Towards a Natural Language Dialog System for Mobility Service Platforms

David Thulke[1,*], Felix Schwinger[1,2] and Karl-Heinz Krempels[1,2]

[1] *Information Systems, RWTH Aachen University, Aachen, Germany*

[2] *Fraunhofer Institute for Applied Information Technology FIT, St. Augustin, Germany*

Keywords: Mobility Service Platforms, Natural Language Processing, Dialog Systems.

Abstract: Due to a rise in novel mobility modes, urban transportation systems have become more heterogeneous and complicated in recent years. Mobility Service Platforms integrate different mobility services to offer integrated travel information, booking, and travel assistance, regardless of mobility provider or mode. Traditionally, users access these information systems through graphical user interfaces. Especially for the older population, such a sophisticated information system for a complex problem is problematic. Therefore, in this paper, we propose an approach and a prototype for a natural language interface for Mobility Service Platforms. The natural language interface allows access to the Mobility Service Platforms' information systems and integrates other domains, such as event and place information into natural language queries. To this end, we introduce a simple unified data model for travel, event, and Point of Interest domain and design an interaction model for the natural language interface. We evaluate the prototype in a case study with potential users. The evaluation shows that most users are more comfortable interacting with a mobility service platform using natural language instead of using different graphical user interfaces providing similar functionality.

1 INTRODUCTION

Finding optimal itineraries in urban transportation networks is becoming increasingly complex. Motorized private transport causes challenges like congestions, air pollution, scarcity of parking spaces, and greenhouse gas emissions. Simultaneously, alternative mobility services like car-, bike-, or ride-sharing are emerging and gaining popularity. These alternative mobility services may help to alleviate the first-mile/last-mile problem. This problem occurs with public transportation. People have trouble managing their first mile to a public transit station and from their last-mile from destination stop to their final destination. Hence, alternative mobility services are also of great political importance to provide sustainable transportation means by allowing people to access public transportation more efficiently.

An increasing amount of real-time data on traffic flow and delays in public transport networks became available in recent years for all these mobility modes. For travelers, it becomes increasingly difficult to find their preferred journeys in such heterogeneous information systems. Furthermore, due to mobility modes' heterogeneity, the shortest time, price, weather conditions, reliability, and sustainability are critical factors to consider for travelers when choosing their itinerary. Manually comparing different possibilities is becoming a challenging task since the necessary information is often only available in specialized applications. Optimal itineraries may even be intermodal, i.e., they consist of a mix of multiple different modes of transportation during a single trip. In this case, reserving and booking legs requires dealing with different pricing schemes and booking processes. Additionally, travelers have to check each leg for delays individually manually and are accountable if they miss their connection due to a prior delay. One approach to solve these issues are Mobility Service Platforms (MSPs) (Beutel et al., 2018) which implement the concept of Mobility-as-a-Service.

MSPs try to integrate different transportation modes to provide travelers with a unified interface for routing, booking, and travel assistance. Mobility-as-a-Service describes the notion that people stop relying on private vehicles for their mobility needs, but rather turn to service providers. To make this feasible,

* David Thulke is now affiliated with the Chair of Human Language Technology and Pattern Recognition, RWTH Aachen University, Aachen, Germany

Thulke, D., Schwinger, F. and Krempels, K.
Towards a Natural Language Dialog System for Mobility Service Platforms.
DOI: 10.5220/0010526507060713
In *Proceedings of the 7th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2021)*, pages 706-713
ISBN: 978-989-758-513-5

standardization of interfaces and business processes between different stakeholders is necessary. Even though MSPs attempt to ease the complexity of combining multiple transportation modes, new users may easily be overwhelmed by the sheer amount of different options available to them (Schulz et al., 2020). One of the political goals of MSPs are to enable more people to use sustainable mobility modes for their daily mobility needs, i.e., allowing a switch from private transportation to public and shared transportation. Therefore, people require information on how to realize their mobility needs with the available modes without being drowned in the mass of heterogeneous information.

Users primarily interact with these systems using Graphical User Interfaces (GUIs), like web pages or smartphone apps. An alternative to these are Natural Language Interfaces (NLIs). They allow users to interact with a system using natural speech or text. There is a distinction between question-answering and dialog systems. The former only replies to a single user utterance, whereas, in contrast, the latter takes the previous conversation into account when answering user queries. This kind of interaction has some advantages for travel information systems compared to GUIs. Using speech for the interaction makes the system accessible in situations where a screen can or should not be used, e.g., while driving or walking or even for users who have difficulties operating GUIs due to disabilities (Pradhan et al., 2018). Additionally, interacting with the system using natural language may make it more accessible to users with low system knowledge or inexperienced users with limited knowledge about interacting with computers at all (Dodd et al., 2017). It is easier to formulate a complex query using a single sentence than selecting the options in a GUI. Another advantage of dialog systems is that it is easy to switch domains or interact with the system across multiple areas. For example, users could ask for a restaurant recommendation and then plan an itinerary for the selected result. On the other hand, NLIs make it harder to discover functionalities, may not meet the language understanding capabilities expected by users, and often result in privacy concerns due to off-device processing of language and data annotation requirements.

This work aims to do the first step towards a dialog system for an MSP to lower its usage complexity in Mobility-as-a-Service scenarios. For this, we contribute in the following way:

Domain Data Model: To allow users seamless interaction between the domains travel, point of interest, and events, we designed a corresponding data model. The connection of these domains helps travelers to find itineraries suitable to their needs.

Interaction Model: For the NLI, we designed a prototype interaction model capturing intents belong to the three domains of the data model. The user can query information in a distinct domain and formulate queries that mix information from different domains.

Context Model: In a cross-domain scenario, it is complex to handle the context of a conversation as utterances may refer to a multitude of different entities. For this, we developed a cross-domain context model that captures relevant context information in the different domains.

With these contributions, we attempt to ease the usage of an MSP and address the problem of the flood of information for new users.

Section 2 gives a brief overview of related work. In Section 3, we elicit general requirements, propose a system architecture, and design an initial prototype for such a system. Based on this, Section 4 presents a simple prototype of the system. The case study, where potential users used and rated the general concept and an initial prototype, is discussed in Section 5. Finally, Section 6 concludes the paper and gives an outlook on future work.

2 RELATED WORK

This section introduces and discusses related work and state-of-the-art travel information systems and natural language dialog systems. Already in 1995, Aust *et al.* developed a spoken dialog system for train timetable information in Germany. The system was restricted to this single domain but allowed users to correct and update their queries. It was callable via the telephone network and used stochastic context-free grammars for speech understanding. Turunen *et al.* (2005), Raux *et al.* (2005), Dušek *et al.* (2014) developed similar systems.

The Deep Map project (Malaka and Zipf, 2000) was to build a tourist guide for the City of Heidelberg. Users could query information like sights, hotels, restaurants, and finding itineraries through a natural language interface. The SpaceBook project (Bartie et al., 2018) developed a similar system for the City of Edinburgh. In contrast to the system proposed in this paper, they focused on supporting tourists during their trip.

Braun *et al.* (2018) developed a system integrating different mobility services accessible via a natural language interface. It enables intermodal routing by building a meta-model for different mobility services. In contrast to other approaches, they only use the publicly available APIs of the mobility services and do not rely on their cooperation. Other services of a mobility service platform, such as reserving or booking itineraries, are currently not supported by their approach. In contrast to the system developed in this service, their interface is limited to travel information.

In the NLMaps project (Lawrence and Riezler, 2016), the goal was to develop a question-answering system for the OpenStreetMap (OSM) database. The system translates user utterances to queries to the database. Besides the system, the authors present the corresponding NLmaps corpus in (Haas and Riezler, 2016) containing 2,380 natural language questions in German and English paired with queries to OSM in the custom query language.

3 SYSTEM DESIGN

For the requirements engineering and implementation process, we followed a user-centered approach (Lowdermilk, 2013). To guide the system's design, we start by defining multiple scenarios as user stories that we have also validated in a Wizard-of-Oz experiment (Dahlbäck et al., 1993). In a Wizard-of-Oz experiment, users interact with a system without knowing that a person operates the system. These user stories outlined how potential users may interact with the system. We used these scenarios to derive requirements and to guide the design and development of our approach. Next, a common data model for the individual domains covered by the system is defined. Based on this, an interaction model is developed, describing the interaction between users and the system. Next, a context model is discussed, which the system has to consider during a dialog. Finally, we give an overview of the architecture of the whole system.

3.1 Domain Data Model

The proposed system should combine data from different sources. A data model of the relevant domains has to be specified to give reasonable responses.

The main objects in the domain are shown in Figure 1 and are *POIs*, *events*, and *itineraries*. A *POI* is a subclass of a *Place*, which is a generic description of an object on a map. A *Place* has a *Geography* and may have an *Address*. The former is an abstract type representing a geographical shape and may be a *Coordinate*, a *Line*, like a street, or an *Area*, like a city.

An *Event* has a start date and may have a specific end date or no defined end. It has a name and a description and is located at a particular *Place*. Finally, an *Itinerary* is a possible trip between two places. It has an ordered non-empty list of *Legs*. Each *Leg* represents one part of the trip and has a specific start and end place and a start and end date. Finally, each *User* has a home, work, and current location, which are references to a specific place.

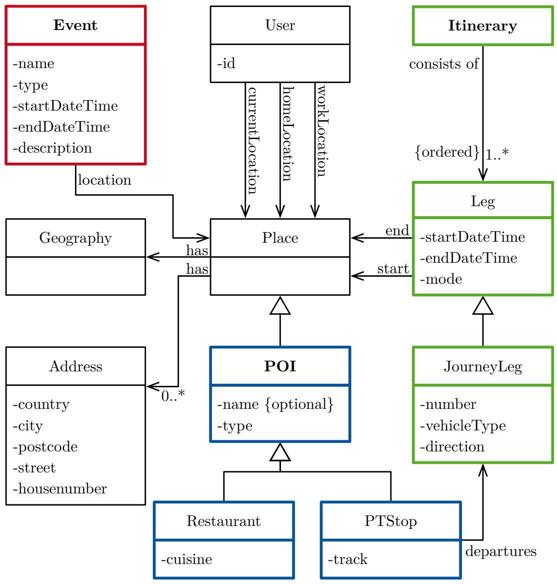

Figure 1: Data Model of the individual domains. The objects in the travel domain are in **green**, in the Point of Interest (POI) domain in **blue** and in the event domain in **red**.

3.2 Interaction Model

As a next step in the system design, we define the interaction model between the user and the system. The interaction model consists of the set of dialog acts, i.e., the intents and their corresponding entities, the system can understand. Intents are classes of user utterances describing their intention or goal.

We gathered an initial corpus of possible dialogs and user utterances to define the set of intents and entities systematically. For this, we firstly generated a corpus by hand consisting of manually generated utterances based on the results of a small Wizard-of-Oz experiment (Dahlbäck et al., 1993), and secondly, incorporated the NLMaps corpus.

Thereby the following main intents were identified:

- `get_pois`: utterances that search for POIs.
- `get_events`: utterances that search for events.
- `get_departures`: utterances asking for departures at public transport stops.
- `find_itineraries`: utterances that ask for an itinerary.

Another set of utterances are those where users want to change or correct a previous request (e.g., "I meant near A") or respond to a clarification request of the system (e.g., "What is your destination?"). These were classified under the intent `update_slot`. The purpose of some entities depends on the context of the utterance. A reference to a place could be a `source`, `destination`, or `direction` entity. A time could be a `start_time` or an `arrival_time`. The time in the utterance "I meant at 2 pm" could either be a departure or arrival time depending on which one of them the user specified in a previous utterance (e.g., "I want to depart at 3 pm" or "I want to arrive at 3 pm").

For references to places or dates, the user context can be used (e.g., the home location). Users might want to query the current state of the context (e.g., "Where am I?" or "Where do I work?") or might want to update this state (e.g., "I work in the computer science center."). For these utterances, the intents `get_*_location` and `update_*_location` were added. Similar intents for dates could be added in the future.

3.3 Context Model

To adequately reply to a request, it is often not sufficient to understand the user utterance, but the additional context must be considered. Bunt (1999) defines context as "factors relevant to the understanding of communicative behavior". He categorizes context into dialog, user, and world context.

The dialog context consists of the relevant information of the current dialog with the user. This includes the history and the current goal of the dialog. These are used in the dialog manager to decide on the next action of the system. Users may refer to objects mentioned in earlier utterances (e.g. "How do I get *there*?"). This co-reference has to be resolved from the dialog history.

The user context contains information on the user, which is relevant to the dialog. This work includes the current, home, and work location of the user. Besides, this can include personal information like the user's preferences.

Finally, the world context includes information independent of the current dialog and user but could still influence the dialog.

3.4 System Architecture

Figure 2 gives an overview of the individual components of the proposed system. The user interacts with the User Interface (UI). This interaction could either be via speech or text. In the former case, an Automatic Speech Recognition (ASR) component transcribes the user utterance. The Natural Language Understanding (NLU) component receives the user utterances as text. Its task is to transform the text into a dialog act consisting of an intent and corresponding entities defined in the context model. Based on this dialog act and the current context, the dialog manager's task is to predict the system's following action. Actions are either system dialog acts or forwarded to the task manager. An example of the latter is to find an itinerary. Based on the action and the current context, the task manager creates a query to the MSP. Its response is passed back to the dialog manager to update the current context. A system dialog act is transformed into text by the Natural Language Generation (NLG) component. This text is then displayed or spoken to the user via the UI. Besides, the UI may send additional context updates, e.g., the current user location, directly to the dialog manager.

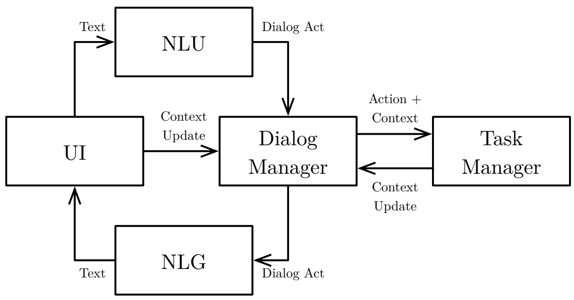

Figure 2: Abstract overview of the system architecture and the interaction of the individual components.

4 IMPLEMENTATION

This section gives an overview of the implementation of the individual components discussed in Section 3.4. Even though the system currently only supports German, examples are given in English or translated into English if there is a difference between them. The individual components are containerized using Docker. This segmentation increases the portability of the system by facilitating deployment to a new environment.

4.1 Natural Language Understanding

Extracting dialog acts from user utterances, as described in the interaction model, can be divided into the sub-tasks of intent classification and slot filling. For both tasks, we used the default implementation, pipelines, and models provided by the Rasa NLU framework[1]. As a first step, the user utterance is tokenized and converted to a sequence of GloVe embedding vectors (Pennington et al., 2014). These vectors are averaged to create a representation of the complete utterance. Then support vector machines (SVMs) are used to classify the intent based on this representation. For the slot filling task, a feature-based conditional random field (CRF) model (Lafferty et al., 2001) is used. Bocklisch *et al.* (2017) discuss this approach in detail.

Even though these approaches do not achieve state-of-the-art performance on the respective tasks, they are very efficient to train and work reasonably with low amounts of training data, making them suitable to bootstrap the system.

The next step was to collect and annotate the necessary training data. As freely available utterances for NLI for the travel information domain is sparse, we trained the system initially with the data from the NLMaps corpus (about 1.500 utterances). As no open corpus exists for cross-domain NLI we extended this corpus with further data. For this, we performed a Wizard-of-Oz experiment (Dahlbäck et al., 1993) and included about 150 relevant utterances. Next, we manually fine-tuned the corpus and included 400 more possible utterances. This fine-tuning included, among others, replacing words with synonyms or similar utterances with slightly different nuances. Finally, we augmented the corpus's utterances by replacing entities with different values to increase the corpus size further.

Afterward, we needed to annotate the utterances for each intent in the corpus. For this, we manually annotated about 5 to 20 utterances per intent for a total number of 56 intents. To annotate the remaining utterances, we trained an initial version of the natural language understanding system with the initial data. Then the system was used to classify and recognize the next chunk of utterances automatically. As not all utterances were correctly classified, we manually corrected the system's classification errors, and then we added the data to the training corpus. We repeated this process until all utterances were correctly labeled.

4.2 Dialog Manager

The Rasa Core[2] module, which is based on Hybrid Code Networks (Williams et al., 2017), is the basis for the dialog manager. Its task is to handle the current dialog state. On a high level, the dialog manager consists of a state tracker, a policy, and a set of actions. The tracker stores the latest messages as a list of intents and the current dialog state. The dialog state is represented as a key-value store. Depending on their type, values are converted to features to be used in the prediction. The values can either be entities extracted from user utterances, added by actions (e.g., the results of a routing request) or added externally (e.g., the user's current or home location). Based on the state of the tracker, the policy predicts the next action, which should be executed. A set of sample dialogs has to be provided to train the model of the policy. These consist of a list of user dialog acts, i.e., the intent and corresponding entities, the system's actions, and relevant dialog state updates. We generated sample dialogues directly from user utterances based on the Wizard-of-Oz experiments transcripts and manually interacted with the system. This way, we gathered user utterances on how they would interact with the system and then added these utterances to our corpus.

4.3 Task Manager

The role of the task manager is to execute actions or tasks triggered by the dialog manager. Incoming requests from the Rasa Core module are processed and handed to the corresponding action handler. Each action handler performs the following steps: The first step is to parse entities and resolve references. This parsing includes analyzing dates given in natural language, resolving places to their geography, and resolving references to the dialog history. The dialog state is validated, and a query to an external system is built using these parsed entities. Next, the task manager executes this query. The result is either a successful query, a check back, or an error. Afterward, the dialog state is updated accordingly. For parsing and queries, the task manager may call external services that provide the data integration.

4.4 Data Integration

The previously introduced task manager's role is also the data integration. Typically, all required information should be provided by an actual MSP. As no system with such interfaces exists to our best knowledge, we mocked it using a set of external services providing the system's functionality. The task manager may either query external web services or query an internal Postgres database for information retrieval.

The Postgres database with the spatial extension PostGIS implements the point of interest and event part of the data model introduced in Figure 1. For event information, the system scraped the database of a local magazine publishing local event information.

[1] http://legacy-docs-v1.rasa.com/nlu/about/

[2] http://legacy-docs-v1.rasa.com/core/about/

For POI information, we queried the Overpass API of the OSM project to store relevant POIs in the local database. The geocoding of locations mentioned by the user in a query was translated into coordinates by the open-source geocoders *Nominatim* and *Photon*. Travel information is retrieved by querying an external web service provided by the local transport provider. Travel information is time-dependent and may change over time. It is not stored in the database but rather recalculated every time the user requests travel information.

4.5 Natural Language Generation

The system uses a template-based approach for NLG. Thereby, the system differentiates between predefined and generated responses. The former are simple templates that can include placeholders that the system can fill with slot values. Rasa Core handles them and is either invoked directly by the dialog manager (as an utter action) or triggered by the task manager. Examples for these are responses to small talk utterances or check backs like “What is your destination?”.

Generated responses are more complex responses (e.g., a list of objects or an itinerary description). It takes as input the system dialog act and the current context.

5 EVALUATION

Potential users evaluated the approach and the implemented prototype to assess whether it matches our requirements and the described scenario. The evaluation is still a work-in-progress and only provides initial non-representative feedback for a cross-domain NLI. We first present our methodology and then introduce the most relevant results.

5.1 Methodology

We evaluated the system with an online survey, structured into five distinct parts. The first one inquired about general demographic data from the participants. In the second part, the survey inquired participants about their previous experience with NLIs. It asked them whether they are using NLIs and, if so, which systems they use, in which domains, and in which areas they think, NLIs are useful. The third part presented participants with the concept of a dialog system for an MSP, covering all phases of a trip, including related domains. It asked the participants whether they think that such a system would be useful and whether they would use it. This approach allowed

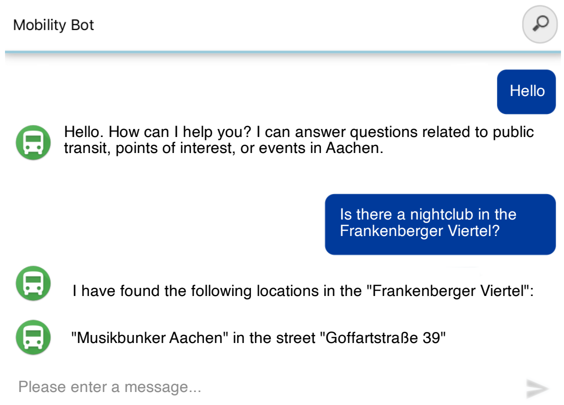

Figure 3: Translated screenshot of the chat interface of the prototype application.

gathering feedback on the system’s actual goal without influencing participants by letting them already use the current prototype. In the fourth part, participants should use the system via a chat interface. A screenshot of the chat interface is shown in Figure 3.

First, the survey explained the functionality for participants to get familiar with the system. The next three pages of the survey asked participants to complete the three different scenarios using the system and the systems they usually use in these domains. The following three scenarios were used:

1. Participants were asked to search for a nightclub in a given neighborhood in Aachen. After finding a suitable one, they should search for events at this place on the following weekend and ask for an itinerary to get to one of these events on time.
2. Participants should plan an itinerary by bus from one location to another for the following day.
3. Participants should plan a trip with their children during the next week. Therefore, they should find events during the next week which are suitable for children. After picking one of the events, they should find a suitable parking space nearby.

These scenarios and similar ones were also part of the Wizard-of-Oz experiments to gather system requirements and utterances. As the scenarios are based in the vicinity of the city of Aachen, the knowledge of the participants of the area is also important.

For each scenario, users were asked with which system it was easier to complete the task, faster to complete the task, and which system they preferred overall. Additionally, they were asked whether any errors occurred while using the system.

The final part of the survey asked participants to evaluate the usability of the prototype. Therefore, the framework for questionnaires by the International Telecommunication Union presented in

(Möller, 2003) was used. Möller mainly intended the survey for spoken dialog systems used over the telephone network. We translated the survey questionnaire into German and omitted questions specific to the spoken interaction.

After the survey, we lastly asked the participants for qualitative feedback on what they liked the most and least on the system and whether they have any proposals for change or improvement. No systems with similar scope, i.e., NLI across the three different domains and on a focus on Mobility Service Platforms, exists to our knowledge, Therefore, we decided not to compare the model directly to other approaches but to focus on the described user-centered evaluation and compare the integrated NLI approach to multiple GUI approaches. In future work, the prototype will be compared to state-of-the-art solutions.

5.2 Results

The survey was conducted in the environment of the university and thus the participants were not demographically representative. In total 25 participants completed the survey (10 female, 14 male, and 1 non-binary participants). Most participants were between 20 and 29 years old, but most other age groups were represented. Most participants considered themselves as rather tech-savvy. Two-thirds of the participants (16 out of 25) stated that they have local knowledge of the City of Aachen.

About half of the participants are actively using NLIs, like Google Assistant, Apple Siri, and Amazon Alexa. Though more than 80% of these participants stated that they think that NLIs are useful in the domains of travel information, POIs, and local events, less than 30% are using NLIs in one of these domains. For travel information, most participants are currently using the apps of regional or national transport providers. For POIs most participants are using Google Maps, and for local events, participants mentioned Google Search, Social Media, and Print Advertising most frequently. The concept of an NLI for an MSP was rated positively. About half of the participants stated that they would use such a system, while the other half was unsure.

After completing each scenario with the system and the applications they usually use, they were asked which of them were more comfortable to use, faster to get a result and which one they preferred overall. Most users preferred and thought that the prototype was more comfortable to use in all scenarios and faster to use than the other application in the second and third scenarios. Only in the first scenario, a majority stated that the other applications were faster to use.

Participants were asked whether error or unexpected behavior occurred while using the prototype. For the first scenario, about half of the users affirmed this. For the other scenarios, about one-third of the participants answered yes to this question. This high-error rate could explain why most users could complete the first scenario faster using other applications.

Next, participants were asked to fill out the usability questionnaire (Möller, 2003). Only half of the participants stated that the system understood them perfectly or rater perfectly, which could be explained by the sometimes limited performance of the NLU system. 9 participants stated that the system sometimes did not behave as they expected and that the system made errors either frequently or frequently. Most participants stated that the system reacted flexibly, and they were able to control the dialog as desired. 15 participants were satisfied or somewhat satisfied with the system. In contrast, one participant was unsatisfied, and four somewhat unsatisfied.

Finally, participants were asked for qualitative feedback. The most stated positive points were that the system allows for easy and quick access to information, integrates different domains, and allows referring to previous utterances, either to correct a query or when changing the domain. The negative feedback included that some utterances were not understood correctly. Some participants disliked the textual representation of itineraries and would prefer a more structured one. The system requires too much typing. Consequently, one of the suggested improvements was to allow speech as an input and output modality. Participants suggested using additional visualizations like maps to display location or pictures for events or POIs. Some participants asked for instruction on how to use the system. They were unsure which kind of queries the system is capable of handling. The system partly caused this by mapping unsupported utterances to one of the intents with relatively high confidence.

6 CONCLUSION

This paper presented our current progress towards a natural language interface for a Mobility Service Platform. MSPs give travelers more options on how to reach their destination, ideally resulting in cheaper, faster, more reliable, and more comfortable itineraries. Globally this can result in better resource usage, fewer traffic congestions, and more sustainable mobility behavior. Accessing these platforms using NLIs can increase the ease of use and make them more accessible. An NLI may significantly help people not accustomed to using public or shared transportation. With the NLI and the integration of further information domains, we attempted to increase the benefit of the information systems, particularly for infrequent users.

We proposed a data model integrating different data resources in the individual domains. Furthermore, we designed an interaction model covering the interaction in a limited subdomain. The prototype, which is still a work-in-progress, was implemented based on existing open source components and evaluated by potential users. One of the challenges was to acquire a sufficient amount of training data that would be required to improve the NLU component. Nevertheless, the system was rated already rather positively by users in a preliminary test. Participants emphasized the seamless integration of related domains.

Besides improving the performance by using more training data and more sophisticated models, the system's domain should be extended to cover more services provided by MSPs. The interaction's usability can be improved by offering additional input and output modalities, like speech or visualizations. Finally, we plan to perform a broader evaluation with a larger group of people from different backgrounds, thus making the evaluation representative.

ACKNOWLEDGEMENTS

This work has, in part, been funded by the Federal Ministry of Transport and Digital Infrastructure (BMVI) within the funding guideline "Automated and Connected Driving" under the grant number 16AVF2134B.

REFERENCES

Bartie, P., Mackaness, W., Lemon, O., Dalmas, T., Janarthanam, S., Hill, R. L., Dickinson, A., and Liu, X. (2018). A Dialogue Based Mobile Virtual Assistant for Tourists: The SpaceBook Project. *Computers, Environment and Urban Systems*, 67:110–123.

Beutel, M. C., Gökay, S., Ohler, F., Kohl, W., Krempels, K.-H., Rose, T., Samsel, C., Schwinger, F., and Terwelp, C. (2018). Mobility Service Platforms - Cross-Company Cooperation for Transportation Service Interoperability. In *Proceedings of the 20th International Conference on Enterprise Information Systems*, pages 151–161.

Bocklisch, T., Faulkner, J., Pawlowski, N., and Nichol, A. (2017). Rasa: Open Source Language Understanding and Dialogue Management. Technical report, Rasa.

Dahlbäck, N., Jönsson, A., and Ahrenberg, L. (1993). Wizard of Oz studies—why and how. *Knowledge-based systems*, 6(4):258–266.

Dodd, C., Athauda, R., and Adam, M. T. (2017). Designing User Interfaces for the Elderly: A Systematic Literature Review. In *Australasian Conference on Information Systems*, pages 1–11, Hobart, Australia.

Haas, C. and Riezler, S. (2016). A Corpus and Semantic Parser for Multilingual Natural Language Querying of OpenStreetMap. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 740–750, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lafferty, J., Mccallum, A., and Pereira, F. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 282–289.

Lawrence, C. and Riezler, S. (2016). NLmaps: A Natural Language Interface to Query OpenStreetMap. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 6–10, Osaka, Japan.

Lowdermilk, T. (2013). *User-centered design: a developer's guide to building user-friendly applications.* " O'Reilly Media, Inc.".

Malaka, R. and Zipf, A. (2000). DEEP MAP Challenging IT research in the framework of a tourist information system. In *Information and communication technologies in tourism*, pages 15—-27. Springer.

Möller, S. (2003). Subjective Quality Evaluation of Telephone Services Based on Spoken Dialogue Systems. Technical report, International Telecommunication Union.

Pennington, J., Socher, R., and Manning, C. (2014). GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Stroudsburg, PA, USA. Association for Computational Linguistics.

Pradhan, A., Mehta, K., and Findlater, L. (2018). Accessibility Came by Accident: Use of Voice-Controlled Intelligent Personal Assistants by People with Disabilities. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 459.

Schulz, T., Böhm, M., Gewald, H., and Krcmar, H. (2020). Smart mobility–an analysis of potential customers' preference structures. *Electronic Markets*, pages 1–20.

Williams, J. D., Asadi, K., and Zweig, G. (2017). Hybrid Code Networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 665—-677, Vancouver, Canada. Association for Computational Linguistics.