Spoken and Multimodal Bus Timetable Systems: Design, Development and Evaluation

Markku Turunen, Jaakko Hakulinen, Esa-Pekka Salonen, Anssi Kainulainen and Leena Helin

Speech-based and Pervasive Interaction group, TAUCHI, Department of Computer Sciences
University of Tampere, Tampere, Finland
{Markku.Turunen,Jaakko.Hakulinen,Esa-Pekka.Salonen,
Anssi.Kainulainen, Leena.Helin}@cs.uta.fi

Abstract

We present three speech-based bus timetable systems with different approaches. The first system had open user-initiative dialogues aiming at natural interaction. The user tests revealed several problems with this approach, and we developed solutions from multiple perspectives to overcome the problems. The second system focuses on task-oriented system-initiative dialogues, and contains a multimodal interface for smart mobile terminals. The third has a flexible mixed-initiative dialogue strategy, and offers multimodal integrated tutoring. All systems share a common agent-based architecture. We discuss solutions for design, development and evaluation of this kind of information systems.

1. Introduction

We have developed multiple bus timetable systems in various research projects. The timetable domain is practical, provides plenty of research challenges, and the results can be used in other similar information systems. Timetable services are suitable also for mobile and speech-based interfaces. Existing human-operated systems can be replaced with automatic systems, and new services can be produced for areas that are not possible or economically viable with human operators. These automated systems might be the only possibility for such special user groups as visually impaired people to access certain information.

We present solutions for spoken dialogue system development challenges based on the experiences from the iterative design, implementation and evaluation of Interact, Stopman and Busman systems. The functionality of Interact and Busman is similar to other speech-based timetable services, such as MALIN [1]. Users may query information such as bus routes and their timetables. Stopman serves bus stop specific timetables. All systems can be used with mobile and fixed line telephones. Example 1 demonstrates dialogues with the systems.

Each system has different user interface, and uses different approach to dialogue management. During the development of these systems we have faced several challenges and developed solutions to these using agent-based technologies. In the user tests of the Interact system we found that open user-initiative dialogue strategy based on human-human interaction failed to provide enough information to users and take the initiative when needed. The Stopman system provides a task-oriented, system-initiated interface while the Busman system aims for truly mixed-initiative dialogue management. We have also developed multimodal extensions to the Busman and the Stopman systems to provide guidance and graphical elements to the interface.

The systems have been developed in collaboration with various universities and companies. Stopman has been available to public since 2003. The systems are based on the common Jaspis architecture to be presented next.

2. Common system architecture

The timetable systems are built on top of the Jaspis architecture [2]. Jaspis is based on the agents – managers – evaluators –paradigm, where managers coordinate sets of evaluators to select appropriate agents that actually handle the tasks (e.g., dialogue decisions). Agents in Jaspis-based applications are compact software components. Managers represent the high level organization of tasks that are handled within spoken dialogue applications. All components share all the information via Information Storage.

Figure 1 illustrates the components of the timetable systems. All three bus timetable applications use this high-level structure. The systems contain standard Jaspis components (Communication, Dialogue, Presentation and Input managers), and an additional manager, the Database Manager (Task Manager in Interact), which is used to communicate with the timetable database.

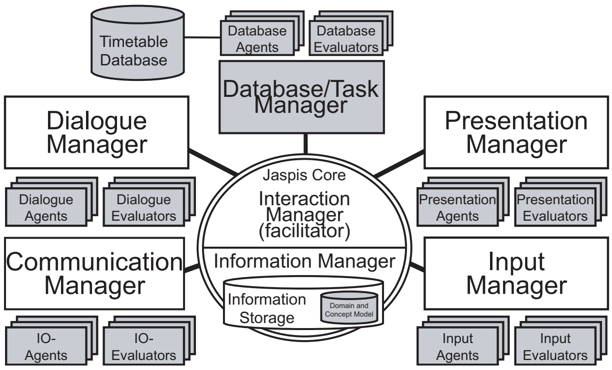

Figure 1: General system architecture.

The differences in the systems are in the structure of the Information Storage (modeling of dialogue and domain information), and in the agents that use that information to carry out interaction tasks. Next, the features of each system are presented. We go through the systems starting from the Interact system, after which the Stopman and the Busman systems, and their multimodal extensions are presented.

3. Natural interaction - the Interact system

The main purpose of the Interact project was to investigate natural interaction methods. Development took advantage of a corpus recorded from real calls to the human operated Helsinki local traffic information service. Adaptivity was the

	Interact:	Busman:	Stopman:
S_1	Welcome to the Helsinki automatic bus timetable service. How may I help you?	Welcome to Busman, the Tampere bus information service! How may I help you?	Welcome to Stopman. You can ask help by saying "tell instructions". Please tell the name of the bus stop, for example Central Square.
U_1	*When does bus number 6 leave from Hakaniemi to Hietalahti?*	*I'm leaving from Central Square, and I'm going to Hervanta.*	*Alexander Chuch.*
S_2	At 10:59.	Sorry, I didn't recognize that. Please tell me where you are leaving from.	Do you want to retrieve timetable for stop Alexander Chuch?
U_2	*When is the next bus from downtown to Munkkiniemi which leaves after 10:58?*	*From Central Square.*	*Yes.*
S_3	Unfortunately bus 44 does not go from downtown to Munkkiniemi.	What is your destination?	Wait while the time table for Alexander Chuch is retrieved. The time is 10:10. Today, the next buses leaving from bus stop Alexander Chuch are to Vehmainen number 5, now (list continues)… Give your command, for example tell instructions.
U_3	*When does leave the next, what lines go from downtown to Munkkiniemi?*	*Hervanta.*	*Select line.*
S_4	Sorry I don't understand.	From Central Square to Hervanta use lines 13 and 20. Do you want to hear timetables?	Please tell the line number. You can a list the lines by saying “list lines”.
U_4	*What lines go from downtown to Munkkiniemi?*	*Yes.*	Thirteen.
S_5	At 10:57.	From Central Square to Hervanta on weekdays at 11:15 bus in line 23. (list continues)…	Next number thirteen buses from Alexander Chuch are to Hermia in 1 minute, to Ikuri, in (list continues)…

Example 1: Sample dialogues with all three systems (Translated from Finnish).

main theme of the project, and this was explored in various ways, including the combination of rule-based and statistical approaches to interaction management [3]. The Jaspis architecture was used to integrate different approaches and techniques into a working spoken dialogue system.

The dialogue management strategy in the Interact system is user-initiative with clarification questions, i.e., the system asks for missing information. Dialogue state is modeled using the following concepts: speaker, dialogue act, topic, new information, goals and expectations. These concepts are present in each dialogue turn. Domain specific knowledge is modeled using a form. Dialogue agents model dialogue acts. The system contains dialogue agents for asking missing information, confirming possibly misunderstood concepts and answering to the queries.

We conducted a scenario-based user study with the Interact system. The user played the role of a visitor traveling around in Helsinki using buses and trams [4]. The study revealed serious usability problems with open user-initiative dialogues and brief system outputs. Most importantly, the system failed to provide sufficient feedback to the users in error situations, and take the initiative when needed. In addition, the recognition accuracy was unacceptable. These problems are demonstrated in Example 1.

The agent-based approach of the Jaspis architecture was used to overcome the problems. In order to provide more feedback to the users, we added a set of new agents which implemented more informative outputs than the main generator agent. By using compact agents it was possible to add new outputs and experiment with them without the need to modify the main generator agent, which was built by other project partners. We also found it very useful to utilize the dialogue and task knowledge from the shared Information Storage to improve speech outputs without modifying dialogue agents.

The experiment revealed problems with the corpus-based language model. Human-human interaction was used as the basis of the recognition grammar and input analysis components [5]. However, the language used by the users in the experiment was different from the language used in conversations with human operators: people used shorter sentences, terser style, and in overall different words and structures, such as non-grammatical sentences similar to universal commands [6]. We addressed this problem by adding a set of new agents to take care of those inputs that the original, corpus-based input interpretation components were not able to handle. In this way, we were able to include “universal commands” to the system without affecting the main natural language understanding agents. We have used this approach in the e-mail domain as well [2].

The final system also contained a special agent for detecting the topic of the utterance. This was done by using two recognizers, one having a limited task oriented grammar and the other having a more open grammar. The open grammar identifies whether the system can handle the overall topic of the utterance. The content of the utterance is then recognized using the task oriented grammar. The open grammar covers a large set of words relevant to the domain. The topic agent uses document maps, a methodology based on the Self-Organizing Maps algorithm [7] for mapping utterances to topics. Document maps were trained with conversations between customers and human operators [8].

The cases presented show how iterative, user-centered development of speech applications can be achieved with compact agents to overcome interface problems and add new functionality to existing spoken dialogue systems. Still, the resulting interface was found to be inefficient - the average number of user utterances for each task of the user study was five, although only one or two would have been needed. We addressed the problems of the Interact system in Stopman and Busman systems with different approaches. Most users can be satisfied with a robust, system-initiative interface. This is the approach used in the Stopman system. On the other hand, when natural interaction is used, the system must support truly mixed-initiative dialogues, and offer guidance to the users. We addressed these issues in the Busman system. In both systems, multimodal additions are used to make the interaction more efficient. Next, we present these systems and their extensions.

4. Task-oriented system-initiative dialogues – the Stopman system

The functionality of the Stopman system covers queries about timetables of single bus stops. The system is targeted for regular bus travelers: it provides stop specific timetables for each of the about 1200 bus stops in Tampere area. In addition to speech interface Stopman offers a smartphone interface.

The aim of the system is to satisfy most of the callers with the first timetable listing, as presented in Example 1. The most fundamental information is included in the initial listing, which explains the length of the prompt shown in the example. At the beginning of the call, the system requests the user to give a bus stop name. After this, the rest of the functions are available. Functionality includes navigation in the timetable, selection of specific bus lines, and specifying a certain time. For example, the user may request timetables for the bus line 30 passing the main library on Saturday after 16.25.

The Stopman system has been publicly available since August 2003. All calls to the system are recorded, and the most interesting ones have been analyzed. In addition, different evaluations have been done to the system to make it more efficient and pleasant to use. These range from a formal usability study to experiences collected from the users.

Concluding from the recorded dialogues it seems that the users are almost always happy with the first timetable listing. For many, the first arriving bus is all they need. Clearly, there is no use to inform users about the additional options by default and thus lengthen the dialogue. Guidance is available in the context-sensitive help messages, which are spoken when requested by the users, when the user is silent, or when there are errors present.

Analysis of recordings supports the findings from the Interact system regarding the interaction style. The interaction between computer and human is very different from human-human interaction. For example, users are not polite towards the system, and almost always the call is ended by hanging up. The usage also changes after the initial experiences. The average duration of calls diminished from almost three minutes to one minute in six months. There are two very likely explanations. The users learn to use the system more effectively, and after a while hoax calls disappear [9].

Concluding from the usage experiences, most problems relate to technical restrictions, such as lack of barge-in and problems with the voice activity detection. This makes it hard to get used to the rhythm of the interaction: to speak after the signal tone and not to keep long pauses between words. These problems are shared with the other systems. To overcome the problems, we developed an interactive tutor to teach the users how to speak to the system, among other topics. This will be presented in the next section.

Another problem concerns long listings and menus. For example, in the Stopman system users found them boring and complicated. The users easily forget what was said previously. To address this problem we developed a multimodal interface to be used with smartphones.

4.1. Using smartphones for multimodal interaction

In the timetable domain different lists and menus are an essential part of the systems outputs. To overcome the problems mentioned we have extended the Stopman system to use smartphone displays to make listings and menus more pleasant and efficient. Menus for item selection are presented also in graphical form. All system outputs feature a small menu where universal commands, such as “Quit” and “Cancel” are always presented. In addition, some actions, e.g., the selection of a certain bus line, can be made with menus. The interface is illustrated in Figure 2.

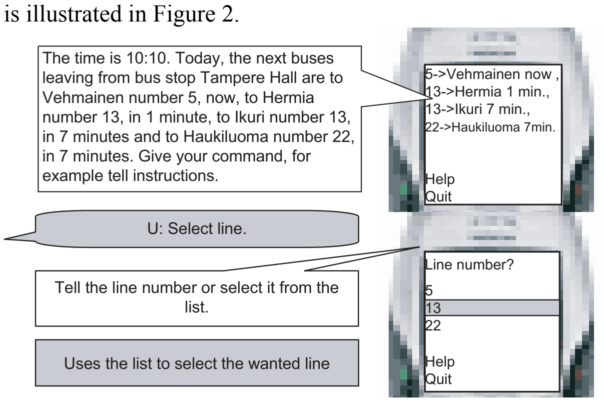

Figure 2: Smartphone interface.

The smartphone interface distributes parts of the dialogue management to various devices using VoiceXML descriptions [10]. In the Stopman system, both the speech-only telephone interface and the multimodal smartphone interface are generated from the same VoiceXML descriptions. Descriptions are generated by the dialogue agents, and modality specific information is embedded to the descriptions. Some of the resources, such as speech recognizers, can be shared between the devices. This general principle can be used in other applications and with other devices as well.

5. Towards truly mixed-initiative interaction – the Busman system

In the Busman system we addressed the challenge of user-initiative dialogues. The system allows same kind of queries as the Interact system, for example, “which bus goes to Hervanta” and “when does the next one leave”. Like other systems, it uses a form for dialogue control. The system is based on a general dialogue model that is released as an extension of the Jaspis architecture and has also been used in an e-mail application [2]. The model uses generic agents for dialogue management and natural language understanding and generation. Applications are constructed by defining concept and domain models in the system configuration file. All parts of the system operate by using the same concepts. Thus, there is no incompatibility between the system components, and the language used in the system is consistent. The general approach makes it possible to define concept and domain information in one place.

Most cases, such as general confirmations do not require domain specific knowledge and generic agents are used. However, specialized agents can be written as necessary. For example, one was developed when we encountered various problems caused by recognition errors in user tests. Analysis of errors revealed a common pattern and we were able to develop a specialized agent to remove the likely misrecognitions.

The second main feature of the Busman system is the agent-based modeling of initiative. In addition to asking for missing information Busman is capable of taking the initiative in other situations as well. Most importantly, the initiative is modeled in a way which allows adaptive, truly mixed-initiative dialogues to take place. Each dialogue agent has an initiative attribute, which indicates the level of initiative the agent implements. In the current implementation one dialogue evaluator is used to give preference to user-initiative agents when the interaction proceeds normally, and system-initiative agents when errors are present. The behavior is shown in Example 1, where the system changes the dialogue strategy to system-initiative after speech recognition errors, while the Interact system is only capable of asking the lacking information needed for the current task.

While the improvements presented made the interface more efficient, we were still facing the problem of how to guide the users to use the system more efficiently. We addressed this problem with integrated multimodal tutoring.

5.1. Integrated and multimodal tutoring

In our opinion, the user and his capabilities as a learner are not used enough in spoken dialogue applications. It is also a known phenomenon that the users do not read manuals [11]. In the users tests of the Interact system we learned that the users quickly bypass general help prompts. To overcome this issue we came up with the idea of integrated, context-sensitive tutoring. In the Busman application the tutor is implemented as a multimodal application to make the capabilities of the system visible and help the user to learn how to use the system [12]. In particular, we have found that integrated tutoring is a good way to familiarize the user to the rhythm of the interaction. As reported in the previous section, this is a real problem when users start using spoken dialogue systems.

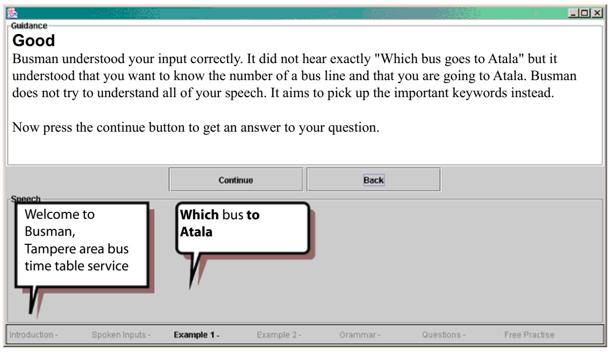

Figure 3: Multimodal tutoring of Busman

As illustrated in Figure 3, the multimodal tutor comments and explains the speech-based interaction in real context. The tutor monitors Busman, visualizes it and provides guidance accordingly. Busman works as it does without the tutor.

6. Discussion and conclusions

We have presented three timetable systems with different interaction management approaches. Concluding from the tests with the Interact system we found that it is not preferable to use human-human data, as such, to design spoken dialogue systems. This was shown in the inefficiency of the initial design of Interact. The system-initiated Stopman system is able to serve most users with a single prompt and usage changes towards even briefer interaction, which is shown in the shortening of call durations over time. The Busman system, on the other hand, solves the inefficiency by using software tutors that familiarize users to the system and by using a truly mixed-initiative user interface.

Although the data from usage is not comparable between systems, we have demonstrated how the problems of open-ended user-initiative dialogues can be solved using agent-based technologies as a part of the user-centered iterative development process. In particular, we have shown how robust interaction can be achieved using both system-initiative and mixed-initiative approaches, and how multimodality can be used to improve usability when speech-only interfaces are inefficient. The technologies that we have developed are based on the same agent architecture, and implemented as add-ons to the existing systems. Most of the solutions are generic, and only minor changes, if any, are needed to the existing systems.

References

[1] Dahlbäck, N. & Jönsson, A. "Knowledge Sources in Spoken Dialogue Systems", *Proc. of Eurospeech'99*, 1999.

[2] Turunen, M., Hakulinen, J., Räihä, K.-J., Salonen, E.-P., Kainulainen, A. & Prusi, P. "An architecture and applications for speech-based accessibility systems", *IBM Systems Journal, Vol. 44, No. 3, 2005*. (to appear).

[3] Jokinen, K., Kerminen, A., Kaipainen, M., Jauhiainen, T., Wilcock, G., Turunen, M., Hakulinen, J., Kuusisto, J. & Lagus, K. "Adaptive Dialogue Systems - Interaction with Interact", *Proc. of the SIGdial Workshop on Discourse and Dialogue*, 2002.

[4] Turunen, M. "Jaspis – A Spoken Dialogue Architecture and its Applications", *PhD dissertation, Department of Computer Sciences, Report A-2004-2*, 2004.

[5] Jauhiainen, T. "Using existing written language analyzers in understanding natural spoken Finnish", *Proc. of Nodalida '01*, 2001.

[6] Rosenfeld, R., Zhu, X., Shriver, S., Toth, A., Lenzo, K. & Black, A. W. "Towards a Universal Speech Interface", *Proc. of ICSLP 2000,* 2000.

[7] Kohonen, T. *Self-organizing maps*. Berlin; Heidelberg; New-York: Springer; 1995. ISBN: 3-540-58600-8.

[8] Lagus, K. & Kuusisto, J. "Topic identification in natural language dialogues using neural networks", *Proc. of SIGdial Workshop on Discourse and Dialogue*, 2002.

[9] Johnsen, M. H., Svendsen, T., Amble, T., Holter, T. & Harborg, E. "TABOR - a norwegian spoken dialogue system for bus travel information", *Proc. of ICSLP 2000,* 2000.

[10] Salonen, E-P., Turunen, M., Hakulinen, J., Helin, L., Prusi, P., & Kainulainen, A. "Distributed Dialogue Management for Smart Terminal Devices", *to appear in proc. of Interspeech 2005,* 2005.

[11] Carroll, J. M. & Rosson, M. B. "Paradox of the Active User", in Carroll John, M. (Ed.), *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction*. Cambridge, MIT Press, MA, 1987.

[12] Hakulinen, J., Turunen, M. & Salonen, E-P. "Software Tutors for Dialogue Systems.", *to appear in proc. of Text, Speech and Dialogue (TSD),* 2005.