The CU Communicator: An Architecture for Dialogue Systems[1]

Bryan Pellom, Wayne Ward, Sameer Pradhan

Center for Spoken Language Research
University of Colorado, Boulder
Boulder, Colorado 80309-0594, USA
http://cslr.colorado.edu

ABSTRACT

This paper presents our recent work towards development of the University of Colorado (CU) Communicator, an interactive dialogue system for airline, hotel and rental car information. The CU Communicator integrates speech recognition, synthesis and natural language understanding technologies using the DARPA Hub Architecture to allow users to converse with an automated travel agent. During a typical telephone-based interaction, users can retrieve up-to-date travel information such as flight schedules, pricing, along with hotel and rental car availability. The CU Communicator has been under development since April of 1999 and represents our test-bed system for developing robust human-computer interactions where reusability and dialogue system portability serve as two main goals of our work. This paper will focus on describing our recent system improvements and will present an analysis of dialogues received and lessons learned.

1. INTRODUCTION

The CU Communicator system is a Hub compliant implementation of the DARPA Communicator task[1,2].The system combines continuous speech recognition, natural language understanding and flexible dialogue control to enable natural conversational interaction by telephone callers to access information about airline flights, hotels and rental cars. The system connects live over the web to get real up-to-date air travel, hotel and rental car information.

Our approach to robustness in natural language understanding is to use robust parsing strategies and event driven dialogue strategies. These strategies stress semantic content and coherence over syntactic form. We have extended the Phoenix[3] system as the basis for our robust parsing. This system uses a hierarchical frame parse representation. For our "event driven" dialogue manager, the developer creates a set of hierarchical frames, representing the information that the system and user interact about. Prompts are associated with fields in the frames. The dialogue manager decides what to do next from the current system context, not from a script or network.

2. SYSTEM OVERVIEW

The CU Communicator uses a DARPA Hub compliant architecture; the system is composed of a number of servers that interact with each other through the DARPA Hub. The system is composed of the hub and seven servers;

1. Audio – receives signals from telephone and sends to recognizer. Sends synthesized speech to telephone
2. Speech Recognition – Takes signals from audio server and produces a word lattice
3. Natural Language Processing – Takes word lattice from recognizer and produces the best interpretation
4. Dialogue Management – Resolves ambiguities in the interpretation, estimates confidence in the extracted information, clarifies with user if required, integrates with current dialogue context, builds database queries (SQL), sends data to NL generation for presentation to user, prompts user for information
5. Database Interaction – Receives SQL queries from Dialogue Manager, interfaces to SQL database and returns data, Retrieves live data from the web.
6. Text-to-Speech Synthesis – Receives word strings from NL generation, synthesizes them to be sent to the audio server.
7. Natural Language Generation – Generates text strings to be sent to Text-to-Speech.

Servers do not interact directly with each other, but pass frames to the hub. The hub acts as a router which send the frames to the receiving server. A simple script controls hub actions.

2.1 Speech Recognition

We are currently using the Carnegie Mellon University Sphinx-II system[4] in our speech recognition server. This is a semi-continuous Hidden Markov Model recognizer with a class trigram language model. The recognition server receives the input vectors from the audio server and produces a word lattice from which a single best hypothesis is picked and sent to the hub for processing by the dialogue manager.

[1] This work supported by DARPA through SPAWAR under grant # N66001-00-2-8906

2.2 Natural Language Parser

We use a newly modified version of the Phoenix[3] parser to map the speech recognizer output onto a sequence of semantic frames. A Phoenix frame is a named set of slots, where the slots represent related pieces of information. Each slot has an associated context-free semantic grammar that specifies word string patterns that can fill the slot. The grammars are compiled into Recursive Transition Networks, which are matched against the recognizer output to fill slots. Each filled slot contains a semantic parse tree with the slot name as root.

Phoenix has been modified to also produce an extracted representation of the parse that maps directly onto the task concept structures. For example, the utterance

"I want to go from Boston to Denver Tuesday morning"

would produce the extracted parse:

```
Flight_Constraint: Depart_Location.City.Boston
Flight_Constraint: Arrive_Location.City.Denver
Flight_Constraints:[Date_Time].[Date].[Day_Name].tuesday
                   [Time_Range].[Period_Of_Day].morning
```

2.3 Dialogue Manager (DM)

The Dialogue Manager controls the system's interaction with the user and the application server. It is responsible for deciding what action the system will take at each step in the interaction. The Dialogue Manager has several functions. It resolves ambiguities in the current interpretation; Estimates confidence in the extracted information; Clarifies the interpretation with the user if required; Integrates new input with the dialogue context; Builds database queries (SQL); Sends information to NL generation for presentation to user; and prompts the user for missing information.

We have developed a flexible, event driven dialogue manager in which the current context of the system is used to decide what to do next. The system does not use a dialogue network or a dialogue script, rather a general engine operates on the semantic representations and the current context to control the interaction flow.

The Dialogue Manager receives the extracted parse. It then integrates the parse into the current context. Context consists of a set of frames and a set of global variables. As new extracted information arrives, it is put into the context frames and sometimes used to set global variables. The system provides a general purpose library of routines for manipulating frames.

The system developer creates a task file that specifies the system ontology and templates for communicating about nodes in the hierarchy. The templates are filled in by the values in the frames to generate output in the desired language. This is the way we currently generate SQL queries and user prompts. An example task frame specification is:

```
Frame:Air
 [Depart_Loc]+
   Prompt: "where are you departing from"
   [City_Name]*
           Confirm: "You are departing from $([City_Name]).
                       Is that correct?"
           Sql: "dep_$[leg_num] in (select airport_code from
           airport_codes where city like '!%' $(and
           state_province like '[Depart_Loc].[State]' ) )"
   [Airport_Code]*
```

This example defines a frame with name Air and slot [Depart_Loc]. The child nodes of Depart_Loc are are [City_Name] and [Airport_Code]. The "+" after [Depart_Loc] indicates that it is a mandatory field. The Prompt string is the template for prompting for the node information. The "*" after [City_Name] and [Airport_Code] indicate that if either of them is filled, the parent node [Depart_Loc] is filled. The Confirm string is a template to prompt the user to confirm the values. The SQL string is the template to use the value in an SQL query to the database.

The system will prompt for all mandatory nodes that have prompts. All mandatory nodes must be filled for the frames to be complete. Users may specify information in any order, but the system will prompt for whatever information is missing until the frame is complete.

Our mechanism does not have explicit "user initiative" and "system initiative" modes. If the system needs information, then it asks for it. The system never requires that the user respond to the prompt. The user can respond with anything and the system will parse the utterance and set the focus to the resulting frame.

3. DATA COLLECTION

3.1 Metrics

Sites in the DARPA Communicator program agreed to log a common set of metrics for their systems. The proposed set of metrics was: Task Completion, Time to Completion, Turns to Completion, User Words/Turn, System Words/Turn, User Concepts/Turn, Concept Efficiency, State of Itinerary, Error Messages, Help Messages, Response Latency, User Words to Completion, System Words to Completion, User Repeats, System Repeats/Reprompts, Word Error, Mean Length of System Utterance, and Mean Length of System Turn.

3.2 University of Colorado Data Collection

The Center for Spoken Language Research maintains a dialup Communicator system for data collection. Users wishing to use the dialogue system can register at our web site [5] and receive a PIN code and system telephone number. To date, our system has fielded over 900 calls totaling approximately 11,500 utterances from nearly 300 registered users.

3.3 NIST Multi-Site Data Collection Effort

During the months of June and July of 2000, The National Institute of Standards (NIST) conducted a multi-site data collection effort for the nine DARPA Communicator participants. Participating sites included: AT&T, IBM, BBN, SRI, CMU, Colorado, MIT, Lucent, and MITRE. In this data collection, a pool of potential users were selected from various parts of the United States by a market research firm. The selected subjects were native speakers of American English who were possible frequent travelers. Users were asked to perform nine tasks. The first seven tasks consisted of fixed scenarios for one—way and round trip flights both within and outside of the United States. The final two tasks consisted of users making open-ended business or vacation trips that they might have recently taken or were going to take.

Subjects were instructed to obtain scenario instructions from a website maintained by NIST. For the fixed scenarios, a tabular template was used as shown in Table 1. After obtaining the task scenario, subjects then would read specific instructions related to the Communicator system they were about to dial. This sometimes included registering at the the local Communicator system website. Users then dialed a centralized computer at NIST known as the Communicator Data Collection System (CDCS). The CDCS forwarded the phone call to one of the nine Communicator systems and recorded the entire conversation. Individually, each Communicator system logged the individual system and user utterances along with a standard set of metrics. After each call, users were asked to fill out a short questionnaire which probed their satisfaction with using the system. Each of the nine Communicator sites received roughly 72 telephone calls during the period of the data collection. In the next section, we review the metrics which were obtained from the CU Communicator.

Preferred Airline:	Alaska Airlines
Flight	
Starting Date	Monday October 23, 2000
Starting Location	Las Vegas, Nevada
Destination	Seattle, Washington
Preferred Departure Time	Late Morning
In Seattle	
Rental Car:	No
Hotel:	No
Return Flight:	None (trip is one-way)

Table 1: Example task-scenario template.

4. NIST JUNE 2000 DATA ANALYSIS

4.1 Task Completion

A total of 72 calls from NIST participants were received by the CU Communicator system. Of these, 44 callers were female and 28 were male. Each scenario was inspected by hand and compared against the scenario provided by NIST to the subject. For the two open-ended tasks, judgement was made based on what the user asked for with that of the data provided to the user. In total, it was judged that 53/72 (73.6%) of the tasks were completed successfully. Table 2 summarizes the reasons for the 22 failed calls,

Category	# of Failures	Percent
Audio Server	8	42.1%
City Name Recognition	6	31.6%
Inattentive User	2	10.5%
Dialogue Manager	2	10.5%
Correct City, Wrong Airport	1	5.2%

Table 2: Summary of system failures by category

The majority of failures were due to problems with the audio server. The audio server used by the CU Communicator is the common audio server distributed as part of the core Communicator architecture. In several of the failures, we noted that the server would fail to end-point user speech causing control to not be returned to the dialogue manager.

The second source of call failures resulted from speech recognition errors in decoding of city names. We noted problems with confusable city-pairs such as Austin/Boston and Asheville/Nashville. Since users could not get their city names recognized, the call failed with the user often hanging up the telephone. The final source of call failures was due to "inattentive users". In these cases, the user confirmed incorrect information causing the system to not be able to retrieve the data needed to complete the task.

4.2 Speech Recognition Word Error Rate

A total of 1327 utterances were recorded from the 72 NIST calls. Of these, 1264 contained user speech. Since the CU Communicator system does not implement voice-based barge-in, one source of error was due to users who spoke before the recording process was started. Even though a tone was presented to the user to signify the time to speak, 6.9% of the utterances contained instances in which the user spoke before the tone. Since all users were exposed to several other Communicator systems which employed voice barge-in, there may be some effect from exposure to those systems. The CU Communicator recognition system uses separate male and female models running in parallel. Table 3 summarizes the word error rates for the system. Overall, the system had a word error rate of 26.0% (24.7% when truncated utterances are removed from the analysis). The WER for the male callers was 19.1% and that of female callers was found to be 27.8%. The usage of parallel decoders utilizing male and female acoustic models reduced the female WER from 34.0% to 27.8% (an 18.2% relative reduction in error).

Condition	Word Error Rate
SI models	29.8%
Male + female models	26.0%
Male + female models, Without speech before tone	24.7%

Table 3: CU Communicator Word Error Rates

User Survey Question	Mean (1)	CU (2)	CU (3)
Were you able to complete your entire task?	62.0%	77.8%	85.9%
It was easy to get the information that I wanted	2.9	2.1	1.9
I found it easy to understand what the system said	2.2	1.5	1.4
I knew what I could say or do at each point in the dialogue	2.5	1.8	1.6
The system worked the way I expected it to	2.9	2.3	2.0
I would like to use this system regularly	3.4	2.6	2.4
Average	2.8	2.1	1.9

Table 5: System results for NIST callers. Overall Mean for the 9 fielded Communicator systems (1), CU Communicator system (2) and CU Communicator excluding audio server failures (3). User satisfaction survey items 2-6 are encoded as "1" for Agree Completely through "5" Disagree Completely.

4.3 Evaluation Metrics

Table 4 summarizes results obtained from metrics derived automatically from the logged timing markers for the calls in which the user completed the task assigned to them. We can see that the average time to task completion is 260 seconds (4 minutes and 20 seconds). During this period there are an average of 19 user turns and 19 computer turns (37.6 average total turns). The average response latency was 1.86 seconds. This includes the 500 msec needed by the audio server to adapt the noise energy settings for endpoint detection and time needed for NL parsing, speech synthesis computation, dialogue management processing, and NL generation. The response latency also includes the time required to access the data live from the internet travel information provider.

The system reprompted the user an average of 2.4 times per dialogue session. However, we point out here that many of the reprompts occur when the user spoke before the tone causing a speech recognition error or audio server time-out.

Item	Min	Mean	Max
Time to Completion (secs)	120.9	260.3	537.2
Total Turns to Completion	23	37.6	61
Response Latency (secs)	1.5	1.9	2.4
User Words to Task End	19	39.4	105
System Words to End	173	331.9	914
Number of Reprompts	0	2.4	15

Table 4: Dialogue system evaluation metric

4.4 User Satisfaction Survey

After each telephone call, users were asked to fill out a short questionnaire related to their experience with interacting each Communicator system. Table 5 summarizes results obtained from the CU Communicator with respect to the mean (column 1) values obtained from all 9 fielded systems.

It is interesting to note that while 77.8% of the callers said thought they had completed the task successfully, the true completion rate was only 73.6%. For the remaining items (2-6) users were asked to answer the question on a 5-point Likert scale where a value of 1 represents "Strongly Agree", a value of 3 represents "Neutral", and a value of 5 represents "Strongly Disagree". We see here that the CU Communicator scored better than the mean values in all categories (lower scores are better).

5. SUMMARY

We have presented the metrics gathered by our system in the first common data collection for the DARPA Communicator project. We, along with the other sites in the program, will be using the data to try to determine which metrics are most correlated with user satisfaction and system performance.

6. REFERENCES

[1] http://fofoca.mitre.org

[2] Seneff, S., Hurley, E., Lau, R., Pao, C., Schmid, P., Zue, V., "Galaxy-II: A Reference Architecture for Conversational System Development," *Proc. ICSLP-98*, Sydney Australia, Vol. 3, pp. 931-934, 1998.

[3] Ward, W., "Extracting Information From Spontaneous Speech", *Proc. ICSLP-94*, September 1994.

[4] Ravishankar, M.K., "Efficient Algorithms for Speech Recognition". Unpublished Dissertation CMU-CS-96-138, Carnegie Mellon University, 1996

[5] http://communicator.colorado.edu