

---

# IMPLEMENTATION OF A STATISTICAL DIALOGUE MANAGER FOR COMMERCIAL CONVERSATIONAL SYSTEMS

---

PREPRINT

**Pablo Cañas, David Griol**

Software Engineering Department, University of Granada, Spain  
dgriol@ugr.es

This is a pre-print version of the chapter: Cañas, P., Griol, D. (2021). Implementation of a Statistical Dialogue Manager for Commercial Conversational Systems. In: Herrero, Á., Cambra, C., Urda, D., Sedano, J., Quintián, H., Corchado, E. (eds) 15th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2020). SOCO 2020. Advances in Intelligent Systems and Computing, vol 1268. Springer, Cham. [https://doi.org/10.1007/978-3-030-57802-2\\_37](https://doi.org/10.1007/978-3-030-57802-2_37) ([https://link.springer.com/chapter/10.1007/978-3-030-57802-2\\_37](https://link.springer.com/chapter/10.1007/978-3-030-57802-2_37))

This preprint follows Springer Self-archiving policy for non-open access books and chapters (<https://www.springer.com/gp/open-access/publication-policies/self-archiving-policy>): “authors may deposit a portion of the pre-submission version of their manuscript (preprint) in a recognised preprint server (...). This portion of the pre-submission manuscript (preprint) may be deposited and made publicly available at any point.”

# Implementation of a statistical dialogue manager for commercial conversational systems\*

Pablo Cañas<sup>1</sup> and David Griol<sup>2</sup>

<sup>1</sup> École Polytechnique Fédérale de Lausanne (EPFL), Route Cantonale, 1015  
Lausanne (Switzerland)

`pablo.canascastellanos@epfl.ch`

<sup>2</sup> Dept. of Languages and Computer Systems, University of Granada, Periodista  
Daniel Saucedo Aranda sn, Granada (Spain), `dgriol@ugr.es`

**Abstract.** Conversational interfaces have recently become an ubiquitous element in both the personal sphere by improving individual’s quality of life, and industrial environments by the automation of services and its corresponding costs savings. However, designing the dialogue model used by these interfaces to decide the next response is a hard-to-accomplish task for complex conversational interactions. In this paper, we propose a statistical-based dialogue manager architecture, which provides flexibility to develop and maintain this module. Our proposal has been integrated with DialogFlow, a natural language understanding platform provided by Google to design conversational user interfaces. The proposed architecture has been assessed with a real use case for a train scheduling domain, proving that the user experience is of a high value and it can be integrated for commercial setups.

**Keywords:** Conversational systems · Dialogue management · Machine Learning · DialogFlow.

## 1 Introduction

Conversational interfaces are systems that emulate interactive conversations with humans [9, 2]. These systems use natural language to provide dialogue capabilities with different purposes, such as performing transactions, responding to questions, or simply to chat.

These interfaces have become a key research subject for many organizations that have understood the potential revenue of introducing it in society’s mainstream. Virtual personal assistants, such as Google Now, Apple’s Siri, Amazon’s Alexa or Microsoft’s Cortana, allow users to perform a wide variety of tasks, from setting an alarm to updating the calendar, finding nearest restaurants, preparing a recipe or reporting news [7].

---

\* The research leading to these results has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 823907 (MENHIR project:<https://menhir-project.eu>).

In addition, a growing amount of entities are using conversational interfaces to automate their services while increasing customer experience and satisfaction. Among others, these agents are being used for making appointments, providing legal advice, self-help therapy, and answer FAQ about the COVID-19 pandemic [1]. Such systems are making these range of services more efficient without the need for human resources, hence generating a potential billion-dollar industry around them [2].

Spoken conversational interfaces are usually made up of four different components: an automatic speech recognizer (ASR), which records the sequence of words uttered by the speaker; a natural language understanding module (NLU), which obtains the semantics from the recognized sentence by performing morphosyntactic analysis; a dialogue manager (DM), which decides the next response of the system, interpreting the semantic representation of the input in the context of the dialogue; and a text-to-speech synthesizer (TTS), which transforms the response in natural language into synthesized speech.

The dialogue manager is the main object of study of this paper. This module considers the information provided by the user during the history of the dialogue, the dialogue context and the results of the queries to the data repositories of the system to return a meaningful response.

Early models for dialogue strategies were implemented using rule-based methods and predefined dialogue trees [8]. This methodology consists of manually determining the response that the agent should retrieve to each of the user inputs. Such approach, which is still broadly used nowadays, can be appropriate for very simple use cases; for instance, systems answering a reduced set of isolated frequently asked questions. However, more complex dialogue systems usually require several user-system interactions for a successful interaction, thus making a rule-based approach unfeasible both in terms of maintainability and scalability.

As a solution to this problem, new methodologies for statistically dialogue modeling have been proposed during the last years [4]. Recent literature includes proposals based on Partially Observable Markov Decision Processes [11] and Reinforcement Deep Learning [3], which generate user-system interaction simulations to learn the appropriate response for every input. Supervised-learning-based solutions have also been proposed, including the use of Neural Networks [5], Stochastic Finite-State Transducers [6], and Bayesian Networks [10].

There currently exist several frameworks that ease the task of building industrial conversational agents, being Google’s DialogFlow<sup>3</sup> one of the most popular ones. Most of these toolkits allow specifying tree-based implementations for the dialogue manager, in which the system will respond to the specified user utterances [7]. However, some toolkits, like DialogFlow, also allow developers to integrate their own statistical model of the dialogue manager for the agent implementation. This brings a huge potential to develop and maintain such module for commercial and industrial setups.

To achieve this objective, in this paper we propose a practical framework to develop statistical-based dialogue managers that can be easily integrated in

<sup>3</sup> <https://dialogflow.com/>

toolkits like Dialogflow. As a proof of concept, we have implemented a practical conversational system for a train scheduling domain, in which we use the functionalities provided by DialogFlow for natural language understanding and a statistical dialogue manager developed using our proposal with a dialogue corpus acquired for the task.

The remainder of the paper is as follows. Section 2 describes the main features of the DialogFlow platform to create conversational interfaces. Section 3 presents our proposal to integrate statistical dialogue management functionalities in a conversational system designed using this platform. In Sections 4 and 5 we describe the application of this proposal to develop a conversational system providing railway information and the results of its integration and preliminary evaluation. Finally, Section 6 presents the conclusions and future research lines.

## 2 The DialogFlow platform

DialogFlow allows the development of conversational interfaces by automatically implementing the natural language understanding module by means of providing training phrases for each intent (end-user’s intention for a conversation turn) and defining the dialogue manager by means of the use of context conditions for each intent and the responses to return to the end-users for each one of them.

DialogFlow currently supports 32 languages and dialects<sup>4</sup>. Conversational interfaces developed using this toolkit can be integrated into wearable devices, cars, intelligent speakers, web plugins, and other mobile applications.

Regarding the NLU module development, DialogFlow has three basic primitives:

1. **Intents:** An intent is a specific action that users can invoke by using sentences that match their NLU model. Developers must provide a set of training phrases for each intent. As a result, depending on the user input, the agent maps each user response to a specific intent in order to provide a system response. Therefore, each intent represents a dialogue turn within the conversation.
2. **Entities:** An entity represents a term or object that is relevant for the intents and that provides a specific context for them. The entities are usually keywords used for identifying and extracting useful data from user inputs. DialogFlow provides a wide variety of predefined system entities, such as dates, times, cities, colors, or units of measure, but developers can also define their own domain-dependent entities. An entity consists of an entity type (e.g. fruit) and entity values (e.g. banana, strawberry, orange).
3. **Contexts:** They represent the current state of the interaction and allow agents to carry information from one intent to another. They can be combined to control the conversational path in order to define conditions required to access an intent (input contexts) or defined after accessing them (output contexts).

---

<sup>4</sup> <https://cloud.google.com/dialogflow/docs/reference/language>

Developing a dialogue manager in DialogFlow implies defining a set of possible responses for each intent. However, this set of responses is static and hence limits the flexibility of the chatbot. As an example, let's imagine that a user requests information to buy a train ticket. The user could start the interaction querying for different pieces of information: origin and destination cities, departure and arrival dates, price range, duration, services, train type, etc. A possible option is to define a single intent for these requests, as all of them are related to the same user's intention to book a train ticket. However, the number of combinations of parameters to consider becomes exponentially large for practical domains. Moreover, if the user does not provide all the information pieces, the agent needs to ask for the remaining data, and the range of context possibilities to take into account becomes huge. Another option is to define unique intents for each of the pieces of information, but again taking into account all the different combinations makes the dialogue management definition unfeasible.

By means of the fulfillment functionality provided by DialogFlow it is possible to connect natural language understanding and processing for each intent to any kind of business logic, such as querying databases, access third party APIs, or using a Machine-Learning-based model to predict the most adequate response given the dialogue context. In our proposal, we use this functionality to integrate more scalable statistical dialogue models for the dialogue manager.

### 3 Proposed framework for statistical dialogue management

Figure 1 shows the architecture for the proposed statistical DM framework. As it can be observed, the framework integrates Firebase applications<sup>5</sup> to provide cloud functions, realtime databases, and hosting. Nonetheless, other internal or third party services can be used to facilitate these services.

DialogFlow's NLP module is used to select the user intents and the entities in their utterances. Instead of defining a tree-based model, intents are not used to retrieve a predefined response, but to extract the context information to appropriately feed the statistical DM model with the dialogue history. The context is sent to the cloud function, which will first obtain the dialogue state from the previous interaction with such user.

The statistical dialogue model selects the next system response according to the dialogue state defined by the context (for instance, confirm a particular piece of information, request for additional information pieces, or inform about the results of a query). Depending on the type of response, the framework could require accessing a third party or internal database for completing the request (e.g., to inform about the ticket price for a specific train).

The dialogue state is updated with the data gathered and crafted during the interaction, in order to be ready for the next user input. The response selected by the statistical model is sent to DialogFlow so that the TTS module concludes the dialogue cycle.

<sup>5</sup> <https://firebase.google.com/>

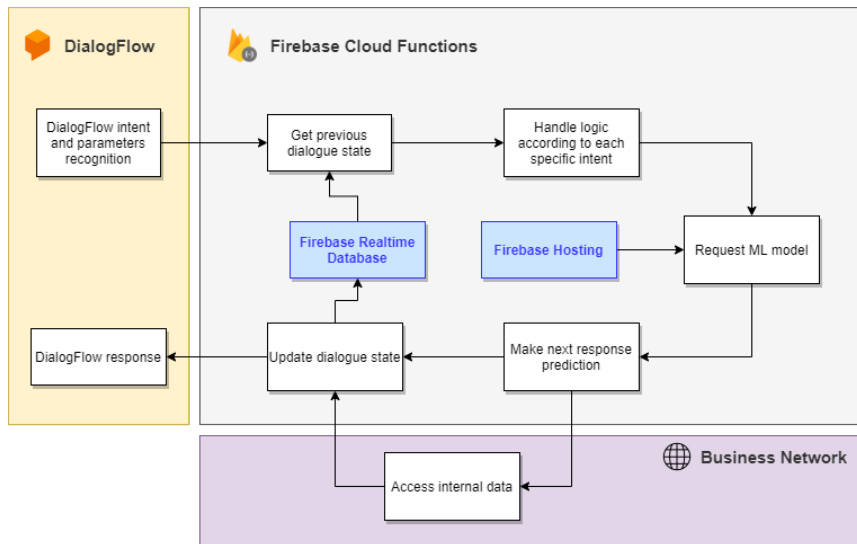


Fig. 1: Architecture of the proposed framework for the statistical dialogue manager implementation

The architecture of the framework provides modularity, scalability, speed, domain-independence, ability to handle complex and long interactions, and easiness for assembling with the rest of the modules required by complex conversational systems.

#### 4 Use case: train scheduling domain

As a proof of concept, the proposed framework has been used to develop a conversational system for a train scheduling domain, in which users can ask the system about information to plan a rail trip around Spain. The corpus that has been used to train the statistical dialogue manager is described in [5]. It consists of 4,006 training instances, which are labelled with one of the 23 possible system responses.

The number of input features used to define the context for the dialogue states is 19. They are encoded as Figure 2 shows: the previous action taken by the system; five attributes encoded as 0 or 1 depending on the type of request the user is asking for (timetables, price, train type, order number or service list); ten attributes denoting the type of information mentioned in the dialogue, encoded with 0 if the concept is unknown, 1 if the concept is known with a high confidence score, or 2 if the concept is known with a low confidence score; and three task-independent attributes denoting acceptance, rejection or not understanding.

The statistical model used for training was a standard Multilayer Perceptron architecture, suitable for this task as found in previous work [5], implemented in

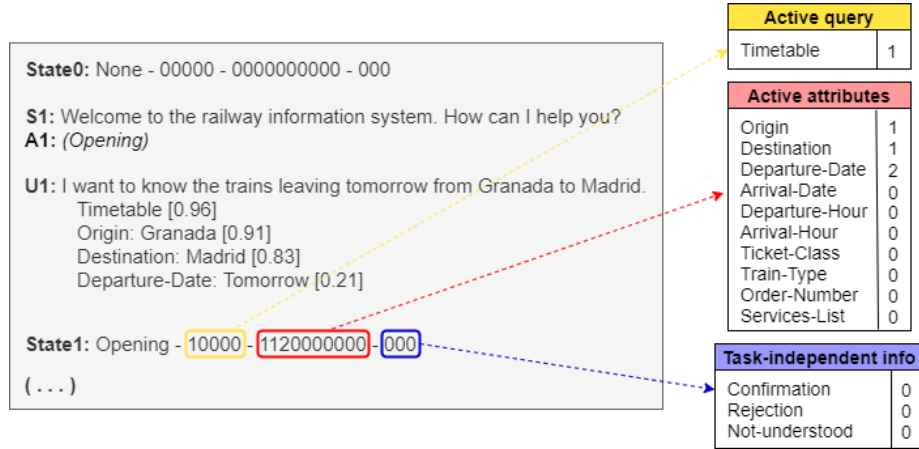


Fig. 2: Example of encoding for the train scheduling domain corpus

TensorFlow<sup>6</sup>. It consisted of 1 densely-connected layer of 256 neurons, trained using a step size of  $\alpha = 0.0005$ , ADAM optimizer, and early stopping to avoid overfitting. The trained model was stored as a JSON object in a Firebase Hosting instance.

DialogFlow’s NLP module was created by defining the set of intents, parameters and entities required for the use case. A total of 13 intents were defined, each of them related to a specific request or piece of information that the user could ask. Table 1 shows an example of some of the training phrases that were defined for two of those intents, the one providing information about the departure schedule, and the one asking for the duration of the trip.

Intent Name	Training Phrases	Parameters
Say-Departure-Date	Para mañana (For tomorrow)	
	Me gustaría salir el 2 de abril (I would like to depart April the 2nd)	
	Para mañana a las 3 (For tomorrow at 3)	
	Salgo el 4 de marzo a las 8 de la tarde (I depart March the 4th at 8 pm)	departureDate departureHour
	Me gustaría coger el tren a las 5 y cuarto de hoy (I would like to take the train today at quarter past 5)	
Ask-Route-Length	Me gustaría salir el 2 de abril a las 16:00 (I would like to depart April the 2nd at 16:00)	
	Sí, ¿cuál es la duración del trayecto? (Yes, what is the route length?)	
	¿Cuál es el tiempo de recorrido? (What is the route length?)	
	Sí, me gustaría saber el tiempo que se tarda (Yes, I would like to know how long does it take)	
	¿Cuánto se tarda? (How long does it take?)	
	¿Cuánto tarda el tren en llegar? (How long does the train take to arrive?)	

Table 1: Intent examples for the train scheduling domain.

<sup>6</sup> <https://www.tensorflow.org/>

The set of entities corresponds to the 10 attributes that Table 2 shows. Seven of these entities were already predefined in DialogFlow.

Parameter Name	Entity Type	Entity Values
origin destination	city (system)	Madrid, Barcelona, Vigo...
departureDate arrivalDate	date (system)	2020-05-04, tomorrow...
departureHour arrivalHour	time (system)	09:30, 4pm, noon...
ticketClass	ticketClass (crafted)	tourist, preference
trainType	trainType (crafted)	AVE, Alvia, Cercanías...
services	services (crafted)	cafeteria, wifi, newspaper...

Table 2: Parameters and entities defined for the train scheduling domain

A specific handler for each of the different DialogFlow intents was defined for the cloud function. Following the previously described architecture, the first step is to access the Firebase Realtime Database instance to obtain the previous system response, as well as all the information that it was already stored for the interaction. After this, depending on each specific intent, new information is added to the state (e.g. for the *Say-Departure-Date* shown in Table 1, departure schedule data).

The dialogue state is then encoded and sent to the statistical dialogue model, that uses this information as input to predict the next system response. Depending on the type of response (e.g., to provide the schedule for a train route), a new request to a third party or internal business layer can be required to inform about the trains fulfilling the conditions required by the user.

After this, the updated state is inserted in the Firebase Realtime Database, together with the system response, so that this information is available for the next interaction. Finally, the new system response is sent to DialogFlow.

## 5 Integration and evaluation

One of the main reasons that makes DialogFlow ideal for industrial applications is its straightforward integration with a wide collection of popular chat environments that allows to deploy real conversational user interfaces instantly. Integrations include Facebook Messenger, Twitter, Slack, Skype, Telegram, Google Assistant, and Amazon Alexa, among others, as well as embeddings for web applications.

A preliminary evaluation process was followed to validate the architecture implemented and the quality of the conversational agent. As a result, a total of



20 people from diverse ages, gender, educational and technological background were selected and interviewed to evaluate the train scheduling chatbot.

First, an objective evaluation of the system was completed, analyzing seven different metrics extracted from the interactions between the user and the system. Table 3 shows the metrics and results of the evaluation. The results show that the success rate is very acceptable and the turn coherence is high. The percentage of unique dialogs also points that the system is quite versatile.

Metric	Evaluation Value
Dialogue success rate	80 %
Turn coherence rate	78 %
Average #turns	7
Average #requests	2.89
#Turns shortest dialogue	10
#Turns longest dialogue	5
% Different dialogs	55 %

Table 3: Results of the objective evaluation

Users were also asked to provide their subjective opinion on the system’s performance with seven questions, scoring from 1 (lowest) to 5 (highest). Results, presented in Table 4, show a positive perception of the application. While the weakest point is the error recovery capability, users believe that the interaction with the system is clear and fast. The overall satisfaction is also high, with a large percentage of returning customers. This information validates the viability of the proposed solution for industrial purposes.

Question	Avg. score	Std Dev.
How well did you understand the system messages?	5	0
How well did the system understand you?	3.80	0.83
Was it easy for you to get the requested information?	4	1.12
Was the interaction with the system quick enough?	4.60	0.71
If there were system errors, was it easy for you to correct them?	3.30	1.41
In general, are you satisfied with the performance of the system?	4.40	0.73
Would you use this system to schedule your future train rides?	4.20	0.68

Table 4: Results of the subjective evaluation

Figure 3 shows an example of a successful dialogue extracted from one of the tests. Although the user speaks with a colloquial wording, providing unnecessary extra information and other tags such as *'more or less'* or *'everything has become clear to me'*, the system is able to retrieve very accurate responses, and successfully complete the interaction.

**U1:** Muy buenas tardes. (*Good afternoon.*)  
**S1:** Bienvenido, ¿en qué puedo ayudarte? (*Welcome, how can I help you?*)  
**U2:** ¿Podría decirme los horarios de trenes de Madrid a Zaragoza, por favor? (*Could you please tell me the train schedule from Madrid to Zaragoza?*)  
**S2:** ¿En qué fecha desearías viajar? (*In which date would you like to travel?*)  
**U3:** Para mañana, si es posible. (*For tomorrow, if possible.*)  
**S3:** Buscando los trenes disponibles, espere un segundo... Hay un tren disponible de Madrid a Zaragoza el día 26/5/2019 a las 16:00. ¿Puedo ayudarte en algo más? (*Searching for available trains, please wait... There is an available train from Madrid to Zaragoza on the 26/5/2019 at 16:00. Can I help you in anything else?*)  
**U4:** ¿Cuánto me costaría ese billete? (*How much would that ticket cost me?*)  
**S4:** El trayecto seleccionado tiene un coste de 31 euros en clase turista. ¿Necesitas algo más? (*The selected route has a cost of 31 euros in tourist class. Do you need anything else?*)  
**U5:** ¿Cuánto tiempo dura el viaje, más o menos? (*How long does the trip take, more or less?*)  
**S5:** El tiempo de duración del tren seleccionado es de 90 minutos. ¿Te puedo ayudar en algo más? (*The selected train duration is 90 minutes. Can I help you in anything else?*)  
**U6:** ¿Y qué tipo de trenes hay disponibles en ese billete? (*And what type of trains are available with that ticket?*)  
**S6:** El tipo de tren del vehículo seleccionado es AVE. ¿Tienes alguna otra consulta? (*The train type of the selected vehicle is AVE. Do you have any other enquiry?*)  
**U7:** No, muchas gracias, me ha quedado todo muy claro. Muy amable. (*No, thank you, everything has become clear to me. Very kind of you.*)  
**S7:** Espero haber sido de ayuda. Que tenga un buen día. (*I hope I have been helpful. Have a nice day.*)

Fig. 3: Example of a successful dialogue extracted from the evaluation

## 6 Conclusions and future work

In this paper we have presented a framework to automatize the processes to develop commercial conversational interfaces by means of statistical dialogue models. We have integrated this framework within the DialogFlow platform, easily assembling the set of components in the interface, and deploying a ready-to-use industrial application which can be integrated into different environments and devices.

We have validated our proposal developing and evaluating a dialogue system based on a real use case, a train scheduling domain. This implementation has covered the end-to-end process of developing a personalized conversational interface learning a statistical dialogue manager for the task and integrating it with DialogFlow using our framework. The results of the evaluation show the vi-

ability and potential value of our proposal to develop commercial conversational systems.

This document contains some limitations that will be addressed in future research work. As more advanced algorithms for text processing have recently emerged, it would be interesting to address their suitability for dialogue management tasks. Moreover, it would be valuable to further explore the scalability of our proposal for larger and more complex datasets, as well as detailed comparisons to other similar work for such scenarios.

Future work also includes the completion of a more detailed evaluation of our framework integrating other statistical approaches for dialogue management, and the automation of the processes required for creating the structure of intents and entities in toolkits such as DialogFlow.

## References

1. Androutsopoulou, A., Karacapilidis, N., Loukis, E., Charalabidis, Y.: Transforming the communication between citizens and government through ai-guided chatbots. *Government Information Quarterly* **36**(2), 358 – 367 (2019)
2. Bavaresco, R., Silveira, D., Reis, E., Barbosa, J., Righi, R., Costa, C., Antunes, R., Gomes, M., Gatti, C., Vanzin, M., Junior, S.C., Silva, E., Moreira, C.: Conversational agents in business: A systematic literature review and future research directions. *Computer Science Review* **36**, 100239 (2020)
3. Cuayáhuitl, H., Keizer, S., Lemon, O.: Strategic Dialogue Management via Deep Reinforcement Learning. *CoRR* **abs/1511.08099** (2015)
4. Gao, J., Galley, M., Li, L.: *Neural Approaches to Conversational AI*. Now Publishers (2019)
5. Griol, D., Callejas, Z., López-Cózar, R., Riccardi, G.: A domain-independent statistical methodology for dialog management in spoken dialog systems. *Computer Speech & Language* **28**(3), 743–768 (2014)
6. Hurtado, L., Planells, J., Segarra, E., Sanchis, E., Griol, D.: A Stochastic Finite-State Transducer Approach to Spoken Dialog Management. In: *Proc. 11th Annual Conference of the International Speech Communication Association (InterSpeech'10)*. pp. 3002–3005. Makuhari, Chiba, Japan (2010)
7. Janarthanam, S.: *Hands-On Chatbots and Conversational UI Development: Build chatbots and voice user interfaces with Chatfuel, Dialogflow, Microsoft Bot Framework, Twilio, and Alexa Skills*. Packt Publishing (2017)
8. Lopes, J., Eskenazi, M., Trancoso, I.: From rule-based to data-driven lexical entrainment models in spoken dialog systems. *Computer Speech & Language* **31**(1), 87–112 (2015)
9. McTear, M., Callejas, Z., Griol, D.: *The Conversational Interface: Talking to Smart Devices*. Springer (2016)
10. Thomson, B., Yu, K., Keizer, S., Gasic, M., Jurcicek, F., Mairesse, F., Young, S.: Bayesian dialogue system for the Let’s Go Spoken Dialogue Challenge. In: *Proc. of IEEE Spoken Language Technology Workshop (SLT'10)*. pp. 460–465. Berkeley, USA (2010)
11. Young, S., Gašić, M., Keizer, S., Mairesse, F., Schatzmann, J., Thomson, B., Yu, K.: The Hidden Information State model: A practical framework for POMDP-based spoken dialogue management. *Computer Speech & Language* **24**, 150–174 (2010)