MultiPro: Prototyping Multimodal UI with Anthropomorphic Agents

Philipp Kulms[1], Herwin van Welbergen[1], Stefan Kopp[1]

Social Cognitive Systems Group, CITEC, Bielefeld University, Germany[1]

{pkulms,hvanwelbergen,skopp}@techfak.uni-bielefeld.de

Abstract

Modern user interfaces (UI) often provide natural and multimodal interaction, sometimes modelled in the form of a conversation with an anthropomorphic agent embedded in the system. Designing such interfaces is a challenging task. Furthermore, it is often not clear which user scenarios will profit from a social, conversational interaction paradigm and how it could be integrated with classic paradigms like direct manipulation interfaces. Our multimodal prototyping framework MultiPro helps designers in rapidly designing UIs to explore these questions. It allows the easy prototyping and evaluation of UI using a mix of anthropomorphic (e.g., human-like appearance, speech) and classic graphical elements. We illustrate how designers can specify the look and interaction flow with MultiPro, how MultiPro supports user-based evaluation of the designs, and how MultiPro is used in HCI research applications.

1 Introduction

Human-centered design includes the idea of rapidly testing and evaluating UI prototypes. While there is an abundance of tools enabling rapid prototyping of classic direct manipulation UI, tool support for the prototyping of multimodal UI for early design stages is sparse. We developed MultiPro, a framework for multimodal UI prototyping, to address this need. MultiPro enables the efficient prototyping and evaluation of UI using a combination of anthropomorphic and classic graphical elements. The framework offers support for full multimodal system design: MultiPro provides an interface for input and output modalities, supports multimodal fusion and fission, interaction flow management, and knowledge management. With MultiPro, designers can build and evaluate interactive, high fidelity prototypes. UI are evaluated by tracking how users interact with them, including interaction performance. Early in the design process, evaluations are typically mostly qualitative in nature. This includes, for instance, observing how users navigate an unfamiliar UI, analyzing specific pain points in the navigation, or identifying at which design elements users direct their gaze. Later in the process, more quantitative approaches can be employed, such as measuring task success, time on task,

Veröffentlicht durch die Gesellschaft für Informatik e. V. 2018 in
R. Dachselt, G. Weber (Hrsg.):
Mensch und Computer 2018 – Tagungsband, 02.–05. September 2018, Dresden.
 https://doi.org/10.18420/muc2018-mci-0236

and error rates. Evaluation often involves comparing multiple prototype alternatives, for example to see what added value an anthropomorphic agent brings to a UI. MultiPro supports all these evaluation activities. In this paper, we first discuss the requirements that motivated MultiPro. We then present the design of MultiPro and illustrate how designers can specify the look and feel as well as interaction flow with MultiPro, and how MultiPro supports user-based evaluation of such designs. Finally, we show how MultiPro is used in HCI research applications.

2 Related Work

Early work on anthropomorphic agents provided tools to embed them in graphical UI such as websites (e.g., André et al., 1998). However, the focus has mostly been on multimodal information representation, only limited interaction and input processing capabilities were provided. Today, conversational and social elements are increasingly used in UI that combine such interactions and direct manipulation (e.g., Siri). Current development tools of UI with anthropomorphic agents or speech technology focus on conversational interface design, without explicit support to embed graphical UI elements (e.g., Skantze & Al Moubayed, 2012). If and how agents should be embedded in a UI has been part of ongoing discussion (see e.g. Shneiderman & Maes, 1997). Although anthropomorphic agents leverage more natural interaction and allow the delegation of tasks to algorithms that may be more capable of solving them, they may not be the most efficient tool to solve any task, may give unrealistic expectations, and may diminish user control. A famous example that is mostly associated with a negative user experience is the Microsoft Office Assistant (Clippy). Swarts (2003) explored several design issues of Clippy and concludes that "[…] designing effective user interface agents is hard: many factors – task, situation, behavior, appearance, label – influence users' responses. However, there seem to be sufficient benefits to using such agents to justify continued research to explore how these factors work." (p. 51). We agree with this sentiment and offer a design tool that helps in a systematic exploration and evaluation of the many factors that influence the evaluation of UI with and without anthropomorphic agents. Several toolkits support the prototyping of multimodal interfaces (see Cuenca et al., 2013). However, most, if not all of them limit their functionality to supporting the integration of recognizers, fusing their input, and managing the interaction flow. MultiPro additionally supports multimodal fission, a knowledge source, a blackboard for interaction specific data, synthesizer integration (e.g., for agent behavior, TTS), and uses statecharts as graphical representation of the interaction flow.

3 Requirements

With MultiPro, we aim to support the iterative design process for multimodal UI that combine multimodal/conversational elements such as anthropomorphic agents and graphical UI elements. We therefore support both the design and evaluation of such interfaces. Anthropomorphic agents are typically employed to socially assist users through persuasion, establishing trust, and task delegation. The social presence of agents and their perceived trustworthiness and competence depends critically on both looks and behavior (see e.g. Bailenson et al., 2005).

Therefore, the design of interfaces with agents typically requires employing *high fidelity* prototypes from the start. According to a survey (Myers et al., 2008), in the development of high-fidelity prototypes, designers require tools that allow them to *design not only the 'look', but also the interaction* at a detailed level of UI, that allow *easy exploration*, offer *side-by-side comparison of behavior alternatives*, and support *iterative design*. MultiPro supports the design of the look and interaction of UI. It enables easy comparison and exploration of different interface versions, by swapping out interaction logic (statecharts), 'looks' (screens) and content (from the database). For UI design with anthropomorphic agents, this allows designers to select the most suitable places in the interaction to insert the agent, and to perform direct side-by-side comparisons between agents and graphical UI for a specific interaction. Multimodal UI design tools require *interfaces for input and output devices, multimodal fission and fusion, interaction flow management, and managing and interacting with knowledge sources* (Cuenca et al., 2013). All of these capabilities are supported by MultiPro. In addition to input and output interfaces, MultiPro provides support for the exact component setup, that is: specifying which components to use and how they are configured, including the selection on which sensors are connected to the prototype and which agent is used (i.e., looks and voice). Of special importance for agents is multimodal fission and specifically the *synchronization between modalities* (e.g., gesture, speech) which is essential to convey the meaning of the agent's behavior (Habets et al., 2011). We have embedded a realizer for BML (Behavior Markup Language) to take care of this and extended it to allow synchronization of the agent's behavior with actions on graphical UI elements (e.g., to synchronize pointing gestures with UI highlights). UI can be evaluated by observing user interactions, measuring the usability using metrics, and asking users about the subjective interaction experience, for instance using questionnaires. MultiPro provides tool support for all of these evaluation activities. It supports the analysis of observations by providing functionality to *record* interactions including user, desktop, and eye gaze recordings, and automatic *annotation* of these recordings with the underlying UI-state. The annotations and recordings are exported to the ELAN annotation tool (Wittenburg et al., 2006), where they can be further enhanced with manual annotations. MultiPro keeps track of the sequence of UI-states and the time spent in the UI-state for each interaction. This information can be used to automatically calculate *usability metrics* such as task success, time on task or error rate. MultiPro allows the storage of information entered or selected in the UI. This can, for example, be used to extract information from questionnaires embedded in the UI.

4 MultiPro Architecture

The MultiPro architecture (see Fig. 1) consists of:

1. A statechart that models the interaction flow of a prototype;
2. A blackboard that stores knowledge on the ongoing interaction in key-value pairs;
3. An external database that stores and queries persistent interaction data;
4. Input devices (trigger events in the statechart or write information to the blackboard);
5. Output devices (activated through statechart actions, can present blackboard data).

MultiPro offers a framework to prototype the interaction flow, multimodal fusion and fission, and knowledge management of multimodal systems. Communication between these components is managed by the ipaaca middleware (incremental processing architecture for artificial conversational agents).

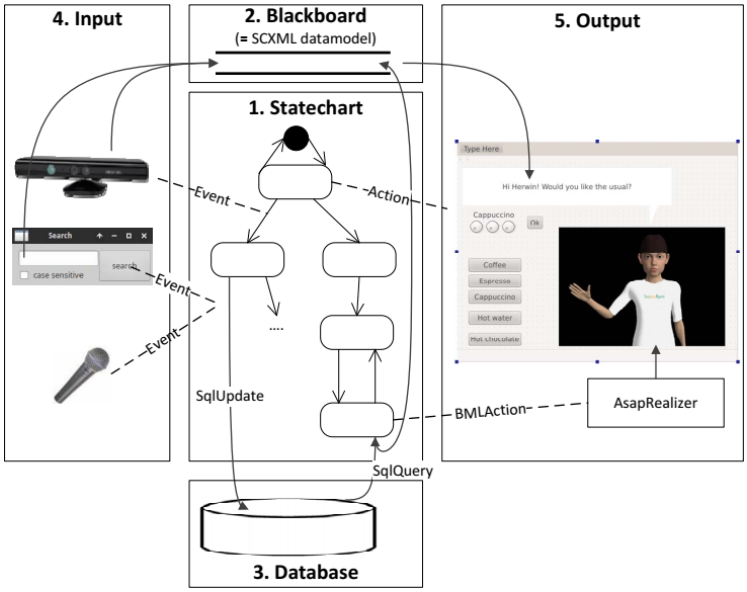

Figure 1: The MultiPro architecture

4.1 Interaction flow management

In a MultiPro prototype, the interaction flow is modeled using a statechart. Statecharts enhance finite state machines with hierarchical composition and concurrency. State transitions are triggered by events. Transitions may contain guards: logical expressions that, when evaluated to be false, prevent the transition from occurring. Actions can be specified to occur in a statechart whenever a transition is triggered, or a state is entered or exited. We use the SCXML standard to specify statecharts and Apache Commons SCXML for running them. Events are triggered by input modalities, while actions are typically used to send information to output devices. An SCXML statechart includes a datamodel and with it offers the capability of storing, reading, and modifying a set of data that is internal to the state machine. The interaction flow in MultiPro is thus specified by designing a statechart that models the relationship between user or other external input, the internal datamodel, and system output.

4.2 Embedding knowledge sources

We connect the datamodel from the statechart with an ipaaca blackboard that can be written on by other ipaaca components such as input and output modules. The blackboard serves as

shared map for the statechart and input and output modalities in MultiPro. The datamodel may be used in logical expressions for guards, to parameterize actions, in direct data transfer from input (e.g., text in a textfield is stored directly in the datamodel), in direct data transfer to output (e.g., text in textfields or labels can be directly set through the datamodel), and in transferring information to and from the database. The database is used to store persistent information about user interactions. We provide an SqlUpdate action to perform an update on the database directly and an SqlQuery action to query the database.

4.3 Multimodal fusion and fission

Multimodal fusion entails assigning a certain meaning or interpretation to a combination of unimodal events, that is, the construction of meaningful compound events out of unimodal events. Such compound events are constructed out of sequential, parallel, alternative or iterative event combinations (Cuenca et al., 2014). Statecharts can represent these relationships in a compact way. Although multimodal fusion and interaction flow are specified in the same statechart, a separation between these functionalities is achieved by using hierarchies. We embed the statecharts for compound events as parallel subcharts in each dialog state from which they should trigger a transition, allowing us to track the final state of compound event charts upon which new transitions to the next dialog state can be specified. Multimodal fission refers to the arrangement and synchronization of information output. Output devices may be directly steered by actions in the statechart or by changing content on the blackboard. UI-elements such as labels and textareas are connected through the blackboard and change instantly on blackboard updates. Output actions include loading new UI-screens, changing part of a UI-screen, or sending ipaaca messages to custom output devices. More fine-grained output fission can be specified by using an action to send out ipaaca messages in BML (Kopp et al., 2006). BML is the de facto standard for the specification and synchronization of behaviors such as speech, gesture, gaze, and facial expression of anthropomorphic agents. We use AsapRealizer as BML realizer because its configuration flexibility allows us to easily add new modalities and select a set of desired output modalities for prototypes (Reidsma & van Welbergen, 2013). The latter means that multimodal fission through BML does not require a mandatory agent, but can be limited to synchronized speech and UI-highlights. Ipaaca messages can be used as additional behavior, allowing us to synchronize such messages with agent behavior. These messages may be used to activate output or to update the blackboard. For example, the agent may point at a specific UI-element, and as soon as its finger is directed at it, an ipaaca message is sent to highlight the UI-element pointed at.

4.4 Supported input, output, and analysis devices

MultiPro can use input devices supported by ipaaca. We have implemented a Qt plugin to connect UI-widgets and touch gesture recognition from Qt such that button presses and touch gestures generate events. The content of content-bearing widgets (e.g., textareas, comboboxes) is linked to the blackboard. Custom ipaaca messages (e.g., from application specific devices) can also be used to trigger state transitions. On the output side, we provide actions to load a UI-screen, insert a UI-screen into the existing one, or enable/ disable UI-elements. The layout

and content of UI-elements can be modified through the blackboard (e.g., changing label content, button highlighting). An action is provided to send custom ipaaca messages, for instance to application specific devices. BML actions allow the expression of coordinated agent gesture, gaze, facial expression, posture, speech, audio, and ipaaca messages. We also support several analysis tools which are configured through actions, including starting and stopping desktop, user, and eye-tracking recordings, starting and stopping state tracking, making a snapshot of selected parts of the blackboard, and combining selected recordings into ELAN files.

5 Prototyping and Evaluation with MultiPro

MultiPro provides a project structure for iterative UI design. Projects contain a prototype and may also contain alternatives for its look and interaction flow. A typical design flow (see Fig. 2) goes from the joint design of the UI-screens and interaction flow, to evaluating the design, to analyzing the results, which then enables the designer to improve the prototype. Here we discuss how these activities are supported by MultiPro.

5.1 Designing the user interface and interaction flow

UI-screens are designed using Qt Creator. This includes deciding whether or not the specific screen contains an agent and if so, where the agent is to be positioned. To this end, MulitPro provides a custom Qt widget for the agent. In addition to loading different UI-screens, MultiPro supports changing only part of a screen, for example to show or hide the agent. The interaction is designed using the open source scxmlgui statechart editor (Fig. 2, bottom left). A typical interaction design first loads a UI-screen using a load action. Loading a UI-screen automatically makes all buttons in that screen available as events in the statechart and connects content-bearing widgets such as textareas, labels, and spinboxes to the blackboard. The content of such widgets can be dynamically changed by updating their corresponding variables in the blackboard. Reversely, any updates in content-bearing elements (e.g., entering text in a textarea) updates the corresponding variable in the blackboard. The blackboard is also used to alter UI-element properties such as color. This can be used to highlight certain UI-elements such as those explained by the agent. The interaction flow is further specified by the events that trigger transitions to new states, and the actions that may occur upon such events. Events include button presses, recognized gestures, and blackboard updates. Actions include changing UI-elements, sending BML to the realizer, sending ipaaca messages and querying or updating the database. To cater for design alternatives, multiple statecharts may be developed concurrently. Designing the interaction includes deciding which data is to be stored for further analysis of the interaction. To this end, MultiPro provides actions to start/stop screen, webcam, and eye-tracking recordings, actions to start/stop tracking the statechart state sequence, and an action to compose an ELAN file based on this data. MultiPro also provides actions to store the current content of the blackboard in a .csv file for further analysis in statistics software.

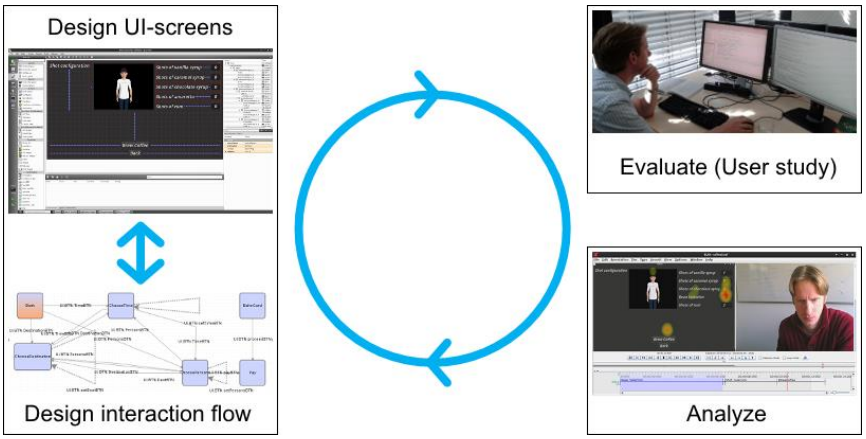

Figure 2: The MultiPro design flow

5.2 Evaluating the user interface and analyzing interaction data

In the evaluation stage, designers test the interaction and run the interaction in user studies. In MultiPro, evaluation involves selecting which design alternative (i.e., which statechart) to run, which input and output components to be used, and how they should be configured. Different evaluation instances can also be created by providing alternative database content. As an evaluation session runs, interaction data is recorded, as specified in the interaction flow. After a session, the recorded data can be combined into an ELAN file. MultiPro allows designers to configure which recorded information should be included exactly. In the ELAN annotation tool, designers can examine the recorded eye-tracking, user, and desktop recordings, annotated with UI-state (see Fig. 3).

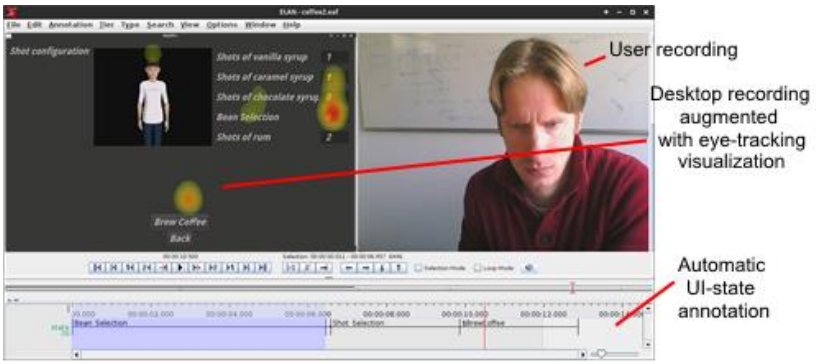

Figure 3: Annotated, recording-based prototype analysis in ELAN

The annotations can be further enhanced manually. The desktop recording is augmented with gaze plots demonstrating where the user looked on the screen. MultiPro allows designers to choose the visualization technique. Currently implemented plots include heat maps (see Fig.

3). Heat maps assign different colors to areas that received low (green) versus high (red) attention, helping designers to monitor the user's visual information search. MultiPro supports quantitative data analysis, for instance by providing actions to append the content of blackboard elements to a .csv file or database. Such .csv files can easily be imported in statistics software for further analysis. MultiPro tracks the state sequence of user interactions and how long users spent in each state. This allows the computation of key usability measures like task success (did the user arrive in the right state with the right content of the blackboard), time on task, error rate (deviation from ideal state-sequences), and learnability (improvement of previous metrics for the same user over multiple interactions).

6 Use Case: Designing HCI Experiments with MultiPro

Using MultiPro, we realized a framework to analyze human-agent cooperation (Kulms et al., 2015). Motivated by the need to investigate trust in intelligent agents capable (vs. not capable) of showing social cues, we designed a UI for an interactive cooperation game used in several experiments (Buchholz et al., 2017; Kulms & Kopp, 2016; Kulms & Kopp, 2018). The application has a graphical UI for the game and a widget for the agent partner (see Fig. 4). The agent can give task-related advice and show emotion displays such as regret or anger after joint goal failure. The game is managed by a custom engine connected to ipaaca, allowing us to use actions in the statechart to send out messages to the engine, for example to set up configurations (e.g., with vs. without on-screen agent) and manage turns. The game sends information over ipaaca messages (e.g., score, turn information) which trigger transitions in the statechart. The collaborative interaction scheme is also managed by the statechart. The framework is thus able to load different interaction schemes corresponding to different experimental manipulations by simply executing different SCXML files. The statechart specifies the agent behavior, controlling its verbal utterances, gestures, and facial expressions. The UI of each study were designed in two iterations. In the first iteration, the goal was to check whether players can solve the game puzzle together with the agent in reasonable time. The focus of this iteration was on evaluating the interaction with the agent, checking the difficulty of the goals, and evaluating the overall usability. This design was tested in pilot evaluations with a small number of participants. For instance, the first analysis revealed users' need for navigating the UI without a mouse. For the second iteration (the actual user study), the controls were optimized according to the results of the first iteration, and the UI was further extended with additional game rounds and the display of game statistics. In sum, setting up the study materials with MultiPro allowed for quick evaluations of the envisioned interaction with the agent across multiple studies. Importantly, the barrier for conducting pre-studies with functional prototypes were significantly decreased with MultiPro. This use case shows how enriching a regular direct manipulation UI with human-like social cues works with MultiPro. To the best of our knowledge, MultiPro is the only framework supporting this kind of prototyping, making it a useful tool for investigating the use of agents in task environments, for instance by comparing two or more interfaces with vs. without on-screen agent, or comparing different agent behaviors.

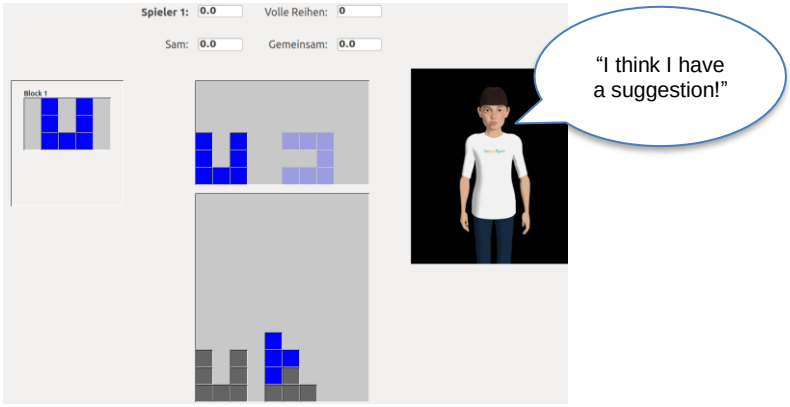

Figure 4: User interface of the collaborative HCI game with anthropomorphic agent as partner

7 Conclusion

MultiPro enables designers to explore whether and where a specific UI will profit from a conversational and/or social interaction paradigm, and how it could be integrated with more classical paradigms like graphical interfaces with direct manipulation. To this end, it contributes the first multimodal prototyping framework which supports the easy and iterative design of UI that use a mix of anthropomorphic agents and graphical UI-elements. We have illustrated MultiPro's capabilities in a use case which shows us that MultiPro may contribute to long standing questions on embedding agents in UI. Our further work focuses on extending the range of sensors and synthesizers that can be plugged into MultiPro, and improving MultiPro's design tools. The latter includes allowing users to design reusable templates for the statechart and UI, for example to setup interaction patterns and provide the functionality to insert such templates into a statechart easily.

Acknowledgments

This research was supported by the German Federal Ministry of Education and Research (BMBF) within the Leading-Edge Cluster 'its OWL', managed by the Project Management Agency Karlsruhe (PTKA), as well as by the Deutsche Forschungsgemeinschaft (DFG) within the Center of Excellence 277 'Cognitive Interaction Technology' (CITEC).

Bibliography

André, E., Rist, T. & Müller, J. (1998). WebPersona: A lifelike presentation agent for the World-Wide Web. *Knowledge-Based Systems, 11*(1), 25-36.

Bailenson, J. N., Swinth, K., Hoyt, C., Persky, S., Dimov, A. & Blascovich, J. (2005). The independent and interactive effects of embodied-agent appearance and behavior on self-report, cognitive, and behavioral markers of copresence in immersive virtual environments. *Presence: Teleoperators and Virtual Environments, 14*(4), 379–393.

Buchholz, V., Kulms, P. & Kopp, S. (2017). It's (not) your fault! Blame and trust repair in human-agent cooperation. *Kognitive Systeme 2017-1*.

Cuenca, F., Vanacken, D., Coninx, K. & Luyten, K. (2013). Assessing the support provided by a toolkit for rapid prototyping of multimodal systems. In: *Proceedings of the 5th ACM SIGCHI symposium on Engineering Interactive Computing Systems*. New York, NY: ACM Press, pp. 307-312.

Cuenca, F., van den Bergh, J., Luyten, K., & Coninx, K. (2014). A domain-specific textual language for rapid prototyping of multimodal interactive systems. In: *Proceedings of the 2014 ACM SIGCHI symposium on Engineering interactive computing systems*. New York, NY: ACM Press, pp. 97–106.

Habets, B., Kita, S., Shao, Z., Ozyurek, A. & Hagoort, P. (2011). The role of synchrony and ambiguity in speech-gesture integration during comprehension. *Journal of Cognitive Neuroscience, 23*(8), 1845-1854.

Kopp, S., Krenn, B., Marsella, S., Marshall, A. N., Pelachaud, C., Pirker, H., et al. (2006). Towards a common framework for multimodal generation: The behavior markup language. In: *Intelligent Virtual Agents*. Berlin, Heidelberg: Springer, pp. 205-217.

Kulms, P. & Kopp, S. (2016). The effect of embodiment and competence on trust and cooperation in human–agent interaction. In: *Intelligent Virtual Agents*. Berlin, Heidelberg: Springer, pp. 75-84.

Kulms, P., & Kopp, S. (2018). A social cognition perspective on human--computer trust: The effect of perceived warmth and competence on trust in decision-making with computers. *Frontiers in Digital Humanities*, *5*.

Kulms, P., Mattar, N. & Kopp, S. (2015). An interaction game framework for the investigation of human-agent cooperation. In: *Intelligent Virtual Agents*. Berlin, Heidelberg: Springer, pp. 399-402.

Myers, B., Park, S.Y., Nakano, Y., Mueller, G. & Ko, A. (2008). How designers design and program interactive behaviors. In: *Symposium on Visual Languages and Human-Centric Computing*, pp. 177-184.

Reidsma, D. & van Welbergen, H. (2013). AsapRealizer in practice – A modular and extensible architecture for a BML realizer. *Entertainment Computing, 4*(3), 157-169.

Shneiderman, B. & Maes, P. (1997). Direct manipulation vs. interface agents. *interactions, 4*(6), 42-61.

Skantze, G. & Al Moubayed, S. (2012). Iristk: A statechart-based toolkit for multi-party face-to-face interaction. In: *International Conference on Multimodal Interaction*, pp. 69-76.

Swartz, L. (2003). Why people hate the paperclip: Labels, appearance, behavior and social responses to user interface agents. Unpublished manuscript. Stanford University, Palo Alto, CA.

Wittenburg, P., Brugman, H., Russel, A., Klassmann, A. & Sloetjes, H. (2006). ELAN: A professional framework for multimodality research. In: *5th International Conference on Language Resources and Evaluation*, pp. 1556-1559.