the priorities and goals of the exercise.

CommandTalk translates spoken utterances from human exercise participants into Command and Control Simulation Interface Language (CCSIL) commands for synthetic entities to process. Likewise, synthetic entities send CCSIL messages to other synthetic entities and to human participants, which CommandTalk translates into appropriate natural language statements. We first describe CCSIL and CommandTalk, and then focus on the role, usage, and error feedback mechanisms of the CommandTalk system during the STOW-97 exercise. Finally, we propose future directions for CommandTalk in distributed interoperability simulations.

Keywords: speech recognition, speech synthesis, HCI, simulation, STOW, CommandTalk, distributed, CCSIL, dialogue management, error-feedback, multi-modal interface

2. Introduction

CommandTalk is a bi-directional spoken language human computer interface that allows natural language communication between humans and synthetic entities. In the STOW-97 distributed interoperability simulation exercise, these entities are part of the Modular Semi-Automated Forces (ModSAF or SAF) simulation. The SAF simulation system uses the Command and Control Simulation Interface Language (CCSIL) for entities to communicate with one another (MITRE 1996). CCSIL defines structured, formatted command and control messages for each of the four armed services (Air Force, Army, Navy, and Marine Corps). CommandTalk (Moore et al. 1996, Bratt et al. 1996) consists of state-of-the-art spoken language technology (speech recognition, speech synthesis, and natural language understanding) designed specifically for commanders and exercise participants to interface with synthetic entities.

Figure 1 depicts the overall system in which a commander interacts with a SAF simulation using the CommandTalk speech interface and the SAF's graphical user interface (GUI). The CommandTalk speech interface includes both recognition and synthesis components that use service-specific spoken language grammars. Commands that originate from GUI interactions affect entities only within a particular SAF, while commands that originate from the CommandTalk spoken language interface affect entities across all SAF simulations. This interoperability capability occurs because CommandTalk translates spoken language commands into CCSIL messages (Goldschen

[*] This work is supported by SSC (formerly NRaD) for STOW.

1997). CommandTalk bi-directionally translates spoken language messages into CCSIL messages and translates CCSIL messages into spoken service-specific (natural language) statements heard with a speech synthesizer.

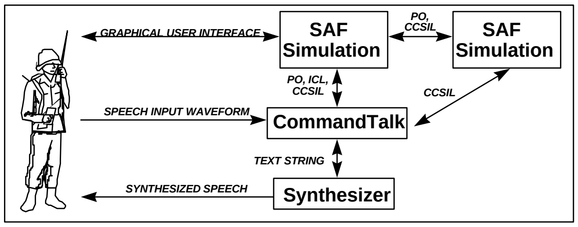

Figure 1 Commander, CommandTalk, and Multiple SAFs

2.1 CCSIL

The Command and Control Simulation Interface Language (CCSIL) facilitates interoperability between and among entities in a simulation exercise (MITRE 1996). CCSIL allows for synthetic entities to exchange command and control information, such as orders, directives, status reports, and intelligence reports (MITRE 1996). CCSIL messages contain a specific message type and message header. CCSIL supports ground, sea, air, and tactical operations. A ground operation, for example, might require that a platoon move to particular location. The CCSIL message header contains sender, receiver, and radio number fields. The *sender* field contains the name of entity sending the CCSIL message. The *receiver* field contains the name of the entity (or names of the entities) that the CCSIL message is intended for. The *radio number* field corresponds to the radio frequency selected by the sender. All virtual and human entities tuned to a particular frequency receive all CCSIL messages transmitted on that frequency.

2.2 CommandTalk

CommandTalk was initially conceived as a training tool for the Marine Corps Leathernet program (Moore et al. 1996, Bratt et al. 1996). CommandTalk provides the STOW-97 exercise with spoken language human computer interface technology for human commanders to interact with entities in SAF simulations. CommandTalk uses grammars developed for each of the four military services. These grammars allow two types of spoken language input. The first type of spoken language input is for exercise (or scenario) manipulation. An example of this kind of command is *'Create an M1A1 company at 950 960 facing northeast'*. The second type of spoken input is for issuing command and control orders such as *'First platoon assault and secure objective alpha'*. The commander always has the option to issue commands with the SAF GUI. CommandTalk, however, eliminates the need for commanders to learn how to issue complex exercise objectives from the SAF GUI.

CommandTalk incorporates into a SAF simulation an Open Agent Architecture (OAA) and speech-related technologies (Cohen et al. 1994, Dowding et al. 1993, Dowding et al. 1994). The Open Agent Architecture provides the infrastructure for distributed processes (agents) to communicate. These agents coordinate, interact, and plan using the Interagent Communication Language (ICL), a Prolog-style language (Dowding 1996). CommandTalk agents translate spoken commands directly into SAF simulation commands. The speech input display provides error feedback relating to the spoken language input. This display indicates whether the speech recognizer is ready or busy, a command translates properly, and provides the text of the spoken utterance.

Figure 2 depicts CommandTalk agent interaction after the issuance of a platoon halt by the commander. This figure shows that commands directed towards a SAF simulation, e.g. Marine Corps SAF (MCSAF) entity, originate either from the SAF GUI or from the CommandTalk ModSAF agent.

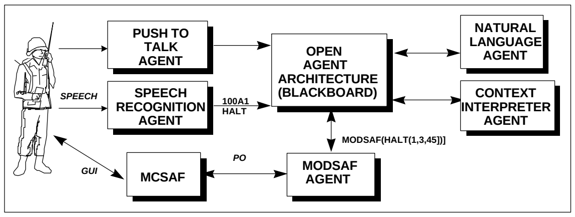

Figure 2 Commander halting a platoon in CommandTalk

CommandTalk spoken language input requires a push-to-talk button and microphone not shown in Figure 2. The speech recognition system (Nuance 1996) processes the spoken language input into a text string for the Gemini natural language (NL) parser (Dowding et al. 1993). Next, the NL parser produces a semantic logical form for the Context Interpreter (CI). The CI resolves under-specified information such as pronouns and speaker reference.

Figure 2 illustrates the following example. After the speaker pushes the microphone and utters the statement ‘100A1 Halt’, the *Speech Recognition (SR) Agent* recognizes the statement and places the text string *‘100A1 Halt’* on the blackboard. The *Natural Language Agent* accepts this message and places a context-independent structured interpretation on the blackboard corresponding to the SR Agent text string input. The purpose of the NL agent is to perform natural language parsing and semantic interpretation of the recognized word string (Dowding et al. 1993, Dowding et al. 1994). The *Context Interpretation Agent* picks up this message and places on the blackboard one or more MCSAF-specific ICL statements corresponding to the context-independent structured interpretation.[1] The Natural Language and Context Interpretation Agents ensure, in this example, that platoon 100A1 is currently performing an MCSAF-specific task. The Context Interpretation Agent specifically places on the blackboard the MCSAF statement *‘ModSAF(Halt([1,3,47]))’*. Persistent object (PO) protocol provides the mapping of the name *‘100A1’* to the persistent object, in this case, *‘[1,3,47]’*. Finally, the *ModSAF Agent* updates the MCSAF simulation from this MCSAF statement. If platoon 100A1 does not exist in the simulation, the speech interface displays a message (relayed from the context interpreter) that 'unit 100A1 does not exist in the simulation’, and CommandTalk does not send the command to the SAF simulation.

[1] The key difference between the functionality of the Natural Language Agent and the Context Interpretation Agent is that “*the structured interpretation [from the NL Agent] encodes only information directly expressed in the word string (or utterance), while the Context Interpretation Agent applies contextual information to produce a complete interpretation*” (Moore et al. 1996).

2.3 CommandTalk and STOW-97

Rather than customizing a specific *ModSAF Agent* for each SAF simulation in STOW-97, CommandTalk provides two additional agents to support human and entity communications (see Figure 3). Goldschen (1997) defines the CCSIL Agent (CA) as a process that translates *externally generated messages* into CCSIL formatted messages which may be sent to entities in any SAF simulation participating in the interoperability exercise. These external messages originate from the speech-input component of the CommandTalk interface. A second process, the CCSIL-TO-ENGLISH (C2E) Agent translates CCSIL messages generated by synthetic entities into service-specific (natural language) statements for the speech synthesizer.

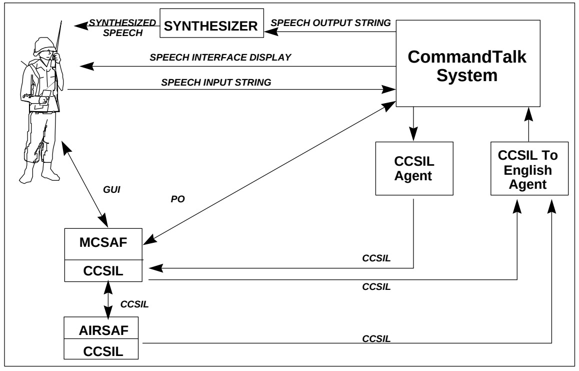

Figure 3 STOW-97 CommandTalk System

Figure 3 illustrates a commander using both the ModSAF GUI and CommandTalk to interface with entities in the MCSAF. Synthetic entities within the MCSAF generate CCSIL messages to interact with entities in distributed SAF simulations. Thus, CommandTalk provides, by using CCSIL messages, a means for humans to interact with entities in distributed SAFs.

2.3.1 Commander to Entity Communication

The ability for a commander to send command and control messages to other entities in a distributed interpretability simulation exercise defines the role of the CCSIL Agent (CA) (Goldschen 1997). The CA translates utterances from a commander into CCSIL messages that are sent to entities in different SAF simulations. As Figure 3 depicts, the CA attaches to (or represents) a human commander interfacing with MCSAF in an interoperability exercise.

The CA translates only command and control messages into CCSIL. The ModSAF Agent processes non-command and control messages that affect entities only in the MCSAF. With the CA, for example, a Marine commander in MCSAF can request entities in the Air Force SAF to provide fire support using the CCSIL Fire Support message (MITRE 1996). Before sending a CCSIL message, the CA determines the common radio number (frequency) shared by sender and receiver. Since the CA, in this example, attaches to an entity in MCSAF, messages which the CA sends use the CCSIL software components of MCSAF. All entities tuned to a given frequency and within the distributed interoperability simulation exercise receive the CCSIL message. Entities listed in the receiver field of the CCSIL message respond as appropriate (based on their internal SAF-specific algorithms).

2.3.2 Entity to Commander Communication

The ability for a commander to receive spoken language messages from synthetic entities defines the role of the CCSIL-To-English (C2E) Agent. The C2E Agent translates CCSIL messages from different SAF entities into service-specific statements for the speech synthesizer (Black et al. 1997) to process. Speech synthesis allows entities of multiple STOW-97 simulations to communicate on specific radio frequencies. The CommandTalk speech synthesis system selects different voices for entities so that listeners may distinguish among the multiple entities broadcasting on the same frequency. A commander or exercise participant may select one or multiple radio frequencies to monitor or eavesdrop on CCSIL message traffic. Eavesdroppers hear all message traffic over a given radio frequency regardless of whether a message originates from a synthetic entity or a human. The text translation that the C2E forms from a CCSIL message depends on the military service (i.e., service-specific grammar) of the sender.

2.3.3 CommandTalk System Feedback

Figure 3 illustrates that CommandTalk provides three different system-to-commander feedback mechanisms. Feedback originates from the CommandTalk speech interface display, from the SAF GUI, and from synthesized CCSIL messages.

First, the speech interface display provides user feedback that indicates the success or failure of spoken language input. An utterance, although considered valid by the speech recognition grammar, may not be valid for the current simulation state. For example, the speech recognizer may recognize the name of a unit in an utterance and consider the utterance valid. However, if the unit does not exist in the current simulation, the context interpreter rejects this utterance.

To provide the user feedback about the validity of a statement, the speech interface display uses both visual (color) and audio (sound) information. For example, this interface displays yellow when initializing, green when ready, and red when an error occurs. The speech interface displays, if possible, the reason for the error. CommandTalk provides audio cues such as a short beep when a spoken utterance is successfully translated and a short ding when an error occurs.

Second, the SAF GUI provides visual feedback in response to a spoken command. For example, if the commander says 'Zoom in on Boomer one', the SAF zooms in and centers on unit Boomer1. If unit Boomer1 does not exist, the speech interface displays an error message, and no message is sent to the SAF.

Third, synthetic entities provide spoken language feedback in response to commands from other entities and the human controller. When an air controller says, *'Lancer zero one, vector two seven zero,'* CommandTalk translates and synthesizes Lancer01's response as *'Roger. Vector two seven zero.'* Spoken language feedback indicates to the controller that the entity has received and understood the command.

3. STOW-97

STOW-97 consists of prototype high fidelity computer simulation software training tools. By providing the ability to quickly create, execute, and assess realistic joint training exercises, STOW-97 helps to support Joint Task Force missions. STOW-97 fosters CommandTalk as an advanced technology that provides exercise participants with a more familiar, realistic interface and reduces the number of training support personnel. Prototype computer simulation training exercises, typically, require operators to use an unfamiliar computer interface and additional support personnel to manipulate the exercise. This violates a standing goal to 'train as you fight'.

The purpose of CommandTalk for the STOW-97 exercise is to provide a realistic and familiar spoken language interface to SAF simulations for exercise participants. CommandTalk allows these participants to interact with synthetic entities using the same commands as for directing human forces. Military personnel from the Air Force, Army, Marine Corps, and Navy interact directly with the simulation without learning specific skills for operating the simulation system. Thus, CommandTalk enables the participants to concentrate on the priorities and goals of the exercise.

CommandTalk was successful in its uses during STOW-97 for the Air Force, Army, and Marine Corps exercise participants. Navy participants did not use CommandTalk.

3.1 Training Audience

Military personnel from the four services, Air Force, Army, Marine Corps, and Navy composed the training audience for the STOW-97 exercise. For the Army and Marine Corps, the CommandTalk grammar supports company and platoon commanders of mechanized platforms. This grammar allows commanders to move units, change speed, modify formations, and assault enemy locations. The Army grammar also includes a number of engineering support functions. The Navy grammar allows commanders to steer ships, change courses, set movement speeds, and fire missiles. Finally, the CommandTalk grammar supports Navy and Air Force Weapons Controllers (AWC). AWC's are members of an air combat crew and assist in air combat missions as ground-based 'eyes and ears' of the team. Controllers assist the fighters by directing them to targets and relaying information, such as, air contacts, air base status, and observed tactics. The CommandTalk AWC grammar comprises a subset of real-world Weapons Controller language, and the C2E Agent synthesizes a subset of Air Force and Navy combat pilot language.

CommandTalk supports scenario management, such as creation of synthetic forces, placement of control measures, and SAF display manipulations (e.g. panning and zooming). During STOW-97 support personnel used CommandTalk to quickly create scenarios and to manipulate the SAF GUI.

3.1.1 Scenario Examples

The CommandTalk grammar development uses military verbal and radio communication from real military scenarios. We specifically incorporated realistic scenarios to identify the STOW-97 Navy and Air Force language requirements for Defensive Counter Air (DCA) and Close Air Support (CAS) missions. DCA missions typically require more monitoring and intervention than CAS missions.

3.1.1.1 Defensive Counter Air

DCA missions require coordinated activities between controllers and air crew. The DCA scenario (Harper et al. 1997) defines language requirements for similar and dissimilar air tactics to counter active air threats such as other fighters or ballistic missiles. These missions use air or surface-based radar controllers to monitor threats with operational radar and intelligence information. During STOW-97, controllers used CommandTalk spoken language technology to provide synthetic aircraft entities with tactical air information such as bearing, range, and altitude about targets. Controllers spoke to these entities with detailed advisory information about threats in the rapidly changing environment and in response to synthetic entity requests.

The DCA scenario provides a basis for understanding the operational use of common brevity terms and for identifying the air message structure. The air grammar, constrained by CCSIL message limitations, did not fully mirror the real-world human controller and human pilot communications. Statements CCSIL did not support were removed from the DCA scenario. These items include certain types of pilot readbacks, ambiguous receiver, and target location by reference off another target.

3.1.1.2 Close Air Support

CAS missions described by (Jackson et al. 1996) and mapped into CCSIL (Goldschen et al. 1996) contain the language requirements to support realistic CAS scenarios. These missions consist of air actions using fixed and rotary-wing aircraft against hostile targets in close proximity to friendly forces. These scenarios require detailed communications among controllers, pilots, and ground forces.

Although the CommandTalk language supports CAS missions, the STOW-97 exercise did not use these missions with CommandTalk. Two qualified military controllers were on position at all times and did not need to intervene nor take direct control of pre-planned CAS missions.

3.2 Lessons Learned

STOW-97 provides an opportunity to critically observe CommandTalk and examine its key technical and operational strengths.

3.2.1 Spoken Language Input

The following operational and technical lessons relate to the speech recognition portions of CommandTalk.

- Speech is a natural, realistic, and effective means for enabling new users with adequate domain knowledge to interact with the simulation system.
- The *find* speech command provides a faster method to locate entities and points than the SAF GUI.
- The speech recognizer performs well with a noise canceling microphone. Background noise during STOW-97 did not seriously degrade the recognition accuracy.
- The push-to-talk button is an awkward interface, especially for the AWCs since they sometimes use both hands to operate the simulation system.

The following lessons relate to the spoken language grammars used for the STOW-97 exercise.

- Written checklists that list commands by desired action are helpful to the unassisted commanders and controllers. Checklists and positional guides are familiar to military personnel since operator workstations require them.
- Use of service-specific grammars allow exercise participants to speak naturally into the CommandTalk interface.
- CommandTalk, during STOW-97, did not have the capability to dynamically add new elements (entity names, points) to the speech recognition grammar. Entities with unexpected names could not be addressed during the exercise. Software changes took at least 24 hours, which is a disadvantage in a rapidly changing exercise environment.

3.2.2 Spoken Language Output

The following lessons pertain to spoken language output.

- In conjunction with speech recognition, speech synthesis enables realistic human computer interfaces.
- Multiple synthetic voices allow exercise participants to distinguish different entities.
- Exercise participants find it useful to simultaneously monitor multiple frequencies.
- Entities responses were difficult to hear through audio speakers. Instead, these responses should be heard through operator headsets to reduce the noise level in the operations room.
- Some messages from the speech synthesizer were too slow to keep pace with the action.
- Speech from the synthesizer sounds too machine-like and is sometimes difficult to understand.
- CommandTalk did not synthesize all CCSIL properly. The C2E Agent algorithm limits how certain messages translate from CCSIL into natural language statements. Furthermore, exercise participants did not always agree about the proper translation of a CCSIL message to natural language statements.
- It is confusing for commanders to hear their own synthesized messages.

3.2.3 CCSIL

These issues relate to CCSIL's role as the communication mechanism between the CCSIL Agent, C2E Agent, and SAF.

- CCSIL communication allows for the command and control of entities in different SAF simulations.
- To increase the speed and number of simulation entities per machine, CCSIL support was removed from AirSF simulation for the STOW-97 exercise. This modified SAF is AirSF-lite. However, CommandTalk and the CCSIL Agent could not communicate with entities in AirSF-lite. As a solution, the CCSIL Agent connected to a non-participating exercise entity on an AirSF-full station. This connection allowed AWCs to control any entity in the exercise.
- CCSIL message types and structure definitions limit the usage of the language. For example, an air controller may say 'push it up' to request an aircraft to increase speed during an intercept. There are no corresponding CCSIL messages for this message. Any speed changes must, therefore, use vector messages, which contain a mandatory field for direction (e.g., *vector west*).

3.2.4 Feedback Mechanisms

We describe a post-hoc analysis of user error-feedback during the STOW-97 exercise. Messages from CommandTalk to the SAF are either for scenario manipulation (GUI-related) interactions or commands to synthetic entities. This section focuses on commands sent between humans and cooperating synthetic entities. CommandTalk agents communicate information about speech input using the speech interface GUI. If a message passes successfully through the speech input system, synthetic entities respond upon receiving a valid message.

Since commanders during STOW-97 spoke to synthetic entities with the same language as they would to human counterparts, these commanders expected *natural, predictable interactions* with synthetic entities. These interactions, however, were not natural because there was no familiar means for resolving potential mis-communications.

We describe some error feedback mechanisms to support natural interaction between human and synthetic entities. We present these mechanisms by *level of linguistic error*, and we reference specific CommandTalk and SAF simulation components for resolving the error. The CommandTalk OAA consists of agents that accept and process messages, and submit new messages for other agents to process. Each CommandTalk agent examines a message to process information at a specific linguistic level. The Speech Recognition Agent converts speech signals into a string of words. The Natural Language Agent verifies the syntactic and semantic structure of this word string. The Context Interpreter Agent examines this new structure in the context of larger chunks of information. For each level of linguistic analysis, a specific CommandTalk agent processes the message and identifies potential errors.

Table 1 provides an eight-level breakdown of the types of errors from the use of CommandTalk during STOW-97. This analysis maps the type of error to a level of interpretation failure from Duff et al, 1996. Levels of interpretation failure correspond to levels of error detection in the cognitive model described by Clark et al., 1989. This model serves as a useful means for analyzing system failures as they pertain to communication failures between human and synthetic entities. Table 1 lists the type of linguistic communicative failure, the CommandTalk agent most likely to resolve the error at a given level, a functional description of the error, and an example from STOW-97 CommandTalk. We summarize each entry of Table 1 with an air weapons controller example from the STOW-97 exercise.

Level 0: Synthetic entities did not respond upon receiving a command that they determined invalid. This lack of response caused confusion for controllers trying to communicate with these entities. As a consequence, these controllers deviated from standard procedures (e.g., query that the agent received and understood the spoken message). Controllers would repeat the same command several times, and then give up communicating with the agent until a later time.

Level 1: Using the speech interface display, controllers recovered easily from speech recognition errors such as noisy, broken, or garbled transmissions.

Level 2: Controllers watched the speech interface display for possible recognition errors. This interface displayed the string 'invalid command' for commands not structurally valid. This type of feedback allows controllers to repeat the command.

Level 3: Controllers sometimes unknowingly stated an ill-formed command causing mis-recognition. In these cases, controllers did not know what was wrong with the spoken message since display feedback was insufficient. For example, the speech interface did not indicate parts of the message structurally or semantically ill-formed.

Level 4: CommandTalk correctly parsed and sent commands to synthetic entities, which did not always exhibit correct behavior. Often, synthetic entities did not execute a given command since that command required the entity to exceed model domain constraints. These entities partially executed commands without informing the controller that the *exact* command was not executed. For example, a fighter entity receiving a message to exceed its valid speed parameters, responds by moving at the highest speed possible -- not at the requested speed. This entity does not inform the controller that it is not completely obeying the command.

Level 5: Controllers did not receive feedback about potentially *conflicting* commands. Conflicting commands can cause unpredictable behavior. For example, when an air controller says "*left 270*" and the shortest turn to 270 is right, the synthetic entity accepts the heading and ignores the direction. The entity assumes that the controller meant to say "*right 270*". The controller, however, might mean to *say* "*take the long way around to 270 and turn right*", a perfectly valid command. The entity should either question the turn direction or obey the complete command.

Level 6: Controllers did not receive feedback for perfectly valid commands that entities could not execute. This situation typically arose when the software behaviors for the command did not exist in the SAF simulation (the behavior had not yet been implemented). The controller did not have the proper feedback to know why a command was not executed.

Level 7: Since it is possible that an unintended message can successfully pass through the speech input system, controllers monitored text strings in the speech interface display. This capability allowed controllers to re-state the intended message to the receiving entity.

Level	**Linguistic Level**	**CT component**	**Error Description**	**Examples in CommandTalk**
Level 0	Discourse	(DM) bi-directional monitor	No speech received following prompt to user	Synthetic entity: "Say again Line 9?" No response Human air controller: "What State?" No response. Respondent error.
Level 1	Acoustics	SR	SR has no hypothesis for current acoustic input	Unrecognized or garbled input (e.g., broken transmission, user abort, stutter) SR or human error.
Level 2	Syntax	NL	Invalid syntactic structure	Human air controller recognized as: "Contact two knight four zero" …actually said "contact two nine zero" SR error. Due to mis-recognition this was determined an invalid command.
Level 3	Sentential Semantics	NL/CI	Semantic type conflict: ill-formed CCSIL command	Human air controller said: "Climb angels" Human error. Missing mandatory argument. Should have been "Climb angels *altitude*"
Level 4	Domain KB	Agent domain model	Violation of static Domain Model	Human air controller said: "Set speed mach 2 point oh" Human error. This is an impossible speed.

Level 5	Contextual Felicity	Agent state	Infelicitous in current discourse context	Human air controller said: "left 270" Human error. Statement should have been "right 270"
Level 6	Pragmatics	Agent behavior rule-base	Pragmatics	Human air controller: "RTB in five minutes" No rule exists in agent behaviors to interpret this command.
Level 7	Speaker Detection	User	Error not detected above	Human said, "Ice 25 RTB" Recognized as "Ice 29 RTB" SR error. Valid system input. Error caught by the speaker.

Table 1: Eight Categories of Miscommunication

4. Future Directions

Due to the experience gained from the STOW-97 exercise, we identify the following items to enhance and complement capabilities of CommandTalk.

4.1 Spoken Language Input

CommandTalk should provide an automatic mechanism to add specific entity names into the spoken language grammar. Commanders should be allowed to address by name anything within the simulation. This technology exists and would remove the requirement that names be known in advance of an exercise.

4.2 Spoken Language Output

We recommend that CommandTalk use the following features for spoken language output. First, we should add "radio realism" into the simulation. Anthony et al. (1997) describes the integration of a spoken language interface that uses a SINGCARS radio model. This provides a means of adding realistic noise into the signal. Second, the system should provide the capability to dynamically select a synthesis grammar. For example, Air Force pilots are accustomed to hearing a message phrased as '*bogey two-seven-zero fifteen'* for contact reports, while Navy pilots prefer to hear *'bogey bears two-seven-zero for fifteen miles*'. Finally, CommandTalk should offer the ability to control message synthesis based on the dynamic situation state of the simulation. The current system uses a one-to-one mapping between CCSIL messages and corresponding natural language statements. In reality, message form and content differs greatly depending on current situation state. For example, during intercepts, controller and aircraft transmissions at long range are more detailed than at shorter ranges.

4.3 Error Feedback

Most of the errors between controllers and synthetic entities during STOW-97 occurred because humans had insufficient information about the actions and intents of synthetic entities. We recommend an approach that minimizes these types of user error feedback. This approach uses a cognitive model to distinguish among levels of understanding in human-human dialogue (Clark et al. 1989).

We suggest two general strategies to achieve effective user error feedback in accordance with the model. The first uses a centralized Dialogue Manager that accesses all bi-directional message traffic, domain knowledge, and system state information. The Dialogue Manager is responsible for diagnosing errors, choosing and executing a repair strategy (Duff et al. 1996). An alternative strategy expands the current CommandTalk agents to account for the types of errors (as Table 1illustrates). When necessary, these agents should be able to query and negotiate with other agents to resolve specific errors.

5. Acknowledgment

We thank the DARPA STOW Project Manager Cdr. Peggy Feldmann for her support of CommandTalk.

6. References

Anthony, E., C. Bowen, M. Peet, and S. Tammaro (1997), "Integrating a Radio Model with a Spoken Language Interface for Military Simulations," Proceedings of the 5th

European Conference on Speech Communication and Technology, Rhodes, Greece, 22-25 September, 1997, pp. 1823-1826.

Black, A., A. Taylor, R. Caley, (1997) "The Festival Speech Synthesis System," 1997, available from http://www.cstr.ed.ac.uk/projects/festival.html

Bratt, H., J. Dowding, M. Gawron, Y. Gorfu, and B. Moore (1996), "Technical Overview of the CommandTalk System," SRI International Report, January 30, 1996.

Clark, H. and E. Schafer (1989), "Contributing to Discourse," Cognitive Science, volume 13, pages 259-294, 1989.

Cohen, P., A. Cheyer, M. Wang, and S. Baeg (1994), "An Open Agent Architecture," in AAAI Spring Symposium Series, Software Agents, Sanford, California, pp. 1-8, 1994.

Dowding, J. J. Gawron, D. Appelt, J. Bear, L. Cherney, R. Moore, and D. Moran (1993), "Gemini: A Natural Language System for Spoken-Language Understanding," in Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics, Columbus, Ohio, pp. 54-61, 1993.

Dowding, J., R. Moore, F. Andry, and D. Moran (1994), "Interleaving Syntax and Semantics in an Efficient Bottom-Up Parse," in Proceedings of the 32nd Annual Meeting of the Association for Computations Linguistics, Las Cruces, New Mexico, pp. 110-116, 1994.

Dowding, J., (1996) "ModSAF Agent Language," SRI International Report, September 8, 1996.

Duff, D., B. Gates, and S. Luperfoy (1996), "An Architecture for Spoken Dialogue Management," ICSLP 96, vol. 2, available from http://www.asel.udel.edu/icslp/cdrom/vol2.htm.

Goldschen, A. and L. Harper, (1996) "Mapping of Close Air Support Demo Commands into CCSIL Agent Commands," MITRE Technical Report, in progress, Draft dated October 14, 1996, (not in public domain).

Goldschen, A., (1997), "The Role of the CCSIL Agent for Distributed Simulations," 1997 Spring Simulation Interoperability Workshop, Orlando, Florida, March 1997.

Harper, L. and A. Goldschen (1997), "Mapping of Defensive Counter Air Commands into CCSIL Commands, Draft MITRE Technical Paper, (not in public domain).

Jackson, C. and C. Petersen (1996), "Command and Control Simulation Interface Language: Close Air Support Demonstration," BMH Draft Document Report, March 3, 1996, http://www.bmh.com/bmh/CAS/CAS_article.html.

The MITRE Corporation, (1996), Command and Control Simulation Interface Language (CCSIL), MITRE Informal Report, Modeling and Simulation Technical Center, volumes-VI, Release 3.0, October 7, 1996, (not in public domain).

Moore, R., J. Dowding, H. Bratt, J. Gawron, Y. Gorfu, and A. Cheyer (1996), "CommandTalk: A Spoken Language Interface for Battlefield Simulations," SRI International Report, May 30, 1996.

The Nuance Speech Recognition System (1996), Version 5, Nuance Communications, Menlo Park, CA, 1996.

7. Authors' Biographies

Alan J. Goldschen was the Lead Engineer for the CommandTalk STOW-97 effort. He is a Lead Engineer in the MITRE Signal Processing Center and holds a Ph.D. in Electrical Engineering from George Washington University.

Lisa D. Harper, a former Air Force air weapons controller, is an AI Engineer for the Washington Artificial Intelligence Center at The MITRE Corporation. She holds an M.S. in Theoretical Linguistics from Georgetown University. She is currently a Ph.D. candidate with particular interests in natural language processing for dynamic 3D environments.

E. Richard Anthony was the STOW CommandTalk project leader during the STOW-97 ACTD. Currently, he is a Senior Signal Processing Engineer with The MITRE

Corporation and holds a M.S. in Electrical Engineering from Stanford University.