The Role of User Error-Feedback in the STOW-97 TacAir-Soar CommandTalk System*

Lisa D. Harper

Computational Linguistics
Department of Linguistics
Georgetown University
harperl@gusun.georgetown.edu

The MITRE Corporation
1820 Dolley Madison Blvd.
McLean, VA 22102
lisah@mitre.org

1. Abstract

This paper describes the role of user error-feedback in the TacAir-Soar CommandTalk system during the 1997 Synthetic Theater of War (STOW-97) interoperability exercise. The purpose of TacAir-Soar CommandTalk for STOW-97 was to provide a spoken language interface to the Air Simulated Forces (AirSF) simulation. The TacAir-Soar CommandTalk system provides human Air Weapons Controllers the capability to interact directly with simulated pilot entities using knowledge of standard radio communications procedures. This eliminates the need for military controllers to learn simulation-specific skills and to concentrate on mission goals and objectives. Issues are raised concerning the application of error detection, diagnosis and recovery strategies employed by human pilots and controllers to the TacAir-Soar CommandTalk system.

Keywords: HCI, STOW, CommandTalk, CCSIL, dialogue management, error-feedback, multi-modal, collaboration, situation awareness, repair dialogues, grounding.

2. Introduction

CommandTalk is a bi-directional spoken language interface to battlefield simulations that allows natural language communication between humans and synthetic entities. In the STOW-97 distributed interoperability simulation exercise, human exercise participants communicated with synthetic entities in the Modular Semi-Automated Forces (ModSAF or SAF) via spoken dialogue. For the air battle, the ModSAF simulation was modified to be an Air Simulated Forces (AirSF) environment. The purpose of CommandTalk in the STOW-97 AirSF environment was to allow human radar intercept controllers to interact with simulated pilot entities in a manner consistent with the way they communicate with human pilots. By using natural language to communicate with simulation entities, human exercise participants were able to focus on the goals and objectives of the exercise without spending large amounts of time and effort learning how to operate simulation-specific tools and interfaces.

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

* This work was accepted as an extended abstract in the 7th Conference on Computer Generated Forces (CFG) and Behavioral Representation, May 12-14, 1998, Orlando, Florida. Content from this paper appears in *The Role of Speech in Distributed Simulation: The STOW-97 CommandTalk System* (Goldschen, Harper, Anthony, 1997) in the 7th Conference on CFG and Behavioral Representation.

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, natural language parsing and context interpretation components) designed specifically for commanders and other exercise participants to interface with synthetic entities. CommandTalk system components interface with CCSIL to allow humans to send spoken messages into the simulation. Since synthetic entities also communicate with one another via CCSIL messages, all messages may be synthesized and heard over loudspeakers. In STOW-97, this capability enabled exercise participants to not only hear specific entity responses on a given operating simulated radio frequency, but also allowed *broadcasting* of any entity-entity message traffic on any known simulated radio frequency. This capability injected an unexpected element of realism into the STOW-97 exercise since frequencies could be passively monitored for simulation entity status reports without necessitating a constant "eyes on screen" during the 48-hour exercise.

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. 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 1997).

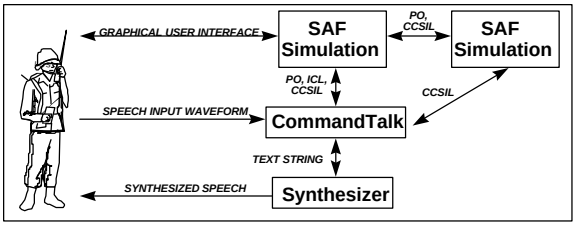

Figure 1 Commander, CommandTalk, and Multiple SAFs

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 supports ground, sea, air, and tactical operations. All virtual and human entities tuned to a particular simulated 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 provided the STOW-97 exercise with spoken language human computer interface technology for human commanders and radar intercept controllers to interact with entities in SAF simulations. CommandTalk uses specific 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 messages*** to simulation entities (e.g., *'Kill bandit 240 50 miles'*). The user always has the option to issue GUI-specific commands from the SAF GUI. CommandTalk, however, eliminates the need to learn how to issue complex GUI manipulations 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.

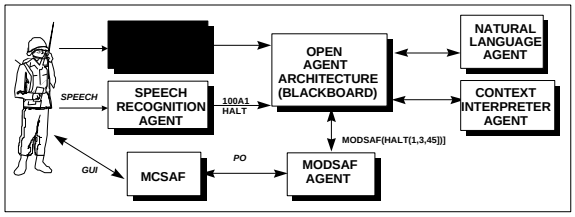

Figure 2 Commander halting a platoon in CommandTalk

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. The CI agent is also aware of some relevant non-linguistic context such as which entities are present in the simulation at any moment.

In figure 2, 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.

Rather than customizing a specific *ModSAF Agent* for each SAF simulation in STOW-97, CommandTalk provided 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.

[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).

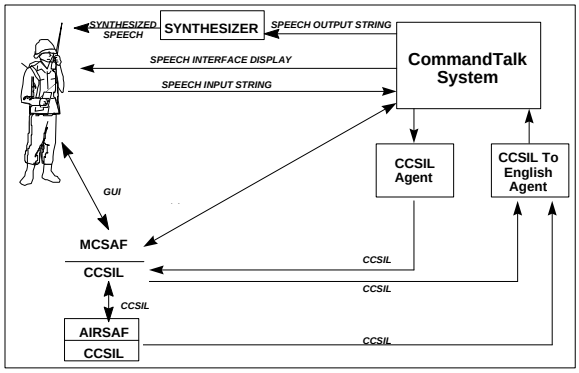

Figure 3 STOW-97 CommandTalk System

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 synthetic entities in *distributed* SAFs.

2.3 TacAir-Soar

The Soar group located out of the University of Michigan provided computer-generated air forces for STOW-97 AirSF. The Soar software, or Intelligent Forces (IFOR) architecture, provided STOW-97 with the ability to generate large-scale distributed entity-level simulations which required little human intervention. During STOW-97, fifteen types of Soar entities flew over ten different types of missions including air-to-air, air-to-ground, reconnaissance, and refueling, escort, and airborne early warning, etc. During the 48-hour exercise, over 700 fixed winged aircraft (FWA) were flown and approximately 30 to 80 planes were airborne at any one time (Laird, et al., 1997).

TacAir-Soar is the Soar system for flying FWA missions and interfaces with the AirSF simulation system via a Soar-ModSAF-Interface (SMI) (Laird, et al., 1997). The SMI serves as an interface between TacAir-Soar behaviors and the underlying AirSF simulation environment. In order to communicate with Soar entities via the CommandTalk interface, Soar was modified to send and accept CCSIL messages to and from the underlying AirSF simulation.

3. STOW-97

STOW-97 consisted 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 helped to support Joint Task Force missions. STOW-97 fostered CommandTalk as an advanced technology that provided exercise participants with a more familiar, realistic interface designed to reduce the number of training support personnel. Typically, prototype computer simulation training exercises 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 TacAir-Soar CommandTalk system was designed as a 'train as you fight' training tool.

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.

For the TacAir-Soar CommandTalk system, grammars were developed to support Air Force and Navy Air 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.

3.2 Scenario Examples

BMH Associates, Soar group, and The MITRE Corporation created storyboards and dialogues critical for the development of grammars and lexicons. Storyboards are an effective means for identifying the kind and scope of interactions in given scenarios. 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.

3.2.1 Defensive Counter Air

DCA missions require coordinated activities between controllers and air crews. 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 collaborated with entities by providing 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 which could not be mapped into CCSIL were removed from the DCA scenario. These items include certain types of pilot readbacks, ambiguous receiver, and target location by reference off another target. Specific problems of this nature will be addressed in section four below.

3.2.2 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 precise integration and preplanned coordination by supporting units. The nature of the interaction between controllers and pilots is highly scripted and primarily consists of aircraft hand-offs between adjacent control agencies. The primary collaborative effort between controllers and pilots occurs between the Forward Air Controller (FAC) coordinating operations on a visible target location and ingressing air-to-ground fighters. Since CAS is pre-coordinated, as long as mission parameters have not changed, ingressing fighters may not need to deviate from a prescribed target run.

Although the CommandTalk language supports CAS missions, the STOW-97 exercise did not use CommandTalk in support of these missions. 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.

3.3 System Capabilities

The TacAir-Soar CommandTalk spoken-language interface includes a push-to-talk button and microphone similar to that used by radar intercept controllers. Operators are able to the view the battlespace in the simulation in a manner analogous to viewing an actual radar console. Operators also are able to monitor simulated radio frequencies in a manner similar to the way Ultra High Frequencies (UHF) are monitored in air operations centers. The key difference for communications in the TacAir-Soar system is that the controller is speaking not with human pilots, but with synthetic pilot entities. A Secondary difference is that the controller also uses speech to communicate with the AirSF GUI.

3.3.1 Communication with the GUI

The types of voice commands used to control the AirSF GUI include functions commonly performed by controllers on radar scopes such as zooming in and out on a target, “selecting” a digital piece of data, centering in on a point, creating, deleting or manipulating digital overlay data. By using the CommandTalk

interface, all of these "hands-busy" activities are enabled by voice. Whether or not this ability is more or less efficient or desirable than by hands-on GUI manipulation was not evaluated.

One possible benefit to having voice-activated displays involves coordination of complex actions. For example, when determining the distance between a fighter and a target via ModSAF GUI controls, first one needs to manually roll the cursor onto the fighter, push a button to select the fighter, and then scroll a cursor out to a target. This action is made complicated by the necessity to sometimes zoom in and out to select and de-select targets. By voice one could potentially ask, '*range from fighter x to target y*.' In actual radar systems, continuous information about focus items may be automatically displayed. However, there exist many complex manipulations that might be made more efficient by voice commands.

It is important to note that as the language for GUI manipulation becomes more complex and information-laden, problems for novice users increase. Also there is a greater chance for miscommunication and corresponding need for more sophisticated system error diagnosis and repair mechanisms. Currently, GUI feedback in the CommandTalk ModSAF system is simple command execution or speech recognition failure.

3.3.2 Human to Synthetic Entity Communication

The ability for a commander to send command and control messages to other entities in a distributed interoperability 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 SAF in an interoperability exercise. The CA translates only command and control messages into CCSIL. By using 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).

3.3.3 Synthetic Entity to Controller 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.

3.3.4 CommandTalk System Feedback

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.

4. Pilot-Controller Error Detection, Diagnosis and Recovery

The FAA in 1993 attributed as many as 23 percent of all operational errors to be caused either directly or indirectly by miscommunications (Federal Aviation Administration, 1993). Types of linguistic errors include phonetic similarity, ambiguous phraseology, transposition (e.g., number sequences), recipient not monitoring, etc. (Cushing, 1996). Though statistics for military aviation miscommunications are not available for this paper, there is good reason to believe that this number is much lower for military operations – particularly during air-to-air combat.

Under military control, three main strategies are employed to reduce miscommunication during air-to-air combat missions: adherence to MCM 3-1 operational brevity guidelines, levels of control, and strict communications flow.

4.1 Operational Brevity Terms and Phraseology

MCM/TACP/PACFM/USAFEM 3-1, Vol. I contains a glossary of unclassified operational brevity words and terms used to provide a common understanding and to minimize radio transmissions by aircrew and controllers. These terms are not all inclusive and plain English is used to communicate when no appropriate term or phrase exists. Correlative terms occur in allied forces terminology which may sometimes conflict. This is a problem area for joint force operations.

4.2 Levels of Control

Levels of control refers to the level of pilot autonomy in a mission. There are four levels: close control (highly directive), tactical control (directive/advisory) advisory control (suggestive/informative) and broadcast control (broadcast informative). Levels of control vary depending upon real or simulated states of readiness, controller saturation, and airspace restrictions. In general, the amount of pilot autonomy increases stepwise from close, tactical, advisory, and broadcast control. The importance of level of control in pilot-controller dialogues relate to command *intent*. For example, a vector directive by a controller under close control is an order while the same directive under advisory control is a suggestion. Under tactical control the controller will issues command directives for safety purposes, but may issues directives during intercepts which are advisory in nature. Thus, "left 240" in one context may be an order, while in another context only a suggestion.

4.3 Communications Flow

Communications flow in controller-directed air-to-air missions are general structured as:

1	(a) **controller** directive/statement/query (b) **pilot** readback/reply (c) **controller** listen and verify

2	(a) **pilot** request/statement (b) **controller** readback/reply (c) **pilot** listen and verify

These two blocks illustrate basic units in military aviation discourse structure. Statement one is functionally a *presentation* phase while statements two and three comprise an *acceptance* phase. Diagnoses and repair dialogues are initiated in the acceptance phase. These basic units of conversation incorporate a mechanism for both preventing and repairing errors. This model of initiation and acceptance is consistent with Clark and Schaefer, 1987. Clark and Schaefer hypothesize two levels of communication. At one level, speakers and listeners address a topic and at another level they establish grounding of beliefs about that content. The basic units of dialogue presented here provide a mechanism which serves to satisfy the Grounding Criterion: '*the speaker and addressees mutually believe that the addressees have understood what the speaker meant to a criterion sufficient for current purposes'* (Clark and Schaefer, 1987).

Dialogue initiation is dependent upon task and situation. However, the listener is required to verbalize a readback if a command (advisory or directive) has been issued. (If the initiator is requesting information, the readback will be omitted and a response transmitted.) This readback stage is critical since the readback confirms that the command has been understood and will be executed. Readbacks may be partial, such as "roger", "roger, 240", or full such as "roger, left 240". Partial readbacks are a potential sources for mis-understanding. If a partial readback is issued and the initiator believes that command has been fully understood – but in fact the initiator and responder have two different ideas of what the command is – then error can result. Therefore, a third step is required. The initiator must listen and verify that the command was understood. In so doing, if the initiator has any doubt that the command was not fully understood, a repair dialogue is initiated.

Communication in air-to-air combat is frequently multi-party discourse such that communication flows in a coordinated manner between a lead pilot and controller, pilot and wingman, and often wingman and controller on the same frequency. Military discourse is very different from that of civilian control. Civilian air communications generally flows in multiple channels from a single controller to multiple aircraft. Aircraft flight crews passively monitor communications between the controller and other aircraft. In military aviation discourse, true multi-party discourse may exist. Regardless, the ability to monitor other conversations is important towards effective situation awareness of all participants within a local area.

As previously mentioned, a repair dialogue is initiated if a readback is absent or incorrect. Keywords such as "confirm" or "negative" following a readback indicate that a repair dialogue has been initiated. However, if the communications flow is non-standard, repair dialogues may not be initiated. Non-standard communications flow may occur if workload is very high or the radio signal is very poor. Under these conditions speakers may switch to English vernacular to troubleshoot problems and players may take multiple turns in succession. The potential for error increases during periods of non-standard communications.

In summary, three general discourse strategies are applied in military aviation radio communications for the purpose of error prevention. MCM 3-1 ensures that a common set of terminology and phraseology is understood. Levels of control serve to establish intent when directives are issued. Finally, a standard communications flow serves as a mechanism for reducing miscommunications and facilitating timely error diagnosis and repair.

5. System Feedback Mechanisms

This section analyzes the TacAir-Soar CommandTalk system error-feedback and recovery mechanisms in terms of a cognitive model of interpretation. Messages from CommandTalk to the SAF are either for scenario manipulation (GUI-related) interactions or for communication with synthetic entities. This section primarily focuses on messages sent between humans and cooperating synthetic entities.

5.1 Cognitive Model of Error Detection, Diagnosis, and Repair

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

Described below are error diagnosis mechanisms required to support more natural interaction between human and synthetic entities. These mechanisms are presented by *level of linguistic error,* and specific CommandTalk and SAF simulation components are referenced as responsible agents for resolving a given level of linguistic 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 and Schaefer, 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.

Level 0: *System does not respond to a query or command.* In STOW-97, 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). During STOW-97, controllers tended to repeat the same command several times, and then give up communicating with the agent until a later time.

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

Level 2: *NL Syntactic Error.* 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 informed controllers that an invalid or mis-recognized command was issued. If the command was simply mis-recognized, the user could easily repair the situation. If the command was structurally invalid but thought to be structurally valid, display feedback provided insufficient information to enable repair of the situation.

Level 3: *NL Sentential Semantic Error.* Controllers sometimes unknowingly input a semantically ill-formed command (e.g., '*climb angels*' is missing an altitude argument). The issues for controller diagnosis and repair or the same as for level two, above.

Level 4: *Domain Knowledge Violations.* 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: *Contextual Felicity Errors.* Controllers did not receive entity 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: *Pragmatic Errors.* Controllers did not receive feedback for perfectly valid commands that entities were not pre-programmed to understand. 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: *Speaker Undetected Errors.* Since it is possible that an unintended message can successfully pass through the speech input system, errors must be diagnosable in subsequent dialogues. For example, if the controller directs that an aircraft descend to Flight Level 240 and the pilot believes that he has been directed to take a heading of 240, the error may not be detected till some moments later. In this case, there must be a mechanism by which the controller can query an entity about previous statements and actions. This capability did not exist in STOW-97.

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

5.2 Unresolved Issues

In this section several key unresolved issues central to "naturalness of communication" and user error-feedback and recovery are raised.

5.2.1 Interface Language Specifications

The Command and Control Simulation Interface Language (CCSIL) is a language developed to facilitate interoperability among entities in a simulation exercise. In TacAir-Soar, pilot entities communicate with one another and with human exercise participants via CCSIL messages. The CCSIL Agent facilitates the process of converting externally generated messages into CCSIL formatted messages. Not included in CCSIL message formats were discourse-related information relevant to dialogue turns, message intent (e.g., level of control), and speech act information. This type of information was not relevant towards building an interface language for simulated entity communication. Though a Context Interpreting Agent may be able to evaluate some types of discourse-level and contextual information during speech input, communicating discourse-level information to synthetic entities is important towards negotiating turns and initiating repair dialogues. One aspect of turn negotiation is the expected flow of communication. If synthetic entity feedback is not sensitive to dialogue structure inherent in controller-pilot communications, the familiar system of dialogue error-diagnosis and repair is disrupted.

Other factors limiting "naturalness of communication" include point-of-view references not included in CCSIL message definitions. Controllers and pilots strategically pass more succinct information as situation awareness increases. Following a standard flow of communication, callsigns are often omitted between speakers and addressees. In conjunction, references to previously established target entities are made relative to other established entities. For example, if a pilot has radar contact on one bandit, a second bandit may be identified as "trailer two miles (from the first bandit)". These kinds of dynamic descriptive references were not considered in the specification of CCSIL for air-to-air missions.

5.2.2 Dialogue Management and Error Diagnosis

Although many speech and system errors were effectively diagnosed and repaired by users of the TacAir-Soar CommandTalk system, a central dialogue manager sensitive to dialogue principles discussed in section four, and able to access situation, task and domain-specific knowledge could help facilitate error-recovery in levels four, five and six in the table above. As an alternative, if Soar agents were designed to be sensitive to dialogue structure and error-diagnosis mechanisms used in pilot-controller discourse, error-recover in interpretation levels four, five and six might likewise be improved.

5.3.3 Collaboration with Simulated Entities

As shown in section four, pilots and controllers collaborate in conversation by contributing to conversation in presentations and acceptances in an effort to present information and ground that information as the conversation progresses. Human controllers require the active participation of synthetic entities in this process in order to accomplish goals and objectives using standard communications procedures.

5.3.4 Situated Knowledge and Natural Language Generation

In STOW-97, synthetic entities communicated with human exercise participants via synthesized speech. Messages were prescribed and rigid with no sensitivity to changing contexts and dialogue goals. One common complaint during the STOW-97 exercise was that it was difficult to know what the Soar agents were "thinking". Part of this criticism is directed towards the *form* of entity feedback. Goldschen et al. briefly discuss advantages towards injecting situated knowledge into generated synthetic entity voice messages.

6. Future Directions

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. An approach that minimizes miscommunication would include knowledge of discourse strategies employed by controllers and pilots. Furthermore, an approach using a

cognitive model to distinguish among levels of understanding in human-human dialogue has been found useful in a post-hoc examination of functional requirements for error-diagnosis and repair in this system (Clark et al. 1989).

Two alternate strategies are proposed to achieve effective user error feedback in accordance with the Clark and Schaefer model presented. 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. This second alternative is non-architecture specific.

7. Acknowledgments

Thanks to Alan Goldschen (MITRE) for providing much guidance and support during the STOW-97 exercise and for his comments and advise during many discussions of the CCSIL, CommandTalk, and error-feedback capabilities of the TacAir-Soar CommandTalk system. Thanks also to Mark Gawron (SRI) and Paul Nielson (Soar Technologies) for many discussions regarding integration of SRI speech components and the TacAir-Soar simulation system.

8. References

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 (1987), "Collaborating on Contributions to Conversations," Language and Cognitive Processes, volume 2:1, pages 19-41, 1987.

Clark, H. and E. Schaefer (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.

Cushing, S. (1995). Pilot-Air Traffic Control Communication: It's Not (Only) What You Say, It's How You Say It.

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.

Federal Aviation Administration (1993). Data Link: Making Air Traffic Control Safer. FAA Technical Center: Atlantic City, N.J.

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.

Goldschen, A., L. Harper, R. Anthony (1997). "The Role of Speech in Distributed Simulation: The STOW-97 CommandTalk

System", 7th Conference on CFG and Behavioral Representation, 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.

Laird, J.E., K. Coulter, R. Jones, P. Kenny, F. Koss, P. Nielsen (1997). "Review of Soar/FWA Participation in STOW-97. http://ai.eecs.umich.edu/ifor/stow-review.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. Author Biography

Lisa D. Harper, an inactive reserve Air Force Air Weapons Controller, is a Sr. Artificial Intelligence 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 visual environments.