LARRI: A LANGUAGE-BASED MAINTENANCE AND REPAIR ASSISTANT

Dan Bohus
Carnegie Mellon University
Pittsburgh, PA, 15213
dbohus@cs.cmu.edu

Alexander Rudnicky
Carnegie Mellon University
Pittsburgh, PA, 15213
air@cs.cmu.edu

Abstract LARRI (Language-based Agent for Retrieval of Repair Information) is a dialog-based system for support of maintenance and repair domains, characterized by large amounts of documentation and by procedural information. LARRI is based on an architecture developed by Carnegie Mellon University for the DARPA Communicator program and is integrated with a wearable computer system developed by the Wearable Computing group at Carnegie Mellon University. LARRI adapts a dialog-management architecture developed and optimized for a telephone-based problem solving task (travel planning), and applies it to a very different domain—aircraft maintenance. The system was taken on a field trial on two occasions where it was used by professional aircraft mechanics. We found that our architecture, AGENDA, extended readily to a multi-modal and multi-media framework. At the same time we found that assumptions that were reasonable in a services domain turn out to be inappropriate for a maintenance domain. Apart from the need to manage integration between input modes and output modalities, we found that the system needed to support multiple categories of tasks and that a different balance between user and system goals was required. A significant problem in the maintenance domain is the need to assimilate and make available for language processing appropriate domain information.

Keywords: Multi-modal systems, maintenance and repair domain, task representation, field trials

Introduction

LARRI (the Language-based Agent for Retrieval of Repair Information) is a dialog-based system for support of maintenance and repair activities for aircraft mechanics. LARRI is based on an architecture developed by Carnegie Mellon University for the DARPA Communicator program and is integrated with a wearable computer system obtained from the Wearable Computing group at CMU (see, e.g., Smailagic et al., 2001).

LARRI is interesting in that it takes a dialog-management architecture developed and optimized for a telephone-based problem-solving task (travel planning), and applies it to a very different domain - aircraft maintenance. On the positive side, we found that our architecture, AGENDA (Rudnicky et al., 2000;Xu and Rudnicky, 2000), extended readily to a multi-modal and multi-media framework. At the same time we found that certain assumptions built into the system that were reasonable in a services domain turn out to be inappropriate for a maintenance domain. For example, the time-scale of activity in the maintenance domain is significantly different. In the travel-planning scenario the primary goal is to help the user rapidly find a satisfactory solution. In the maintenance domain, rapid and accurate solution is no longer the main goal; rather, accurate task execution and easy access to relevant maintenance-related information are the main goals. Moreover the system needs to provide two different usage modes: browsing and guidance. In the former, the user expects to be able to search for information and examine procedures; in the latter, the user and the system need to cooperatively carry out the step sequence associated with a given procedure. In both cases the underlying domain knowledge is the same, implying a need for a clearer separation between task and dialog representations.

We continue by describing LARRI's capabilities and functionality, based on a sample interaction with the system. Subsequently, Section 2 presents the architectural and implementational details behind the various system components. In Section 3 we describe a field study with US Navy F/A-18 mechanics, and discuss our current and planned future efforts for improving the LARRI system, in light of the feedback obtained during these field trials. Finally, Section 4 concludes the paper.

1. LARRI: System description

LARRI (**L**anguage-based **A**gent for **R**etrieval of **R**epair **I**nformation) is a multi-modal system for support of maintenance and repair activities for aircraft mechanics. The system implements a Level 4/5 IETM (Interactive Electronic Technical Manual) (iet, 2000), that is, semanticallyannotated documentation. While the use of such documentation is not currently widespread, it represents the next step in the evolution of documentation for complex systems, particularly military ones. LARRI explores the potential integration between language-based interfaces and such materials.

LARRI integrates a graphical user interface for easy visualization of dense technical information (i.e. instructions, video-streams, animations, annotations) with a spoken dialog system that facilitates information access and offers task guidance and mentoring in this environment. The graphical interface (illustrated in Figure 1) is accessible via a head-worn display connected to a wearable client computer. A rotary mouse (dial) provides direct access to the GUI elements.

The natural language spoken-dialog component enhances GUI functionality (which provides only for simple selection), with support for natural language commands and for direct information access. The preponderance of hands- and eyes-busy situations in this domain makes natural language dialog a very attractive interaction modality. Commu-

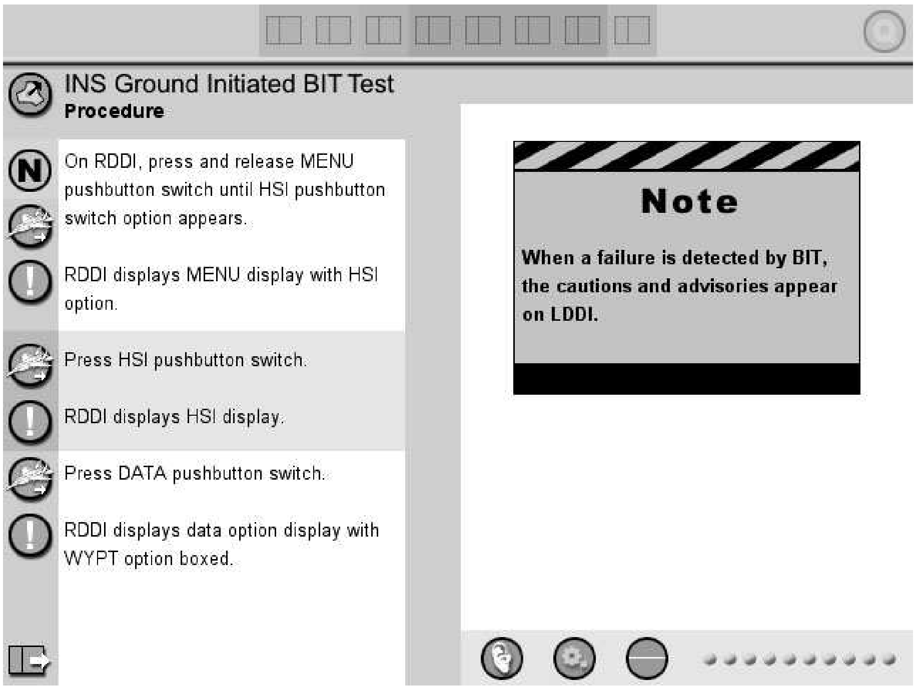

Figure 1. LARRI's Graphical User Interface

1 L:	Hi, this is LARRI. State your name please	***Login phase***
2 *U:*	*Hi, my name is John Doe*	*Profile retrieval*
3 L:	Hello John Doe. Before you start on your scheduled work, we need you to finish an urgent task	
4 *U:*	*OK*	
5 L:	Let's reprioritize your schedule: first you will perform the INS Ground Initiated BIT Test on plane 301. You last performed this task 3 months ago. Is that correct ?	*Task list negotiation*
6 *U:*	*Yes, that's right*	
7 L:	Then you will do the BXF radar check on plane 304. You last performed this task 2 weeks ago. Is that correct ?	
8 *U:*	*Affirmative*	
9 L:	When you are ready to start working on the INS Bit Test, please say ready to work	
10 *U:*	*Ready to work*	
	... ...	*System presents*
	... ...	*task summary*
11 L:	Step A: On RDDI press and release the MENU pushbutton switch until HSI pushbutton switch option appears	***Task guidance*** *(Step execution)*
12 *U:*	*Done*	
13 L:	Does the RDDI menu display the HSI ?	*Outcome verification*
14 *U:*	*Affirmative*	
15 L:	Step B: Now press the HSI pushbutton switch	
16 *U:*	*Ok. RDDI displays the HSI*	
17 L:	Ok. Step C: Press the DATA pushbutton switch	
18 *U:*	*<long timeout>*	
19 L:	Did you push the DATA pushbutton ?	*State clarification*
20 *U:*	*Yes, I just did*	
21 L:	Does the RDDI display DATA option display with the WYPT option boxed ?	*Outcome verif.*
22 *U:*	*Yes*	
23 L:	Ok. Step D: Select WYPT 0	
24 *U:*	*Okay. WPT is 0 and the latitude is local*	
25 L:	Good. Skipping to step J. On RDDI press and release MENU pushbutton switch until BIT pushbutton switch option appears.	*Branch on outcome*
26 *L:*	... ...	

Figure 2. A sample interaction with LARRI

nication becomes possible in parallel with manual work, and unwieldy interface devices (such as keyboards) can be eliminated. The use of natural language also provides opportunities for increased efficiency and flexibility: the user can interact with the system both on a simple command level (manipulating the GUI or simply stepping through a procedure) and on a goal-oriented level (where the user describes desired end-states and the system translates these into the corresponding command sequences). Furthermore, evidence from areas such as intelligent training and tutoring systems indicates that the use of speech and sound to convey information improves learner/user performance (Goodman, 1993).

Figure 2 illustrates a sample interaction with LARRI. Both the user (U:) and system utterances (L:) are presented, together with annotations on the right-hand side describing some specific system behaviors associated with those turns.

The LARRI scenario consists of a login phase, followed by the aided execution of several tasks and finally a logout phase. Once the user identifies himself, LARRI will retrieve the user's profile and current schedule from a backend agent (turns 1-2). Subsequently the schedule is displayed and LARRI engages in a negotiation/confirmation dialog to establish the tasks for the current session (turns 3-8). Once the login phase is completed, the system retrieves from the backend the set of maintenance tasks assigned to the user, and presents a brief summary of the state of the current task (in case it was left incomplete).

The typical maintenance task consists of a sequence of steps, which are to be performed by the aircraft maintainer in a certain order. The large majority of the steps (*basic steps*) contain a set of instructions, optionally followed by a set of verification questions in which the possible outcomes of the step are discussed. The various outcomes of a step can lead to different branches, either within the same task, or into a new (sub)task. Basic steps can also include animations or short video sequences which are to be displayed to the user, or various annotations introduced by previous maintainers which are available upon request. A secondary, and less frequent class of steps are *notes*, *warnings* and *cautions*. These are also visually present, and can be read to the user either at the system's own initiative or at the user's request (depending on the severity of the situation). Overall, maintenance tasks fit well into the AGENDA architecture, which uses a dynamically modifiable tree structure to represent the user's task.

Turns 11-26 in Figure 2 illustrate LARRI's behavior in guidance mode. However, the maintainer can also perform random access to the documentation/task, either by directly accessing the GUI elements with the dial or by issuing a set of spoken language commands such as "goto step 15", etc.

Once a maintenance task has been completed, the system provides again a brief summary of the completed activity, updates the corresponding information on the back-end side, and moves to the next scheduled task. Finally, upon completion of all the scheduled tasks, the user is logged out of the system and the session is concluded.

2. LARRI: An architectural view

LARRI is implemented as a distributed client-server architecture, shown in Figure 3. When tested in the field the system runs on three separate computers, one worn by the user and two stationary servers. For development purposes, the system runs on a single desktop computer. The wearable platform hosts clients for audio input and output, and the graphical user interface. Other components (i.e. speech recognition, dialog management, language generation and synthesis, task-specific agents, etc.) run on the server side. Communication between components is accomplished over an 802.11b link, using (TCP/IP) socket connections.

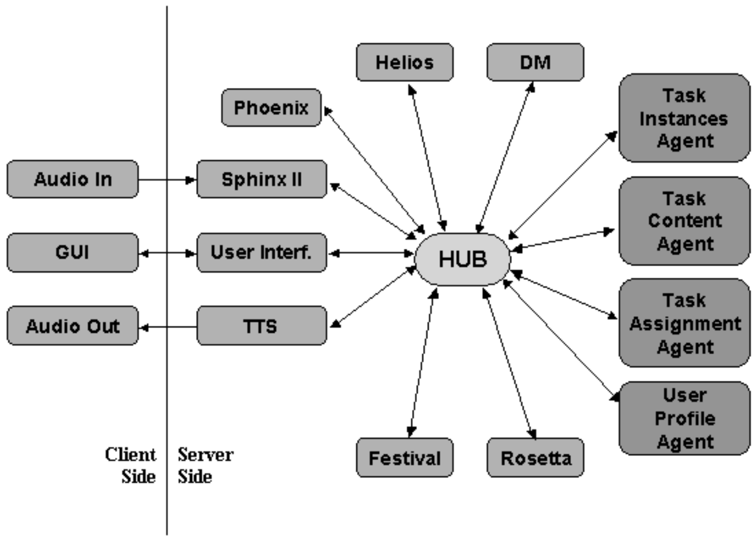

Figure 3. LARRI's Galaxy-based Architecture

The wearable computer is worn in a belt that enables participants to move the location of the computer to a comfortable area along their torso. It consists of a Pentium 233 MMX processor, with 64 MB RAM and a 680 MB hard drive, equivalent to an IBM ThinkPad 560X, but weighing 400 grams and measuring 26 x 80 x 120 mm. The head-worn display, developed at IBM's Watson Research Center, has a small liquid crystal chip with an image transmitted by a prism and redirected to the user's eye. With the magnifying optics contained in this device, it appears to users that they are reading a full-sized desktop computer screen with VGA resolution. LARRI uses a Gentex 100902 noise-canceling microphone, which is boom-mounted on a cranial [helmet] worn by the mechanic. The Gentex provides a high level of acoustic noise cancellation and allows the system to operate in the hangar environment (where the ambient noise is 85-95 dBA). Speech is acquired using a Telex H-551 USB digitizer and is sampled at 11.025 kHz and 16 bits.

LARRI makes use of the Galaxy (Seneff et al., 1998) architecture. The system is composed of a series of modules, implemented as parallel processes (i.e. speech recognition, understanding, synthesis, dialog management, etc.) that interact with each other by passing messages through a central programmable traffic-router—the Galaxy hub (see Figure 3).

Speech recognition is performed via the SPHINX II (Huang et al., 1993) decoder, using semi-continuous HMMs. Acoustic models were trained using the WSJ0 corpus, and a class-based, trigram language model was constructed and used for recognition. The current vocabulary, covering the INS BIT Test procedure and general system commands, contains 408 words. A semantic grammar was written for the INS BIT Test domain and is used by the Phoenix robust parser to create a semantic representation of user input, which is then forwarded to the Dialog Manager. Similarly, the events from the graphical User Interface are semanticized, integrated with the spoken input and passed to the Dialog Manager.

The Dialog Management component is based on the AGENDA architecture (Xu and Rudnicky, 2000), and makes use of 23 handlers to manage task and discourse phenomena. On the back-end side, the Dialog Manager communicates with 4 separate domain agents that handle user profiles, the task library, and task instantiation and assignment information.

Currently, the maintenance tasks and procedures are described in terms of a special-purpose declarative language (TML, Task Markup Language). This language is used both for defining *task types* (i.e. the structure of certain tasks, in terms of phases, steps, and the relationships between them), as well as *task instances*, which include additional information pertaining to a specific task instantiation (i.e. the completion status of each phase/step, how much time each step took, various maintainer annotations, etc). The three task domain agents retrieve this representation from a database, and then provide it to the dialog manager which uses it to construct the appropriate dialog handlers. The two forms of representation can be seen to correspond to the use of the data for browsing and for guidance purposes. For illustrative purposes, we show in Figure 4 an excerpt from an actual task instance, as represented by this formalism.

```
{
step_id 0003
step_gui_id INS.1.5
step_name c
step_type basic_step
instructions On GND PWR control panel assembly, set and
             hold 1 and 2 switches to B on for 3 seconds
estimated_time 30
default_execute_pointer next_step
outcomes_array :  1
   {
      {
      outcome_nr 1
      outcome_gui_id INS.1.5
      outcome Switches latch and remain on.
      execute_pointer_yes next_step
      execute_pointer_no task 00009
      result 1
      }
   }
start_time 010912-20:13:10
end_time 010912-20:14:02
completed 1
}
```

Figure 4. An excerpt from a task instance representation: the italicized material shows the instance-specific elements added to the basic type when it is instantiated for purposes of guiding the user through a procedure. Note the use of an `estimated_time` parameter that allows the system to trigger reprompts at the appropriate time (when in guidance mode).

LARRI uses the ROSETTA natural language generation module, but only in template-based mode, using both general prompts and prompts based on material in the task representation (e.g., `instructions` in Figure 4). Synthesis is performed using a Festival-based unit-selection synthesizer (Black and Lenzo, 2000). A limited domain voice was created especially for this task, with a fallback on a diphone voice.

3. Experiments and Results

3.1 Field evaluation

To better understand how the various elements of the system perform under realistic conditions, we evaluated LARRI on two separate occasions, once in July 2001 and once in December 2001. The evaluations took place at the Naval Strike Aircraft Test Squadron, Pautuxent River Naval Air Station, Maryland, with the participation of AE (Aviation Electrical) technicians, all Navy personnel. The evaluation built on previous work with a non-speech version of the system (Siegel et al., 2000), thus task materials as well as a well-defined and tested experimental protocol were available and were incorporated into the current evaluation (with modifications to take into account the use of a speech system). The materials used were modeled on level 4/5 IETMs (Interactive Electronic Technical Manuals) and the task used was the Inertial Navigation System (INS) Built In Test (BIT) for the F/A-18C/D fighter plane. We selected this particular task because of the availability of appropriate IETM material and existing task analysis information. Since this was a heuristic evaluation of a proof-of-concept prototype our focus was on understanding the experience of the mechanics. Thus each mechanic performed the task once and was then interviewed about their experience. The testing procedure included a period of instruction, to familiarize the mechanic with the wearable equipment and with the characteristics of the speech system, followed by actual performance of the task (which took place in the cockpit of an F/A-18C aircraft). Five maintainers participated. Three of the five mechanics were able to complete the task but gave the system divergent ratings, either very high or very low (a range of 1 to 4 on a 5-point scale). We attribute this outcome to problems in both interface design and in operational difficulties in the hangar environment, specifically the interaction between the computational requirements of the prototype and the difficulty of operating a wireless network in an environment high in RF noise (involving unpredictable outages in connectivity). Nevertheless both field evaluations provided valuable information that was used for system improvement.

While users commented favorably on the language-based interface, analysis of the sessions and user comments identified several issues. These include: better feedback on system state; a more flexible balance between the spoken and graphical outputs and improvements to the GUI design. In the next subsection, we discuss these issues in more detail, and present our current and future planned efforts addressing them.

3.2 Lessons learned and future plans

During the field trials, several problems were caused by a lack of proper feedback on (speech) system state. The user's focus is on the task, but he requires clear yet unobtrusive indication of whether speech input is being processed and understood correctly. Although the initial design included a VU meter and status indicators (see Figure 1), the experiments revealed that a more comprehensive state feedback is needed, both at the signal level (line quality, signal integrity, clipping, etc), and at the understanding level (recognition success). A successful feedback system would both communicate that a problem exists and would provide an intuitive guide to how it might be remedied.

A second important observation concerns creating an optimal balance between spoken and graphical output. Since a graphic display is available, the role of speech output changes correspondingly: speech is no longer the principal medium for transmitting information to the user. While one solution is to statically assign message types to output medium, we believe integration will be more valuable. The current system uses paraphrase and meta-comments to augment screen-based information.

A related, task-level observation is that the most experienced maintainers were actually slowed down by the system taking the time to go through the verification questions for each step. This underlined the importance of tailoring the system speech output to the situation at hand, and to the user's level of expertise. Users familiar with the task will not need to have it read to them step-by-step, while inexperienced mechanics could benefit from this redundant presentation. As a first solution to this issue, we added a "terse" vs. "verbose" mode for the system. Switching between the modes is achieved at the user's request, with the initial settings (reflecting the user's level of expertise) contained in the user profile. A more sophisticated model could take into account other factors such as the dynamics of the interaction and the user's history with the current task.

Based on the feedback and observations from the field trials, we have designed a new graphical user interface, which provides several improvements over the original one. Changes include: better system state feedback, popup help windows (i.e. on misunderstandings, a list of possible commands acceptable to the system at that point can be popped up on the screen to prime the user), better distinction between steps and verification questions, and in general a cleaner design which conveys more clearly the user's current position and focus in the task.

From a practical perspective, a critical issue encountered in the maintenance domain is the problem of preparing source materials for use in a language-based interface. Documentation for the F/A-18 system, on paper, involves many thousands of pages of text, as well as diagrams and other graphical materials. Moreover, this documentation is revised on a regular basis. None of this material, understandably, is currently prepared with a view to being accessible through a spoken-language interface. The unresolved problem is how to automatically prepare such materials for (intelligent) spoken-language based access. This is not a problem that is adequately addressed by current toolkit approaches to spoken language interface development (which at best simplify the prototyping of small closed-domain applications) but one that necessarily needs to be addressed for large and constantly mutating domains such as aircraft maintenance.

Currently, we use a declarative language (TML, Task Markup Language) for maintenance tasks, from which dialog handlers for the Agenda dialog manager are generated dynamically, as needed. Nevertheless, both the semantic grammar for the Phoenix parser, and the system's prompts covering the INS BIT Test had to be handcrafted. Given that IETMs for new systems are created in XML form, we anticipate using an XSLT program to render the IETM into the corresponding TML; however the process needs to in addition automatically generate a grammar for the parser as well as appropriate prompts for generation. To drive the recognition system we also need to generate a corresponding language model and pronunciation dictionary. Our previous experience with designing such a process (for a movie information service that provided current information) indicates that a fully automatic process is difficult to create. Yet such a process would be essentially given the impracticality of manually creating and maintaining the requisite knowledge base.

4. Conclusion

We have described a language-based system for a complex maintenance and repair domain. This domain differs significantly from service-oriented domains such as travel planning in several respects. First, the nature of the activity violates common assumptions made in services domains in that precision and completion are the primary goals, and simple speed of execution is less important. The criterion for success is jointly determined by the system and the user rather than by the user only (who, for example, is the only judge of whether a particular itinerary is satisfactory). The maintenance domain involves multiple categories of interaction such as guidance and browsing, that imply different relationships with essentially the same domain information (thus task and domain representations need to be explicitly differentiated, even when the domain information effectively describes a task). As well, users may be engaged in several comparable maintenance activities, meaning that the system needs to track individual tasks and be able to manage the user's change of focus between them.

Second, the nature of the interface is no longer uni-modal and the system must provide for both gestural input and graphic output. In our particular domain, which is hands-busy, manual input does not appear to be a major factor in the interface. In particular, voice provides a more appropriate and flexible input for navigation, for data entry and for information access. Nevertheless in situations of excess noise, gesture becomes the only viable input. On output, we have found, unsurprisingly, that speech and graphic outputs need to be coordinated and that care must be taken to avoid unnecessary redundancy between the two channels. At the same time the user needs to be given control over the details of how outputs are distributed over modalities. For example, in an eyes-busy situation a verbal rendering of screen content is essential but cannot be anticipated a priori.

Finally, the maintenance and repair domain stresses the importance of automating the assimilation of domain information into a form that can be used by a language-based system. In many dialog systems this knowledge is handcrafted; this is possible because the size or complexity of the domain is manageable. We believe that effective interfaces to maintenance documentation require the extraction of substantive structure relevant to the tasks performed; the challenge is to extract this information automatically without the aid of complex world models.

5. Acknowledgements

This research was sponsored in part by the Space and Naval Warfare Systems Center, San Diego, under Grant No. N66001-99-1-8905. The content of the information in this publication does not necessarily reflect the position or the policy of the US Government, and no official endorsement should be inferred.

We would like to thank the following individuals for their various contributions to this work: Dick Martin, Jane Siegel, Brian Gollum, Jack Moffat and Tom Hawley from the CMU Wearables Group; Rong Zhang, Alan Black, Kevin Lenzo, Ananlada Chotimongkol, Tina Bennett, Kayur Patel and Mike Seltzer from the Speech Group.

References

(2000). IETM Central. http://www.dcnicn.com/IETMCentral/default.asp.

Black, A. and Lenzo, K. (2000). Limited domain synthesis. *Proceedings of ICSLP*.

Goodman, B. (1993). Multimedia explanations for intelligent training systems. *Intelligent Multimedia Interfaces*, pages 148–171.

Huang, X., Alleva, F., Hon, H.-W., Hwang, M.-Y., Lee, K.-F., and Rosenfeld, R. (1993). The SPHINX-II speech recognition system: an overview. *Computer Speech and Language*, 7(2):137–148.

Rudnicky, A., Bennett, C., Black, A., Chotimongkol, A., Lenzo, K., Oh, A., and Singh, R. (2000). Task and domain specific modeling in the Carnegie Mellon Communicator. *Proceedings of ICSLP*.

Seneff, S., a. H. E., Lau, R., Pao, C., Schmid, P., and Zue, V. (1998). Galaxy-II: A reference architecture for conversational system development. *Proceedings of ICSLP*.

Siegel, J., Hyder, E., Moffett, J., and Nawrocki, E. (2000). IETM usability: Using empirical studies to improve performance aiding. (Technical Report CMU-CS-01-131).

Smailagic, A., Siewiorek, D. P., and Reilly, D. (2001). CMU wearable computers for real-time speech translation. *IEEE Personal Communications*, 8(2):6–12.

Xu, W. and Rudnicky, A. (May 2000). Task-based dialog management using an agenda. *ANLP/NAACL Workshop on Conversational Systems*, pages 42–47.