Development of a toolkit handling multiple speech-oriented guidance agents for mobile applications

Sunao Hara, Hiromichi Kawanami, Hiroshi Saruwatari and Kiyohiro Shikano

Abstract In this study, we propose a novel toolkit to handle multiple speech-oriented guidance agents for mobile applications. The basic architecture of the toolkit is server-and-client architecture. We assumed the servers are located on a cloud-computing environment, and the clients are mobile phones, such as the iPhone. Huge amounts of servers exist on the cloud-computing environment, and each server can communicate with other servers. It is difficult to develop an omnipotent spoken dialog system, but it is easy to develop a spoken dialog agent that has limited but deep knowledge. If such limited agents could communicate with each other, a spoken dialog system with wide-ranging knowledge could be created. In this paper, we implemented speech-oriented guidance servers specialized to provide guide information for confined locations, and implemented a mobile application that can get information from the servers.

1 Introduction

Spoken dialog systems based on server-client-type architecture have been widely investigated. In particular, many of the systems have been used on the World Wide Web (WWW)[2, 4, 8]. Some voice search applications focusing on mobile usage have been created by Google [9], Microsoft [6], and Yahoo, among others.

Currently, location information can easily be measured using GPS on a mobile phone, and this function is expected to be efficiently used for mobile applications. Among applications that marry Automatic Speech Recognition (ASR) and location information, a speech-guidance system for local information is one of the most promising and useful. In this study, we developed a toolkit for multi-agent server-client spoken dialog systems. Each agent has only a small function as a guidance

Sunao Hara, Hiromichi Kawanami, Hiroshi Saruwatari, Kiyohiro Shikano
Graduate School of Information Science, Nara Institute of Science and Technology, 8916-5, Takayama-cho, Ikoma, Nara, Japan, e-mail: {hara,kawanami,sawatari,shikano}@is.naist.jp

system in a confined location, e.g. restaurant, museum, community center, railway station. However, if the agents can collaborate with each other, they can be viewed as a single dialog system with large-scale knowledge. In this paper, we propose a toolkit called *"Tankred on Rails (ToR)"*, and its client software called "*iTakemaru*." This software is based on "*Takemaru-kun*" [7] and its related systems [3]. Details and a demonstration are shown in later sections.

2 Dialog management toolkit for mobile applications

2.1 Speech-oriented guidance system for community center

The *Takemaru-kun* system has been implemented at the entrance hall of a community center since Nov. 2002 [7]. A newer system, *Kita-chan* and *Kita-robo*, have been in place at *Gakken Kita-ikoma* railway station since March 2006 [3]. The base system of *Takemaru-kun, Kita-chan* and *Kita-robo* had a major alteration made by Cincarek [1]. He rewrote the whole system in Ruby and maintained it as "a Dialogue System Toolkit written in Ruby", which is called *Tankred.* [1]

This toolkit consisted of several modules: Automatic Speech Recognition (ASR), Dialog Management (DM), Text-to-Speech (TTS), Internet browser, and Computer Graphic Agent. The ASR module used Julius [5]. The DM module included functions of domain classification and response generation [10]. The TTS module used was a commercially available version. The toolkit was used on Debian GNU/Linux 4.0 (etch) with Ruby 1.8.

2.2 Speech-guidance information service software

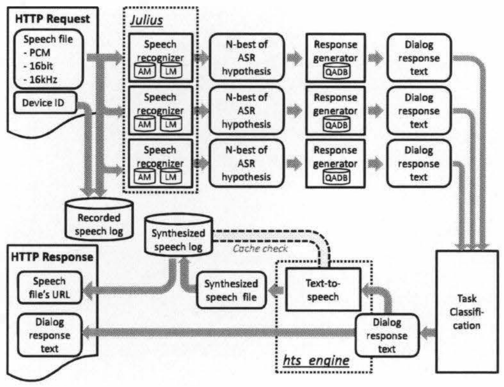
We extended the "Tankred" system for WWW service software, i.e. Saas. Core modules such as ASR, DM, and TTS were implemented on the server-side, while interface modules such as display of the text and agent, speech input, and speech output were implemented on the client-side. The server system was implemented by *Ruby on Rails*,[2] which is one of the most popular frameworks for WWW development. We called it "*Tankred on Rails (ToR)*".

A flowchart of the server system is shown in Fig. 1. The client system received the user's speech input through its microphone, and converted the speech input to a speech file (PCM, 16 bit, 16 kHz, WAV-RIFF). Then, the speech file was sent to the server system as an HTTP POST request. The server system recognized the speech using Julius and generated the N-best recognition result. The recognition result was sent to the DM modules, and it selected an appropriate response from each DM

[1] "*Tankred*" is an old German name meaning "the thinking adviser."

[2] http://rubyonrails.org/

Fig. 1 Flowchart of server system



module. The response was sent to the TTS module, which generated the speech response as a speech file with the file's URL. This response text and the speech file's URL were sent back to the client as an HTTP Response. The server recorded the speech file, recognition result, response result, an application ID generated in the client application, and its usage time.

The software used only HTTP protocols for the Internet. Therefore, the reversed HTTP proxy software, such as Apache httpd with mod_proxy, was used for the front-end of "ToR" to ensure high scalability.

The DM module was implemented with three types of dialog functions: a) simple Q&A with large vocabulary continuous speech recognition (LVCSR), b) simple Q&A with rule-based grammar speech recognition, and c) two stage Q&A. For the system answer selection, one module was selected based on the ASR score value that is the weighted summation of an acoustical likelihood and a linguistic likelihood. Type a) used the question-and-answer paired database, called QADB. First, all of the input sentences were compared to the question text in the QADB. Then, the most similar question was selected [10]. Finally, the answer text paired with the selected question was presented. Type b) simulated a mathematics calculator. It used a simple speech grammar for a mathematics calculator of four arithmetic operations; e.g. the system could accept "What is the sum of three plus four?", and answered "seven." Type c) simulated a WWW search task. It used the grammar for the first stage and the LVCSR for the next stage. In the first stage, the system accepted magic sentences, e.g. "Start WWW search," and in the second stage, the system accepted the word for WWW search.

The ASR module used was Julius [5], and it used the same acoustic/language models as "Takemaru-kun." For the TTS module, open-source software was used: *Open JTalk* [3] and its back-end system *hts_engine*. [11] [4]

[3] http://open-jtalk.sourceforge.net/

[4] http://hts-engine.sourceforge.net/

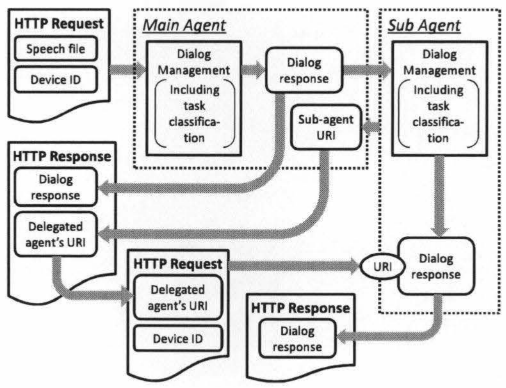


Fig. 2 Flowchart of server-to-server connection handling multiple agents

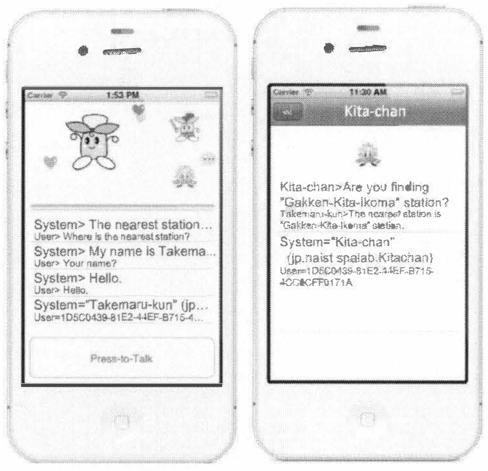
2.3 Speech guidance service with multiple agents

The system was extended to handle multiple agents. A flowchart of the system is shown in Fig. 2. The whole system was constructed with one main agent and several sub agents. The user of a client system talks with the main agent. At the same time, the main agent communicates with the sub agents, which is implemented as server-to-server communication. Responses of the main agent include information about which sub agent has been delegated a task from the main agent; therefore, the client can obtain more information from the sub agent. Server-to-server communication is assumed to be faster than client-to-server communication; therefore, this architecture is expected to be efficient for mobile, which has a lot of limitations, such as calculation speed, amount of memory, and connection speed. This architecture also has high affinity with cloud computing. The user talks to just one agent but can get more information, as if he/she were talking with a huge amount of agents.

2.4 iTakemaru: client software for mobile phone

The example client system was implemented with Objective-C, and it could be used on the iPhone 4S. It could generate HTTP requests and interpret HTTP responses as described in Fig. 1, and it could guide users through Takemaru-kun, which is placed at the community center [7]. We called it "*iTakemaru*". A screenshot of the client system is shown in Fig. 3. The system accepted user's speech as Press-to-Talk architecture; that is, the speech was recorded while the user pushed buttons on the interface.

"iTakemaru" could handle one main agent and multiple sub agents. In the current implementation, the agents were selected by users before using the system. It is



System> The nearest station...
User> Where is the nearest station?
System> My name is Takema...
User> Your name?
System> Hello.
User> Hello.
System="Takemaru-kun" (jp...
User=1D5C0439-81E2-44EF-B715-4...

Kita-chan>Are you finding "Gakken-Kita-Ikoma" station?
Takemaru-kun>The nearest station is "Gakken-Kita-Ikoma" station.
System="Kita-chan" (jp.naist spalab.Kitachan)
User=1D5C0439-81E2-44EF-B715-4CC8CFF0171A

Fig. 3 iTakemaru: client software handling multiple agents.The left-side figure shows the main agent "Takemaru-kun" and sub agents "SENTO Takemaru-kun" and "Kita-chan." Takemaru-kun is used at a community center and "Kita-chan" is used at a rail station. "SENTO Takemaru-kun" was originally used at an exhibition site developed as a special version of "Takemaru-kun". The image on the right shows that the sub agent proposed more detailed information than the main agent.

more convenient to select the appropriate agents automatically based on their usage location, but this implementation remains as future work.

In the left side of Fig. 3, the user used "Takemaru-kun" as the main agent and he/she mainly talked with "Takemaru-kun". The user selected two sub agents, and the main agent was connected to the sub agents in their background server systems. One is "Kita-chan [3]", which is a guidance system at *Gakken Kita-ikoma* railway station in Nara, Japan. The other is "SENTO Takemaru-kun", which was previously used as a guidance system for the "Heijo-Sento 1300th anniversary", which was held in Nara from July to October 2010.

In the dialog example in the figure, "Takemaru-kun" knows "*Gakken Kita-ikoma* railway station is the nearest station from his location" but does not know details about the station. The sub agent "Kita-chan" is a guidance system of the station and of course knows details about the station. "Kita-chan" always monitored the conversation between the user and "Takemaru-kun," and he notified the user by balloon icon that he could give more detailed information about their conversation topic than "Takemaru-kun." The right side of Fig. 3 is the screen for "Kita-chan," and the user could get more information from him about the railway station.

3 Conclusions

In this study, we developed a speech service software for speech-oriented guidance systems with multiple agents. This software will be used for the spoken dialog service of a mobile phone, and will also be distributed as a developer's toolkit. We hope that a lot of dialog agents will be produced by non-professionals, and they will be widely used as one of methods for transmitting information.

If the software is to be widely used in a lot of organizations, we should consider a sharing method for the speech, text, and other resources between the organizations. For example, investigation of the training method of the acoustic model for the distributed resources is an important issue.

Acknowledgements This work was partially supported by CREST (Core Research for Evolutional Science and Technology) at the Japan Science and Technology Agency (JST).

References

[1] Cincarek T (2008) Selective training for cost-effective development of real-environment speech recognition applications. PhD thesis, Nara Institute of Science and Technology
[2] Gruenstein A, McGraw I, Badr I (2008) The WAMI toolkit for developing, deploying, and evaluating web-accessible multimodal interfaces. In: Proceedings of ICMI 2008, pp 141–148
[3] Kawanami H, Takeuchi S, Torres R, Saruwatari H, Shikano K (2011) Development and operation of speech-oriented information guidance systems, kita-chan and kita-robo. In: Proceedings of APSIPA Annual Summit and Conference (APSIPA-2011)
[4] Lau R, et al (1997) WebGALAXY - integrating spoken language and hypertext navigation. In: Proceedings of Eurospeech-97, pp 883–886
[5] Lee A, Kawahara T (2009) Recent development of open-source speech recognition engine julius. In: Proceedings of APSIPA-ASC 2009, pp 131–137
[6] Levit M, Chang S, Buntschuh B, Kibre N (2012) End-to-end speech recognition accuraty metric for voice-search tasks. In: Proceedings of ICASSP 2012, pp 5141–5144
[7] Nisimura R, Lee A, Saruwatari H, Shikano K (2004) Public speech-oriented guidance system with adult and child discrimination capability. In: Proceedings of ICASSP 2004, pp I-433–I-436
[8] Nisimura R, Miyake J, Kawahara H, Irino T (2009) Development of speech input method for interactive voiceweb systems. In: Proceedings of Human-Computer Interaction, Part II, Springer, pp 710–719
[9] Schalkwyk J, Beeferman D, Beaufays F, Byrne B, Chelba C, Cohen M, Kamvar M, Strope B (2010) Google search by voice: A case study. In: Neustein A (ed) Advances in Speech Recognition: Mobile Environments, Call Centers and Clinics, Springer, chap 4, pp 61–90
[10] Takeuchi S, Cincarek T, Kawanami H, Saruwatari H, Shikano K (2008) Question and answer database optimization using speech recognition results. In: Proceedings of Interspeech 2008, pp 451–454
[11] Zen H, Nose T, Yamagishi J, Sako S, Masuko T, Black AW, Tokuda K (2007) The HMM-based speech synthesis system (HTS) version 2.0. In: Proceedings of ISCA SSW6, Bonn, Germany, pp 294–299