# Tell Me How to Cook

**Itika Gupta** and **Sabita Acharya**
Department of Computer Science
University of Illinois at Chicago, Chicago, Illinois
igupta5@uic.edu, sachar4@uic.edu

## Abstract

In this paper, we present a system initiative spoken dialogue system that helps the user while cooking a recipe. Similar to cookbooks, our system provides details like ingredients, preparation time, servings, procedure, difficulty level and some useful tips for a recipe. In addition, our system also provides the user with an option to select a recipe based on time or difficulty level and customize it based on the number of servings. Unlike existing cooking dialogue systems, our system aims to provide companionship to the user while cooking and thus, waits for the user to finish each step before proceeding. To handle the erroneous conditions caused by imperfect Automatic Speech Recognition (ASR) systems, we allow the user to provide text input after multiple incorrect ASR hypothesis. We make use of recipes available online in the USENET cookbook as our dataset. The evaluation of the system shows that irrespective of the amount of information displayed in the Graphical User Interface (GUI), users are easily distracted when a dialogue system has a visual interface. Hence, for domains that require the user to concentrate on a particular task, having a GUI might not be a good idea. Our results also show that the users do not prefer a cooking dialogue system to pause after each instructional step.

## 1 Introduction

Spoken dialogue systems (SDSs) are computer programs that take speech as input and produce speech as output. Many SDSs has been developed for the purpose of accomplishing a task by conversing with the user. Initially, they were developed to help the user with simple tasks such as train schedule information and air travel information (Hempel, 2008). But nowadays, they are being used for various complex tasks such as tutoring systems (Boyer et al., 2010), in-car applications (Kuhn et al., 1999) (Pellom et al., 2001), scheduling appointments (Hodson, 2014) and personal assistants e.g. Google Now, Cortana and Siri.

Until now, there is a little prior work that addresses the problem of dialogue systems in cooking domain. Most of the existing systems assume that user would like to have all the information before hand. While this assumption simplifies the processing, it fails to model a real time interaction a user would have with the chef. It also requires the user to go back and forth between the system and the cooking task as it is hard to remember all the information. Also, the existing systems are distractive. They require users hands and eyes as a mode of interaction. Thus, they limit the user's ability to multi-task while using the system.

Therefore, in this paper we introduce our system which tries to model a more natural human to human interaction and provides companionship while cooking. Our system is primarily speech based. As speech is the most natural modality for human to communicate with each other and other interactive agents, it is less distractive and allows the user to focus on the primary task of cooking. But since the existing ASR systems have high word error rate, we also incorporate text input in case the system fails to understand the user's speech even after multiple

trials.

Apart from companionship and less distractive speech based interface, our system provides the user with four options at the beginning of the application. The four options consist of: (1) select a recipe based on the recipe name, (2) ask for the list of recipes, (3) ask for the list of recipes based on preparation time, and (4) ask for the list of recipes based on difficulty level. Later during the conversation, the system asks the user if he/she wants to customize the recipe based on the number of servings before providing the amount of ingredients. Customization is possible due to the separate column of amount present in the database. Recently, techniques have been developed to extract the structured information from the recipes using conditional random fields which can be efficiently used by the dialogue systems [1].

We investigate whether our speech based system, which waits for the user to complete each step during the procedure, would be a better companion to learners than the current state-of-the-art systems. Our system is built on two hypotheses: (1) If users are provided with GUI, they tend to focus more on visual content rather than trying to understand user's spoken utterances. Thus, having speech as primary mode of communication can be less distractive. (2) Users will appreciate a dialogue system which waits for them while cooking over the non-interactive ones i.e. the ones which recite the whole recipe at one go.

The remainder of the paper is structured as follows: In Section 2, we present an overview of the existing dialogue systems in the cooking domain. Section 3 presents the foundation for the kind of data our dialogue system handles and the corpus used for recipe information. In Section 4, we provide detailed description about each component of our dialogue system pipeline. Section 5 presents the experimental setup and its results and finally conclusions are drawn in Section 6.

## 2 Related work

Apart from the popular applications like train or air-travel information systems (Cheyer and Martin, 2001), ticket reservations (Erdogan, 2001), and home banking [2], in the recent years, spoken dialogue systems have also been developed for some other tasks like survey interviewing (Johnston et al., 2013) and guiding tourists (Bonneau-Maynard and Devillers, 1998) (Misu et al., 2010). However, not much research effort has been oriented towards developing a spoken dialogue system for cooking application. Wasinger (2001) developed a home cooking assistant based on a multi-modal design. It was a system initiative system that could function using three different sets of modalities: speech only; touch and visual only; speech,touch and visual. Laroche et al. (2013) also built a system initiative multimodal dialogue system using VoiceXML framework. This application was developed for tablet devices and could comprehend swiping and tilting actions. The dialogue system developed by Pardal et al. (2011) was based on Olympus framework and used ontologies to model and conceptualize domain information. It was a user initiative system that could also play relevant video clips during conversation. A cooking assistant application for smartphone and tablet devices was developed by Ulrich et al. (2013) could also answer questions during cooking. Unlike the earlier approaches, the authors made use of shallow natural language techniques like part-of-speech tagging , morphological analysis, and sentence boundary detection on the domain corpus, which allowed them to quickly access and utilize domain knowledge instead of modeling everything by hand.

Even though we can observe the gradual development in the methodologies that have been used in developing these systems over time, all of these earlier systems just read out loud the ingredients and cooking instructions step-by-step. In such a case, the user has to either listen to the entire recipe steps at once or pause the system in between and resume it after they complete a particular step. This makes such systems no different than a youtube video which just spills out information. In contrast, our application waits for the user to accomplish particular steps and resumes when they are ready to proceed.

## 3 Data

Two types of data were essential for the development of our application : a) a corpus of recipes

[1] http://open.blogs.nytimes.com/2015/04/09/extracting-structured-data-from-recipes-using-conditional-random-fields/

[2] http://www.nuance.com/index.htm

**Laura:** Okay lets work on our dough. First thing I am going to do is dissolve my one packet of yeast in one quarter cup of nice warm water. If its too hot, it slows the yeast. I am getting a pinch of sugar which is yeasts favorite food.
**Julia:** Good.
**Laura:** Okay now I am going to let that nice and happy over there. Now I need my mixing bowl and here I have two cups of again, warm water. Anytime you put liquid into it, it needs to be about the same temperature.
**Julia:** Yes.
**Laura:** I am going to 3 table spoons of vegetable shortening which helps to give my bagels the texture.
**Julia:** Oh yes they were softy chewy.
**Laura:** Okay I am going to put into this 1 table spoon of salt and I need 1 table spoon of sugar and color and flavor.

**Figure 1:** Example of the conversation between the learner and the chef.

and b) a source to provide an insight about the kind of conversation that takes place between two people; where one is providing guidance to the other about cooking. We obtained corpus of recipes from an online source called USENET CookBook (Reid, 1988). The USENET CookBook contains over 500 recipes for main dishes, desserts, and lunch and has been used as the source of recipes information for most of the existing cooking dialogue applications (Wasinger, 2001) (Laroche et al., 2013). A typical recipe consists of the following details:

- Recipe name
- Short and long description of the recipe
- Number of servings
- Ingredients
- Procedure
- Additional notes
- Difficulty level and time required to cook the recipe

In order to model the dialogue system for our application, we collected the videos where the chef talks with the learner about the steps involved in cooking a particular dish. In particular, we referred to the YouTube channels of two television shows: Baking with Julia Child [3] and Trophy Cupcakes [4]. The videos from Cooking Matters website informed us about the terminologies that are frequently used by professional chefs while giving instructions [5]. We transcribed around 10 minutes of conversation from the Baking with Julia Child show. The transcript (a small portion is shown below) consists of 20 turns of conversation between the chef and the learner, with the chef speaking around 450 words compared to 85 words spoken by the learner in the entire duration. Figure 1 shows an example of the conversation between the learner and the chef while making bagels.

By analyzing these conversations, we were able to make the following observations:

- While giving instructions about cooking a recipe, the instructor drives the entire conversation while the learner acknowledges at each step. This helped us in deciding that a system initiative approach is appropriate for this domain.
- The instructor usually begins the conversation by giving some additional information about the recipe. The information include some do's and don'ts for the recipe or the instructors previous experience while making it. Since our recipe corpus already had similar notes, we decided to use them at the beginning of the conversation.
- We also realized that amount of servings is an essential piece of information because it helps in determining the quantity of ingredients required. This motivated us to include an option for allowing the user to customize the number of servings.

[3] Croissants with Esther McManus https://www.youtube.com/watch?v=xps55MQ2Vgo

[4] Trophy Cupcakes on Martha Stewart https://www.youtube.com/watch?v=k_877FkaGLk

[5] Cooking Matters: http://cookingmatters.org/

## 4 Methodology

In this section, we first explain our system architecture in detail and then, we provide a list of tools we used to accomplish the task. And finally, we provide a step-by-step flow of our system and techniques used to achieve the same.

### 4.1 Architecture

Our application follows the general spoken dialogue system architecture as shown in figure 2. The Speech Recognition (SR) and Language Understanding (LU) components recognize and interpret the user's speech input with a certain confidence score. For cases when the SR is not able to understand the user even after several attempts, a text box gets activated in the screen and the user is allowed to give a text input. The interpreted text is then sent to the dialogue manager. Dialogue manager consists of two subcomponents: dialogue control and dialogue context model. The dialogue context model has access to the recipe database and keeps track of the contents that have been shared with the user. Given the information in the dialogue context model, dialogue control determines the next system action. System response is generated by using templates, which is finally converted to speech by the text to speech component.

### 4.2 Tools

Based on our experience with the dialogue system tools that we had explored, we decided to use VoiceXML for developing our application because its speech recognition component was excellent and the tool itself was easy to use. On further research, we came across another tool developed by Voxeo called Aspect Customer Experience Platform (CXP) toolkit. Unlike VoiceXML, this tool provides a drag-and-drop functionality that saves the developer from encoding the rules in XML. However, since CXP Developer does not have a detailed documentation, we had to spend a fair amount of time in figuring out how to use the tool. In particular, establishing a database connection for the application was very difficult and thus, we had to move to JAVA for developing our application. For Text-To-Speech (TTS) synthesis and Automatic Speech Recognition (ASR), we use the J.A.R.V.I.S. Speech API [6]. J.A.R.V.I.S. provides a simple and efficient way to access the speech engines created by Google. The recipes from our toy corpus are stored in a MySQL database. Figure 3 shows an example of the type of interaction that takes place between the user and the system.

### 4.3 Approach

We created four tables in a database, one each for recipe, ingredients, notes, and procedure. These tables were then manually populated with the information extracted from our recipe corpus. Since our system allows user to customize servings and make selection based on the time requirement, we have a separate column for the quantity of ingredients needed and the amount of time required before, during, after a preparing the recipe. Since we were working only on a toy corpus, we did not explore automated methods for populating the database. However, this process can be automated by using scripts to scrape the contents from web pages and automatically insert contents in the database. Recently, techniques have also been developed to extract the structured information from the recipes using conditional random fields, which can be efficiently used by the dialogue systems.

Our application typically begins by asking the user to select one the following four methods for getting recipe details: 1) select a recipe based on recipe name, 2) ask for the list of recipes, 3) ask for the list of recipes based on preparation time, and 4) ask for the list of recipes based on difficulty level. The detailed steps involved in each of these options are shown in figure 4. The user responds by saying the method number they are interested in. The SR can recognize the users response as either a number (e.g. 1) or a word (e.g. one ). In order to deal with both the cases, we make use of a mapping between numbers and words. In addition, we also collected a list of positive and negative words and made use of regular expressions in order to identify the variations of positive and negative acknowledgments.

## 5 Experimental design

Our dialogue system was evaluated with 7 subjects, all of whom were graduate students. All of them

[6] https://github.com/lkuza2/java-speech-api

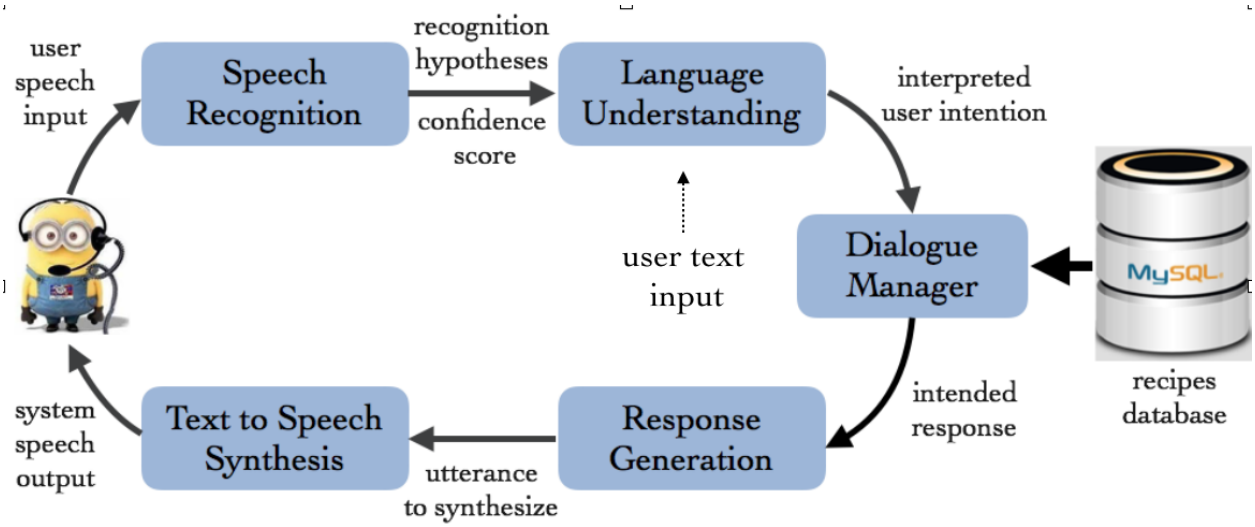


**Figure 2:** Basic spoken dialogue system srchitecture.

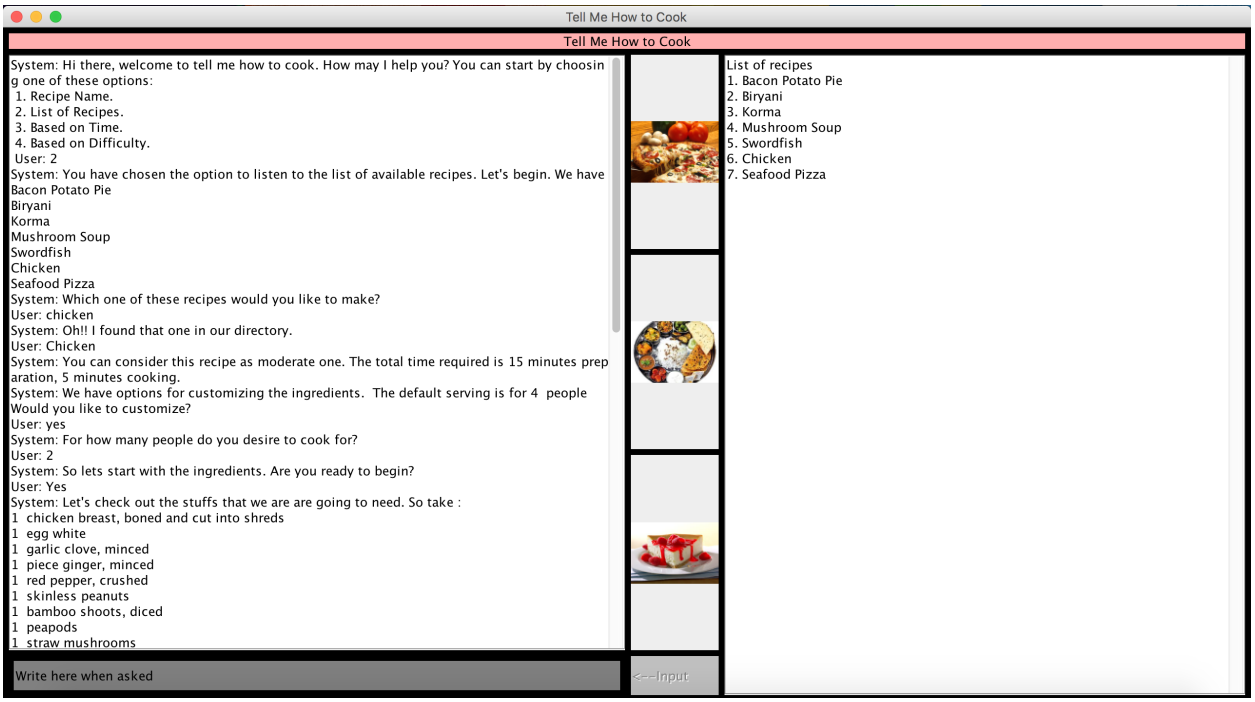


**Figure 3:** The spoken dialogue system for cooking.

were non-native speakers. The test subjects had no previous experience with the proposed system. The evaluation task included two parts. First, the subjects were asked to use the system as long as they want. On an average, they spent 10 minutes interacting with the system. They were optionally asked to provide a subjective feedback about the system. Second, they were asked to fill a questionnaire after they finished using the system. The questionnaire consisted of questions concerning the performance of the system which was rated on a five point Likert scale (1-Strongly Disagree, 2-Disagree, 3- Neutral, 4-Agree, 5-Strongly Agree) . Due to time limitations, we were not able to perform an extensive evaluation of the application. Questions used for evaluation are provided in figure 5.

**CASE 1:**

- User provides a recipe name.
- System checks if it is present in the database.
- System prompts the user until a valid selection.
- Allow text input after multiple wrong ASR attempts.
- Gives information about the difficulty level and time requirements.
- Gives user an option to customize the serving size.
- Provides information about the ingredients based on the serving size.
- After every three ingredients, asks the user for acknowledgement.
- After ingredients, system asks the user if they are ready to proceed with the procedure.
- The system waits until users affirmation.
- Provides information about the procedure step-by-step.
- After each step asks if the user was able to understand.
- If yes, asks if the user wants it to wait until he/she finishes the current step. (Companionship: This feature distinguishes our application from the existing ones. Unlike the existing systems, our application waits for the user until they are ready to proceed to the next step.)

**CASE 2:**

- System reads out all the recipes present in the database.
- Asks the user to select a particular recipe.
- Follows steps from Case 1.

**CASE 3:**

- System asks the user to select a difficulty level.
- Filters out only those recipes that have the same difficulty level as selected by the user.
- Asks the user to select a particular recipe.
- Follows steps from Case 1.

**CASE 4:**

- System asks the user about his time availability.
- Sums the time required before, after, and during preparation of the recipe from the database.
- Informs the user about the recipes that can be prepared within that time frame.
- Asks the user to select a particular recipe.
- Follows steps from Case 1.

**Figure 4:** Task flow for option/case 1, 2, 3 and 4.

## 6 Results and discussion

Figure 6 shows the average response of the subjects to the evaluation questions. We can see that the average values for Q2 and Q14 are below 3 (3 represents neutral) while the average values for other questions vary between 3.5 and 5. While most of the participants were satisfied with the performance of the TTS, they did not consider ASR to be good enough. This result was expected because even while developing the application, we had a lot of trouble in mak-

Q1. The system was easy to understand. (Text-To-Speech Performance)

Q2. The system understood what you said. (Automatic Speech Recognition Performance)

Q3. Having a Graphical User Interface (GUI) was helpful. (GUI helpfulness)

Q4. The systems speech was natural. (Naturalness of the dialogue)

Q5. It was easy to follow the steps provided by the system and accomplish the task. (Task Ease)

Q6. You accomplished the task of cooking. (Task completion)

Q7. The pace of interaction with the system was appropriate. (Interaction Pace)

Q8. You knew what you could say at each point of the dialogue. (User Expertise)

Q9. The system worked the way you had expected it to. (Expected Behavior)

Q10. From your current experience with using the system, you think you would keep using the system regularly for learning new recipes. (Future Use)

Q11. You frequently encountered problems while using the system. (Number of interaction problems)

Q12. The system was able to handle the errors. (Error handling adequacy)

Q13. You are satisfied with the overall performance of the system. (User satisfaction)

Q14. You liked the wait/pauses between the recipe steps. (Companionship hypothesis)

**Figure 5:** Questions used for evaluation task.

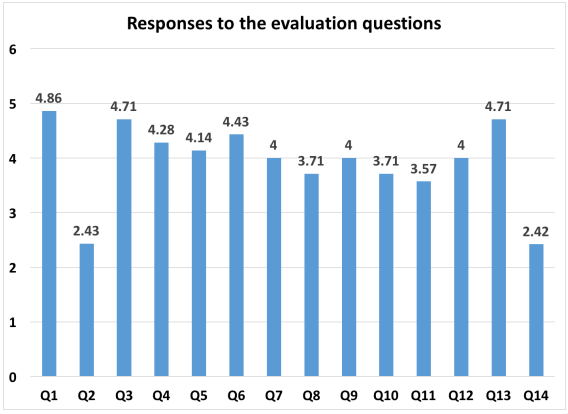


**Figure 6:** Responses to the evaluation questions.

ing ASR understand simple words like two, yes and no. Due to the poor performance of ASR, we were not able to make the system only speech based and had to include an option for giving text input.

Another interesting observation that can be made from these results is that even though our application is mostly speech based and the GUI only displays history of the conversation, most of the participants agree that the GUI was helpful. They were also observed looking at the GUI for major part of their conversation. This verifies our first hypothesis that visual interfaces are usually distracting and thus, the speech based dialogue systems can be one way to lower the distraction (provided the performance of the ASR system is improved).

The low value for Q14 indicates that the users did not want the system to wait for them after giving instructions about each step of the recipe procedure. Even though this result violates our second hypothesis, we need to take into consideration that this system is meant to assist the users in real time. While the user is actually cooking, it might be helpful if the system allows them to complete a particular step before giving details about the next step. But since the evaluations were not performed in a real cooking scenario, the subjects might have disliked the systems interruption after each step. Therefore, we will need to perform a real time evaluation to verify if our assumptions are correct.

There are several limitations of our approach. First, due to the poor performance of the ASR, we have restricted the number of spoken user responses that the system allows. In addition, since we are using a rule-based approach, the responses given by the system remains same in every iteration. Language generations techniques might have helped in introducing variations in responses; but due to time limitations, we did not explore that option. Second,

the sample conversations that we referred to for obtaining domain knowledge might not be representative of an actual speech only dialogue system. Since both of the dialogue participants in the YouTube videos were able to see each other, they have an advantage of using gestures or facial expressions to express emotions such as acknowledgement. However, for a completely speech based dialogue, the kind of conversation might have been different. Third, our system does not handle the cases when the users dialogue act differs from what the system expects at that instance. For example, if the system is expecting the user to say the name of the ingredient and the user says the amount of servings or the name of the recipe, the system is not able to comprehend that information. Finally, since our system frequently asks the user for acknowledgement, they might get frustrated. This was also observed while performing the experiments.

## 7 Conclusion and future work

In this paper, we have presented a system initiative spoken dialogue system that helps in cooking a recipe. The system allows the user to customize the recipe based on servings. They can also choose a recipe based on time or difficulty level. Our aim was to build a more natural and interactive model which can act as a companion to the user while cooking. The system was evaluated by 7 subjects. The evaluation results showed that users do get distracted with the presence of GUI and thus, speech based dialogue systems can help to maintain an undivided attention of the user. Also, users do not prefer pauses after each instructional step in a cooking dialogue system.

Our evaluation results show that users do get distracted with the presence of GUI and thus, speech based dialogue systems can help to maintain an undivided attention of the user.

In future work, we intend to experiment with mixed initiative strategy and evaluate whether mixed initiative or system initiative is better for cooking domain. We also intend to include an option of selecting a recipe based on ingredients. Finally, we plan to use resources such as Cooks Thesaurus [7], a cooking encyclopedia, in order to suggest substitutes for a given ingredient.

[7] http://www.foodsubs.com/


## Acknowledgments

We would like to thank Abhinav Kumar, Sanket Gaurav, Joyti Arora, Manas Nyati, Prakash Paudyal, Arayna Sharma and Mehrdad Alizadeh for evaluating the system and providing the helpful feedback.


## Appendix

We both collaborated on the project and report equally. We started by exploring the dialogue system toolkits such as Aspect Customer Experience Platform (CXP) and Opendial. We faced difficulty in connecting our application with the database in CXP. Therefore, we moved on to using the J.A.R.V.I.S Speech API and tried to make it work together. Once we had all the components working separately, Itika Gupta started building the Graphical User Interface (GUI) and populating the database and Sabita Acharya started implementing the different modules of the dialogue system. Then, we both sat together and brought all the bits and pieces together to build the final application which included adding more features like customizable servings and selection based on time and difficulty level.

## References


H Bonneau-Maynard and L Devillers. 1998. Dialog strategies in a tourist information spoken dialog system. *SPECOM98*.

Kristy Elizabeth Boyer, Robert Phillips, Amy Ingram, Eun Young Ha, Michael Wallis, Mladen Vouk, and James Lester. 2010. Characterizing the effectiveness of tutorial dialogue with hidden markov models. In *Intelligent Tutoring Systems*, pages 55–64. Springer.

Adam Cheyer and David Martin. 2001. The open agent architecture. *Autonomous Agents and Multi-Agent Systems*, 4(1):143–148.

Hakan Erdogan. 2001. Speech recognition for a travel reservation system. *IC-AI 2001*.

Thomas Hempel. 2008. *Usability of Speech Dialog Systems: Listening to the Target Audience*. Springer Science & Business Media.

Hal Hodson. 2014. Meet your ai assistant. *New Scientist*, 224(3000):17.

Michael Johnston, Patrick Ehlen, Frederick G Conrad, Michael F Schober, Christopher Antoun, Stefanie Fail, Andrew Hupp, Lucas Vickers, Huiying Yan, and Chan Zhang. 2013. Spoken dialog systems for automated survey interviewing. In *the proceedings of the 14 th*

*Annual SIGDIAL Meeting on Discourse and Dialogue*, pages 329–333.

Thomas Kuhn, Akhtar Jameel, Matthias Stümpfle, and Afsaneh Haddadi. 1999. Hybrid in-car speech recognition for mobile multimedia applications. In *Vehicular Technology Conference, 1999 IEEE 49th*, volume 3, pages 2009–2013. IEEE.

Romain Laroche, Jan Dziekan, Laurent Roussarie, and Piotr Baczyk. 2013. Cooking coach spoken/multimodal dialogue systems. *Cooking with Computers*.

Teruhisa Misu, Chiori Hori, Kiyonori Ohtake, Etsuo Mizukami, Akihiro Kobayashi, Kentaro Kayama, Tetsuya Fujii, Hideki Kashioka, Hisashi Kawai, and Satoshi Nakamura. 2010. Sightseeing guidance systems based on wfst-based dialogue manager. In *Spoken Dialogue Systems for Ambient Environments*, pages 194–195. Springer.

Joana Paulo Pardal and Nuno J Mamede. 2011. Starting to cook a coaching dialogue system in the olympus framework. In *Proceedings of the Paralinguistic Information and its Integration in Spoken Dialogue Systems Workshop*, pages 255–267. Springer.

Bryan Pellom, Wayne Ward, John Hansen, Ronald Cole, Kadri Hacioglu, Jianping Zhang, Xiuyang Yu, and Sameer Pradhan. 2001. University of colorado dialog systems for travel and navigation. In *Proceedings of the first international conference on Human language technology research*, pages 1–6. Association for Computational Linguistics.

Brian K Reid. 1988. The usenet cookbook - an experiment in electronic publishing. *Electronic Publishing*, 1(1):55–76.

Ulrich Schäfer, Frederik Arnold, Simon Ostermann, and Saskia Reifers. 2013. Ingredients and recipe for a robust mobile speech-enabled cooking assistant for german. In *KI 2013: Advances in Artificial Intelligence*, pages 212–223. Springer.

Rainer M Wasinger. 2001. Dialog based user interfaces featuring a home cooking assistant. *University of Sydney, Australia*.