A Multimodal Dialogue Framework for Cloud-Based Companion Systems

Matthias Kraus, Gregor Behnke, Pascal Bercher, Marvin Schiller, Susanne Biundo, Birte Glimm and Wolfgang Minker

Abstract *Companion systems* are cooperative, cognitive systems aiming at assisting a user in everyday situations. Therefore, these systems require a high level of availability. One option to meet this requirement ist to use a web-deployable architecture. In this demo paper, we present a multimodal cloud-based dialogue framework for the development of a distributed, web-based *companion system*. The proposed framework is intended to provide an efficient, easily extensible, and scalable approach for this kind of systems and will be demonstrated in a do-it-yourself assistance scenario.

1 Introduction

Nowadays we expect technological devices to be intelligent and to adapt their functionalities to our individual needs. *Companion Technology* fulfils these requirements (Biundo & Wendemuth, 2016; Biundo, Höller, Schattenberg, & Bercher, 2016). A *companion system* is understood as a cognitive system which models its behaviour according to the individual user's needs, preferences, capabilities, and also takes into account the current situation. Therefore, it requires cognitive abilities like planning, reasoning, and the ability to conduct a dialogue. By providing a high level of cooperation capability and reliability, even in complex tasks, *companion systems* aim at being an everyday competent and trustworthy partner. For this purpose, those systems need to be always available and be accessible

Matthias Kraus, Wolfgang Minker
Institute of Communications Engineering, Ulm University, Germany,
e-mail: {matthias.kraus, wolfgang.minker}@uni-ulm.de

Gregor Behnke, Pascal Bercher, Marvin Schiller, Susanne Biundo, Birte Glimm
Institute of Artificial Intelligence, Ulm University, Germany,
e-mail: {gregor.behnke, pascal.bercher, marvin.schiller, susanne.biundo, birte.glimm}@uni-ulm.de

from everywhere. Hence, the exploitation of distributed, scalable, and most importantly web-based interaction design concepts has to be considered. In literature there are numerous examples for the development and implementation of such systems. For example, Fuchs, Tsourakis, and Rayner (2012) describe a scalable architecture for web-deployment of spoken dialogue systems. The integrated dialogue service relies on grammar-based language processing and rule-based dialogue management combining the Open Source Regulus Platform (Rayner, Hockey, & Bouillon, 2006) and the Information State approach by Larsson and Traum (2000). In order to connect user input to the respective dialogue service, a speech router relying on a session ID is used. Ramanarayanan, Suendermann-Oeft, Ivanov, and Evanini (2015) present a cloud-based dialogue system relying on VoiceXML. In order to manage the dialogue, the open-source dialogue-authoring tool OpenVXML (https://github.com/OpenMethods/OpenVXML) is applied. This framework allows to specify dialogue workflows using a graphical user interface. Zhao, Lee, and Eskenazi (2016) present a multimodal web-based dialogue framework allowing for multi-agent conversation with real users. The proposed framework (DialPort) serves as a platform for the interaction with various remote resources, such as external end-to-end dialogue systems or knowledge bases. For dialogue management they implement a dialogue engine relying on a hierarchical policy in combination with a "Knowledge Ontology" for encoding domain knowledge.

Contrary to the approaches described above, which mostly aim at providing web-services for hosting various dialogue systems and knowledge APIs, we developed a modular and easily extensible cloud-based framework for integrating cognitive capabilities, like Artificial Intelligence (AI) planning. The design of the web-service is structured in the style of the REST (REpresentational State Transfer) paradigm, but not to a full extent, as the REST architectural constraints demand the server to be stateless. Due to the purpose of a dialogue manager to maintain dialogue state and history, our design could not comply with this limitation. In the following, the application scenario for demonstration of a first cloud-based protoype is described. Furthermore, we will provide insight into the underlying architecture and the functionality of the proposed system.

2 Application Scenario

Our current work addresses the implementation of a *companion system* in the do-it-yourself (DIY) home improvement domain. The system is intended to assist novice users in the performance of home improvement projects that require knowledge in the use of power tools (electric drills, saws, etc.). For this reason, the system provides support for the user while performing the task in the form of instructions as step-by-step assistance and in the form of assistance on demand, in case the user asks for specific support. By doing so, the user gains experience in the usage of individual power tools and should be encouraged to employ them in different projects.

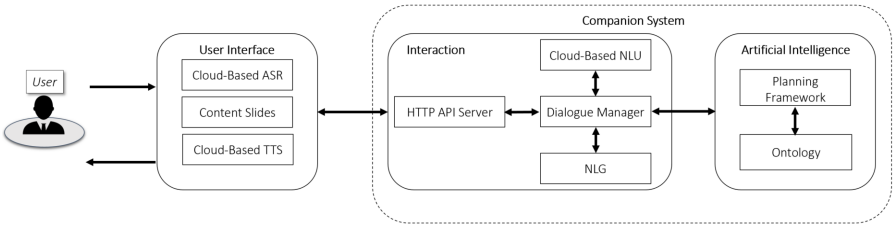

Fig. 1 Modular architecture of the cloud-based *companion system*. The user interacts with the system through a multimodal interface. As controlling entity of the architecture serves a dialogue manager interacting with the modules for planning and the ontology.

3 Cloud-Based Architectural Concept

In order to make our *companion system* web-deployable, a client/server approach is used. On the client side the user interacts with the system through an interface. In this work, we implement a multimodal graphical interface, which is designed as a web service and applicable from every web browser. Therefore, the javascript-based framework *Vue.js* (https://vuejs.org) is used, supporting three different input modalities: speech, text, and touch input. Due to the need of hands-free communication in complex task scenarios, speech input serves as a primary modality. For Automatic Speech Recognition (ASR) we use the Google Chrome web browser's own speech-to-text service in combination with the *annyang.js* (https://www.talater.com/annyang) javascript library. The *annyang.js* library enables the system to allow for continuous speech detection while avoiding the processing of off-topic talk by only transmitting content to the dialogue system after a certain keyword has been spotted, similarly to the activation of current personal assistants. Dialogue output is presented depending on the user's request. As a response to planning requests, a sequence of plan slides using the *vue-slick.js* library is visualised. Explanation and background information are conveyed in the form of text and synthesized speech, also using Chrome's speech service.

The entry to the cloud-based *companion system* forms a HTTP API Server that handles the CRUD requests from the interface server and forwards the conveyed user input to the dialogue component. For each user of the system, there exists a resource on the HTTP API Server in the form of *example.com/companion/user/*$\{id\}$, each containing an individual instance of the dialogue manager. Hence, a user needs to authenticate himself before having access to the system. For this we make use of a token-based authentication approach (https://en.wikipedia.org/wiki/Access_token). This allows for handling multiple, parallel user dialogues.

For semantic analysis of the user requests a statistical-driven Natural Language Understanding (NLU) module is applied. In our architecture, Microsoft's cloud-based *Language Understanding Intelligent Service* (LUIS) (Williams et al., 2015) is used, as it provides a fast and comfortable way of generating machine-learning-

powered NLU models which can easily be extended. In order to determine if the user input is addressing the planning framework, the ontology or the dialogue itself, we develop an individual NLU model for each of them. In doing so, we leverage the full power of statistical NLU that facilitates controlling the dialogue flow and creates an efficient way for interaction processing. When new user input arrives it is forwarded to all three models. After the individual LUIS models have analysed the user input, the response from the model with the highest confidence is selected and transformed into an abstract user act and forwarded to the Dialogue Manager (DM).

The task-independent dialogue management module mediates the interaction between user and the artificial intelligence components of the system and keeps track of the dialogue's state and history in order to provide context-related system output. The current implementation of the DM is based on a version of the classical Information State approach by Larsson and Traum (2000). This approach relies on the three concepts *dialogue move* (user input), *information state* (dialogue state), and *dialogue action* (system output). After each user act the dialogue state (plan- and dialogue-related context information, e.g. sequence of plan actions, previous user input) is updated and a transformed act is forwarded to the respective AI component (planning framework, ontology) of the system. Dialogue-related user acts, like affirmation of information, are directly processed in the DM.

For equipping the system with more intelligence and competence, a planning component provides procedural knowledge, i.e. it generates a valid plan, which consists of a sequence of actions a user has to take in order to reach his goal. The AI planning module of the system applies a hybrid planning formalism, which combines both consequence-based and hierarchical reasoning (Biundo & Schattenberg, 2001). It allows for stating both the effects of actions on the physical world as well as the interconnection of actions. Plans are generated using a novel translation of these planning problems into SAT, which has been shown to have a suitably high performance (Behnke, Höller, & Biundo, 2018).

Furthermore, we make use of an ontology containing domain-specific information which hence is capable of providing declarative background knowledge to the user. It is formulated in the web ontology language OWL (https://www.w3.org/OWL). For a more detailed description of the ontology used in this work and its relation to the planning module see Schiller et al. (2017). Natural Language Generation (NLG) makes use of texts obtained from the ontology, including instructions for each plan action and descriptions and explanations for concepts in the DIY domain generated using text patterns (Schiller, Schiller, & Glimm, 2017) .

The response of the queried module is transformed into an abstract system output and returned to the client, where the interface interprets the message and presents the content in visual and/or spoken form. In Fig. 1 the structure of the system is depicted. The interaction between user and system is based on HTTP messages using the JSON format for data exchange. The communication of the individual components of the cloud-service is handled via socket connections, while also using the JSON data format. In the following, we take a closer look on interaction design.

4 Demo Interaction Design

For demonstration, we will show how the user is accompanied starting with planning a DIY project up to its completion. Each interaction starts with the user stating a planning objective, e.g. “I want to build a keyrack.”. The purpose of this first dialogue is to define a task which forms a valid planning problem for the planner to solve. Once the planner has found a solution, the result is passed back to the DM. As the plan is represented as a sequence of actions with no further information besides the name of the task and the involved materials and tools, the ontology is accessed for more information on each action. This information comprises instruction texts and references to multimedia content (image, video) visualising the content. Afterwards, the enriched plan is presented to the user as a sequence of slides assisting in the performance of the task. Throughout the interaction with the system, the user may ask for background information and explanations regarding certain concepts available in the ontology. This functionality intends to provide the user with further assistance. The current implementation of the system is able to handle three different kinds of conceptual knowledge-based requests: encyclopedic requests, media request, and availability requests. In our application scenario, the user may request further information on DIY items, such as materials and tools. Encyclopedic requests concern the appearance or the purpose of a material or tool (e.g. to tell the user about the properties of a particular class of tool). For receiving an image or a video of a DIY item, a media request can be used. In order to check the availability of certain items, an availability request can be posed.

5 Conclusion

We have implemented a multimodal dialogue framework for the integration of AI components within a cloud-based architecture. In the next iteration of the proposed *companion system*, we aim at including individual user-dependent assistance which takes into account, among other features, the user’s preferences and his expertise. Furthermore, proactive situation-dependent support based on user monitoring is planned as well as integrating the user in the planning process using mixed-initiative planning techniques.

Acknowledgement

This work is funded by the German Research Foundation (DFG) within the technology transfer project “Do it yourself, but not alone: Companion Technology for Home Improvement” of the Transregional Collaborative Research Centre SFB/TRR 62. The industrial project partner is the Corporate Research Sector of the Robert Bosch GmbH.

References

Behnke, G., Höller, D., & Biundo, S. (2018). totSAT – Totally-ordered hierarchical planning through SAT. In *Proceedings of the 32th AAAI conference on AI (AAAI 2018)*. AAAI Press.

Biundo, S., Höller, D., Schattenberg, B., & Bercher, P. (2016). Companion-technology: An overview. *Künstliche Intelligenz*, *30*(1), 11–20. (Special Issue on Companion Technologies) doi: 10.1007/s13218-015-0419-3

Biundo, S., & Schattenberg, B. (2001). From abstract crisis to concrete relief – a preliminary report on combining state abstraction and HTN planning. In *Proceedings of the 6th european conference on planning (ECP 2001)* (pp. 157–168). AAAI Press.

Biundo, S., & Wendemuth, A. (2016). *Companion*-technology for cognitive technical systems. *Künstliche Intelligenz*, *30*(1), 71–75. (Special Issue on Companion Technologies) doi: 10.1007/s13218-015-0414-8

Fuchs, M., Tsourakis, N., & Rayner, E. (2012). A scalable architecture for web deployment of spoken dialogue systems. *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, 1309–1314.

Larsson, S., & Traum, D. R. (2000). Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Natural language engineering*, *6*(3-4), 323–340.

Ramanarayanan, V., Suendermann-Oeft, D., Ivanov, A. V., & Evanini, K. (2015). A distributed cloud-based dialog system for conversational application development. In *Proceedings of the 16th annual meeting of the special interest group on discourse and dialogue (SIGDIAL)* (pp. 432–434). Association for Computational Linguistics.

Rayner, M., Hockey, B. A., & Bouillon, P. (2006). *Putting linguistics into speech recognition: The regulus grammar compiler*. Center for the Study of Language and Information.

Schiller, M., Behnke, G., Schmautz, M., Bercher, P., Kraus, M., Dorna, M., ... Biundo, S. (2017). A paradigm for coupling procedural and conceptual knowledge in companion systems. In *Companion technology (ICCT), 2017 international conference on* (pp. 1–6).

Schiller, M., Schiller, F., & Glimm, B. (2017). Testing the adequacy of automated explanations of EL subsumptions. *Proceedings of the 30th International Workshop on Description Logics (DL)*, *CEUR workshop proceedings vol. 1879*.

Williams, J. D., Kamal, E., Ashour, M., Amr, H., Miller, J., & Zweig, G. (2015). Fast and easy language understanding for dialog systems with microsoft language understanding intelligent service (LUIS). In *Proceedings of the 16th annual meeting of the special interest group on discourse and dialogue (SIGDIAL)* (pp. 159–161). Association for Computational Linguistics.

Zhao, T., Lee, K., & Eskenazi, M. (2016). DialPort: Connecting the spoken dialog research community to real user data. In *Spoken language technology workshop (SLT), 2016 IEEE* (pp. 83–90).