In Proceedings of ICSLP, pp. 122-125, Beijing, China, 2000

THE AT&T-DARPA COMMUNICATOR MIXED-INITIATIVE SPOKEN DIALOG SYSTEM

E.Levin, S. Narayanan, R. Pieraccini, K. Biatov, E. Bocchieri, G. Di Fabbrizio, W. Eckert, S. Lee, A. Pokrovsky, M. Rahim, P. Ruscitti, M. Walker

AT&T Labs – Research
180 Park Avenue, Florham Park, NJ, 07932
{levin,shri,biatov,enrico,pino,sungbok,andrei,mazin,ruscitti,walker}@research.att.com

ABSTRACT

The design and implementation of the AT&T Communicator mixed-initiative spoken dialog system is described. The Communicator project, sponsored by DARPA and launched in 1999, is a multi-year multi-site project on advanced spoken dialog systems research. The main focus of this paper is on the issues related to the design of mixed-initiative systems. In addition to describing our architecture and implementation of the complex travel task, the paper reports some preliminary evaluation results.

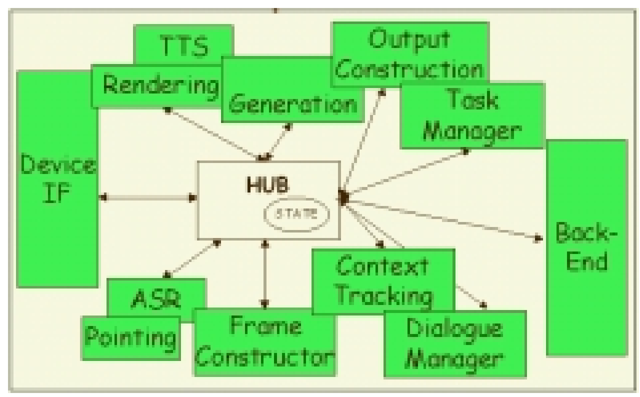
1. INTRODUCTION

Providing spoken language interaction capability as a part of multimedia user experience is believed to add naturalness, and perhaps, efficiency to human-computer interactions. Numerous commercial spoken dialog systems are currently being deployed, primarily for access to information over the telephone. There are, however, major open research issues that challenge deployment of completely natural and unconstrained spoken language interactions even for limited task domains. These primarily arise because the state-of the-art in automatic speech recognition (ASR) and spoken language understanding is far from being perfect. In practice, as a means of dealing with these limitations, spoken language systems are typically implemented by imposing constraints on the range and scope of user input allowed at any point during an interaction: both through well-designed prompts directing the user to answer specific questions and by concurrently limiting the scope of the underlying language models and grammars for ASR. Since constraining the scope of user input (i.e., the *initiative* that a user may take) compromises the apparent flexibility and naturalness of an interaction, dialog system designers tend to take a middle ground by allowing varying degrees of user initiative. This paper, using the example of the AT&T Communicator travel system, addresses the problem of the design and implementation of mixed-initiative spoken dialog systems for handling fairly complex tasks.

2. COMMUNICATOR ARCHITECTURE SPECIFICATIONS

The DARPA Communicator dialog architecture is hub centric as shown in Fig. 1 [1,2]. The hub is a programmable traffic router that is responsible to invoke the different servers in the system and routes the messages between them. The messages are represented by Galaxy frames [1,2]. The architecture does not define the functionality but just provides standard APIs. Therefore the servers depicted in Fig. 1 is a particular instantiation of the Communicator architecture. The servers

Figure 1. Galaxy Communicator Reference Architecture



operate through callback functions that are invoked by the hub.

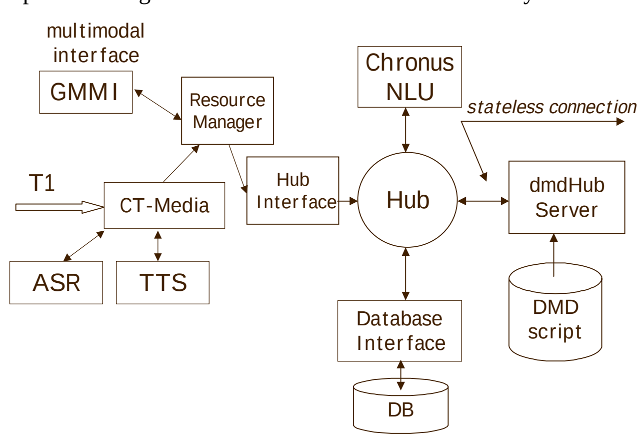


Figure 2. AT&T Communicator Architecture

The hub itself is event driven: upon receiving a new frame message, it finds and invokes the appropriate callback functions and passes the frame to the relevant servers. The callback up mechanism requires the servers to be “state-less” i.e., the state variables that control the operation of the servers have to be stored outside the callback functions. The architecture also provides the means for centralized logging of events at the hub level.

3. THE AT&T COMMUNICATOR COMPLIANT ARCHITECTURE

Figure 2 shows AT&T's implementation of the communicator-compliant architecture. The computer telephony is handled by an ECTF-standards compliant CT Media platform. We used an application independent middleware, called the Application Resource Manager (ARM) [9] that controls the I/O devices (telephony, multimodal interfaces) and I/O resources (ASR, TTS, GUI managers). The CT Media platform is versatile and supports multi-channel, multimodal services as well as IP and traditional telephony. In our implementation the hub controls the dialog manager, spoken language understanding, backends and other servers (e.g. timer) and talks to I/O resources and devices through the ARM.

3.1 SYSTEM COMPONENTS

ASR and TTS: We used the AT&T Watson continuous speech recognition engine[4] that supports audio barge-in capabilities. We used generic context-dependent acoustic models for telephone speech. The Watson recognizer supports both stochastic and finite-state grammars. We also used the AT&T Next Generation text-to-speech system [5] for generating the system responses on the fly.

Backend: One of the goals of the Communicator architecture is to accommodate "plug-and-play" i.e. exchange and sharing of system components. The AT&T system uses the flight server developed at the University of Colorado [3]. The server gets real-time travel data by scraping a commercial website and stores in a local database (SYBASE). The caching mechanism reduces the amount of time spent on web access during an interaction.

Timer: A timer server was implemented to output frames to the hub at specified time intervals. This server is invoked when the dialog manger requests the hub to make a database query. If the output of the timer precedes the results of the database query, the dialog manager can take an appropriate action eg., inform the user the status of the database query.

Spoken Language Understanding (SLU): We used the CHRONUS SLU system [7] that was adapted as a hub server.

Dialog Manager: The dialog manager controls the application. It decides when to invoke other modules (understanding, I/O decisions, database queries) etc. The dialog manager is implemented as an interpreter for a scripting language that permits easy manipulations of frame-like structures. The nature of the dialog management is inherently context dependent i.e., the dialog manager's current action may depend on events in the history of the dialog. But since the hub servers are invoked through callback functions special effort has to be made by the dialog manager server to preserve the history of the dialog to be used in the next callback in that session. This could be accomplished manually by the application designer, for example, saving key dialog variables in a global memory and reading them during the next callback invocation. Our dialog manager provides this feature automatically. The interpreter keeps track of its own state e.g., the value of variables, the execution stack etc. And saves it automatically when it returns. During the next callback, the interpreter loads the same state, and resumes the execution from it left in a seamless manner.

4. MIXED-INITIATIVE DIALOG STRATEGIES

System-initiative vs. Mixed-initiative dialogs. System initiative dialogs are ubiquitous in numerous IVR and several recently deployed commercial speech services. In system initiative dialogs, the flow of the interaction is completely controlled by the system. At each point in such dialogs, the system expects and will accept only a limited number of possible responses. The other extreme is user-initiated dialog systems. In such systems (for example, ATIS), the system responds to any user request without trying to constrain the expected input from the user in any way. For example in original the air travel domain (ATIS) system, a correct response to a user's query "show me the flights" would be to retrieve and show the user the entire flight database. The idea of mixed-initiative systems is to combine the flexibility of a user-initiative system with the problem-solving nature of system-initiative system. For example, a reasonable response to the query "show me the flights" could be "please tell me where you would like to fly".

Designing dialog strategies. The most common model for implementation of system-initiative dialogs is a tree wherein the root node is the opening prompt. The number of branches from each node corresponds to the number of different types of response the system allows the user of input at that point in the dialog. Since the mixed-initiative systems permit the user to change the course of the dialog at any point, the number of possible inputs at any point during the dialog (and hence the branching factor of the tree) is prohibitively large. We implemented the mixed-initiative strategy using the sequential decision process model [8]. This model is based on the definition of dialog state and dialog actions. Dialog actions corresponds the system interactions with the outside world e.g. users, backends and timer server. Dialog state represents the knowledge that the dialog manager keeps (value of all relevant variables) at any point in the dialog in order to determine the next action i.e., what to say to the user and what to expect from the user. The sequential decision process describes the operation of the dialog manager as follows:

Initialization: start from initial state

Iterate until done (final state is reached)

NextAction: Choose and perform next action

Get new input

NextState: Update state with new input

In the implementation of the communicator system the actions were interactions with the user of the backends; the inputs came either from ASR (user input), database query results or the timer server (when query results were not ready). The development of the strategy focused on the two main functions: NextAction and NextState.

ASR Performance. In addition to a more complex dialog strategy design, mixed-initiative system pose challenges for ASR

. In system-initiative dialogs, the grammar (or language models) can be constrained to the few possible inputs expected at this point in the dialog thereby increasing the underlying ASR performance. Mixed-initiative systems, in general, need to be able to process a wider range of inputs capturing possible user initiatives. This means larger and more complex language models and hence reduced ASR performance. Given the state-of-the-art in ASR technology, mixed-initiative system design needs to trade-off between the degree of initiative allowed and the ASR performance.

4. THE TRAVEL TASK

The main application for the DARPA Communicator project was the implementation of the complex travel task. The goal of the travel system is to provide a wide range travel-related services including multi-leg flights, hotel and car arrangements. We implemented the system following a mixed-initiative model as explained in the previous section.

Dialog Strategy. The functional flow of the dialog is as follows:

Sign in: In this stage, the user can sign in using his pin (obtained through web registration) wherein the user's profile will be retrieved which has information about departure location and user preferences for airline, hotel and car rentals. The user is also provided an option to sign in as a guest user.

Flight Planning consists of the following two stages for each leg.

Information gathering: The system solicits from the user mandatory information required for enabling a database dip: departure and arrival locations, date and time of the flight. Although at each turn the system will request information about one of these attributes, the user can take initiative and provide more than one piece of information at a time. For example the following is valid interaction:

SYSTEM1: *Welcome guest user! Where are you leaving from?*

USER1: *from Boston to Denver one-way tomorrow on United*

SYSTEM2: *Leaving from Boston to Denver. Flying on June fifteenth. United flight. One way. And, what time did you want to leave?*

In response to the request for his departure airport, the user provided multiple concepts that the system was successfully able to incorporate into its state. Since the only the mandatory unfilled was time information, the system automatically requested the preferred time in the next turn. Under normal dialog conditions, the systems provides an implicit confirmation of the concepts it processed in the previous dialog turn as illustrated in the turn marked SYSTEM2 in the above example. Since in a mixed-initiative system the user inputs are not constrained, this confirmation has to be generated dynamically. Further, the system will incorporate in its state also the non-mandatory attributes (such as airline, meal and seat preferences) provided by user's initiative.

Flight presentation and negotiation: Once the system has gathered all the mandatory data, it launches a database query and informs the user of the same. If the database results are delayed, using the information from the timer, the dialog manager generates "hold-on" messages to the user. If exact match to the user's required could not be found, the system takes initative in relaxing the airline and/or time preferences and informs the user about this. For example, the system may respond with "*Sorry there no flights with Delta leaving at that time. However, I found three other flights from Boston to Denver on June fifteenth....*"

When the retrieval resulted in multiple flights, the flights are sorted, by default, based on their price. The user is provided a brief summary of the number of flights together with the information about the first flight in the list. The user has the option of selecting the presented flight or browse through the list of flights using commands such as "next option", "the fifth option" etc or filter the list of flights providing additional constraints such as airline, different departure time etc. If the presented flight option is complex e.g., has multiple stopovers, the systems provides only minimal information while browsing and the prompts the users for further details, should they be interested.

Roundtrip flights are treated differently for efficiency. The information gathering stage covers both the outbound and return legs and the presentation/negotiation is for the full itinerary.

Ground arrangements are optional – the system prompts the user if they are interested in making hotel or car bookings. The information gathering is implemented similarly to the one in flight planning – system solicits for preferred hotel name, location, car rental company and car type. Should the user not express preferences, the system would suggest an option.

Itinerary summary presentation (optional upon user request)

User satisfaction polling and Closing remarks

Controlling the degree of initiative.

As we discussed above, there exists a trade-off between the initiative allowed for the user and the achievable ASR performance. This trade-off is dynamically controlled in our strategy as follows. The system assumes "normal" dialog conditions at the onset of the interaction wherein the maximum possible initiative for the user is allowed as described in the previous section. If the system detects "trouble" conditions (e.g., repeated request for the same attribute), the system gradually reduces the allowed scope of user input by applying more and more constrained language models together with more and more specific prompts. After several attempts, the system will switch to a strict system initiative mode where it will explicitly confirm each piece of information gathered until this point and continue to prompt the user for one piece of information at a time.

Meta functionalities. Our system provides meta functionalities: CANCEL, STARTOVER, REPEAT and HELP. Detailed context-sensitive help is to be implemented in the next version of our system.

5. PRELIMINARY EVALUATION RESULTS

A formal evaluation of Communicator travel systems for all participants was conducted in June/July 2000. NIST coordinated this effort by facilitating anonymous recruited users to call the various systems and conduct a scenario-based dialog with the systems and rate their experience by completing a web-based survey. The AT&T system received a total of 81 calls in this period from 81 different users. A summary of the results for our system is provided below (Table1): 71.6% of the calls (58/81) resulted in task completion (according to the user's judgment). Of remaining 23 calls, 21 were a result of call disconnection due to system-related problems including those of back-end, ASR, TTS and audio servers.

It's interesting to see that Mean-User-Words-per-turn is greater for TaskIncomp cases, which may imply that a frustrated user may produce longer utterances. This may render the situation even worse, especially if the degree of initiative that the user is allowed is concurrently reduced.

The mean number of concepts per turn conveyed by the user was around 1.3. A concept, roughly corresponds to a slot in the form, and was calculated automatically from transcribed user utterances by the SLU module (so called, conditional concept accuracy). In the data obtained from the 81 subjects The number of concepts per turn was closer to 1 if utterances corresponding to dates were ignored (typically subjects tended to convey date information in the month-day number-year format specified in the scenarios). This speculation was supported by the mean concept/turn measure for the open scenario (travel task parameters decided by the user) vs. fixed scenario (user provided specific destination, dates etc) tasks in the evaluation: 1.18 for open scenario vs. 1.31 for fixed scenario (significant difference, $p < 0.001$; data from 60 completed calls used: 48 fixed, 12 open). These numbers indicate that the users did not take any initiative and largely followed system directives providing only requested concepts while the system was capable of processing a fairly large number of concepts at any given turn during a normal interaction.

Measures	**TaskComp**	**TaskIncomp**
Word Accuracy	72.73	63.25
Mean-User-words-per-turn	2.28	2.97
Mean-Syetem-Utterance-Dur	9.11	7.33
Mean-System-Turn-Dur	9.47	7.78
Mean-System-Words-per-Turn	23.55	19.43
Response-Latency	1.34	1.23
OnTask-Dur	279.89	174.69
Total-Task-Dur	294.33	189.01
Prompt-percentage	67	60
Turns-To-Taskend	42.07	30.09
User-Words-To-Taskend	45.81	35.77
System-words-to-Taskend	518.50	321.05
Number of User Utterances	20.52	15.00

Table 1

Table 2 shows the summary of subjective user response (Likert scale 1-5, where a score of **1 is BEST**) provided by NIST based on the web-surveys. The mean and the median are for all the nine systems that participated in evaluation:

Question topic	**Score**	**Mean**	**Median**
Ease of use	2.27	2.88	2.8
Ease of understanding the system	2.2	2.23	2.1
Knew what to say	1.89	2.54	2.5
Worked the way user expected	2.41	2.95	2.9
Will Use system regularly	2.86	3.36	3.3

Table 2

SUMMARY

In this paper we discussed the AT&T implementation of DARPA communicator compliant architecture. We focused our discussion on the characteristics and implementation issues of mixed initiative dialog, and described the complex travel task application implemented according to the outlined principles. The results of preliminary evaluation are inconclusive with regards to the importance of allowing the user initiative and control over the dialog. This could be and artifact resulting from the dialogs being scenario based and generated by first time users using the system only once.

6. REFERENCES

[1] Seneff, S. et al, "Galaxy II: A reference architecture for conversational system development", Proc. of ICSLP, Sydney, Australia, pp. 931-934, 1998.

[2] http://fofoca.mitre.org/doc.html Galaxy Communicator

[3] Ward, W., and Pellom. B, *"The CU Communicator system"*, Proc. of IEEE ASRU Workshop, Keystone, CO, pp. 341-344, Dec. 1999.

[4] Sharp, R. D., Bocchieri, E., Castillo, C., Parthasarathy, S.,Rath, C., Riley, M. and Rowland, J. *"The Watson speech recognition engine."* Proc. ICASSP 97, 4065-4068, 1997.

[5] Beutnagel, M., Conkie, A., Schroeter, J., Styliano. Y, and Syrdal, A. *"The AT&T Next Gen TTS System."* Proc. Joint Mtg. ASA, EAA and DEGA, Berlin, 1999.

[6] Pieraccini, R., Levin, E. and Eckert, W., *"AMICA: the AT&T Mixed Initiative Conversational Architecture"*, Proc. of EUROSPEECH 97, Rhodes, Greece, Sept. 1997.

[7] Levin, E. and Pieraccini, R., *"CHRONUS: the next generation"*, Proc. of 1995 ARPA Spoken Language Systems Technical Workshop, Austin, Texas, Jan. 1995.

[8] Pieraccini, R., Levin, E., Eckert, W. and Narayanan. S, *"Spoken dialog systems: From theory to practice"*, Proc. of IEEE ASRU Workshop, Keystone, CO, Dec. 1999.

[9] Di Fabbrizio, G., Ruscitti, E., Narayanan, S. and Kamm. C, *"Extending computer telephony and IP telephony standards for voice-enabled services in a multi-modal user interface environment"*, Proc. of Interactive Dialogue in Multimodal systems, Kloster-Irsee, Germany, pp.9-12, Jun. 1999.