2007

Conquest – An Open-Source Dialog System for Conferences

Dan Bohus  
*Carnegie Mellon University*

Sergio Puerto  
*Carnegie Mellon University*

David Huggins-Daines  
*Carnegie Mellon University*

Venkatesh Keri  
*Carnegie Mellon University*

Gopala Krishna  
*Carnegie Mellon University*

See next page for additional authors

Follow this and additional works at: [http://repository.cmu.edu/compsci](http://repository.cmu.edu/compsci)
Conquest – an Open-Source Dialog System for Conferences

Dan Bohus, Sergio Grau Puerto, David Huggins-Daines, Venkatesh Keri, Gopala Krishna, Rohit Kumar, Antoine Raux, Stefanie Tomko

School of Computer Science
Carnegie Mellon University
{ dbohus, sgrau, dhuggins, vkeri, gopalakr, rohitk, antoine, stef }@cs.cmu.edu

Abstract

We describe ConQuest, an open-source, reusable spoken dialog system that provides technical program information during conferences. The system uses a transparent, modular and open infrastructure, and aims to enable applied research in spoken language interfaces. The conference domain is a good platform for applied research since it permits periodical redeployments and evaluations with a real user-base. In this paper, we describe the system’s functionality, overall architecture, and we discuss two initial deployments.

1 Introduction

Conducting applied spoken language interface research is generally a costly endeavor. Developing, deploying and maintaining real-world spoken language interfaces requires an existing infrastructure, a significant amount of engineering effort, and can greatly benefit from the availability of certain resources such as transcribed in-domain data.

In an effort to enable applied research and to lower this high cost of entry, we have developed ConQuest (Conference Questions) an open-source spoken dialog system that provides access to schedule and technical program information during conferences. We believe the conference domain has a number of good properties for applied research: it includes a number of tasks of different complexities, it provides regular access to a real-world user population; it permits periodical deployments and evaluations and therefore can provide a natural common evaluation task for the spoken language interfaces community.

The ConQuest system is constructed on top of the open, transparent and modular Olympus dialog system framework (2007), and can be easily reused across different conferences. To date, the system has been deployed in two conferences: InterSpeech 2006 and IJCAI 2007. Together with corpora collected from these deployments, the system is freely available for download (Conquest, 2007).

We begin by describing the ConQuest functionality in the next section. Then, in section 3 we provide an overview of the system architecture and discuss the development process. In section 4 we briefly discuss the two deployment efforts. Finally, in section 5 we discuss related work and draw a number of conclusions.

2 Functionality

As Figure 1 illustrates, ConQuest is a mixed-initiative spoken dialog system that provides access to schedule and technical program information during conferences.

Users can browse the schedule and find details about various papers or sessions of interest by providing identifying information, such as topics, session names, special events, paper titles, author names, specific dates and times, specific locations, or a combination thereof (e.g. turns 2, 4, 14). Additionally, the system also allows users to listen to current announcements and to cast their votes for a Best Paper Award (e.g. turns 10-17).

The interaction is mixed-initiative; at any point, the user may switch the current focus of the conversation (e.g. turn 8). ConQuest employs an adaptive information presentation strategy that allows
users to easily navigate the schedule (see turns 3, 5
and 15). The system uses a rich repertoire of error
recovery strategies to handle potential errors, in-
cluding several fall-back strategies (e.g. turn 13).

3 System Architecture

The ConQuest system was built using RavenClaw/
Olympus (2007), an open-source framework that
facilitates research and development in task ori-
ented conversational spoken language interfaces.
Olympus consists of a collection of components
for recognition, language understanding, dialog
management, language generation, speech synthe-
sis, etc., and the corresponding communication
infrastructure. To date, Olympus has been used to
develop and deploy a number of other systems
spanning different domains and interaction types (Bohus and Rudnicky, 2003).

A key characteristic of the Olympus framework
is a clear separation between the domain independ-
ent programs (or components) and domain specific
resources. This decoupling promotes reusability
and significantly lessens the system development
effort. In ConQuest, the authoring effort was fo-
cused on developing resources such as the lexicon,
language model, grammar, dialog task speciﬁca-
tion, etc. Some interesting, unanticipated engineer-
ing challenges we faced during development were
dealing with foreign names and accented charac-
ters and performing text normalization on various
fields (e.g. Alex Smith and Alexander Smith are
the same author), while at the same time ensuring
consistency between these various resources. Be-
low, we brieﬂy comment of each component and
the corresponding resources. Figure 2 provides a
top-level architectural view.

Speech Recognition. ConQuest uses a recogni-
tion server coupled to a set of parallel recognition
engines: two SPHINX-II decoders (Huang et al.,
1992) that use gender-speciﬁc acoustic models,
and a DTMF (touch-tone decoder). Each recogni-
tion engine uses class-based (e.g. paper titles, au-
thor names, etc.), state-speciﬁc trigram-language
models. We started with an initial language model
built using data collected with an early text-only
prototype. We then internally deployed a speech
based system, collected more data, transcribed it,
and used it to retrain the language models. The

Figure 1. A sample interaction with ConQuest
final language models used during the InterSpeech deployment were trained from on a corpus of 6350 utterances. The system operated with a lexicon of 4795 words, which included 659 lexicalized (concatenated) paper titles, and 1492 lexicalized author names, and 78 lexicalized session names. The pronunciations were generated using CMU Dictionary and later manually corrected.

**Language understanding.** The system uses the Phoenix (Ward and Issar, 1994) robust parser to extract concepts from the recognition results. A domain-specific shallow semantic grammar was developed and concatenated with a domain-independent grammar for generic expressions like [Yes], [No], [Date], [Time], etc.

**Dialog management.** ConQuest uses a RavenClaw-based dialog manager (Bohus and Rudnicky, 2003). We developed a dialog task specification for the conference schedule domain, expressed as a hierarchical plan for the interaction, which the RavenClaw engine uses to drive the dialog. In the process, the RavenClaw engine automatically provides additional generic conversational skills such as error recovery strategies and support for various universal dialog mechanisms (e.g. repeat, start-over, what-can-I-say, etc.)

**Backend/Database.** A backend agent looks up schedule information from the database (stored as a flat text file). The backend agent also performs domain specific pre-lookup normalization (e.g. mapping author names to their canonical forms), and post-lookup processing of the returned records (e.g. clustering papers by sessions). The database file serves as starting point for constructing a number of other system resources (e.g. language model classes, lexicon, etc.)

**Temporal reference resolution agent.** Apart from the database agent, the dialog manager also communicates with an agent that resolves temporal expressions (e.g. tomorrow at four p.m.) into canonical forms.

**Language generation.** ConQuest uses Rosetta, a template-based language generation component. The authoring effort at this level consisted of writing various templates for the different system questions and information presentation prompts.

**Speech synthesis.** ConQuest uses the Cepstral (2005) speech synthesis engine, configured with an open-domain unit selection voice. We manually checked and corrected pronunciations for author names, various technical terms and abbreviations.

4 Development and Deployment

The first development of ConQuest system was done for the Interspeech 2006 conference held in Pittsburgh, PA. The iterative development process involved regular interaction with potential users i.e. researchers who regularly attend conferences. Seven developers working half time participated in this development for about three months. An estimated one man-year of effort was spent. This estimate does not include the effort involved in transcribing the data collected after the conference.

Two systems were deployed at the Interspeech 2006 conference: a desktop system using a close-talking microphone placed by the registration desk, and a telephone-based system. Throughout the conference we collected a corpus of 174 sessions. We have orthographically transcribed the user ut-
terances and are currently analyzing the data; we plan to soon release it to the community, together with detailed statistics, the full system logs as well as the full system source code (Conquest, 2007).

Following Interspeech 2006, ConQuest was redeployed at IJCAI 2007 conference held in Hyderabad, India. The second deployment took an estimated two man-months: three developers working half-time for over a month. The significant parts of the second deployment involved incorporating scheduling data for the IJCAI 2007 and implementing two new requirements i.e. support for workshops and Indian English speech recognition. The IJCAI development had fewer iterations than the first effort. The two desktop systems set up at the conference venue collected 129 sessions of data. This data is currently being transcribed and will soon be released to the community through the Conquest website (Conquest, 2007).

Through these two deployments of ConQuest the system specifications have been refined and we expect the development time to asymptote to less than a month after a few more deployments.

5 Discussion and Conclusion

Our primary goal in developing ConQuest was to enable research by constructing and releasing an open-source, full-fledged dialog system, as well as an initial corpus collected with this system. The system is built on top of an open, transparent and modular infrastructure that facilitates research in spoken language interfaces (Olympus, 2007).

There have been a number of other efforts to collect and publish dialog corpora, for instance within the DARPA Communicator project. A more recent project, that operates in a domain similar to ConQuest is DiSCoH, a Dialog System for Conference Help developed by researchers at AT&T, ICSI and Edinburgh University, and deployed during the SLT-2006 workshop (Adreani et al., 2006). While their goals are similar, i.e. to enable research, DiSCoH and ConQuest differ in a number of dimensions. Functionality-wise, DiSCoH offers general conference information about the venue, accommodation options and costs, paper submission, etc., while ConQuest provides access to the technical schedule and allows participants to vote for a best paper award. DiSCoH is built using AT&T technology and a call-routing approach; ConQuest relies on a plan-based dialog management framework (RavenClaw) and an open-source infrastructure (Olympus). Finally, the DiSCoH effort aims to develop a richly annotated dialog corpus to be used for research; ConQuest’s aim is to provide both the full system and an initial transcribed and annotated corpus to the community.

The conference domain is interesting in that it allows for frequent redeployment and in theory provides regular access to a certain user-base. It should therefore facilitate research and periodical evaluations. Unfortunately, the dialog corpora collected so far using DiSCoH and ConQuest have been somewhat smaller than our initial expectations. We believe this is largely due to the fact that the systems provide information that is already accessible to users by other means (paper conference program, web-sites, etc.). Perhaps combining the functionalities of these two systems, and expanding into directions where the system provides otherwise hard-to-access information (e.g. local restaurants, transportation, etc.) would lead to increased traffic.

References


