Supporting Wizard of Oz Experimentation for Language Technology Applications

Stephan Schlögl

Thesis submitted for the Degree of Doctor of Philosophy

School of Computer Science & Statistics

Trinity College

University of Dublin

17 March 2013

Declaration

I declare that this thesis has not been submitted as an exercise for a degree at this or any other university and it is entirely my own work. Wherever there is published or unpublished work included, it is duly acknowledged in the text.

I agree to deposit this thesis in the University's open access institutional repository or allow the library to do so on my behalf, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgement.

_____

Summary

Wizard of OZ (WOZ) is a well-established method used by researchers and product designers to simulate the functionality and user experience of future systems. Using a human wizard to mimic possible operations is particularly useful in situations where extensive engineering effort would otherwise be required to explore a given design space. The method has been widely used in connection with speech and language technologies, but advances in sensor technology and pattern recognition as well as new application areas such as human-robot interaction have made WOZ increasingly relevant to the design of interactive systems.

Even though it is recognised as a valuable prototyping technique, surprisingly little effort has been devoted to exploring WOZ from a wizard perspective. The goal of this thesis is to expand upon the existing knowledge by presenting a systematic investigation and analysis of the potential design space for WOZ prototyping as well as an empirical exploration of appropriate support features for wizards through an iteratively developed web-based prototyping platform. The presented research employed a systematic analysis approach in which the construction and consequent evaluation of a maturing system created new insight into WOZ support and some of its challenges. As part of these evaluations, WOZ characteristics such as wizard workload, wizard consistencies as well as the creation of WOZ experiments, were analysed. The results show that wizards gradually adapt to their task, as would be expected, resulting in faster response times. Consistency, however, is more difficult to achieve and therefore requires additional support. Furthermore, it was found that researchers/designers are generally able to create WOZ experiments, even without upfront training, validating our efforts of providing a platform for more generic support.

In summary, this thesis expands the existing body of knowledge on WOZ prototyping by presenting a structured analysis of the method. Informed by both the literature and a set of investigations (including a series of interviews with researchers from industry and academia), it demonstrates the multitude of motivations for using WOZ, highlights its challenges and investigates potential solutions. Being undertaken alongside the development of a new WOZ prototyping tool, this thesis furthermore explores technical problems and presents possibilities for more generic tool support. Even though the presented work has a strong focus on evaluating language technologies, we believe that the generated insight is applicable to the whole domain of WOZ prototyping, and therefore should be seen as a substantial contribution to the field of Human-Computer Interaction.

Acknowledgement

First I would like to thank all those who took part in the different experimental studies I conducted as part of this research agenda. This concerns colleagues from Trinity College Dublin as well as researchers from University College Dublin and Dublin City University. In particular I would like to thank Alfredo, Erwan, Gerard, Hector, Ilana, Lilly, Martin, Oscar, and Roman for their contributions. I really appreciate their commitment as I know how hard it is to find the time and motivation to take part in other research projects when one's own work is piling up.

Next, I would like to thank those researchers from industry and academia who were so kind to talk to me about their experience with Wizard of Oz prototyping, and the problems they had to solve when running their own experiments. A lot of what they said is usually excluded from scientific publications, and so I am very grateful for their honesty. Their feedback and advice really helped in shaping this work. A big thank you goes furthermore to João and Anne who's actions I was able to analyse over the course of several experiments. Anne's contribution needs to be especially highlighted as she was not only an observed study participant but also an amazing colleague and friend throughout the 4 years we spent together in Dublin. We went through bad times and good ones, always being aware of the fact that our office was the nicest in the building.

Three of those four years we were accompanied by two other colleagues I would also like to give kudos to. Dr. Ielka van der Sluis and Dr. Nikiforos Karamanis were both mentors and friends without whom my path of exploring research would have been a really stony one. Thank you both for inspiring discussions, helpful advice and numerous acts of motivation and support. Also, I would like to thank Hilary for her valuable feedback on the thesis, and of course the Centre for Next Generation Localisation (CNGL) for essentially funding my research. A special thank you goes to my two supervisors, Dr. Gavin Doherty and Dr. Saturnino Luz. They both had always time for me when I needed help, and made me look at things from different perspectives when I was stuck. A supervision meeting was never a pain but always enlightening, and usually brought me a significant step further in my work. And last but not least I would like to thank my parents as well as my sister. Without their support, both mentally and financially, I probably would not have been able to finish this doctoral degree.

Related Publications

- Cabral, J. P., Kane, M., Ahmed, Z., Székely, É., Zahra, A., Ogbureke, K. U., Cahill, P., Carson-Berndsen, J., & Schlögl, S. (2012). Using the Wizard-of-Oz Framework in a Pronunciation Training System for Providing User Feedback and Instructions. *Proceedings of the IS ADEPT International Symposium on Automatic Detection of Errors in Pronunciation Training*. Stockholm, Sweden: June 6-8 (Demo Paper).

- Cabral, J. P., Kane, M., Ahmed, Z., Abou-Zleikha, M., Székely, É., Zahra, A., Ogbureke, K., Cahill, P., Carson-Berndsen, J., & Schlögl, S. (2012). Rapidly Testing the Interaction Model of a Pronunciation Training System via Wizard-of-Oz. *Proceedings of the LREC International Conference on Language Resources and Evaluation*. Istanbul, Turkey: May 23-25 (Full Paper).

- Schlögl, S. (2011). Sketching Language: User-Centered Design of a Wizard of Oz Prototyping Framework. *Proceedings of the IFIP INTERACT Conference on Human-Computer Interaction*, Lecture Notes in Computer Science, 2011, Volume 6949/2011, pp. 422-425. Lisbon, Portugal: September 5-9 (Doctoral Consortium Paper).

- Schlögl, S., Schneider, A., Luz, S., & Doherty, G. (2011). Supporting the Wizard: Interface Improvements in Wizard of Oz Studies. *Proceedings of the BCS HCI Conference on Human-Computer Interaction*. Newcastle, UK: July 4-8 (Work in Progress Paper).

- Schlögl, S., Doherty, G., Karamanis, N., Schneider, A., & Luz, S. (2010). Observing the Wizard: In Search of a Generic Interface for Wizard of Oz Studies. *Proceedings of the Irish HCI Conference*, pp. 43-50. Dublin, Ireland: September 2-3 (Full Paper).

- Schlögl, S., Doherty, G., Karamanis, N., & Luz, S. (2010). WebWOZ: A Wizard of Oz Prototyping Framework. *Proceedings of the ACM EICS Symposium on Engineering Interactive Systems*, pp. 109-114. Berlin, Germany: June 19-23 (Late Breaking Results Paper).

- Schlögl, S. (2010). Sketching Experiences with Language Technology. *Interfaces 85*, pp. 22-23 (Magazine Article).

Contents

List of Tables

List of Figures

List of Abbreviations

ASR Automatic Speech Recognition

AR Action Research

AT Activity Theory

CSCW Computer-Supported Collaborative Work

DM Dialogue Management

ECA Embodied Conversational Agent

GT Grounded Theory

GUI Graphical User Interface

HCI Human-Computer Interaction

LTC Language Technology Components

MT Machine Translation

NLG Natural Language Generation

NLP Natural Language Processing

NLU Natural Language Understanding

SDM Systems Development Methodology

SUS System Usability Scale

TTS Text-to-Speech Synthesis

WER Word Error Rate

WOZ Wizard of Oz

UCD User-Centred Design

Chapter 1

Introduction

"There's a cyclone coming, Em, I'll go
look after the stock"
– Uncle Henry

Early and ongoing feedback is important for building high quality interactive systems. Gould and Lewis [1985] list 'Iterative Design' as one of their three key principles for developing usable products and argue that problems and design faults can be discovered and consequently fixed through early and ongoing user testing. Prototypes, either physical or in the form of software, are valuable instruments for eliciting this sort of user feedback. Examples include paper prototypes (e.g. Bailey et al. [2008]), sketches (e.g. Kieffer et al. [2010]), and wire-frames (e.g. Li et al. [2010]) as well as 3D prototypes (e.g. Séquin [2005]) and more advanced mock-ups (e.g. Aleksy et al. [2010]).

Combining novel speech and language technologies into systems across diverse domains poses significant software engineering challenges. Assuring the usability of the resulting applications is an important, and somewhat under-researched, issue in this area. The use of Language Technology Components (LTC), i.e. for example Automatic Speech Recognition (ASR), Machine Translation (MT) and Text-to-Speech Synthesis (TTS), has significantly increased in recent years as their performance has improved. Examples include speech-based interaction in cars to keep a driver's attention on the road and the use of web-based translation tools such as GOOGLE TRANSLATE[1] and MICROSOFT BING TRANSLATOR[2] to understand text written in a foreign language.

Due to the fallible nature of the technologies involved in such applications (e.g. the low recognition accuracy of an ASR system triggering the wrong system action, an ambiguous MT result due to context-free interpretation of a source text, or the metallic sounding voice of a TTS function mixed with pronunciation errors that lead to misunderstandings), one must ensure that usability and human factors are adequately accounted for. As with applications based on a

[1] http://translate.google.com/ [Accessed: July 16^{th} 2012]
[2] http://www.microsofttranslator.com/ [Accessed: July 16^{th} 2012]

Graphical User Interface (GUI), software using LTCs needs to be tested early in the design process. Whereas low-fidelity prototypes assessing GUI applications such as sketches and wireframes can be built relatively quickly and inexpensively, the development of prototypes evaluating the usage of LTCs can be both cost and time intensive. In order to be able to obtain basic user feedback on an envisioned interaction, those systems would usually require at least several hours of recorded speech, valid transcriptions, an implemented framework of rules as well as a solid error-recovery strategy. Hence, a more efficient way of prototyping this type of technology is desirable.

One technique that has been used to test applications involving LTCs is Wizard of Oz (WOZ), where a human, the so called 'wizard', mimics the functionality of a system. Since the technical requirements for such a prototype can be reduced to a minimum, this technique is particularly useful for early stage evaluations and thus a good candidate for addressing the lack of low-fidelity prototyping methods supporting the use of LTCs. However, the task of the wizard is highly demanding, thus supporting it through a usable wizard tool seems crucial. While several wizard tools have been built to date, most of them were designed with specific experiments in mind. The more general issue of understanding the task of the wizard so as to provide a generic WOZ prototyping platform has remained largely unexplored. In order to change this the goal of this thesis has been to gain a better understanding of the purpose of WOZ prototyping and the types of analysis it is used for, and then furthermore explore the task of the wizard and some of its challenges. The research was conducted alongside the development of a new, web-based WOZ prototyping platform, whose goal is to support language-based and audio-visual WOZ experimentation in a more flexible and scenario independent fashion.

1.1 The Wizard of Oz Prototyping Method

Wizard of Oz (WOZ) constitutes a prototyping method that is used by researchers and designers to obtain feedback on functionalities that would require significant resources to be implemented. In a so called 'WOZ experiment' a human 'wizard' mimics the functions of a system, either entirely or in part, which makes it possible to evaluate potential user experiences and interaction strategies without the need for building a fully functional product first (Gould et al. [1983]). The application area of WOZ prototyping ranges from very low-fidelity, where researchers might refer to paper prototyping as being a form of WOZ, to high-scaled simulations in which the wizard only takes over a very distinct functionality of an envisioned system. Similarly, the types of experiments may vary. On the one hand we find set-ups where a test participant is openly aware of the simulation, sometimes even taking over the task of the wizard in order to better convey a constructive argument, on the other hand it is often of high importance that participants believe that they are interacting with a real system, so that their actions and consequently their feedback is just as real. Hence, in a way WOZ experimentation can be used to elicit user feedback throughout the entire development cycle of a new system. It helps at the beginning by expanding low-fidelity prototyping methods such as interface sketches and story-

boards, throughout the development process by simulating missing functionality and offering a way to compare different design solutions, as well as towards the end of a development cycle when it can be used to augment system components whose quality levels are not sufficient and therefore would require additional algorithm training and fine tuning. In addition to these three application levels, which focus mainly on the development and improvement of a system, WOZ is also used as a tool to collect various types of corpus data. The field of computational linguistics, in particular, makes extensive use of the method in order to gather the relevant speech and language corpora underpinning their services. The following sections discuss these different motivations for employing the WOZ prototyping method in more detail and highlight their distinct characteristics.

1.1.1 Exploring the Design Space

While the general idea of WOZ experimentation remains the same no matter at what point of system development it is used, the set-up may vary at different development stages. At the very beginning, for example, the most basic form of the method can use a wizard to interpret mouse clicks and keyboard entries, where the actual setting can be as simple as flipping through a collection of user interface sketches. Here a user would touch on a sheet of paper (to simulate a button press) and the wizard would present the follow-up screen associated with this button press. In a similar experiment, Liu and Khooshabeh [2003] used sketches and post-it notes to represent screens of their envisioned KITCHEN-NET system. The wizard manually updated these artefacts based on spoken user input. Even though in this very basic form of WOZ prototyping a user is aware of the simulation, important feedback can be gathered. In fact, often it is this improvised and rough setting that allows for unconstrained reactions. That is, users might be reluctant to criticise a product which is, at least to some extent, functional and therefore has consumed a certain (undefined) amount of development time. In the case of a paper-based sketch, however, most people would provide unconstrained feedback, since the effort that has been put into building this prototype is perceived as rather low. Similarly a developer would be more likely to make radical changes to an envisioned system functionality if the time and resources spent on building the prototype were low. While one could argue that simple sketches or storyboards would achieve the same result without the effort of running an actual ‘experiment’, it is rather this interactive aspect of WOZ which makes the method so valuable. It helps researchers and designers to better understand an envisioned design space and reflect upon it. That is, it permits them to explore various possible design options and allows for optimisation with respect to functionality, user experience or cost. The user input in this situation serves both, direct feedback on an anticipated functionality as well as a new perspective on an overall concept. Schön [1984] states that a designer ‘reflects-in-action’ when interacting with a sketch or paper-prototype. He argues that the actual process of creating such an artefact already influences and changes initial thoughts. Observing a third person using this artefact (i.e. the prototype) may therefore add an additional dimension to this initial stage of

system design. In this respect early stage WOZ experimentation can be seen as a form of what Soegaard [2010] calls 'Explorative Prototyping' which serves as a communication medium between users and designers. It provides an artefact that is used to discuss pros and cons of an envisioned design idea. The designer in that situation is not necessarily searching for the approval of the presented idea but rather seeks to understand whether the concept is workable. However, the fact that any actual functionality is missing allows, sometimes even asks, for alternative solutions. A participating user is able to freely express his/her thoughts and question the product's functionalities. As the simulation is visible and fully integrated in the discussion it is furthermore possible to adapt to a user's propositions. Alternatives can be explored without changing the test protocol. From an analysis perspective 'Explorative Prototyping' does not aim to collect data for statistical analysis. It seeks the dialogue with the user in the early stages of product design. Not forcing participants into the boundaries of a controlled interaction with a premature prototype (given that at this stage of product development most solutions need to be regarded as untested and premature) following a defined test procedure, creates the stage for this discussion, where the simulation of functionalities can be seen as an instrument. Similar to pen and paper it allows both parties to interact with a medium and to exchange ideas. In such a situation it is not unusual that at some point the participant takes on the role of the simulator, explaining the favoured way of interaction to the designer. As with other 'Participatory Design' instruments (cf. Schuler and Namioka [1993]) the user becomes an integrated part of the development team, where several of these types of discussions eventually help to collect and better understand user requirements. Furthermore theses discussions might trigger awareness of hitherto unconsidered aspects of the design. Hence, without investing a great amount of time and resources, exploratory WOZ prototyping is able to provide valuable insights at the very early stages of product development.

1.1.2 Probing User Actions

In a more advanced product development stage the simulation conducted by the wizard is often hidden from a test participant, so that, at least to some extent, the illusion of a working system is created. With this approach users think they are interacting with an actual product, responses, however, are produced by the wizard who might be sitting in a different room connected through audio, video, and/or different sorts of screen-sharing technologies. Usually this set-up requires a dedicated wizard application interface to be built so that it is possible to simulate envisioned system functions. The resources that are put into building this interface should, however, be kept to a minimum. At this stage the actual product is still a prototype and hence should be treated as such. High development costs at this point may otherwise lead to reluctance in making necessary changes to the design. One might argue that it would be better to save the time and resources that go into building this - what Soegaard [2010] calls an 'Experimental Prototype' - and instead invest the effort into working on the actual product. However, doing so would even further foster the problem of reluctance, where it is unlikely

that an already taken design path would be left after a considerable amount of resources had been spent on it. Treating this WOZ supported prototype as a separate (low-fidelity) system that solely functions as a tool for gathering user feedback, however, cuts the direct connection with the final product, reducing reluctance to implement changes, and consequently acts as an information source based on which design decisions can be taken. On the other hand one might argue that similar information could also be gathered by simply interviewing potential users. While, interviews are a useful and valid instrument for defining the requirements of a future product, they fail when it comes to eliciting non-declarative or tacit knowledge. In other words users would fail to report on actions they are not aware of since they happen unconsciously. Interacting with a (simulated) prototype helps to bring this knowledge to the surface. Furthermore it tackles the so called 'say-do problem' (Goguen and Linde [1993]) which describes the difference between what users say they are doing and what they actually do. Again, the mismatch here is usually not caused by a user's intention but rather linked to the fact that routine tasks are predominantly controlled by tacit knowledge, and so it is difficult to describe these activities in great detail. Observing users interacting with a prototype, however, allows for the probing of certain actions and consequently may provide the designer with relevant insight. As the simulation in this sort of WOZ experimentation is hidden, real user actions can be expected. While one of the problems of experimental user testing, namely the missing context, usually still remains, the gathered data can be used for both qualitative (e.g. user satisfaction, user feedback, etc.) as well as quantitative (e.g. task completion rates, task completion times, etc.) analysis. Compared to traditional prototype evaluations this form of analysis lets the designer also test 'what if' scenarios. That is, while with traditional prototypes the gathered feedback is limited to what the tested technology is capable of, with WOZ it is possible to evaluate better performance. For example, while perfect automatic speech recognition is still not achievable with current technological solutions, WOZ can be used to simulate such a case. Potential interaction scenarios can therefore be sufficiently explored which allows for resources to be spent in a goal-oriented fashion. In summary, this second form of WOZ prototyping can be seen as a valuable tool, not only for designing the product but also for steering the distribution of future resources.

1.1.3 Tuning Technologies

Finally, WOZ experimentation is often employed towards the end of a product development cycle. Here it may be integrated with the actual product and be used to simulate or augment a system function that is either not yet implemented or needs considerable improvement in order to convey the quality that is intended. Similar to WOZ used within the experimental stage, it is usually necessary to build a dedicated interface for the wizard to simulate the envisioned system function. Often this interface is more sophisticated than during earlier stages and its functions are tightly integrated with the test system. Furthermore, as at this point the rest of the product may be of release quality, the task of the wizard is to resemble or even augment an

already working system. Additional training for the human wizard, therefore may be needed in order to assure for correct and consistent wizard actions. While WOZ experimentations in earlier stages of product development serve the exploration of a design space and help to choose the right design path, evaluations towards the end of the development cycle are rather used to fine-tune system functions and evaluate technology components. Novel features, in particular, whose necessary quality level is often hard to judge, can benefit from this type of evaluation. The wizard can be used to augment the system function, which allows for the evaluation of potential user experiences with different levels of quality. Such may help to decide on whether it is necessary to invest additional time in improving the function or if these improvements are not needed as users might not perceive a significant difference (e.g. it can be evaluated which error level is acceptable for a recognition system to be used in a certain context).

In the late stages of the development cycle, where the wizard task at that stage is usually tightly integrated with the rest of the system, the goal of the experiment is to convince a participant that he/she is interacting with a final product. Signs of simulation should not be spotted by the user. While some product development teams might be able to recruit professionals to act as a wizard, it is usually one of the team members who is entrusted with this delicate task. Compared to WOZ conducted early in the development cycle the potential for generating additional design insight is smaller. As the task is more defined it shows less room for creativity and consequently may even be experienced as relatively monotonous. Nevertheless, the added value can be found in the ability to collect sufficient data to be able to deduce significant results. Hence, ground-breaking conceptual changes may not be discovered, but researchers/designers might aim at validating an already taken design path and seek to find those aspects that still need improvement. To do so, WOZ experimentation at this stage usually requires a significantly greater number of participants. In order to be able to produce the relevant data it is furthermore necessary to control for different experimental effects that could potentially bias experiment results. While such a strict experiment design is less important with more exploratory WOZ evaluations where the goal often is to map the design space and search for alternative solutions, here too much flexibility would have the opposite effect. If one, for example, looks for quantifiable results, he/she does not only need to conduct experiments with numerous test participants, but it is also important that these experiments provide a consistent user experience. Otherwise the resulting data may not be valid. However, with more participants the cost of the evaluation increases. While WOZ experimentation at the beginning of a development cycle can be regarded as a cheap design method, using it later in the cycle means dealing with significantly higher costs. Nevertheless, compared to the amount of resources that might be wasted without conducting appropriate system evaluations, the price of WOZ prototyping can be seen as reasonably small. Hence, in summary this third type of WOZ evaluation, which is conducted towards the end of a product development cycle, does help to define the amount of additional improvement that is needed for a function to be acceptable. Furthermore it should produce sufficient data so that a taken design path can be validated and remaining frictions, which might still hamper a good user experience, may be identified.

1.1.4 Collecting Data

In addition to using WOZ to simulate currently missing system functions, researchers/designers also employ the method to gather various types of interaction data. In most cases this concerns speech and language corpora, which are considered essential components for building natural language-based applications (e.g. Life et al. [1996]). Other examples include corpora of gestures and movements (e.g. Zobl et al. [2001]) or facial expressions (e.g. Bevacqua et al. [2010b]). Similar to exploratory studies described earlier, experiments that aim to collect corpora usually happen without having an actual working technology available. Nevertheless the simulation needs to be convincing and possibly consistent as the collected data constitutes the foundation, on which future technologies are built and trained. The quality of the resulting system depends on the number but also on the naturalness of the collected interactions. Hence, the main requirement for WOZ employed as a corpus gathering instrument is to trigger realistic user behaviour. While the previously discussed use cases often focus on the qualitative analysis of experiment results, where interaction strategies and user experiences play a crucial role, for corpus collection the main goal is to generate a sufficient number of iterations so that eventually a mathematical model of the interaction can be built. In other words, while in the previously discussed types of WOZ experimentation the actual product and its state of development is the focus of an experiment, for corpus gathering this direct relation between simulation and final product is not compulsory. The method is rather used as an instrument to elicit natural user data, which is consequently analysed and used to train different sorts of intelligent systems.

1.2 Drawbacks of Wizard of Oz Prototyping

As explained above, WOZ is a valuable prototyping method that can be employed throughout all the different stages of product development. Yet, despite its advantage of simulating functionality and obtaining feedback without upfront investment into building the relevant technology, it should not be regarded as a particularly cheap evaluation method. In particular, the need for a dedicated person that simulates functionality (i.e. the wizard) increases labour cost. While in the early stages of product design, where WOZ is usually employed as an exploratory evaluation method, the designer can take on this role (reducing costs), more advanced simulations might require specialised skills. But even if the designer/researcher performs the simulation task, the effective experimentation costs might still remain high. Throughout an exploratory evaluation phase WOZ serves as a direct design instrument in which the participating parties (i.e. the designer/researcher and an experiment participant) define the frontiers of the potential application space. Hence, at this point it can be regarded as a typical design task. Later on, however, conducting an experiment that realistically simulates a system function with the aim of obtaining user feedback, needs to be seen as a separate task happening in parallel to the actual design process. Here the goal is to effectively produce measurable data from which, after being analysed, design decisions can be derived. WOZ employed in the early stages of product

development therefore serves as a direct design instrument. Similar to a pen sketching different solutions on a sheet of paper, the wizard can simulate varying system functions and discuss with a user their potential. During a realistic simulation, however, the designer's role is limited to passive observation (in case he/she is acting as the wizard the interaction is usually bound by a predefined set of of rules on how to interact with a test participant, which in most cases does not include a direct discussion about the tested design) which constitutes additional time resources spent outside the actual design task. Hence, one could argue as the development of a product advances so the experiment costs for WOZ prototyping increase, with costs arriving at their peak at the end where perfect and consistent simulation requires a highly qualified and trained wizard operating a usually rather elaborated wizard interface, and where furthermore the aim for statistically significant results demands a qualifying number of test participants.

The latter aspect can generally be seen as a drawback of experiment-based prototyping. As with other experimental analysis (e.g. usability testing, A/B testing, etc.), WOZ experimentation results heavily depend on the number of participants tested. While in the early stages of design insight is usually generated through exploration and discussion, in later stages it is preferable to base design decisions on (more or less) hard numbers. Also in the case of a corpus collection task the number of participants represents an important factor, directly influencing the validity of the derived interaction model. Yet, WOZ testing with high numbers of participants seems difficult. It is true that the recruitment of suitable people is a challenge shared by all user testing methods, however, WOZ also needs to deal with restricted flexibility when it comes to time and place. While other test settings are sometimes able to significantly increase the number of participants by employing remote testing services or by simply looking at log files, the complexity of a WOZ set-up usually requires people to travel to a dedicated lab. This furthermore highlights the challenge of organising time-slots, as also the schedule of the wizard (being a separate member of the experiment team) needs to be integrated into the test cycle. New technologies might allow for more flexibility, for example by using web-based WOZ tools, yet the amount of necessary coordination remains high. Furthermore, the wizard, especially during the later stages of system development, needs to be considered as an additional recruitment factor.

A final drawback of WOZ can be found in the general aspects of human simulation. Again the early stages of product development are less affected as here the simulation is visible and rather used as an instrument for probing discussion. Experiments in which the wizard's actions are hidden, however, heavily depend on this simulation aspect. Varying as well as unrealistic wizard behaviour can lead to biased experiment results. That is, on the one hand the wizard faces the task of mimicking a system function which at this point has not been built and for which therefore a clear definition is missing, on the other hand he/she also faces the problem of running the simulation consistently. While sufficient training may help to deal with the latter, it can be difficult to develop the necessary set of rules in order to always mirror realistic system behaviour. The result is therefore often a test setting in which the wizard sometimes unconsciously, sometimes knowingly adapts to a participant's behaviour. Hence, especially

throughout the more advanced stages of product design, where the decisions should be based on measurable results, such variations can lead to incorrect interpretations.

In summary, the main challenge WOZ prototyping is facing can be found in its human-centred nature. The dependency on human participants and the accompanying recruitment process significantly increases experiment costs while decreasing flexibility. Also, the human wizard constitutes an additional cost factor on the researcher side of an experiment and he/she also represents a potential weakness when it comes to controlling for possible side effects. Hence, gathered results are often regarded as less meaningful when compared to those produced by computer-driven experimentations.

1.3 Motivation of this Thesis

While there are surely challenges to overcome, overall WOZ experimentation can be seen as a flexible evaluation method that is applicable in different forms throughout the development process of a new product or service. In the early stages its value lies mainly in exploring the design space by observing users interacting with an unfinished, sketch-like representation of a system. Further along the development road, it is used to 'realistically' simulate system functionality, which prevents the investment of too much time and effort into building something that might not work. Finally, towards the end of the development cycle WOZ can be employed to fine-tune components and evaluate their required level of quality. At all stages the method needs to be seen as an instrument to elicit user feedback, which can be used to eventually build a better product.

Yet, similar to other user-centred evaluation methods, it is often exposed to criticisms as it can require significant amounts of time and resources to be used. WOZ prototypes are usually not part of any final product and here the cost of building them needs to be justified by the design team. Furthermore evaluation results, as with other user-based experimental analysis, heavily depend on the number of participants that can be recruited. Having to source, coordinate and brief participants involved in user-based experiments not only requires time (which can be directly transformed into labour costs) but also a certain amount of flexibility. Having an additional human, i.e. the wizard, being an integral part of the the experimentation further increases the complexity level of the evaluation method. Also, this human element is often seen as a weakness as the produced performance and especially its consistency can vary without sufficient upfront training, leading to unreliable experiment results.

Despite this criticism, numerous examples have shown that the WOZ method is an effective and valuable instrument for informing the design of novel technologies (e.g. Kelley [1983], Gould et al. [1983], Fraser and Gilbert [1991], Salber and Coutaz [1993a], Maulsby et al. [1993], Andersson et al. [2002], Höysniemi et al. [2004], Serrano and Nigay [2010], etc.), and therefore should not be neglected when it comes to building and evaluating systems that heavily depend on complex technology components. In order to improve our understanding of the WOZ method and the challenges that are involved, this thesis therefore presents a

structured analysis of previously conducted WOZ studies as well as the tools and frameworks that have been used to support them. The goal was to identify context specific requirements and explore how more generic tool support might be achieved. Following a methodology which integrated theoretical analyses with system development, the aim was to construct a WOZ tool that reduces the earlier outlined complexity and increases experiment flexibility. Furthermore we wanted to better understand the task of the wizard and its different roles. While the simplicity of its human simulation is generally regarded as the main strength of WOZ, it has also been identified as its biggest challenge, and so we believe that a closer examination of this aspect is required. Consulting with researchers and designers who have experience with WOZ prototyping was one way to collect relevant insight, looking at wizard actions and how they may change over time, was another. In summary the motivation for this work was to expand the body of knowledge on the WOZ prototyping method and decrease its overhead costs by developing a tool that better supports the wizard task and furthermore may be employable beyond the frontiers of a single experimental setting.

1.4 Contributions of this Thesis

This thesis methodically explores previous work in the area of WOZ experimentation as well as the tools that have been employed. In doing so we specifically look for challenges involved in the design and conduct of a study. The goal is not to criticise previous experiments and solutions but rather to highlight how distinct motivations have led to a variety of different tools whose features may to some extent be overlapping.

The main contributions of the first part of this thesis is therefore an extensive discussion of existing WOZ tools, with a special emphasis on language technologies, and how they have been used in the past. Sub-contributions include a derived set of requirements for generic tool support as well as a list of recommendations for successfully running experiments. As such the objective of this part of the thesis is mainly concerned with answering the following question:

"What is the nature and application area of WOZ with respect to language technologies and for which concrete scenarios can it be used as a low-fidelity prototyping method?"

In a next step the gained understanding is used to build a WOZ prototyping platform whose main contribution is to reduce the costs of building a new WOZ tool every time a new use case is to be evaluated. A critical sub-contribution of this platform is a generic and highly flexible architecture for WOZ prototyping that not only offers a way of integrating existing technology components but also handles the different roles a wizard can play when interacting with them. Thus this part of the thesis tackles the following question:

"How can the role of the wizard be supported and what features can be implemented into a generic WOZ prototyping platform that would significantly easy the burden of this task?"

Finally the main contribution of the last part of this thesis is a set of evaluations that employ different releases of the WOZ prototyping platform in order to explore distinct aspects of the wizard task, namely wizard workload, wizard consistency and experiment construction. As such it focuses on answering the following question:

"How does wizard performance vary over time and what experience is necessary to successfully design and run experiments?"

In summary this thesis is concerned with improving our understanding of the field of WOZ prototyping and how it varies between application areas. In doing so our analysis puts a strong emphasis on the simulation of language technologies but also includes other types of WOZ settings. The goal is to provide a structured overview of the method and generate a summary of rules and best-practices for successfully conducting WOZ studies; an aspect that has not been explored so far. The generated results are furthermore used to better understand the task of the wizard as well as the given design space for WOZ experimentation. In particular we are investigating WOZ tool support and how it can be improved. This is mainly motivated by the current lack of software programs and applications that let researchers/designers simulate technology but also by our own demand for a more generic solution. Finally, our initial analysis of the literature has furthermore shown that the task of the wizard and its specific challenges are rarely subject to investigation. Hence a fundamental part of this thesis also explores various aspects of the wizard task (i.e. wizard workload, wizard consistency and the construction of WOZ experiments) and how they may be better supported.

1.5 Structure of this Thesis

The structure of this thesis is as follows: Chapter 2 starts the path of exploration with an extensive analysis of previous work on WOZ. Chapter 3 describes the methodology used to tackle the above listed research questions. Chapter 4 describes the initial analysis and prototyping efforts. Chapter 5 elaborates on the wizard task, its different roles and their influence on a generic tool architecture, and Chapter 6 discusses the final WOZ prototyping platform and reports on a set of evaluations with respect to wizard workload, wizard consistency and experiment construction. Finally, Chapter 7 summarises the results and exemplifies future plans.

Chapter 2

Related Work

"Who are the wizards?
– Dorothy

Wizard of Oz experiments have previously been applied in a variety of different research areas and it is worth highlighting some of this work in order to demonstrate the quality of the WOZ method. We start our exploration of the field by first going back to its very origin and discussing the history of WOZ prototyping (cf. 2.1). From there we more closely look at the various application areas (cf. 2.2) before eventually focusing on Language Technologies (cf. 2.3) and the challenges involved in prototyping this natural form of interaction.

2.1 The History of Wizard of Oz Prototyping

Borrowing its name from the famous novel by Baum [1900], the Wizard of Oz (WOZ) prototyping method was introduced in 1983 by Kelley [1983] who initially referred to it as 'The OZ Paradigm'. At the time 'OZ' was also taken as an acronym for 'Offline Zero' which referred to the fact that a wizard is interpreting a user's input in real time[1]. In his paper Kelley argues that human simulation can be an efficient empirical method for developing user-friendly natural language applications and he supports this argument by showing that only a small number of experiments (i.e. in his study he used 15 participants) and an iterative development methodology (i.e. results were integrated iteratively after every experiment run) are needed to capture most of the vocabulary and dialogue structures required to successfully operate a calendar program using natural language input. Even though Kelley coined the name Wizard of Oz (with respect to prototyping), several researchers before him, used simulation as a means of exploring interactions. Two important examples include Erdmann and Neal [1971] and Bobrow et al. [1977], both of whom tested early concepts for interacting with a flight booking system. While Kelley and his predecessors mainly focused on text-based natural language interaction, Gould

[1] http://www.musicman.net/oz.html [Accessed: April 25^{th} 2012]

et al. [1983] explored the possibilities of a computer processing spoken input. Using a highly trained secretary they simulated an intelligent system that would understand any English sentence and display it on a terminal screen. Unlike earlier studies, Gould and colleagues were, however, mainly interested in the experiences of users and how they felt about using speech as a means for operating a computer system, and therefore they did not build an actual working prototype. Thereafter several researchers employed this new way of simulating interaction. Good et al. [1984], for example, used WOZ to elicit natural user behaviour, based on which they built a user-derived command language. In 1984 the method was the centrepiece for prototyping a multi-lingual message system for the Olympic games (Gould et al. [1987]) where it was used to define appropriate help messages and also used to obtain early feedback on the envisioned system interface. Around the same time Landauer [1987] highlighted its strength as a quick and cheap usability evaluation technique that can be used very early in the development cycle, before Chignel [1990] listed WOZ as a dedicated type of formative evaluation. Subsequently Hill and Miller [1988] and then Carroll and Aaronson [1988] employed it to explore more closely the use of intelligent advisory interfaces. More specifically looking at speech-based interaction, Jönsson and Dahlbäck [1988] used WOZ to point at distinct differences between talking to a computer and talking to a human. Additionally Fraser and Gilbert [1991], who coined the expression 'Pay No Attention to the Man Behind the Curtain (PNAMBIC)' proposed a taxonomy for the simulation of speech systems. Extending the application of WOZ from text- and speech-based interaction to other modalities first Hauptmann [1989] and then later De Marconnay et al. [1993] used it to evaluate gestures and face recognition. This expansion in scope continued with Salber and Coutaz [1993b] who looked at multi-modal interaction, leading to the introduction of multiple wizards. Finally, Dahlbäck et al. [1993] as well as Bernsen et al. [1994] presented general analyses of the WOZ method and Wooffitt et al. [1997] dedicated a complete textbook to the subject. In the following sections we highlight some of that work in more detail.

2.2 Areas of Application

In previous years WOZ experiments have been used for a variety of purposes, including prototyping multi-modal information retrieval (Rajman et al. [2006]), testing speech-based flight booking systems (Karpov et al. [2008]) and simulating a virtual doorman (Mäkelä et al. [2001]). Exploring rather open interaction spaces, Bradley et al. [2009] used WOZ to evaluate a user's experience when interacting with a web-based social companion, Goldstein et al. [1999] employed it to investigate navigation in voice-controlled dialogues, and Davis [1998] tested the advantages of active help for the use of an unfamiliar software application. In terms of scope, the application area of WOZ ranged from designing dialogues (e.g. Howell et al. [2005]) to collecting text and speech corpora (e.g. Benzmüller et al. [2003]) and exploring interaction strategies (e.g. Yang et al. [2000]). As with low-fidelity prototyping methods for software based on GUIs (Graphical User Interfaces), WOZ played a role in shaping application design

and helped to improve the naturalness of an interaction. The method supported designers in producing appropriate dialogue models as well as allowing them to improve their understanding of a domain. A final area in which WOZ was found to be helpful is the exploration of emotions (e.g. Scherer and Schwenker [2008]) and social aspects of human-machine interactions (e.g. Deruyter et al. [2005]).

In order to discuss WOZ examples in a more structured way the remainder of this chapter tries to categorize them in different application domains. While these domains are chosen based on our studies of the literature we would like to note that they are neither exclusive nor clearly defined. Rather, we see the borders between them blurred for which it is not always entirely clear where to place an example. Furthermore we understand that other researchers might use a different classification scheme. However, the purpose of clustering WOZ examples into different application domains is to show the different perspectives of WOZ prototyping and consequently the variety of interests that need to be understood when one seeks to explore the method, rather than to define an analytic classification scheme for WOZ.

2.2.1 Wizard of Oz and (Spoken) Natural Language

From the beginning, WOZ experimentation has been used to study human language and how it can be used as a means to interact with computers. Ever since Kelley highlighted its benefits as part of an iterative design methodology for natural language applications (Kelley [1984]), researchers have employed WOZ simulations to study different aspects of language-based human-machine interaction. The universal understanding at the time was that human-computer dialogues should be modelled after their human-human counterparts, and in order to test this assumption Carbonell [1983], for example, explored discourse pragmatics and ellipsis and Guindon et al. [1987] conducted a WOZ study to compare grammatical structures of text-based human-machine interactions with the ones of human-human dialogue. Dialogue studies continued by putting a great emphasis on identifying different classes of discourse (Stenton and Whittaker [1989]) and how they are best represented by natural language interfaces (e.g. Dahlbäck and Jönsson [1989]; Dahlbäck and Jönsson [1992]). It was then Dahlbäck et al. [1993] who first presented a holistic description of WOZ prototyping with respect to the study of human-machine discourse, further emphasising the argument that people do change their behaviour when they know they interact with a system rather then a human being, and that the method needs to account for that.

At the same time first studies were reported which aimed at expanding the method towards spoken dialogue analysis (e.g. Pilkington [1992]). Here initial results showed that the influence of using distortion mechanisms such as a vocoder to help deceive and make people believe that they are interacting with a machine (i.e. a vocoder transforms a human voice into slowly varying electronic signals which lets the output sound rather metallic) was found to be less influential than, for example, the background knowledge of the people that were hired for the experiments (Dybkjaer et al. [1993]). Quickly researchers like Bertenstam et al. [1995] and

Lamel [1998] discovered the benefits of using WOZ experiments to collect the spoken input of test participants and used this initial corpus as a source to bootstrap the development of the first domain-specific Automatic Speech Recognition (ASR) systems. Alternative methods to build a working system were mainly based on complex algorithms and machine learning processes such as the ones described by Gorin et al. [1997]. However, before these algorithms could have collected sufficient data, a considerable number of real users needed to be exposed to a poorly working system, which sometimes restricted the application of the approach in a business context. By using WOZ in the initial stage of data collection, researchers were able to solve this problem.

In an attempt to feed more knowledge to their spoken dialogue system Ammicht et al. [1999] then moved away from just using WOZ in the early stages of system development and started employing it through all stages of design. From here on in, efforts were mainly devoted to exploring dialogue strategies in different domains and how the insight gained could be used in system design. Pirker et al. [1999] specifically looked at ordering event tickets (in their research they were focusing on cinema), while Whittaker et al. [2002] explored more complex information seeking domains, which led to the development and testing of the first user models. On a different front Hajdinjak and Mihelic [2003] looked at accessing weather information and Krebber et al. [2004] at controlling a smart home. Studies also investigated multi-user environments and how they potentially change the dialogue models (Strauß et al. [2006]), at different entropy levels of the presented information (Winterboer and Moore [2007]), and at the use of referring expressions in situated dialogue tasks (Janarthanam and Lemon [2009]).

Moving the context of interaction from the computer into the car Geutner et al. [2002] focused on using spoken language for navigation as well as context-specific information retrieval while driving. Expanding on this work Lathrop et al. [2004] looked at variations in speech data caused by contextual changes and Cheng et al. [2004] analysed driver utterances in order to find more realistic in-car dialogue behaviour. Similarly, Bertomeu et al. [2006] used a corpus gathered through WOZ experimentation to explore discourse changes influenced by contextual phenomena.

Looking more closely at goal-orientation and the influence recognition errors would have on task completion rates Williams and Young [2004], studied assisted way-finding (i.e. they used a map task scenario in which two dialogue partners - one being an intelligent system - are given two partially incomplete maps, each of which misses different elements. One of the interlocutors then acts as an advisor who helps the other person find the fastest way to a specific target location. The resulting dialogue is used to fill each others missing elements on the map). A wizard mimicked the system and provided for different levels of mis-recognition. A similar setting was used by Koulouri and Lauria [2009] to explore miscommunication and collaborative behaviour. How people change their prosody in a grounding process, as well as how they recover from mis-recognitions were both the focus of additional WOZ studies conducted by Skantze et al. [2006] and Skantze [2005]. Porzel [2006], on the other hand, recommended to radically rethink the envisioned design of dialogue systems and how to model

the interaction with them.

From a more task-focused point of view Lyons and Skeels [2005] studied the way we interact with digital calendar programs. To do so, they used a wizard to simulate an always-on assistant application. The goal was to create a system that would not depend on direct commands but rather run in the background and act upon conversational cues. Trying to push language technologies to a new level, Bälter et al. [2005] used a trained speech therapist as a wizard for a simulated speech training system for children with language disorders. A similar set-up was recently used by Cabral et al. [2012] to help non-native speakers with their pronunciation of English sentences. This brings us to another language-related aspect that has increasingly been explored through WOZ, namely the use of Machine Translation (MT). Explorations in this area range from comparing different MT architectures (Starlander [2007]) to studying the rephrasing and grounding processes achieved through back-translation (Bederson et al. [2010]). Eventually, leaving the goal-oriented dialogue structure Gandhe and Traum [2007], were interested in analysing role-play and what can be learned from that to inform the design of open dialogue systems. Lee et al. [2010] took a similar approach when they used a wizard to control the narrative of a computer game and Dow et al. [2005b] extended the application into the real world by conducting a proof-of-concept study that explored spatial narratives for visitors of a cemetery.

While aspects of speech-recognition as well as speech synthesis have been the focus of some studies, the majority looked at improving natural language understanding and the consequent generation of appropriate output. WOZ experimentations seem particularly helpful in cases where existing technological solutions need to adapt to human behaviour. Skantze and Hjalmarsson [2010], for example, used a wizard to simulate an incremental dialogue manager that adapts its goals whilst a user is talking (i.e. plan-based dialogue management). Others looked at wizards interacting with test participants to construct models of human behaviour which were then used to deduce reinforced learning strategies for dialogue managers (Rieser et al. [2010]). Finally, WOZ was also used to quickly test some of the developed components. For example, Meguro et al. [2011] compared the efficiency of seven different dialogue models by replacing the language understanding part of a spoken dialogue system by a wizard.

To address some challenges of WOZ experimentation, namely the often small experiment size (i.e. experiments usually consisted of one wizard interacting with 10-15 test participants) as well as the dependency on a human wizard, researchers used different approaches. Fabbrizio et al. [2005], for example, used an automated wizard that was modelled to reject or re-prompt user queries with vague or undefined content. Wirén et al. [2007] on the other hand, worked with 10 professional wizards (i.e. call centre employees) over the time-span of 5 weeks to collect a corpus for a potential call-routing application. 42,000 dialogues were recorded. While the number of utterances that were recorded was certainly important for training the system, the author also highlighted that the WOZ method was equally valuable for informing the actual product design[2]. Another example that clearly illustrates the power of WOZ prototyping for

[2]Personal discussion with the author [Date: February 20^{th} 2012]

designing a product based on natural language can be found in 'Aardvark', a company that had the idea to develop a system where people could ask questions using a sort of instant messaging service. These questions would then be routed to their friends, or friends of friends, who might be able to answer them. As the Wall Street Journal reports[3] the Aardvark team used a WOZ set-up over a period of nine months to test their idea before investing any time or money into building an actual system. During this time they not only learnt whether people would be interested in such a service, but also gained insight into the whole process of dealing with people and how to efficiently route questions.

Lessons Learned

The goal for WOZ prototyping employed in the area of Natural Language Processing (NLP) is to build and improve technology that is able to replace a human dialogue partner. Following this motivation WOZ is mainly used for collecting an initial corpus of language, either spoken or text-based, and for exploring dialogue strategies. To a certain extent it is also employed to evaluate user experiences, although the majority of studies would see this as a secondary goal. As the focus in this area lies on producing reliable quantitative data, based on which valid algorithms may be derived, NLP-based WOZ experiments are usually highly controlled. Wizards should be experts, being trained to the point where they act consistently, and bound to a strict test protocol. Employed tools should further improve consistency and control for other confounding factors such as spelling mistakes in utterances or irregular delays. While the method is generally regarded as a low-fidelity approach, it is not perceived as particularly cheap. Training and experiment costs are high, which sometimes may require from researchers/designers to clearly justify them. From a participant's perspective literature has shown that people interact with a system differently than they do with a human interlocutor (e.g. Jönsson and Dahlbäck [1988]), even if their linguistic capabilities were equal (as in theory it is the case in a WOZ setting). They adapt their language similar to when talking to a child or to a foreigner, they use a smaller set of words, and they are more inclined to wait for responses (Dahlbäck et al. [1993]). Researchers designing WOZ experiments should be aware of these aspects so that they take them into account when interpreting results. In summary it can be argued that in NLP the WOZ methods represents an important method for initial data collection whose improvisational aspects should be restricted by a rigorous experimental design and a strict test protocol.

2.2.2 Wizard of Oz and Multi-modal Interaction

While speech and other forms of language-based human-machine communication remains an important area of exploration, in recent years researchers have shown an increased interest in multi-modal interaction paradigms, which gradually expanded the potential application area

[3] http://blogs.wsj.com/venturecapital/2010/04/24/how-a-start-up-grew-by-paying-attention-to-whats-behind-the-curtain/ [Accessed: June 22nd 2012]

for WOZ prototyping. While first explorations started in the 1990's when Oviatt et al. [1992] looked at combined speech and handwriting-based interaction and Life et al. [1996] prototyped a multi-modal service kiosk for train tickets, the majority of them took place in the new millennium when the technological advances increased the likelihood of a real-world implementation. One of the most researched aspects of multi-modal interaction relates to the relationships between modalities and how they influence each other. Starting with a traditional dialogue system setting, Gustafson et al. [2000] for example looked at the influence of an additional mouse-input channel. A user's input was sent to a wizard who controlled the system response. Later, with the advent of virtual worlds such as SECOND LIFE[4] the potential for multi-modal interaction grew further. Corradini and Cohen [2002] for example took speech as well as gestures and explored how people manipulated objects in a computer game. They found that for this very specific form of interaction, a combination of modalities seems beneficial. Qvarfordt et al. [2003] then tried to understand the distinct qualities of the different modalities and found that spoken feedback lets users perceive an interface as more human-like. While spoken feedback might improve the overall user experience, Lee and Billinghurst [2008] highlight that people predominantly dislike talking to a computer. Their WOZ study showed that if people speak with a system, they seldom use whole sentences but rather employ a command-like form of interaction. The influence differently displayed health information has on people's stress levels was subject to another WOZ study conducted by Ferreira et al. [2008]. Their goal was to design an affective health system. Finally, while user behaviour was usually the focus of WOZ studies, analyses of wizard behaviour have also been found useful. In a study by Rieser and Lemon [2010] for example, a WOZ setting was used to gather data on whether to present visual information or to use only speech in clarification requests.

The use of additional modalities also led to more complex experiment set-ups. It became more difficult for one person to simulate several modalities and so Lisowska et al. [2005] used two wizards to test their prototype of a multi-modal (i.e. speech, pointing and text) information browsing and retrieval system. Similarly Melichar and Cenek [2006] used two wizards (one for input and one for output) to study the differences between vocal and multi-modal dialogue management. Looking more specifically at the difficulties of multi-modal WOZ experimentation, Chandler et al. [2002] identified a set of guidelines to be followed. They highlighted that a lack of wizard training might lead to fairly long and inconsistent response times. They also recommended the use of a cheat sheet to help wizards remember commands. Taking into account temporal aspects, Serrano and Nigay [2009] then defined a fusion mechanism for different output modalities which may also lead to a more consistent WOZ simulation. The different modalities in these studies usually included speech and text/mouse on the input side and speech and some sort of visual feedback (i.e. text, a picture or a map) on the output side. Touch-based interaction has also been simulated using a wizard. Examples include Piemot et al. [1995] who simulated the touch-based interaction on a handheld device and Schieben et al. [2009] who explored the use of haptic multi-modal interaction strategies in highly auto-

[4] http://www.secondlife.com [Accessed: June 12^{th} 2012]

mated vehicles. Similarly to pure language-based application scenarios presented earlier, here also WOZ was often used to collect an initial corpus (e.g. Strauß et al. [2008]), based on which automated dialogue environments could be built (e.g. Rieser and Lemon [2008a]). Researchers such as Rieser and Lemon [2008b] were furthermore eager to deduce models for choosing the best presentation modality for a given time and context.

Slowly moving away from traditional input modalities Voida et al. [2005] combined speech with gestures for manipulating 2D objects in the office space. They were mainly interested in understanding movements that seem natural for users. Based on the same interest Höysniemi et al. [2004] used WOZ to collect a corpus of body language and natural gestures to inform the design of a computer-vision-based action video game, and Akers [2006a] [2006b] made WOZ part of a participatory design process for developing a gestural interface helping medical scientists select neural pathways. An obvious next step for the research community was to use the same methods to develop systems that would be able to understand human sign language. Examples include Henderson et al. [2005] and Lee et al. [2005] who employed WOZ studies for the development of an American Sign Language game for deaf children, and McGuire et al. [2004] who emphasised its usage for their plan of building a One-Way American Sign Language Translator. Finally, searching for ways to best control a medical robot, Molin [2004] tested several versions of a graphical user interface through WOZ experiments. At the time WOZ was rarely used for simulating graphical interfaces, so Molin's work highlights several of its qualities. Firstly, being able to adjust responses within the boundaries of a used set-up, which allows for more flexibility in exploring a design solution. Secondly, he highlights that functionalities can quickly be added which supports the rapid and iterative testing of ideas, and thirdly he argues that by informing test participants about the set-up (Note: In Molin's experiments test participants knew about the simulation) one gives the computer a 'human face' and also stimulates user feedback and collaboration (i.e. he reports that in one situation a test participant made a 'computer sound' and so pointed to a missing feature of the prototype). Molin also pointed to some methodological problems of using WOZ. These include inconsistent wizard actions which consequently represents unpredictable system behaviour, the use of 'stakeholders' as test participants which might influence results, and the knowledge of test participants about the simulation for which a hidden wizard in certain cases may be more appropriate (sacrificing the quality of direct user involvement in the design process pointed out earlier). Testing this assumption Bradley et al. [2010] conducted a study focused on the analysis of whether a dialogue changes if test participants know that they interact with a wizard. In half of their experiments, participants knew of the simulation and in the other half, participants were only informed after the experiment had finished. An independent reviewer looked at all the dialogues without knowing which dialogues were produced in the open condition and in which cases the participants did not know about the simulation. Surprisingly in 66% of the cases the reviewer was not able to tell the difference, for which Bradley and colleagues argue that other factors such as, for example, the personality of the participants might be more important than tricking them into the belief that they are interacting with a real system. That the

method can be successful in eliciting important system requirements, even if participants are aware of the simulation, has already been shown earlier, when White and Lutters [2003] conducted a 'Wizard of Oz Field Experiment'. Their goal was to implement a cross-organisational expertise recommendation and organisational memory system. In a proof-of-concept study they simulated knowledge exchange by physically transferring questions and answers between several remote locations. Results were not only used to identify the core functionalities of a future system, but also to better understand the organisational forces that are involved.

Lessons Learned

Despite the fact that WOZ used in multi-modal interaction scenarios is sometimes regarded as an extension to purely language-based applications, the above presented work shows an important methodological difference. In multi-modal applications the analysis often looks at a combination of modalities and how they influence the user experience. Hence, while there is still a strong focus on the technology and how it can be improved, the participant's perspective gains more importance. Therefore, multi-modal applications tend to be more exploratory than pure language-based WOZ experiments. User input may not only be used to train technology components, but also to compare different interaction modalities, identify their challenges, and explore potential coherences. As such the experimental setting can allow for more flexibility. While an overall test script is usually still important, wizards might be told to explore various possible solutions and adapt to a user's input. Hence, in these cases WOZ is not only an instrument for data gathering, but also seen as an important design tool. Besides, a specific challenge of multi-modal WOZ prototyping can be found in the experimental setting. That is, it is significantly more difficult to control for consistency when several modalities need to be simulated. Multiple wizards can help, but the aspect of fusing the relevant data streams remains difficult, leading to additional requirements for the employed wizard tool. Tool support is generally an important issue to be considered when running these types of analysis. Required functionalities range from fusion mechanisms such as the one discussed by Serrano and Nigay [2009] to advanced data logging that may help with analysing user input on different levels. Consequently, WOZ in multi-modal settings may require more resources than single modality studies. In summary it can be said that WOZ for multi-modal applications does not only expand evaluations of language-based scenarios by more interaction channels, but also increasingly moves their focus to more qualitative aspects such as user satisfaction and user preference.

2.2.3 Wizard of Oz and Agents

One of the main applications for natural-language based human-computer interaction is the use of agents and different sorts of intelligent advisers. At the end of the 1980's Hill and Miller [1988] used a human to play the role of a simulated intelligent advisory system. The goal was to test different strategies of how to give advice on the use of a statistical software package. Looking at how users of such a program would formulate questions, Guindon [1991]

found that the generation of simple and rather restricted questions was predominant. Similarly, Pilkington [1992] used WOZ in a text-to-text fashion to provide help to test participants using the UNIX text editor VI. Later, trying to build a model for a professional advisor system, Hill [1993] found that on the one hand, the advice that is given depends on the given or pre-assumed knowledge the wizard has over the knowledge of the advice seeker, and on the other hand, that even if the given advice solves a problem, approximately half of the advice seekers do not use it in an effective and efficient way.

One important aspect of intelligent agents is their potential ability to learn from their interactions with a person. Maulsby et al. [1993] used WOZ to look at this learning process by exploring how people would teach their virtual helper. How pedagogical agents can be used in a virtual learning environment and its influence on group-work was the motivation for a WOZ study conducted by Jondahl and Mørch [2002], and in particular looking at when and how an intelligent advisory system should present feedback Mavrikis and Gutierrez-Santos [2010] found that contextual information plays a crucial role in effective advice giving. They compared three levels of simulation. First, they had the advice giver (facilitator) sitting next to the students. Using this set-up students and facilitator shared the same information space. In a second stage, the facilitator was located in a different room and used a remote desktop system (i.e. open WOZ set-up) to communicate with the students. In this set-up many communicational cues such as the student's face or his/her gestures were lost. Finally, Mavrikis and Gutierrez-Santos used a fully automated advice system that was built upon the results of the previous experiment stages to give advice. They concluded that contextual information is of great importance if one aims at giving effective advice and consequently optimises a student's learning progress. WOZ is an efficient method for building the relevant computer models supporting this agent behaviour (Mavrikis et al. [2012]; Rizzo et al. [2005]). Similarly, Tsovaltzi et al. [2008] and Braun and Rummel [2010] argue that an efficient tutoring system needs to adapt to a students progress. That is, adaptive and context-specific feedback is necessary to direct a learner's interactions with an interactive learning environment to useful ends. Also here both research teams employed a human wizard to learn the relevant cues.

Searching for improvements on a different front Kitamura and Tsujimoto [2003] found that if they use several agents in a simulated information retrieval task, they could broaden people's interests in the topic. Conducting a simulated proof-of-concept study that integrated agent-like conversational abilities into a computer game character, Gustafson et al. [2005] argue that spoken dialogue technology has the potential to enrich a user's experience. Also Dow et al. [2010] showed that blending human and machine control can lead to a richer playing experience.

An additional area of agent research for which WOZ prototyping has seen an increased application is the study of emotional aspects of human-computer interaction. Bradley et al. [2009], for example, used a simulated agent to engage people in a discussion over their personal photo library. The goal was to build a more human-like connection between a user and a virtual character; something for which WOZ simulation seems particularly suitable. On the

other hand, Chen et al. [2010] looked at the difference between introverted and extroverted agents and found that those personality traits influence how users interact with the system, leading to more talkative behaviour with an extroverted version. Again a WOZ experiment was used to control the different modes. Similarly an agent's smiling and other facial cues have been explored. Bevacqua et al. [2010a] and Bevacqua et al. [2010b] report on users partially reflecting a virtual interlocutor's behaviour. Finally, blending on-task dialogue with social conversation (i.e. off-task dialogue), Silvervarg and Jönsson [2010] showed a positive effect on pupils' attitudes and self-efficacy when studying mathematics, and testing new ways for interacting with emotion, Andersson et al. [2002] and Paiva et al. [2002] used a doll to explore emotional input for computer games. Here a wizard interpreted the expressed emotion (judging if the expression was understood based on an instruction sheet of possible expressions) and appropriately controlled the game character. Scherer and Schwenker [2008] on the other hand presented work that extracted emotions from recorded speech. Again, WOZ was used to trigger those dialogues and collect the relevant corpus. Finally, expanding on this, Walter et al. [2010] report on using WOZ experiments for building emotional profiles of users.

Lessons Learned

Based on the discussed literature it can be argued that WOZ prototyping of agents is essentially an extension of what has earlier been categorized as WOZ prototyping for natural language. The main difference is its strong focus on simulating context-aware systems (e.g. Pilkington [1992]), which makes it more difficult to pre-define and control responses. In more recent years experiments were furthermore augmented by a visual interpretation of the simulated agent, turning them into an audio-visual interaction scenario. While this may not influence the wizard perspective of an experiment, it significantly changes the interaction experience for a test participant. Hence, similar to multi-modal interaction scenarios, recent agent-based experiments have usually a strong focus on user experience and how it can be influenced (e.g. Dow et al. [2010]; Chen et al. [2010]). In some cases this can lead to an increased workload for the wizard. For example, a wizard is sometimes not only responsible for choosing appropriate responses but also for the facial expression and/or gesture output with which the agent delivers them. Hence, wizard tools for this type of experiment need to offer appropriate features that help controlling both, the audio and the video channel. In summary, WOZ prototyping of agents can be seen as a hybrid that shares the goals and challenges of simulating language-based interactions scenarios (e.g. collecting language corpora) and combines them with some aspects of multi-modality (e.g. coherence of audio-visual output).

2.2.4 Wizard of Oz and Sensor Technology

In addition to simulating direct interaction modalities such as speech and gestures, WOZ was also used to prototype sensor-based system functionalities. An example is Hudson et al.'s [2003] study on human interruptibility. Their goal was to find measurable elements with which

a system would be able to predict the times when a worker would be available for interruption. Wizards were used to simulate a set of random, but controlled intervals, where they prompted participants to provide self-reports on their interruptibility. Looking at navigational aspects, Liu et al. [2006] used two wizards to provide location-based instructions to cognitively impaired test participants in an indoor way-finding task. A similar set-up was used by Bernhaupt et al. [2007] to simulate an outdoor multi-player game for mobile devices. Krüger et al. [2004] used WOZ in an experiment where they studied knowledge acquisition of pedestrians using a pedestrian navigation system, while Sefelin et al. [2005] lists WOZ as a valuable method for testing the efficiency of guiding concepts. Other examples of WOZ prototyping for location-enhanced applications include Takayama et al.'s [2003] study on people's search behaviour in physical spaces, Dearman et al.'s [2005] work on the influence of location-awareness for meeting a rendezvous partner, and Paay et al.'s [2008] location-based storytelling in urban environments.

The task of the wizard in all these cases was usually to simulate the function of the positioning sensor by manually updating a user's location. Looking at better, more efficient ways for wizards to deal with this task Li et al. [2006] proposed directional crossing and steering on a tablet computer. Their studies with 12 wizards showed a reduced cognitive load for wizards when employing these two interaction methods. Further exploring the design challenges for WOZ prototyping of location-enhanced applications, they also highlighted the necessity of simulating sensing inaccuracy (Li et al. [2007]); an important aspect that makes WOZ experiments more realistic and therefore their results more representative.

Another potential application for sensor technology is monitoring people related parameters. Consolvo et al. [2007] for example, applied WOZ to prototype an intelligent picture frame which conveys information about a relative's medications, activities and mood. Hemmeryckx-Deleersnijder and Thorne [2008] used it to simulate awareness and context sharing between co-located family members, and Rice and Alm [2007] to test a user-driven navigation layout for operating a TV. In all application scenarios, it was the human wizard that adapted the context and therefore allowed for evaluating user experiences before building the relevant technology.

Lessons Learned

Using a human to simulate sensor technology is different from so far discussed WOZ settings in that the task of the wizard is usually not focused on communicative aspects of human-machine interaction (such as natural language or gestures). Rather, he/she is used to provide the type of information that later will be available from a GPS module, a thermometer, or a motion sensor. While with the availability of cheap sensor technologies the price is often not the crucial factor for using a wizard, the missing knowledge on what exactly is to be measured, might be. An example can be found in Hudson et al.'s [2003] work on interruptibility, where the WOZ experiment was used to analyse work processes and identify characteristics that can be used to better steer interruptions. Therefore, an important requirement for tool support in such settings

concerns data logging and analysis mechanisms. Another frequent application of sensor-based WOZ experimentation is the simulation of navigation technology. Here it is mainly the provision of relevant context-depended information that is explored. Researchers/designers may be interested in both, the experience of the test participant as well as the decision of the wizard on when to present which type of information. Tools therefore should not only log the environment but also the actions of the wizard, so that potential computer models may be derived. In summary, WOZ experiments that simulate some sort of sensor technology mainly search for measurable characteristics based on which context-aware computer programs can be built.

2.2.5 Wizard of Oz and Robots

The advent of reasonably priced robot technology opened up an even greater potential for studying emotional aspects of human-machine interaction; one where again WOZ has predominantly been used to test potential behaviour that is difficult to implement. Deruyter et al. [2005] for example simulated the 'social intelligence' of a robotic cat. This included dialogue behaviour as well as facial expressions (i.e. happy face vs. sad face). Similarly, Delaborde et al. [2009] collected audio data from human-robot interaction in order to develop a real-time emotion detection model. These studies often involved children as their inhibition threshold is usually lower than the one found in adults. Saint-Aimé et al. [2011] for example used a wizard-controlled robot teddy bear (stuffed animal) in an experiment with 3-5 year old children, and Villano et al. [2011] report on a simulation where a robot was used to help children with autism spectrum disorders to better socialise. While a set-up with a wizard in the room may allow for better feedback, Read et al. [2005] point out that in such a case children are more aware of the 'fake' situation.

Other research areas that were explored include the use of gestures when interacting with a robot (Mai et al. [2011]) as well as how to deal with interruptions in human-robot interaction (Saulnier et al. [2010]). Also, in order to improve a robot's communication skills, Shiomi et al. [2007] conducted a WOZ study that looked at group attention in public spaces (i.e. in a museum) and how robots can interact with several people at the same time. Finally aiming at describing a classification scheme for the different WOZ methods used in human-robot interaction, Steinfeld and Scassellati [2009] distinguish between 'Wizard of Oz', 'Oz of Wizard', 'Oz with Wizard', 'Wizard with Oz', 'Wizard and Oz' and 'Wizard nor Oz'. In this scheme 'Wizard' stands for the person or system that simulates and 'Oz' defines the user. In the classic form, that is 'Wizard of Oz', human behaviour (i.e. Oz behaviour) is explored using assumed system behaviour. Turning the method around, 'Oz of Wizard' assumes a certain user behaviour and can be used to test technologies. In the intermediate stages one finds 'Oz with Wizard' where robot-centric studies are conducted with human involvement, and 'Wizard with Oz' where human-centric studies are conducted with real technology. In the case where both, human and robot actions are simulated, Seinfeld and colleagues call it 'Wizard nor Oz' and where aspects of wizard and human behaviour are explored, they speak of 'Wizard and Oz'.

Lessons Learned

The simulation of robots represents the last category of this classification scheme. Similar to evaluations of multi-modal or agent-driven scenarios, this type of WOZ experiment investigates interaction paradigms beyond language (e.g. Mai et al. [2011]). Having a physical implementation of an intelligent system, i.e. a robot, allows for a greater number of possible interaction scenarios to be explored. This, however, may also lead to a more complex task for the wizard. While in language-based scenarios a wizard usually focuses on selecting appropriate response utterances, and in a multi-modal experiment he/she might be responsible for controlling certain modalities, a robot can significantly increase the number of potential interaction channels. This includes the movement of body parts as well as more subtle behaviour such as changing the physical proximity between a robot and a test participant. Hence, similar to multi-modal interaction scenarios, robot-based WOZ prototyping usually requires several wizards or advanced tool support, which includes intelligent system behaviour to reduce the workload of the wizard. From an experimental point of view, the majority of those experiments investigates social aspects of human-machine interaction (e.g. Deruyter et al. [2005]; Delaborde et al. [2009]) wherefore they can be classified as rather qualitative. Nevertheless, wizard as well as participant behaviour is often used to derive computer models and so, as with previously discussed WOZ settings, sufficient data-logging mechanisms are important. In summary it can be argued that, giving the wizard this physical appearance of a robot, allows for a new level of exploration, but also significantly increases experiment complexity.

2.3 The Consolidation of Language Technologies

As can be seen from the described examples the WOZ method is employable in a variety of interaction scenarios, ranging from mixed-reality simulations (e.g. Dow et al. [2005a]) to human-robot interaction (e.g. Saint-Aimé et al. [2011]). However, it is mainly in the area of speech and Natural Language Processing (NLP) where the method is regularly used, and where we see an even greater demand for it in the future. The reason for this expected increase is that the use of language technologies such as Automatic Speech Recognition (ASR), Machine Translation (MT) and Text-to-Speech Synthesis (TTS) has significantly increased in recent years. They have been successfully integrated into several applications, with interactive voice response systems receiving widespread commercial deployment. Systems like LET'S GO (Raux et al. [2006]) and the DARPA COMMUNICATOR (Walker et al. [2002]) were used to provide customers with schedule information over the telephone for flights (Karpov et al. [2008]), trains (Lamel et al. [1999]) and buses (Turunen et al. [2005]). Speech technologies have been integrated with MT in prototypes which support multilingual communication during meetings (e.g. Wahlster [2000]), doctor-patient consultations (e.g. Somers [2006]) and travelling (e.g. Paul [2009]). MT for more general purposes is available on-line from companies

like GOOGLE[5] and MICROSOFT[6] which support free translation from/to various languages and usually integrate certain amounts of text analysis (e.g. by automatically recognising the input language) as well as Text-to-Speech Synthesis. In addition to these scenarios, LTCs are also used in settings in which traditional input and output modalities are less appropriate. Hands-busy eyes-busy situations, for instance, require special interaction techniques. In-car navigation, route planning and other services such as electronic car manuals and interactive hotel reservation systems are now increasingly accessible via speech input and combined speech/map output (Geutner et al. [2002]). Moreover, in the areas of tutoring systems and edutainment as well as in the health care sector (e.g. Keskin et al. [2007]) language technologies are used in several multi-modal settings. Here mainly prototypical implementations of Embodied Conversational Agents (ECAs) show interesting application possibilities, including areas such as computer-based mental health care (Coyle et al. [2007]) and e-learning scenarios which help people to learn about computer literacy or physics (Graesser et al. [2001]).

One driver of adoption to this new way of interaction can be found in increasingly ubiquitous access to products and services outside traditional office environments, where in many cases language technology solutions offer distinct advantages such as hands-free and eyes-free interaction (e.g. the use of speech to control a mobile phone). Another contributing factor is the improved performance of these technologies which has opened up new application areas in a variety of different fields. This trend is visible both from an application perspective, in the widespread use of voice dialling, in-car navigation systems with speech interfaces, instant web-based machine translation from mobile devices, and transactions accessed through Interactive Voice Response (IVR) systems, as well as from a research perspective, in emerging areas such as Speech-to-Speech translation (Stüker et al. [2006]) and human-avatar interaction (Bradley et al. [2010]). Nevertheless, the design of appropriate services is difficult and accompanied by various challenges, which includes the lack of adequate prototyping tools.

2.3.1 Dealing with Language Technologies

While the above description shows that the use of language technologies is undoubtedly increasing, several barriers remain to its widespread use. Language technology components are unavoidably imperfect and typically substantial engineering effort (gathering of corpora, training, tuning) is needed before prototypes involving such technologies can deliver a user experience robust enough to allow potential applications to be evaluated with real users; particularly in the case of ASR. Here, state of the art recognition algorithms promise word error rates (WER) as low as 20% (Jurafsky and Martin [2008]) and less than 10% within specified domains (Stücker et al. [2007]). Different accents and dialects, however, can increase this value significantly (Liu et al. [2006]). To improve error rates, examples have shown that domain data can be used to train speech recognisers. Fügen et al. [2006] for example uses previously used material like transcribed speech from meetings and presentations to decrease the WER

[5] http://translate.google.com/ [Accessed: July 16^{th} 2012]

[6] http://www.microsofttranslator.com/ [Accessed: August 2^{nd} 2012]

for public speeches and lectures. A similar approach can be applied to increase the quality of MT. Thus a major challenge in the design and development of language technology applications is the need for domain-specific collections of text (corpora). Application designers and engineers need to invest significant time and effort in building up these resources in order to tune the components to an acceptable quality level. On the other hand, some studies show that even if the used technology is flawed, people may be able to successfully accomplish given tasks. It has been demonstrated that by adapting the dialogue in cases where the ASR is poor, the overall user satisfaction can be increased (Litman and Pan [2000]). While for Graphical User Interfaces (GUIs), methods like sketching and wire-framing allow a designer to test for those effects and adapt the design based on the results, these low-fidelity prototyping techniques do not map well onto systems based around speech and other forms of natural language. Hence, applications that use language technology components such as ASR, MT or TTS as their predominant interaction channel require a different design approach; WOZ can be seen as a prototyping method that offers this means of 'sketching' language-based interaction.

2.3.2 Making a Case for Wizard of Oz

While WOZ has also been employed to prototype other types of interaction, its major areas of application seems to stay in natural language processing (NLP). The reasons for this are varied. On the one hand, we see that recognising, parsing and eventually understanding human language, whether it is in written or in oral form, is still one of the major challenges of modern computing technologies. On the other hand, it represents one of the most natural forms of communication and so researchers in a variety of fields (e.g. NLP, Human-Agent Interaction, Human-Robot Interaction, Multi-modal Interaction, etc.) believe that it is key for providing the perfect user experience. Despite the arguments of some (e.g. Porzel [2006]; Jönsson and Dahlbäck [1988]), which highlight that human-human communication can be radically different from human-machine interaction, the general trend is towards building technology that tries to mimic human communication abilities as close as possible. Relying on the actions of a human simulator for setting the gold standard seems therefore a logical way to go. Not only does it allow for collecting data to tune recognition engines and improve understanding on the input side, but it also represents a valuable instrument for studying dialogue discourse, which is essential for driving the interaction and generating appropriate responses. Here, it is mainly the higher flexibility that a human operator provides compared to a technical solution that lets designers and researchers employ WOZ early on. They are able to test alternatives and run different sorts of evaluations without adjusting or replacing the technology. When it comes to application areas outside the NLP domain we find similar advantages. Yet, here the technological progress is often quicker and sometimes novel solutions quickly exceed the capabilities of a human operator. A system that makes use of some sort of face recognition, for example, can be seen as a potential use case for WOZ evaluation. Having a human taking over this part of the application, which usually would require a significant amount of upfront investment to be built,

would allow for feature-rich testing early on in the development. Human decision processes could be studied and a corpus of image data be collected, similar to the collection of language corpora in NLP. However, given the relatively clear biometric features of the human face it is likely that a working recognition engine may be built without requiring any WOZ experimentation (e.g. Guo et al. [2000] report error-rates between 13% and 3% by simply applying various machine-learning algorithms to a database of images where single faces are shown from 10 different perspectives). It might still be useful to improve and fine-tune such a system but it often adds rather little to the actual recognition accuracy.

This aspect is different with speech and other forms of natural language since here the optimisation process is usually harder and may require significantly more time and resources. Numerous tests at different stages of the development can be necessary to achieve a satisfying result. Also, while the ability to recognise a face or shape can be seen as challenging, it mainly depends on a finite set of characteristics which, when sufficiently defined, can be integrated into the relevant algorithmic models. Accuracy rates therefore increase quickly. With natural language the situation seems more complex. Grammatical variations, sentence perplexity, and word ambiguity as well as accents and regional dialects remain barriers that even the best technology currently available seems to struggle with. Hence, as opposed to other application areas in NLP, experimentations that employ one or more human simulators remain valuable throughout the whole development process and beyond.

Looking at the literature, the use of WOZ for language related interaction scenarios is further reflected. The ACM Guide to Computing Literature[7] for example lists 1,722 hits for the search term 'Wizard of Oz', of which 37% also include the keyword 'Natural Language Processing', 56% the keyword 'Language Technologies' and 59% the keyword 'Speech'. The IEEE Xplore Digital Library[8] lists 702 entries for 'Wizard of Oz', 50% of which also contain the term 'Natural Language Processing', 57% the term 'Language Technologies' and 63% the term 'Speech'. If one focuses on journal publications only the pre-dominance of WOZ used in language-based research is even more apparent with 72% of WOZ related articles listed in the ACM database holding the additional term 'Language Technologies' (looking at ACM Transactions alone this figure is as high as 83% i.e. 30 of the 36 journal articles listed for the search term 'Wizard of Oz' are in the field of 'Language Technologies'). While early applications of WOZ focused on simulating natural language interaction based on pure text or speech, we do see a shift towards multi-modality in recent years, but even within this shift it is the NLP aspect of a study that needs the biggest amount of simulation, as existing technology is simply not mature enough to be used without significant upfront investment.

Based on this language centred point of view we can therefore identify three distinct uses of the WOZ technique for designing interactive systems. Firstly, within interaction design, there is a clear application of the approach to investigate the design of human-computer dialogues. Secondly, it can be used as a means for collecting language corpora (which feeds into

[7] http://dl.acm.org/ [Accessed: April 2nd 2012]

[8] http://ieeexplore.ieee.org/Xplore/guesthome.jsp [Accessed: April 2nd 2012]

both interaction design and engineering work to train and tune technology components), and thirdly researchers developing technology components can employ it as a means for conducting evaluations of their products in specific application areas, without facing the engineering effort of constructing the application itself (which may require more robust components than are currently available).

2.3.3 Challenges of Prototyping Language Technologies

Due to the fact that most LTCs are relatively new to application designers, applications based on such components need to be tested early in the design process. Whereas low-fidelity prototypes assessing standard GUI applications such as sketches and wireframes can be built relatively quickly and inexpensively, the development of prototypes to evaluate the usage of LTCs can be both cost and time intensive. Therefore, an efficient way of creating such prototypes is desirable. WOZ experiments seem to be a good candidate for addressing this issue. However, unlike most standard GUI-based prototypes, WOZ prototypes depend on the actions of a human wizard at runtime. Whereas with GUI commands a specific behaviour can be defined in advance by referring to concrete events like mouse-clicks and keyboard-entries, with LTCs this clear binding is less direct. Here the interaction rather depends on the way a user's input is interpreted. Since in WOZ settings the human wizard mostly mimics system performance, timing aspects may impact on the outcome of the experiment, which puts the wizard under a high cognitive workload (Salber and Coutaz [1993a]). One way of supporting the wizard is to provide an appropriate interface that helps to fulfil this task more efficiently, so that human intervention is imperceptible to the user interacting with the prototype. However, as can be seen from the different settings for WOZ experimentation described earlier, the task of the wizard is diverse and can range from being a sort of dialogue partner at the early stages of product development where the focus lies on openly exploring a design space, to simulating often very defined system functions throughout the entire development cycle. An appropriate wizard interface therefore not only needs to link the human simulator to the prototype (or at the end of a development cycle to the final product) in order to allow for simulating the envisioned system functions, but should also support different levels of fidelity. Furthermore the interface should be intuitive to operate, support accurate as well as consistent wizard actions, and require little resources to be built; the latter making the use of existing prototyping tools, if available, most favourable.

2.4 Summary

In summary one can argue that WOZ experimentation has its roots in the area of NLP research where it was first used to simulate systems that would understand natural language-based input; initially based on text and later based on speech. After that, researchers started combining natural language with other modalities such as handwriting, mouse pointing and later touch

and gestures, which led to an increased application of WOZ prototyping in multi-modal settings. Finally, with the advent of high quality 3D image processing and the availability of affordable sensor and robot technology, wizards were used to simulate emotional aspects of human-machine interaction as well as location and context-specific services. Looking at this multitude of different forms and dimensions, it can also be said that the method often requires considerable tool support in order to be successfully applied. One goal of this thesis was to study this type of tool support and how it can be improved, and so the following chapter will outline in more detail the different methods we have employed to do so. Furthermore, it will describe the methodology we used to foster our understanding of the WOZ method in general, and the task of the wizard in particular.

Chapter 3

Methods

"You must walk. It is a long journey, through the country that is sometimes pleasant and sometimes dark and terrible. However, I will use all the magic arts I know of to keep you from harm."
– The Witch of the North

This chapter describes the approach that was chosen to investigate WOZ tool support and the challenges that are involved. The chapter starts by describing Systems Development Methodology (cf. 3.1) and User-Centred Design (cf. 3.2), and how we used methods from both philosophies in an integrated research approach (cf. 3.3). We then elaborate on the two dimensions of this iterative process, starting with the more practical side of prototyping a tool (cf. 3.4) and then move on to the more theoretical side of exploring Wizard of Oz experimentation (cf. 3.5). Finally, we highlight how iterating between prototyping and experiment exploration drove our research progress (cf. 3.6) and how related methodologies may have led to similar results (cf. 3.7).

3.1 Systems Development Methodology

Research focusing on information systems often concerns the study of hardware and software artefacts shaping human, social and organisational life. While modern information and communication technologies are employed to automate work processes, they also support so called 'knowledge work' (Markus et al. [2002]), where the interplay between people and technologies is seen as a key enabler for producing high quality products. Researchers in this area mainly face the challenge of analysing the effects computer systems have on the way people work and interact with each other and their environment. This, however, can only be achieved if the relevant socio-technical integration between hardware, software, people and processes is in existence and therefore methods such as observations, field studies and use cases can be employed. In situations where the actual construction of the relevant artefact (i.e. the computer

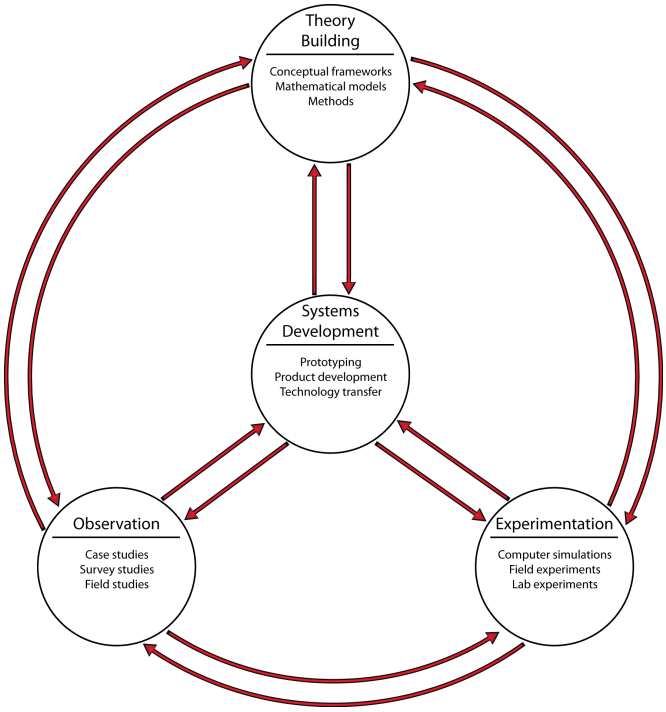

Figure 3.1 – The multi-methodological approach of Systems Development Research described by Nunamaker et al. [1991].

system) is part of the research agenda, we may, however, consider systems development as a research methodology (Nunamaker et al. [1991]). Nunamaker et al. [1991] argue that "*system development as a research methodology fits comfortably into the category of applied science and belongs to the engineering, developmental and formulative types of research*". Applied science is understood as the application of knowledge to problems of immediate concern (Bailey [1982]), engineering as the artistry of design and the spirit of "making something work" (Davies [1973]), development as the use of scientific knowledge directed toward the production of systems (including prototypes) (Blake [1978]), and formulative research (also called exploratory research) as being concerned with the identification of problems for more precise investigation as well as the gain of insight into a certain problem area.

Systems Development Methodology (SDM), as specified by Nunamaker et al. [1991] represents an integrated multi-methodological iterative research approach where Theory Building, Experimentation and Observation orbit the central System Development process (cf. Figure 3.1).

The initial step of this research methodology, Theory Building, underpins the overall development process. It aims at the understanding of a problem space and uses this insight to produce theories and suggest research hypotheses. It does contribute to the body of knowledge by methodically investigating and analysing a research area, but does not produce a system that would employ this new knowledge.

The Experimentation step represents first attempts to test hypotheses and therefore creates a bridge between Theory Building and Observation. It includes both lab and field studies and

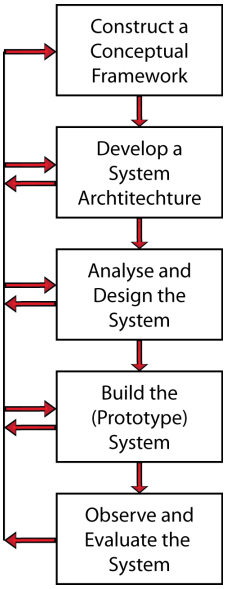

Figure 3.2 – The process of System Development Research based on Nunamaker et al. [1991].

aims for the investigation and validation of distinct aspects of the theoretical framework. Studies are facilitated by (prototypical) system development, which to some extent also explores the feasibility of an envisioned system. Results may be used to adapt and refine requirements and improve systems.

In the Observation step, observations are used to more openly explore the interaction with a system. In contrast to defined experimentation, whose goal is to test hypotheses, Observation aims to expand upon existing theories. The research is conducted in a natural setting, which may produce more ecologically valid results. Methods include unobtrusive lab experiments, field studies as well as surveys.

Throughout the whole development process, the System Development step serves as the focal point around which and through which research methods are integrated in order to form a coherent research agenda. As described by Nunamaker et al. [1991], System Development consists of five steps: concept design, constructing the architecture of the system, prototyping, product development and technology transfer. Concept design concerns the adaptation of new theories into potential applications, for which consequently a system architecture may be developed. Prototypes are then built to evaluate feasibility and define additional requirements. If judged to be successfully prototypes may be expanded into real products. Finally, technology transfer happens when the resulting product has been accepted by organisations and difficulties and constraints encountered during the development process have been sufficiently documented so that the resulting product serves as reference for future implementations. Additionally, difficulties and constraints found can also be used to modify concepts and theories underpinning the product.

Transforming this methodological landscape into an iterative research process, Nunamaker et al. [1991] further discuss five stages of the research approach and their respective research

goals (cf. Figure 3.2). The process starts with the construction of a conceptual framework, which represents aspects of theory building mentioned earlier, and as such deals with the investigation of system functionalities and requirements, and the understanding of the relevant system building processes. This stage is also concerned with the extensive exploration of the field, searching for new ideas, and has the goal of developing meaningful research questions. The second stage then aims to develop a potential system architecture and defines the functionalities of its different components, their interrelationships and interactions. Next system design needs to be developed, including the design of processes to carry out system functionality.

Alternative design solutions also need to be evaluated, before the final design is selected and before an actual prototypical system implementation can be built and tested. Through constructing a prototype one may also learn about potential problems and system complexities. Depending on the maturity of the prototype as well as the nature of the employed evaluation method, the last stage of this process then mirrors either the concept of experimentation, which aims at validating a hypothesis, or the concept of observations, which mainly aims at expanding upon existing theories. In both cases, systems development research needs to be understood as an iterative process where the insight gained from creating and evaluating new functions feeds back into the definition and refinement of existing concepts and theories.

3.2 User-Centred Design Methodology

User-Centred Design (UCD) is a process describing the design and development of new products and services, which specifically looks at and takes into account the needs, wants and limitations of end users. Its goal is to minimize the mismatch between a designer's assumptions of how a product might be used and real-world user behaviour. In doing so, it focuses on what users want and what they can actually do, taking into account psychological as well as physiological limitations such as short-time memory, attention span and human anthropometrics. To achieve this UCD is characterised by a number of mainly qualitative research methods including ethnographic studies, observations, usability testing and prototyping, which are complemented by some quantitative analyses such as surveys, log file analyses, and performance tests. UCD is defined by the ISO 9241 (Ergonomics of human-system interaction), part 210 as Human-centred design for interactive systems[1], and as such leads through the planning, design and development of a new product. Figure 3.3 illustrates UCD's iterative design process which is subdivided into the following steps:

1. Specify context of use: This step focuses on identifying the people who will use the future product, the conditions under which it will be used and the reason why it is used. The goal is to understand a product's purpose of use and all contextual factors influencing its application.

[1] http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=52075 [Accessed: September 11^{th} 2012]

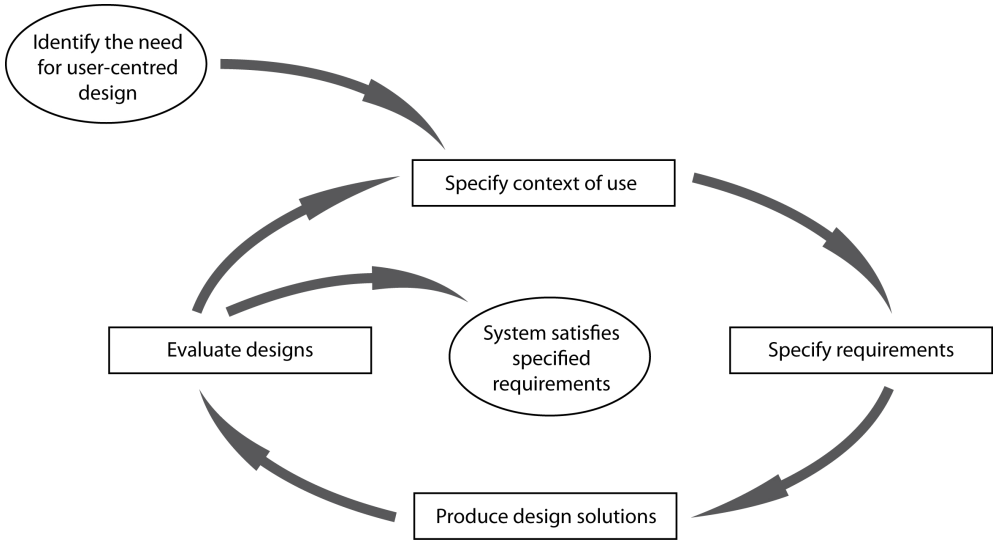

Figure 3.3 – The iterative design process of UCD based on the ISO 9241-210 standard.

2. Specify requirements: Based on the context of use this step defines requirements that need to be met for the product to be successful. Requirements can be separated into business requirements i.e. functionality of the product and user requirements i.e. usability of the product.

3. Produce design solutions: With the product requirements defined design solutions are built. Solutions can range from sketches and rough prototypes to complete designs and are enhanced with any iteration of the design process.

4. Evaluate designs: As with most product development processes the evaluation phase is crucial for the quality of the product. Testing the different design solutions with actual users ensures appropriateness and applicability of the design and fosters its usability.

The UCD model is implemented by a number of design methodologies including Cooperative Design (Greenbaum and Kyng [1991]), Participatory Design (Schuler and Namioka [1993]), and Contextual Design (Beyer and Holtzblatt [1998]), all of which encompass the following key principles:

- The process is driven by an understanding of the target users, the tasks and the context.
- The process involves users throughout the design and development process.
- The process incorporates user-centred evaluation and refinement.
- The process is iterative.
- The process is based on a holistic user experience.
- The design team consists of people from different disciplines, with different skills and perspectives.

The following sections discuss a number of methods that are used to specifically support the UCD process.

3.2.1 Personas

A persona is an archetypical representative of an actual group of users and their needs (Cooper [2004]). In contrast to potential users identified by market segmentation, a persona is not based on an individual person, not based on a customer segment, and not based on demographic data or a job description. Rather, it represents a class of users in context. As such it is not an average person but an exemplar of a person with associated sets of behaviour. A persona helps a designer to focus on a person for whom the product is being designed. It makes specific user requirements tangible and therefore supports UCD. Design teams may also employ a negative persona which represents a class of users for whom the product is explicitly not designed. While personas are an effective instrument for communicating design goals throughout the team, they often lack the required organisational backing to be effective. Also, due to their weak definition the representation of the user may be flawed.

3.2.2 Scenarios

A scenario is 'a story about the use of the designed product' (Caroll [2002]). It represents an example of how a future user (e.g. the persona) is going to use the product in all its context. A scenario is not a list of possibilities but rather a description of a discrete use case. As such it needs to be salient as well as realistic and is iteratively refined throughout the development process. Scenarios may be specified at different levels of abstraction depending on the maturity of the product. Yet, they always include the context of use, the person using the product, the goal that is to be achieved, a description of the task, possible external events influencing the product behaviour, as well as the outcome for the user. Scenarios provide the design team with a dynamic representation of how a future product might work in context. However, they may be based on assumptions and a direct translation into design features may therefore lead to partial or incomplete solutions.

3.2.3 Prototypes

Prototypes serve the purpose of eliciting user feedback on design decisions. On the one hand there are low-fidelity methods such as sketching and storyboarding, which are mainly used to evaluate early concepts and design ideas. On the other hand, more realistic feedback may be collected through high-fidelity artefacts, where mock-ups are used to evaluate the physical form of a product and prototypes are employed to learn about the quality of its envisioned functionality. While the creation of prototypes may be seen as an effective instrument for feedback elicitation, the accompanying costs are often underestimated; especially late in the development process where an extensive set of functionalities needs to be integrated.

3.3 Integration of Methodologies

Aiming for a better understanding of the WOZ method and its tool support we used a combination of SDM and UCD to drive our research progress. That is, while the described work in this thesis generally followed the process for systems development research outlined earlier (cf. Figure 3.2), it also used methods from UCD for both the design as well as the evaluation of prototypes. Furthermore it employed laboratory experimentation as well as observations to more deeply explore the task of the wizard.

In other words, we used SDM as an engineering-based research methodology where the development of a new WOZ tool represented the core of our approach. In order to build the conceptual framework in which this tool should fit in we, however, employed a mix of methods usually found in UCD. That is, we gained a broad insight into WOZ prototyping, its goals as well as its distinct challenges by conducting an extensive interview study with people who had shown sufficient expertise with this method; we used personas and relevant use case scenarios (motivated by both the expert interviews as well as previously published research) to create the context for realistic user requirements; and finally we contentiously sought feedback and ways of improvement through low-fidelity sketching exercises and early (and iterative) prototyping. Hence, one could argue that UCD gave us the relevant instruments that were necessary to more holistically explore the design space of WOZ and all its different dimensions. SDM on the other hand provided us with a suitable research framework that focused more thoroughly on the exploration of the technical challenges that were involved in building WOZ tool support. This mainly concerned questions such as: “What technology platforms are suitable for WOZ tool support?”, “How can different technologies be best integrated?” or “What are the characteristics of a flexible and modular expandable tool architecture?”

In summary the overall goal of this research agenda could therefore be seen as twofold. First, we wanted to explore WOZ tool support and potential solutions that can be used by designers and researchers to explore a variety of different WOZ scenarios; mainly in connection with modern language technologies. Second, we also wanted to extend the existing body of knowledge on WOZ prototyping by analysing some of the challenges researchers/designers face when designing and running experiments. Special attention was given to the task of the wizard and how it can be better supported. By doing so, we wanted to identify explicit features that can help researchers and designers in creating, testing and evaluating software prototypes using different kinds of technologies. Consequently, the goal was to integrate identified features in the different releases of a possible tool. As a result this research was constantly switching between two dimensions:

1. A practical dimension which aimed at a systematic analysis and exploration of tool support for (language-based) WOZ experimentation. An essential requirement for this undertaking was that the resulting tool should be employable beyond the frontiers of a single experiment and therefore tackle one of the main drawbacks of WOZ experimentation, namely the additional resources spent on building a ‘throw-away instrument’.

Other requirements included expandability, usability, flexibility and a low entry barrier for non-expert users.

2. A theoretical dimension whose goal was to more thoroughly explore WOZ experimentation and consequently shed light on the different challenges involved. Special attention was given to the wizard task and how it can be supported.

While in the following sections both dimensions are described separately, we would like to highlight that throughout the execution of the research they were highly interwoven. A clear separation was often not possible. Nevertheless, by discussing them independently we want to emphasise their distinct goals and the methods we employed to approach them.

3.4 Constructing a Tool

On a practical dimension this research was seeking to obtain the relevant knowledge to create a prototyping tool that can be employed beyond the frontiers of a single experiment. We envisioned a tool that tackles the WOZ method on a more generic level and therefore offers high flexibility and ways to customise. While usually a specific research interest triggers the development of such a tool, our motives for this venture were different. We were not driven by a single research question that might require the conduct of a specific WOZ experiment to be answered, but we analysed a variety of different WOZ settings to provide us with the relevant requirements that would need to be supported by a generic tool implementation. Our research therefore relied on third party research interests which gave us the ability to study people's motivations for using WOZ and helped to identify the challenges they were facing by doing so. Previously published work was therefore our starting point for building a robust conceptual framework.

Reviewing existing tools helped to gather a basic set of requirements and provided a broader view on different use case scenarios to be supported. Reviewing the literature also provided a general impression of the type of problems a wizard is facing when running experiments. Previous experiments and the tools that were used to run them gave us an overview of what researchers/designers experienced when confronted with the wizard task, and what tool functionalities were integrated to help them.

However, the literature was only the starting point which led to the definition of an initial set of requirements for the tool. Additional information came from both acting as a wizard (learning by doing) as well as observing other wizards when running real experiments. Since the goal was to create a tool that, while generic enough to be used in a variety of settings, could be customised to support the very specific demands of single wizards, we were highly dependent on feedback provided by those real users employing different versions of our tool in real experiments. Methods from UCD were employed to collect this sort of interaction data. While formal usability tests (cf. Dumas and Redish [1999]) were used to identify severe problems and conceptual misfits with respect to the wizard interface, it was the observation of

longer term interaction with the tool that showed missing functionalities and highlighted those challenges that were difficult to master even with sufficient hands-on experience.

Two different types of evaluations were conducted. Firstly, we employed *within-experiment* analysis, where the feedback was gathered after different experiment runs that belonged to the same experiment setting and whose results were used to improve upon existing tool functionalities. Secondly, we used the differences between experiments (*between-experiment* analysis) to identify new requirements leading to the expansion of tool functionalities. Furthermore we obtained feedback from different perspectives. That is, while at the beginning our own impressions collected from acting as a wizard guided the development process, later on we expanded the tool and integrated functionalities based on observations and discussions with other, third party researchers performing this role.

This iterative development process was employed to move from an initially hard-coded prototype that was able to run a single defined experiment to a platform that supports various forms of WOZ experimentation. Development and evaluation phases of this process were highly integrated so that new functionalities were driven by direct user demands rather then top-down engineering decisions. We may therefore summarise that, triangulating between what was presented in the literature (tools and scenarios), what we experienced ourselves when acting as a wizard, and what we learned from other third party wizards drove the design and development process for this generic WOZ prototyping tool.

3.4.1 Learning from Existing Tools

Before any actual prototyping and development work for the envisioned WOZ tool could start we looked at existing solutions and what features they offer. An essential part of this analysis was to not only enumerate these functions but also to understand why they had been implemented. As most (if not all) work on WOZ tools is grounded in scientifically motivated projects the associated research papers were an obvious starting point for building up this understanding. As already pointed out earlier, most of these applications were built for very specific experimental conditions. Those experimental conditions can be seen as a basis for identifying distinct user requirements. We wanted to understand the reasons for the different experimental set-ups in order to be able to map them to distinct features integrated by the employed tools. While existing tools mainly focus on their specifically associated experiment conditions, our goal was to offer more generic prototyping support. Hence we had to identify and understand a variety of tool requirements which required a comprehensive analysis of applications previously presented. We specifically wanted to learn from the literature what types of functions essentially define a WOZ tool, how they are integrated and what role the wizard plays when using them. Furthermore it was our goal to clearly identify those features that were implemented to directly support the task of the wizard. By understanding those features integrated in existing tools and what motivated them, we were able to define the baseline of the requirements that needed to be supported.

3.4.2 Learning from Previous Experiments

Despite our overall goal of achieving generic support it seemed important to start with specific application scenarios that could be taken as a road map for further expansion. We used personas and scenarios, as described in the UCD process, to find valid use cases for WOZ. These scenarios served as an important information source for successfully running experiments and also helped define the boundaries for our first round of prototype development. Again, examples reported in the literature provided a valid starting point while additional research interests expressed within our research group highlighted additional potential use cases. The overall goal of this step was to define a list of guidelines for WOZ experimentation and to identify a set of typical application areas and concrete scenarios where the method can realistically be employed.

Doing so was not only helpful to better understand the application landscape our tool would need to cover, but also provided us with distinct requirements that were used to drive the design process for the wizard interface. As can be seen from the multitude of tools presented in the past, tools for different scenarios offer different functionalities. Furthermore they show distinct differences when it comes to the layout of their wizard interface. Our goal was to find commonalities. We wanted to define a generic interface that would support wizards in a range of experimental settings. As such we were aiming at general layout concepts that would suit several application contexts and would also allow for the necessary flexibility to adapt to the different stages of WOZ prototyping. Looking at examples presented in the literature and expanding upon them helped to explore the existing design space and provided a pathway, which was consequently used to reveal system requirements that go beyond a possible baseline definition.

3.4.3 Learning from Building Prototypes

Based on what we had learned from existing applications and taking their distinct scenarios as a guideline we were able to start the development of the tool. In accordance with SDM and UCD, we began with paper sketches and other simple prototyping techniques in order to explore initial design ideas. While these early stage concepts were primarily used to compare and reflect upon the different solutions of a possible wizard interface, they also served as an important basis for defining the underlying tool architecture. As a flexible architecture is key for achieving generic support, this aspect of tool design was essential. That is, if the technological base would not allow for easy expansion and customisation it would also be difficult to offer the relevant functionality on an interface level. Existing tools highlighted some of the challenges involved when one aims for flexibility and so constantly re-visiting the literature helped to learn from them. This also shows the iterative nature of this design methodology which requires a continuous dialogue between the tool perspective and the application perspective of software development. While the tool perspective is often found in demonstrations and online documentations, the application perspective is represented by the variety of scenarios that

have been using the WOZ method. They provided the necessary use cases that our tool had to support. Eventually elaborating on one of them, we started building a first working prototype which served as a technology probe.

Starting a first round of evaluation, the technology probe was tested in a realistic setting (i.e. it was employed in a WOZ experiment that aimed at answering a distinct research question) leading to first insights on how the designed interface is operated in real time. Here the realistic scenario was a crucial aspect of this evaluation process as it created the necessary context, which made it possible to obtain valid feedback. Furthermore, as WOZ experimentation usually requires a number of test runs, it also made it possible to see whether a user was able to adapt to new interaction paradigms or if they would keep causing problems over time. A number of iterations involving new prototypes, modifications and extensions as well as varying scenarios and accompanying users tests, were necessary to move towards our goal of generic WOZ support. In doing so, evaluations consisted of both studies of the tool used in different WOZ experiments as well as formal usability tests conducted with participants who were not following a dedicated research agenda. While the former provided the relevant data for improving and expanding upon the functionality of the tool, the latter was seen as a valuable indicator for intuitiveness of use; an important aspect since one of our goals was to make WOZ prototyping more accessible to non-expert users.

In summary, in order to explore WOZ tool support, the first part of this thesis effectively followed an integrated SDM-UCD approach, which grounded its design decisions in information gathered from the literature as well as in the observation of people (including ourselves) using different prototype versions of a potential tool instantiation. The next section will elaborate on that by describing the second, rather theoretical dimension of this integrated research agenda, which more thoroughly focused on generating new insight into WOZ experimentation in general, and the challenges of the wizard task in particular.

3.5 Exploring Wizard of Oz Experimentation

While following a user-centred development approach for building a tool certainly represented an important part of our research agenda, particularly when it comes to evaluating different prototypes and product features, the generated insight was to some extent limited to the feedback gathered from a rather small number of people, the observed wizards, whose actions were evaluated throughout the different stages of the development process. Expanding on this, other research methods were used to generate a broader, scenario-independent understanding of the area. From a methodological point of view we achieved this by using a mixed research approach. Consisting of both qualitative and quantitative analysis methods we followed an integrated exploration path where results were collected and checked from three different perspectives.

First we looked at the user perspective. After considering the different application scenarios for WOZ (cf. Chapter 2) we shifted our focus towards the people conducting WOZ studies,

which let us identify several classes of users showing potential interest in employing the WOZ method. Based on that, a comprehensive interview study with some of those who had recently conducted studies was undertaken. This interview study particularly looked at the reasons for using WOZ, the challenges they had to overcome and the tools they had employed. Furthermore it tried to extract a general attitude towards WOZ prototyping, how it changed over time and where it may be heading. Understanding the roots and seeing further directions was important as our goal was to support a broader application of the method.

A second perspective (i.e. the system perspective) on the method was explored by looking at the required tool support. While to a great extent this analysis was based on results coming from the interview study as well as the literature and the tools that are described there, we put a particular focus on the wizard task and the different roles which it can comprise; especially with respect to language technology. The goal was to build a more solid theoretical foundation based on which an abstract tool architecture could be defined. This route from wizard task to tool architecture was important in order to support a set of different application scenarios, all of which might require different actions from the wizard. The focus was on a modular approach where a flexible configuration of different settings may be possible. Defining the modules, their interplay and how all this is reflected in an application interface for the wizard, was one of the key issues to be solved.

Finally, the third perspective (i.e. the interaction perspective) looked at the interaction with the WOZ tool. It comprised a set of distinct experimentations conducted to better understand the process of WOZ prototyping. This analysis looked at running as well as designing experiments including dedicated studies for both of these tasks. While, similar to tool support, certain aspects of running WOZ experiments could be found in the literature, our goal was to look specifically at those that are related to the wizard task. As already highlighted earlier, we consider the wizard as our user and therefore seek a better understanding of this side of the WOZ experiment process. However, insight could only be gained from the exploration of an actual study setting, and so we used several real WOZ experiments to drive this part of our research. In the following section, the three analysis perspectives are presented in more detail.

3.5.1 The User Perspective

While usually the participants that take part in a WOZ experiment would be regarded as the potential users, our goal here was to explore the WOZ method from a researcher's perspective. In this case, the person running the experiment, i.e. the wizard, constituted our user and therefore had to be analysed in order to understand their distinct requirements. We looked at the user from two different angles. We started with the literature and analysed the types of users that report on WOZ experimentation. We looked at their motivation for using the method and the settings they had applied when doing so. Here, the literature did not only provide enough data to obtain a solid overview on the usage of WOZ prototyping (cf. Chapter 2), but also served as the basis for the second analysis angle, which used an interview study with 20 participating

researchers from academia and industry to uncover some of the aspects of WOZ prototyping that are rarely discussed in publications.

Literature-informed User Perspective

As a first step we re-visited the different studies reported in the literature and analysed what type of researchers might be interested in using the WOZ method. We looked at their motivation for using the method and where they put the emphasis when conducting the experiments. As the related work part of this thesis has already highlighted (cf. Chapter 2), the literature shows a variety of different application areas for WOZ experimentation. While they all share the overall concept of using a human to simulate a system function, the settings between different application areas can vary significantly. Researchers who are interested in exploring emotional feedback, for example, might be less restrictive when it comes to defining the wizard task, than those who want to test certain components of a dialogue system. Our goal was to identify the different possible settings and map them to explicit system functions that had to be supported by the tool.

Elaborating on what we learned from reviewing publications we also thought about potential user groups outside the academic community. While the majority of WOZ experimentation might happen in the course of scientific exploration there is a valid use case for the method in interaction design, where applicants often work in a different contextual setting. Here the presented work conducted by designers in industry offered a first contact point. Additional insight was obtained by the interview study described in the next section. In summary the goal of this literature-based analysis was to obtain a basic understanding of the different user groups and their distinct motivations for employing the method. While some of them had overlapping interests we also found great differences between users. The analysis served as a necessary basis for the following interviews, which aimed at verifying as well as extending the obtained results.

Interview-informed User Perspective

As the literature alone merely served as a starting point for defining user requirements a more elaborate analysis approach was required to obtain more valid results; especially if the gained insight should go beyond simple functionality definitions and instead generate additional knowledge with respect to employing a known prototyping method. In the past researchers have applied different methods for building this sort of understanding of an area of interest. Examples include qualitative research approaches such as focus groups (e.g. Morgan [1997]) and field observations (e.g. Barrett et al. [2004]) as well as more quantitative studies such as traditional surveys and experiments. Although through reviewing the literature we had already built up a certain understanding of WOZ prototyping and what it involves, our goal was to expand beyond the obvious aspects. Hence, we used what we had learned from studying and analysing WOZ experiments and then augmented the results by conducting an interview study.

With this study, we aimed to verify the theories we had developed. The study also gave us the opportunity to explore those aspects of WOZ that are usually suspended when it comes to preparing research results for publication. As such, talking to people about their experience with the method served both, as an instrument of confirmation and verification as well as a tool for new discovery. A similar approach was, for example, used by Clemmensen [2004] to study work practices of HCI professionals. Like focus groups, this form of interview-based analysis tries to generate a broader understanding of an area by not only focusing on a dedicated set of questions but rather allowing for the data to evolve from un- or semi-structured discussions. In our case the knowledge generated from studying the literature served as the source for initial questions. The interview progress itself, however, was steered by the interviewee and the direction he/she wanted to go when talking about their experience. Interviews were fully transcribed and a formal analysis of the data, aimed at painting a more holistic picture of the WOZ prototyping method and its applications areas, was conducted. In summary, the goal of this interview study was to broaden our initial understanding of the field created by the literature and to fill-in the gaps and open questions that had remained.

3.5.2 The System Perspective

The second perspective of this mixed research approach looked at the system and tool aspect of WOZ prototyping. The goal was to identify the different modules needed to offer generic WOZ support, define their task and discuss their relationship to each other. Also, we wanted to find possible ways to integrate them into one platform and respect existing as well as future compatibility. Similar to the user perspective we approached this problem from two different angles. On the one hand we started from the wizard task and elaborated on the different forms and roles it can take on. By sufficiently describing the wizard task and its interplay with technology we were able to define the potential flexibility that the tool should allow for. While these tasks and roles were inspired by the literature and to a large part clarified by talking to other researchers, it was mainly the exhaustive nature of this theoretical analysis that helped to understand the various wizard/technology combinations. On the other hand, we looked at the technological possibilities for developing modern prototyping tools. The technological aspect was important as one challenge of existing WOZ tools is their often missing compatibility with other systems as well as their complex set-up routines. The use of open standards and low entry barriers was therefore a key issue to be solved if the support for WOZ experimentation should go beyond the currently supported single scenario solutions.

Role-based System Perspective

While there was certainly a strong tool focus in the practical part of our research agenda, which is specifically dedicated to developing a WOZ platform (cf. Section 3.4), the theoretical strand also supported this development process. It methodically discusses the different roles a wizard can possibly play when running experiments. To do so we specifically focused on language

technologies and how they potentially can be used in connection with WOZ experimentation. By looking at the LTC pipeline (Note: The LTC pipeline will be described more thoroughly in Chapter 5) and how the interplay between a wizard and the respective technology components can be achieved we were able to paint a comprehensive picture of possible use cases. Even though in our analysis the main focus was on experimentation with respect to language technology, we believe that the underlying principles are applicable to all sorts of WOZ settings.

The result of this analysis was then used to form a set of rules and standards that apply when wizards in their different roles interact with technologies. Integrating those rules as part of a software architecture represented an important part of the foundation for generic tool support and therefore served as an essential contribution for building the envisioned WOZ tool. Furthermore, by focusing on the task and its different characteristics we were able to clearly define this relationship between different technological components and how this interplay is on the one hand connected to the experimental setting and on the other hand depends on the role and task the human wizard has taken on.

In summary, this part of the analysis represented a theoretical expansion of what we had learned from studying the literature and from talking to other researchers. As such it served as an additional underpinning that had the goal to exhaustively define the sort of flexibility that is necessary to support generic WOZ experimentation.

Technology-based System Perspective

Another factor that was discussed with respect to the system perspective is the availability of different technological solutions, their advantages as well as their drawbacks. Different alternatives for implementing a defined software architecture were evaluated. Given that one of the identified problems with existing tools is their dependency on sometimes obsolete technology it was seen as a distinct requirement for this work to put a strong emphasis on future standards. Furthermore, compatibility with existing systems and services as well as its portability to different form factors had to be explored. The latter is particularly important if we think about the advent of language technologies used in phones and other mobile appliances, or their integration into cars and smart living environments. As the main purpose of WOZ prototyping is to evaluate and form future technologies and analyse how they are possibly used, we also need to think about ways of permitting this method to be employed in those places where the technology most likely will appear. This includes the traditional office as well as private and public living environments.

We started our exploration by comparing some of the existing tools and how they are implemented. We looked at their architecture (as far as possible) and the type of support they offer for generic WOZ experimentations. An analysis of possible improvements and how they realistically could be achieved was used as a basis for a discussion about potential technological platforms to be used for implementation. Here it was not only the simplicity of building such a tool that qualified for consideration but also the potential of integrating external compo-

nents. As by definition the wizard only represents an intermediate step towards a technological solution to an interaction scenario, it can be argued that this simulation should resemble the situation as much as possible. The potential for the integration of external services can therefore play a key role when choosing the right technology platform.

In summary, by comparing possible technology platforms for integrating WOZ support and looking at their interoperability with other potential services, we took what we had learned from our previous analyses, focusing on requirements- and task definition, and evaluated which of the current solutions offer sufficient support for building a flexible tool. This discussion was furthermore supported by the knowledge we had gained from a first round of prototyping (cf. Section 3.4.3).

3.5.3 The Interaction Perspective

Finally, the interaction perspective aimed at generating new insight into the challenges of creating and running WOZ experiments. The goal was to focus on the task of the wizard and the challenges a researcher needs to overcome when employing this form of evaluation. While all the previous parts of this research methodology focused on understanding WOZ in order to build a flexible tool the interaction perspective specifically looked at using this tool. As we followed an integrated development process which used realistic experiments conducted with different versions of the tool as a dedicated source of information, the interaction perspective served two purposes. Firstly, it was used to drive the development process and initiate the integration of new functions as well as the optimization of existing ones. Secondly, it aimed at expanding on what we already knew about WOZ prototyping. While all the conducted experiments were motivated by external research interests, they also helped to explore different aspects of the wizard task. In this respect we looked at third parties using different versions of our tool (i.e. one version per experiment) to answer dedicated research questions. This created the necessary context to realistically evaluate wizard workload, wizard consistency as well as the effort dedicated to constructing a WOZ experiment. Following, we describe in more detail why we think that these three areas are important factors having the potential to significantly influence the success of an experiment, and how we were exploring them.

Wizard Workload

Previous research has highlighted that the cognitive effort required to convincingly manage the task of the wizard is highly demanding (Salber and Coutaz [1993a]). To help with this task the wizard usually operates a multi-purpose interface that is customised to fit the particularities of a specific test setting. Our goal was to identify interface elements that might support the wizard beyond the frontiers of a single test scenario. To do so, formal usability tests (cf. Dumas and Redish [1999]) with small groups of users, testing different prototypes of our wizard interface, were used to identify difficulties and suggest improvements. Here the goal was to improve the intuitiveness of our solution without spending too much time on lengthy studies. While con-

ceptual problems were difficult to identify in these walk-up-and-use tests, we wanted to ensure that basic interface glitches were fixed before any in-depth analyses were started. In a way the tests can be seen as a number of trial runs before an actual experiment was conducted. Aiming at more solid feedback we then looked at three real WOZ studies, each of which employed a different wizard over the span of various experiments, and analysed how wizards performed over time. Doing so not only helped to see whether wizards got used to operating the interface but also allowed for identifying those aspects of the wizard task that remained challenging. In summary, this mixture of low-fidelity one-off usability testing combined with analysing long term wizard performance (over the course of a whole study consisting of several WOZ sessions with real test participants) produced results which consist of both direct user feedback as well as observed user behaviour.

Wizard Consistency

A second interesting aspect of running WOZ experiments concerns the consistency with which a wizard simulates system behaviour. While a technological solution would usually perform within a defined margin of consistency, humans are susceptible to changing behaviour; especially if the tasks they are supposed to perform are cognitively demanding. Analysing the log-files of three different WOZ studies we wanted to understand whether more experience leads to better performance. Both timing as well as consistency in terms of simulated system behaviour were evaluated. Statistical measurements were employed to look for learning effects which might lead to faster response times. Furthermore an in-depth analysis of the content wizards produced was conducted, which analysed how this content can vary throughout the course of an experiment. In summary the focus of this round of analyses was on understanding those aspects of the wizard task where experience helps and those aspects where even extensive training sessions fail in harmonizing wizard actions.

Experiment Construction

A final series of evaluations looked at the construction process of WOZ experiments. Software tools that allow for the simulation of system behaviour are often considered as throwaway applications (Dow et al. [2005c]). While resources spent on building test relevant environments seem a necessary investment in order to improve the overall quality of technology, it remains a challenge to convince stakeholders of their importance. Our goal, of building a generic tool for WOZ experimentation, aimed at reducing the amount of developing and set-up time needed for running evaluations. Editing functionalities similar to the ones found in web-based content management systems were meant to offer a straight forward and configurable solution for a variety of different WOZ settings ranging from pure text-based interactions to more sophisticated speech and multi-modal scenarios. In order to evaluate if these functionalities were effective and whether potential wizards were able to use them we ran a series of analysis where participants were asked to design an experiment. Different groups of potential wizard users were

analysed so that we were able to understand their distinct requirements and identify missing functionalities. We looked at their success rate in terms of creating experiments and evaluated the complexity of their creations. Furthermore, we collected their feedback with respect to the perceived usability of the tool as well as the difficulty level of the task. In summary, we aimed for a low-entry barrier and so this analysis looked at whether people were able to create WOZ experiments with our prototyping platform without undergoing a dedicated training session first.

3.6 Combining Methods

While all the different research angles described above integrated their very own research methods endeavouring to answer different research questions, it is the combination of the three perspectives - User, System and Interaction - that aimed at a better understanding of the WOZ method. Led by the user-centred development process of our WOZ tool we sought to identify challenges and problems connected to creating and running WOZ experiments. As opposed to traditional WOZ studies which usually focus on the user that interacts with a simulated piece of technology, we were particularly interested in the person simulating - i.e. the wizard - and aimed at a better understanding of his/her needs. It is important to highlight that the above described analyses were not conducted in a strictly progressive fashion but rather applied iteratively. The tool building process (cf. Section 3.4) was constantly interrupted by experimentation which led to new insight with respect to functionality and missing support. Similarly, the theoretical analyses of the user and system perspective acted as an important expansion to what we had already seen in previous work. Finally, realistic interactions with the different prototype versions helped to verify envisioned solutions as well as allowing for the identification of missing features.

In summary, employing an integrated approach by expanding a Systems Development Methodology using methods from User-Centred Design, we continuously iterated through different product perspectives using the insight garnered from theoretical work to drive the prototype design and the results obtained from prototype analyses to deduce new theories.

If we compare our mixed research approach with Nunamaker et al.'s (Nunamaker et al. [1991]) notion of systems development driven research, we may classify the user perspective as well as our exploration of existing tools and previous experiments, as the stage of defining a conceptual framework. The system perspective as well as the different prototyping efforts conducted is important for both the system architecture and the system design stages. Finally, the interaction perspective can be viewed as the main instrument of evaluation providing, not only feedback with respect to the system functions, but also generating additional knowledge about socio-technical aspects of using it in context.

3.7 Other Related Research Methodologies

Similar to Hasan's [2003] approach when exploring the creation of a web-based groupware system, this thesis is based on an adapted Systems Development Methodology (SDM). That is, we combined theory building, experimentation and observation processes taken from SDM with methods usually found in User-Centred Design (UCD) (e.g. personas, scenarios, early prototyping).

The reason for following this approach was not only to construct a 'usable' WOZ prototyping tool but also to contribute to the body of knowledge both with respect to the development process itself (i.e. what types of technologies are suitable for running WOZ experiments and how can they be integrated) as well as from the point of view of better understanding the various problems the tool aims to solve (i.e. what are the challenges a wizard faces when designing and running WOZ experiments) and the consequences such a solution would trigger (i.e. how does the WOZ tool change the actions of the wizard and how it may such influence experimental results). Development related methodologies such as SDM and UCD are by their very nature aiming for an optimal socio-technical integration and can therefore be seen as valid approaches to drive such an engineering-based research process. In the case of the research presented in this thesis it were mainly the iterative development cycles that were followed by both small-scale as well as more in-depth user studies which generated additional knowledge; in particular with respect to the use of certain tool features.

However, we would like to highlight that other related approaches may have led to comparable results. These approaches are mainly situated in the field of qualitative and interpretative research and might include Action Research (Lewin [1946]) and Grounded Theory (Glaser and Strauss [1967]), or slightly more distant schools of thought such as Constructivism (Glaserfeld [1989]) (if we consider ourself as students of the WOZ method), Pragmatism (Biesta [2003]) (if we consider WOZ prototyping as a valid theoretical framework whose practical instantiation is to be explored), or Activity Theory (Nardi [1995]) (if we consider a WOZ experiment to be an organisational construct in which the human wizard context-driven interacts with various artefacts).

The following sections will elaborate in some more detail on these schools of thought and discuss what distinguishes them from our integrated methodology approach.

3.7.1 Action Research

Action Research (AR) is a research and design methodology which requires an active participation in a team or community of practice (cf. Wenger [1998]) in order to address their distinct needs and problems and find suitable solutions. It is usually conducted by a third party, possibly an external researcher, who is brought into the organisation with the goal of understanding and improving its working practices. The methodology was proposed by Lewin (Lewin [1946]) at the end of the first half of the 20^{th} century and is since then understood as an instrument for organisational development. It follows the underlying principle of including affected peo-

ple in the decision process, arguing that when they are actively involved they are more likely to adopt. AR, as described by Lewin, involves a three step process connected by dedicated feedback loops. In the first step ('Unfreezing') the researcher works together with the team to understand the given situation and identify possible irregularities and problems. Affected parties are actively asked to take part in this exploratory process, whose result may be a joint plan for improvement. In a next step possible actions are compared and evaluated. This usually happens in a workshop where the researcher, together with the other team members, discusses the potential of planned solutions and their possible consequences ('Changing'). Finally, the third step puts actions into practice and evaluates the resulting system change. Feedback may lead to necessary adjustments until the newly created situation achieves satisfaction ('Refreezing').

Throughout its history AR has been subject to several additions and improvements. Action Science (Argyris et al. [1985]) for example focuses on possible actions and how humans design them to satisfy their personal intentions based on a given social context. Heron's Co-operative Inquiry (Heron [1996]) emphasises the knowledge creation process and argues that making organisational members part of the research team leads to better results, and finally Participatory Action Research puts forward a process of collaborative learning, which rejects more traditional forms of 'consulting' where one person (ie. the consultant or teacher) imparts information and advice on others.

While in terms of data collection, analysis and theory building AR and SDM employ similar methods, they differ in their underlying motivation. AR generally has the goal of identifying problems and changing behaviour (hence it is mainly adopted by social scientists), whereas SDM aims at supporting existing processes with effective technological solutions (hence its application is mainly found in engineering sciences). Focusing on this aspect of creating and optimising a new system, SDM differs from AR in that it offers three distinct domains in which additional knowledge may be created: (1) the way a system is built (development technique); (2) the properties of the system itself; (3) the socio-technological aspect of its usage. These domains are particularly important for building high quality software and therefore make SDM a viable methodology for our research program.

3.7.2 Grounded Theory

Grounded Theory (GT) is a research methodology mainly used in the social sciences. It was developed by Glaser and Strauss [1967] who initially referred to it as the constant comparative method (Glaser [1965]). Its goal is the discovery and development of a theory through the repeated analysis and interpretation of data. As such it constitutes a strict contrast to more traditional research approaches where experimentation aims at confirming or rejecting an already existing hypothesis. GT inverts this process by formulating hypotheses that fit the data. This process involves the constant comparison and conceptualisation of new data and ideas until a sufficient understanding of an area of interest is gained. GT can use any kind of data, although most commonly qualitative data collection methods such as interviews and observations are

employed. The analysis usually follows a four stage process. The first stage is referred to as the coding stage. Looking at the initial data the researcher is trying to identify key points (ie. codes) based on which additional data is gathered. Several rounds of data collection and subsequent coding are conducted until new data does not produce any more additional codes. The second stage then groups codes and the related data into higher concepts. Often this grouping leads to unexplored aspects requiring more information to be collected before the third stage defines categories of groups, which consequently constitute the basis for creating a theory. Finally the fourth stage of the process deals with writing a theory that consists of a collection of explanations for the researched area of interest.

After its original definition, GT diverged into two different schools of application whose main difference is rooted in the question of whether the researcher applying the method uses an already pre-defined scheme of codes (cf. Strauss [1987]), or whether new codes are employed as they emerge (cf. Glaser [1978]). The former has the advantage of providing a certain structure for the novice researcher, whereas the latter emphasises induction. While both interpretations of GT have since achieved acceptance as a qualitative research methodology, critics often argue that for the researcher it is impossible to be free of pre-conception, and therefore the validity of the resulting theory may be affected (Thomas and James [2006]).

If we compare GT to SDM we may not find a lot of convergence. That is, SDM highly emphases the aspect of creating an artefact, whereas GT focuses on building a holistic theory of an area of interest. Yet, SDM also comprises a stage of conceptualisation and theory building (cf. Section 3.1). While here the theoretical bases to be explored may be limited to the artefact and its use, it still represents a valid application domain for various GT methods such as data collection, coding and categorization. Similarly Muller and Kogan [2010] highlight this strength of using GT as a data analysis method in Human-Computer Interaction (HCI) and Computer-Supported Collaborative Work (CSCW). Hence, while we find a holistic GT approach as less appropriate for our research agenda, we do see a clear potential in employing some of its elements as a means to better understand our users and their demands.

3.7.3 Other Schools of Thought

Looking at other related research methodologies we do find elements applicable for the exploration of WOZ, yet less suitable for the creation of a new tool, and therefore inadequate as a holistic approach. Constructivism, for example, could provide a valid framework for the exploration of wizard learning strategies, and pragmatism might help to better link WOZ practises and theory, by focusing on wizard work processes and how they are adopted to contextual circumstances. However, both methodologies are theory focused and hence lack the mechanisms for integrating newly gained knowledge with applied prototyping. Similarly, Activity Theory (AT), which constitutes a research framework that places human activities in the centre of its analysis process, falls short on the technical implementation of systems. While tools and artefacts are certainly considered as a fundamental aspect of AT, their development and opti-

mization is usually not the main focus of the methodology. Rather it investigates how they are used to externalise and support mental processes.

Hence, we may summarize that while comparable research methodologies (most notably Action Research and Grounded Theory) can be considered as valid techniques for the exploration of WOZ prototyping and its demand for tool support, we believe that the notion of integrating continuous theory building with iterative system development, as it is emphasised by SDM, represents the most suitable approach for this particular research agenda.

3.8 Summary

This chapter has given an overview of the Systems Development Methodology (SDM) as well as the User-Centred Design Methodology (UCD). Integrating those two methodologies we have outlined our approach of exploring WOZ experimentation, which aims at generating new knowledge in three distinct domains: (1) by looking at WOZ from a user perspective, focusing on some of the subjective challenges of running WOZ experiments; (2) by looking at it from a system perspective, exploring the roles of the wizard and its influence on the system architecture of a tool; and (3) by looking at its interaction perspective, studying wizard workload, wizard consistency and experiment construction.

Having detailed our research methodology the next chapter will continue by discussing the initial stage of this agenda. We start with an extensive exploration of WOZ tools and frameworks that have been used in the past before moving on to a first phase of prototyping. While the focus of this work will mainly lie on systems that allow for running language-based WOZ experiments, we want to highlight once more that the resulting theories are considered generic and therefore may be applicable to other, non-language related experimental settings.

Chapter 4

Basic Requirements for Wizard of Oz Experimentation

"My head is stuffed with straw, you know, and that is why I am going to Oz to ask him for some brains"
– The Scarecrow

This chapter discusses our initial phase of understanding the requirements for building a Wizard of Oz prototyping tool. We start by looking at existing tools and frameworks reported in the literature and how they support WOZ experimentation (cf. 4.1). From there we move on to looking at previous experiments and what we can learn from them (cf. 4.2) before reporting on our initial developing activities leading to a low-fidelity prototypical implementation and sketches (cf. 4.3).

4.1 Existing Tools

Even though there seems to be a clear demand for supporting WOZ experimentation, only a limited range of applications offer adequate functionalities to do so. From the literature, the software tools and frameworks that have been used for WOZ experiments differ greatly between the different application scenarios. Furthermore, many of these tools and frameworks require a considerable amount of set-up time and often depend on obsolete technology. Many also do not appear to be publicly available.

Generally, applications and frameworks that support WOZ exploration can be separated into two categories. First, looking specifically at language-based interaction (as prototyping language-based interaction has earlier been identified as the main application area for WOZ), we find Dialogue Management (DM) tools. These tools focus on the evaluation of various technology components and their primary application lies in the field of Natural Language Processing (NLP) and machine learning. Second, more general WOZ tools, herein referred to

as pure WOZ tools, instead rely completely on human simulation, which makes them more suitable for a variety of exploratory analyses. Following we discuss these two categories of tools in more detail.

4.1.1 Dialogue Management Tools

Two of the better known examples for DM tools are the CSLU toolkit (Sutton et al. [1998]) and the OLYMPUS dialogue framework (Bohus et al. [2007]). Others include the JASPIS dialogue management system (Turunen and Hakulinen [2000]) and the EPFL dialogue platform (Cenek et al. [2005]). DM tools explore the language-based interaction between a human and a system, and aim at improving this dialogue. They usually provide an application development interface which is used by a programmer to specify the dialogue flow and its integration of different LTCs like ASR and TTS. Once designed the dialogue is tested using human participants. In doing so the main focus lies on testing and improving the quality of the employed technology components. Typically, these tools depend on these LTCs, which means that test results will depend heavily on the quality of the currently existing technology. Only crude support is available for human intervention in the form of WOZ.

The CSLU toolkit, for example, offers speech-recognition, natural language understanding, speech synthesis as well as a talking head. Modules are integrated into a stand-alone graphical authoring environment, which permits dialogue flows to be specified. Dialogue elements are dragged onto a canvas where they can be arranged and linked using a flow-chart-like notation that also supports decisions, random generators and loop backs. Input and output can be defined separately for each element so that it is possible to integrate and combine text, spoken and touch-tone based interaction. Even though an integration of WOZ support was planned, to our knowledge, the functionality never made it into any of the final product releases. In general, however, the CSLU toolkit can be seen as a straight-forward prototyping tool that requires little experience, which makes it suitable for both designers as well as NLP researchers. Furthermore, its simple graphical interface increases accessibility for people without a technical background.

In contrast, the OLYMPUS dialogue framework constitutes a powerful client-server environment for implementing and running spoken dialogue systems. The goal of the framework is to provide a highly scalable platform for language technology research, yet its support for quick prototyping is low. Composed of several different components (i.e. an audio server, the APOLLO interaction manager, the PHOENIX grammar parser, the RAVENCLAW dialogue manager (Bohus and Rudnicky [2003]), the ROSETTA language generator, and the KALLIOPE speech synthesiser), none of which provides a graphical interface, it requires a high level of technical know-how to set-up and use. Also, while WOZ experimentation is certainly possible (cf. Bohus and Rudnicky [2005]), it requires the relevant component to be built on a one-off basis and integrated with the rest of the framework. A ready-made WOZ client is not available. Nevertheless, this loose coupling of different technology components allows for a high degree

of flexibility, which makes the framework particularly suitable for evaluating new technologies.

Similarly, the JASPIS dialogue manager provides high adaptability. While predominantly aimed at building working systems, it puts a strong emphasis on information representation. The XML-based output allows for integrating natural language into applications that run on different devices, representing a range of form factors (e.g. Turunen et al. [2005]). Also here WOZ experiments have been conducted in the past (e.g. Mäkelä et al. [2001]), though the integration was achieved through building a one-off interface rather than integrating generic WOZ support.

Finally, the EPFL dialogue platform shows the best support for WOZ integration with language technology. Based on the Rapid Dialogue Prototyping Methodology (RDPM) it automatically creates a graphical wizard interface based on a pre-defined application model (Rajman et al. [2006]). This automatic generation of interfaces makes the platform interesting for researchers that have little technical knowledge. In addition it supports multi-modal as well as vocal designs, extending its application domain beyond purely language-based interaction paradigms. The platform is, however, not publicly available and so a more detailed analysis was unfortunately not possible.

In summary, existing DM tools highlight several important requirements for supporting the prototyping of language-based applications. Graphical, stand-alone tools like the CSLU toolkit provide a 'low barrier to entry' option for non-experts. Also, we see that a high degree of component flexibility, as demonstrated by the OLYMPUS dialogue framework, opens up a wider range of possibilities, especially when it comes to evaluating different technological solutions. Furthermore support for new form factors, as can be found with the XML-based architecture of the JASPIS dialogue manager, seems crucial, particularly when we think about mobile phones, tablet computers and their potential successors. Finally, the dynamic generation of interfaces, whether for wizard or for tested client interfaces, is a feature that helps to significantly reduce the prototyping time.

While the described examples might not cover the totality of DM tools that have been used in the past, they all show that WOZ prototyping has its place in dialogue design - although sometimes separate development work is required to produce the relevant interfaces. Combining some of the features presented with a better WOZ integration might therefore create a new class of product; one that is both, flexible and easy to use without requiring a significant amount of upfront training.

4.1.2 Pure Wizard of Oz Tools

Unlike DM tools, pure WOZ tools try to more fully support low-fidelity prototyping. While these applications offer a higher flexibility than DM tools, they usually do not integrate actual working LTCs. Instead a human mimics the functions of the system, which allows for a less restrictive dialogue design. In addition it facilitates the testing of user experiences that are not yet supported by existing technologies. Pure WOZ tools are, however, scarce and tend to be

only suitable for the one experiment for which they were constructed. Hence, they are often categorized as throwaway applications i.e. they are built for one scenario and only rarely re-used in other settings. Two publicly available tools that allow for more generic experimentation include SUEDE (Klemmer et al. [2000]) and Richard Breuer's WOZ tool[1]. An alternative, yet not publicly available, solution may be found in the NEIMO platform.

SUEDE makes it possible for a researcher/designer to rapidly create prompts and supports the graphical design of a dialogue flow. It lets the researcher/designer record these prompts, arrange them, and play them back to a test participant. During experimentation a participant's responses can be captured and the collected data be made accessible in the form of a browsable HTML document, which can be used as a reference for future design improvements. The advantage of this type of evaluation is that a designer can focus entirely on the interaction, while keeping the quality of the speech output (i.e. the pre-recorded prompts) consistent; something that might vary if actual technology components are used. Also, the exclusion of third party components reduces the complexity of a test set-up, which ultimately leads to less time spent on the configuration of experiments. Two main features of general interest in SUEDE are first, the provision of a simple way of recording, organising and playing back potential system prompts and second, structured access to participants' responses, which is crucial for understanding and consequently improving the overall interaction.

Following a similar goal, namely supporting simple, generic WOZ experimentation, Richard Breuer's WOZ tool puts a stronger focus on more complex dialogue designs. It does not offer a dedicated function to record prompts, however, one can link dialogue elements to stored audio files or make use of an integrated Text-to-Speech function. Furthermore, the tool allows for dialogue flows to be exported as VoiceXML[2] or RTF, so that they can be re-used in third party products. This support of XML standards is an important feature that helps to integrate WOZ experimentation as a fundamental part within the development cycle of new dialogue systems.

Finally, a third application, the NEIMO platform, was mainly used in the 1990's to study the potential of multi-modal user interfaces (Balbo et al. [1993]) and has later been expanded into a multi-workstation usability lab for the analysis of multi-modal interactions (Coutaz et al. [1996]). The demand for evaluating this type of applications has significantly increased since then, and so it is useful to consider support for multi-modal interaction when developing prototyping tools. One important finding of the time, with respect to WOZ prototyping, was that additional modalities also significantly increase a wizard's workload. Hence, NEIMO was one of the first tools to support multiple wizards. Roles could either be split up between the different modalities or were dedicated to the input/output interpretation/generation, on the one hand, and task level processing on the other.

In summary, the distinct feature-set of the pure WOZ tools discussed here ranges from simple graphical interfaces for designing a potentially multi-modal interaction, to powerful logging and export functionalities that make use of industry standards such as VoiceXML and

[1] http://www.softdoc.de/woz/index.html [Accessed: April 7^{th} 2012]
[2] http://www.w3.org/TR/voicexml21/ [Accessed: June 9^{th} 2012]

therefore smooth the path to a possible integration with external systems. Combining these features seems to be the logical next step towards better WOZ support.

However, if we look at more recently created WOZ tools, we find less generic support and a rather high focus on very specific application scenarios, for which the development effort that is required to adapt the tools to different domains is often very high. Examples include MDWOZ (Munteanu and Boldea [2000]) and DIAWOZ (Fiedler and Gabsdil [2002]) for dialogue systems, QUICKWOZ (Smeddinck et al. [2010]) to study embodied conversational agents, WOZ PRO (Hundhausen et al. [2007]) and SKETCHWIZARD (Davis et al. [2007]) for simulating pen-based interaction, and POLONIUS (Lu et al. [2011]) and DOMER (Villano et al. [2011]) to control a robot. WOZ functionality can also be found in TOPIARY (Li et al. [2004]) and BRICKROAD (Liu and Li [2007]), tools for prototyping location-enhanced applications. Finally, Liu et al. [2009] describe a WOZ interface to support the study of information presentation strategies for spoken dialogue systems and, Otto et al. [2011] developed a tool based on the SEMAINE framework[3], a multi-modal dialogue system that aims at sustaining conversations with human users.

An exception to these rather specialised tools can be found in OZLAB (Pettersson and Siponen [2002]), a multi-functional prototyping tool for multiple forms of WOZ experimentation. Here the authors explicitly highlight the re-usability of their tool. However, we were unfortunately not able to evaluate the application in more detail, as it is not publicly available. Finally, focusing on information retrieval tasks, Scherer and Strauß [2008] present a rather flexible WOZ environment for spoken dialogue systems that makes use of other parallel output modalities (e.g. a talking head). But again, an in-depth analysis was not possible due to the lack of availability.

In summary, the majority of existing WOZ tools either suffer from dependencies on obsolete software components and accompanying compatibility problems or pose considerable challenges when there is a need to adapt them to new application scenarios. Also, they are rarely publicly available, which restricts their re-use by other researchers. In addition, a potentially problematic issue with most of the tools is their shift from relying completely on technology to relying completely on the actions of a human. These are both extremes on what we can see as a continuum, where the intervening points represent a mixed-fidelity approach in which imperfect components can be incorporated along with human intervention.

4.1.3 Generic Tools

While the above analysis discussed the two different forms of WOZ support that are currently available (i.e. support through existing wizard functionalities in dialogue management tools and also support through dedicated WOZ applications) one may also conclude that their efficient integration in the product development cycle is difficult. One sign for this argument can be found in the fact that applications that support the simulation of an interaction are com-

[3] http://semaine.opendfki.de/wiki/SEMAINE-2.0 [Accessed: June 19th 2012]

monly seen as throwaway tools that are only used for a certain number of experimentation rounds. While efforts have been undertaken to increase the re-usability of applications with some researchers going as far as aiming for providing a generic evaluation platform, it is the distinct requirements of different test scenarios that often require significant changes to be implemented. Within the same research team these changes are usually implemented, which leads to tools that try to be generic, but to some extent include features that are very specific to a certain application area. When it comes to using the tools of third party providers, adoption may fail due to missing technical documentation or simply because of the amount of time that is needed to become accustomed with the relevant code base, which may be comparable to what it would cost to build a separate tool. We mainly see this when looking at the variation of pure WOZ applications presented in the literature. For example POLONIUS (Lu et al. [2011]) and DOMER (Villano et al. [2011]) are both wizard interfaces that are essential used to control a robot. While the underlying research interest for which they were built might differ, they share the same core functionality, namely sending more ore less pre-defined commands to a remote system. Consequently, given the availability of the code and sufficient amount of documentation, both research teams could each have used the tool of the other team. One could even go a step further and argue that both teams might have been able to use QUICKWOZ (Smeddinck et al. [2010]), a WOZ tool that was presented a year before their respective research results were published. Although the focus of QUICKWOZ lies on avatar-based interaction, the underlying concept remains the same. From an application point of view, the difference between sending commands to a robot or controlling the feedback given through an animated character on a screen is rather small. Similarities can also be found between MD-WOZ (Munteanu and Boldea [2000]) and DIAWOZ (Fiedler and Gabsdil [2002]) as well as between WOZ PRO (Hundhausen et al. [2007]) and SKETCHWIZARD (Davis et al. [2007]). An obstacle for re-using a tool may, however, be found in the programming framework that is used to build it, although in this case a possible solution can be the provision of appropriate software interfaces.

If we look at dialogue management tools, generic support seems more complicated. In this case, as WOZ is often treated as an integrated function rather than being a separate external tool environment, the potential re-usability is limited to the re-usability of the entire dialogue framework. Here WOZ changes from being an evaluation method informing the design to being an integral part of the final product. Nevertheless, by focusing on a modular composition as demonstrated by OLYMPUS (Bohus et al. [2007]), the WOZ function could be liberated and consequently re-used with other comparable frameworks. An advantage of this separation process is not only that a dedicated WOZ module could be integrated with other dialogue environments, but it would also allow WOZ to be treated as as an independent component whose further development could be influenced by multiple parties inside as well as outside the given research team. External feedback would also make it easier to improve and fine-tune existing functionalities as well as allow for gradually integrating novel use cases inspired by new application scenarios.

While interoperability between WOZ tools seems desirable, both to save resources spent on building proprietary solutions and to expand rather than re-build already existing functionality, the applications that have been published in the literature show that the exchange between research teams is rather limited. Despite the fact that numerous examples advertise their high flexibility and easy reconfigurability so that in theory re-use of applications would be possible, researchers solve their specific problems by creating new tools rather than improving existing ones. One reason for this is surely to be found in the varying research interests and their very distinct requirements when it comes to tool support. Another aspects is probably that for applications outside the NLP domain WOZ often plays a minor role for which a quick-and-dirty solution is usually sufficient and building re-usable components might seem unreasonable. Furthermore it is possible that missing access to existing tools as well as insufficient documentation prohibit their use and adoption (Note: SUEDE and Richard Breuer's WOZ tool are the only pure WOZ tools which we were able to download. With dialogue management tools we were restricted to the CSLU toolkit and the OLYMPUS dialogue framework). Finally, little effort has been put into understanding and improving WOZ prototyping to the point where it can be treated as a separate area of competence. After all, WOZ is for most research teams a method to evaluate and improve the design of an envisioned future product. To do so, they come up with solutions that mainly serve their very specific requirements. Tools whose goal is to provide improved generic support for the WOZ method are, however, missing.

4.1.4 Existing Tool Support

Based on our previous analysis it can be said that while WOZ is certainly a valuable prototyping method, the number of tools that support generic experimentation is rather low. From a natural-language point of view, we find a handful of dialogue management frameworks that integrate LTCs such as ASR and TTS, and on the low-fidelity, side allow for designing and testing a certain dialogue flow, and on the high-fidelity side represent full-grown dialogue systems with the accompanying set-up and configuration complexity. On both sides, WOZ experimentation is to some extent supported, however it usually requires additional development and integration effort. In addition to dialogue management tools we find dedicated WOZ tools, which completely replace the functionality of (a) component(s) by a human wizard. These, however, tend to be built for very specific scenarios and are difficult to re-use in other experiments. The WOZ tools that follow a more generic application model are usually based on obsolete software technology and difficult to extend or integrate into existing software environments. In addition it seems that available tools allow for using either a wizard or a technology component. The use of both, where the wizard acts in collaboration with (error-prone) technology, is currently not supported.

From this analysis we can derive that more generic support for WOZ experimentation is missing. Existing tools are either used in isolation, are very specific and hard to adapt, or they require complex client-server configurations to be used. Particularly with respect to lan-

guage technologies we, however, see an increased demand for low-fidelity prototyping (cf. Chapter 2). Support needs to be generic, adaptable, expandable and at the same time easy to configure and possibly capable of being integrated into existing application platforms. In order to achieve this, it seems insufficient to only solve the technical problem of developing a new prototyping platform. We rather need to better understand the WOZ method in general and in particular, we need to better understand the distinct demands of the people employing it, in order to better support them. Here it seems especially important to look at the role of the wizard and explore different aspects that might help make the method more robust and consistent and results, therefore, more representative.

To do so, a series of investigations have been undertaken which will be described in more detail in the following sections. We start with an analysis of previous experiments and their users, their distinct goals as well as the different problems and challenges they were facing. From there we move to defining a set of general requirements for WOZ tool support. Next, a more detailed discussion of the wizard task (with respect to simulating language technologies) and its different configurations leads to the definition of a theoretical tool architecture (cf. Chapter 5), before finally a WOZ prototyping platform, which was built alongside this exploration process and integrates its insights, is evaluated and used for a set of additional experimentations (cf. Chapter 6).

4.2 Previous Experiments

From the very early stages of WOZ prototyping, researchers have tried to find ways of improving the experimental set-up. While at the beginning hiding the human in the loop was the main concern, later applications were dealing with aspects of the validity of a simulation as well as its consistency. From the literature we can therefore define a set of recommendations for WOZ experimentation, that is largely based on previous work. Identified recommendations are illustrated below and summarised in Table 4.1.

4.2.1 Basics

Before running an experiment one needs to clarify whether Wizard of Oz is a suitable method given the desired research interest. Fraser and Gilbert [1991] argue that in order to successfully conduct a WOZ study it must be possible to simulate the future system given human limitations, it must be possible to specify the future system's behaviour and it must be possible to make the simulation convincing. Only if these demands are fulfilled it is sensible to conduct a WOZ experiment.

A second aspect that should be clarified before preparing a study relates to the level and quality of simulation. While usually the goal is to realistically simulate one or more system functionality/ies that is/are currently not available, it can also be desired to use a wizard to equip a system with possibilities that go beyond the current state of the art. As Edlund et al.

[2008] state *"If the wizards are permitted to use whatever means they are given to the best of their ability and any restrictions imposed on them are encoded in the software, then the wizards' actions represent the target the component designer should aim at"*. Having analysed whether a simulation is possible and furthermore defined to what extent this simulation should take place, the next step is to prepare the envisioned WOZ experiment.

4.2.2 Preparation

From reported studies we see that a significant amount of time needs to be put into preparing a WOZ experiment. This starts with the definition of a possible interaction model and ends with recruiting the people that are involved. The first thing to define is the actual interaction scenario that is to be tested. This involves describing the task a participant has to solve, the means that are available to do so (i.e. applications, input-modalities, physical aids, etc.), as well as the given time-frame.

Furthermore, one needs to clearly define the type of people that constitute the main users for the envisioned interaction. While it is often difficult to recruit exact representatives of this target population (i.e. numerous examples in the literature show that participants tend to be university students rather than carefully recruited users representing the exact target group for a product) doing so helps to both fine-tune the scenario as well as interpret the results.

After thoroughly defining the participant's side of the experiment one also needs to pay some attention to the system. More specifically, one needs to define the capabilities and restrictions of this envisioned piece of technology. Taking the example of simulating a spoken dialogue system, this often involves creating a number of possible utterances which can be employed to interact with a participant. Along with this communicative capabilities also comes a set of rules about how they are to be used. While in some studies (e.g. Gould et al. [1983]; Villano et al. [2011]) researchers might have used unrestricted (chat-style) interaction most experiments reported in the literature (e.g. Dahlbäck et al. [1993]; Cheng et al. [2004]; Hajdinjak and Mihelic [2003]) have employed pre-defined (canned) utterances to restrict and control the wizards in their actions. Designing and evaluating these utterances can take a considerable amount of time and the result might have great influence on the success of a study. Furthermore, in the case where the study aims at simulating multi-modal interaction scenarios, utterances also need to be represented in all relevant modalities. A hybrid approach where pre-defined elements are mixed with on-the-fly interactions is possible, but needs to be supported by the WOZ tool employed (e.g. Rajman et al. [2006]).

After the participant as well as the system side of an experiment are defined, the next step is to search for a person to act as the wizard. As the wizard task is a major part of the WOZ method and the person performing it can have significant influence on the results of the experiment, it is advised to put sufficient thought into who to recruit.

4.2.3 The Wizard

One of the most important questions to ask when running a WOZ experiment is: Who will act as a wizard? As Peissner et al. [2001] assert *"The wizard holds a key role for the validity of the study"*. Gould et al. [1983] chose a trained secretary to simulate their 'Listening Typewriter' because she was able to type 80 words per minute, was excellent at following the rules of the simulation, remained cool, did not provide a participant with any help, and was available for several months. While having such an expert acting as the wizard is usually recommended, sometimes it can also be the cause of some difficulties. Wirén et al. [2007] for example, report that using experts (in their case those experts were call centre agents) was problematic because they grasped what a caller wanted and therefore interrupted before a caller had finished speaking, leading to an erroneous simulation. However, this aspect of 'too good' wizards does not often arise as, due to the lack of access to specialists, researchers usually act as wizard(s) themselves, for which the problem of poor wizard behaviour is related to missing expertise rather than to having too much expertise.

Whether confronted with too much or not enough experience, the best way to improve wizard behaviour is to intensively train for the task. Wizard training is therefore a common recommendation that is found in numerous articles and text books describing the WOZ method (e.g. Dahlbäck et al. [1993]; Bernsen et al. [1994]; Wooffitt et al. [1997]). More precisely, it needs to be ensured that the wizard correctly simulates the system for which knowledge in at least three areas is required: the application domain, the system capabilities being modelled, and the tools available to assist in playing this role (Fraser and Gilbert [1991]). Gould et al.'s [1983] typist practised with the simulator for two to three weeks prior to the experiments and Wirén's [2007] customer care agents also had to undergo a one week preparation phase. Only when the wizard is confident performing the simulation should initial pilot tests with the entire system be conducted.

4.2.4 Pilot Tests

Not only should the wizard skills be developed but the whole experiment also needs to be tested using dedicated pilot studies. Dahlbäck et al. [1993] report that they conducted between 20 and 40 test runs before the actual experiments could start. While this run count seems very high, the quality of the experiment results often depends on a flawless set-up and consistent wizard actions. Read et al. [2005] for example, argue that reliability of the findings from a WOZ study can be compromised by a poor experimental set-up, or by a poorly prepared wizard. Conducting several pilot tests to ensure that both the experiment set-up as well as the wizard meet the desired quality level seems therefore crucial for the success of the overall study.

Having looked at previous WOZ experiments and what we have learned from them we now move to describing our initial prototyping stage. This first hand-on experience with the method aimed at a better understanding of the challenges involved in building such an application as well as a further exploration of its potential design space.

Basics	Clarify whether Wizard of Oz is a suitable research method
	Define the level and quality of the desired simulation
Preparation	Define the interaction scenario
	Define target population for the simulated technology
	Define the capabilities and restrictions of the simulated technology
The Wizard	Choose the right wizard
	Train the wizard
Pilot Tests	Run several pilot tests

Table 4.1 – Summary of recommendations for WOZ found in the literature.

4.3 Building Prototypes

Building prototypes is an important task that contributes significantly to the understanding of the different aspects of a product. Often the only way to test whether an idea works is to create a low-fidelity artefact and obtain feedback from a number of potential users. This process of constructing a prototypical implementation of a tool or a service not only helps to gain user feedback but also allows for reflecting on different possible solutions. In order to prototype a WOZ tool we therefore went through different phases of design, development and evaluation. First, to guide identification of distinct tool requirements, we looked at possible application scenarios in which such a tool could be used. Then, before the actual construction work could start, we once more re-examined the literature in order to search for aspects of WOZ that needed particular attention when building this type of tool. While previously we discussed a set of recommendations for WOZ prototyping in general, now we were more specifically looking at technical restrictions, interface concepts and functionalities, which directly refer to the tool building process. After that an initial prototype, which was based on one of the earlier defined scenarios, was built and evaluated. From then onwards we used paper sketches to reflect upon the obtained results. The goal was to move away from a prototype that only supports this single application scenario to one that provides more generic WOZ support. Following we describe this path, from defining scenarios and searching the literature for aspects of tool support, to building an initial prototype and creating sketches in more detail.

4.3.1 Identifying Potential Scenarios

Scenarios, usually in connection with personas, are a widely used technique in interaction design (Cooper [2004]). Identifying hypothetical archetypes of users and describing them helps to define the boundaries of a given design space, whereas a specific context of use keeps the designer focused on the actual goal to be achieved. In our case we used scenarios as a means of requirements gathering. As reported earlier, existing WOZ tools are predominantly built to support very specific experiment settings. In order to collect requirements for more generic support we therefore re-examined the literature and looked for a variety of realistic scenarios

that would help us define the type of settings that our tool would need to support.

Inspired by these examples, as well as research interests expressed by some of our partners, we ended up with four distinct use cases for potential WOZ experimentation (cf. Table 4.2). Based on real research interests, they also reflected our attempts of employing and combining a variety of different language technologies, and therefore could be seen as a valid basis for our initial prototyping efforts.

Scenario 1	A MULTILINGUAL CONVERSATION: *A smartphone application used to translate between a Japanese taxi driver and an English speaking business traveller.*
Scenario 2	LOCATION-BASED STORYTELLING: *A travel device that tells stories to tourists triggered by their current location.*
Scenario 3	THE MULTI-LINGUAL INSTRUCTOR: *A multi-lingual 'Avatar' that talks to a user through the set-up process of a video game console.*
Scenario 4	THE ADAPTIVE HELP-DESK: *A multi-lingual info-point terminal that advises customers about the best internet connection that fulfils their specific requirements.*

Table 4.2 – Potential scenarios inspired by situations in previous WOZ studies and research interest expressed by our partners.

4.3.2 Learning from Existing Tools

In addition to valid scenarios and general recommendations, previous work on WOZ prototyping also taught us some guidelines and requirements for tool support. Specific aspects that were addressed include the technical set-up, the wizard interface as well as a set of functionalities that may help to make a wizard's life easier. Recommendations with respect to tool support are illustrated below and summarised in Table 4.3.

Technical Set-up

The technical set-up of a WOZ experiment usually consists of at least two different stations. One of them is dedicated to the wizard and one represents the system a test participant is interacting with. Unless the experiment is envisioned as an early stage exploration of a possible design space, for which a convincing simulation is not necessary (cf. Section 1.1.1), researchers usually face the challenge of physically separating the wizard from a participant, while keeping as much contextual information as possible. This is typically achieved through creating a network connection between the wizard system and the technology probe used in the simulation. In addition, researchers use video cameras and microphones to capture as much information

as possible, often including close-up front views of participants' faces as well as their screens (Cheng et al. [2004]). The latter can be achieved either through a video splitter (cf. Cheng et al. [2004]) or a dedicated screen capture application such as REAL VNC[4]. The collected data is used for both real-time information for the wizard and post-test analysis. While usually this physical separation between wizard and participant is required to guarantee an effective simulation, in some settings it can be of less importance. Read et al. [2005] for example report on WOZ studies conducted with children in which case having the wizard in the same room was barely noticed by the participants but significantly increased the wizard's awareness of the user's situation and context and therefore allowed for providing more efficient feedback to the system.

In addition to the physical set-up requirements the number and type of hardware components that are used can have a great influence on the quality of the simulation. For example, Davis et al. [2007] note that their wizard interface runs best on a large, high resolution computer screen while Rajman et al.'s (Rajman et al. [2006]) multi-modal WOZ setting requires a total of four different monitors and space for two operators on the wizard side of the experiment. Where on-the-fly speech-output is part of the simulation, one further needs to think about using some sort of vocoder technology in order to achieve what Cheng et al. [2004] call a machine-like 'sonic' sound signature. While previous studies (Bernsen and Dybkjaer [1993]) suggest that voice output filtering might not change a participant's belief in whether he/she is interacting with a real system, it can be important in settings where pre-recorded output is combined with on-the-fly wizard output (cf. Smeddinck et al. [2010]). If both are distorted, either through some sort of audio filter or other forms of online distortion, the difference between prepared and on-the-fly improvised actions can be blurred.

Finally, with regard to the quality of the employed hardware, we would like to note that delays caused by a slow network connection can hinder speedy and accurate wizard responses. Also, poor audio-visual equipment might lead to problems when analysing the recorded data or using it for training technology components. In general it is therefore advised to devote some thought to the physical as well as the technical set-up of an experiment and test it so that these contextual factors do not impact on the study results.

The Wizard Interface

The wizard interface represents the main interaction channel with which a human operator tries to accurately simulate the functions of an envisioned technology. It has been highlighted by several researchers (e.g. Salber and Coutaz [1993a]; Bernsen et al. [1994]) that this task is highly demanding and therefore requires effective tool support in order to be achieved consistently. Dow et al. [2005c] for example argue that the wizard interface, just as any other user interface, should be designed with the wizard operators' perceptual, cognitive, and motor skills in mind. Rajman et al. [2006] furthermore highlight that different modalities require different

[4] http://www.realvnc.com/ [Accessed: July 24th 2012]

cognitive efforts. While a participant's pointing can be processed automatically by the wizard and consequently might lead to a quick response, language input requires interpretation, which naturally takes more time. One goal of the wizard interface can therefore be to accommodate and control such differences.

Usually the wizard interface consists of a finite set of actions that may be triggered based on a participant's input. Again taking the example of a spoken dialogue system, the wizard might choose from a list of pre-generated Text-to-Speech utterances (for simulating multimodal interaction, these utterances might be augmented with related audio-visual feedback produced by a conversational agent). A well designed wizard interface should therefore help in finding the right response. In the past researchers have promoted different solutions for solving this problem. Rajman et al. [2006] used semantic pairs to filter possible responses whereas Cheng et al. [2004] created a 'SentenceSelector' function which allowed for cutting down the number of possibilities by typing only a few salient words into a search box. Munteanu and Boldea [2000] even offered logical comparison and wild-card operators for performing more complicated queries.

Further highlighting the importance of this selection process Bradley et al. [2009], describe how they went from a wizard typing whole responses, to a complex interface consisting of numerous canned text buttons, to finally end up with a drop-down list featuring predictive text. A similar solution was also used by Scherer and Strauß [2008] to help the wizard find appropriate responses. Researchers have also recommended to logical group of responses (Villano et al. [2011]), to offer dedicated 'favourites' that can be pre-programmed by the wizard, and to integrate an event stream that logs previous actions (Davis et al. [2007]). An adaptive interface that changes its content based on pre-defined interaction states has also been found to be advantageous (Lu et al. [2011]).

In contrast to relying on the capabilities of the wizard interface, Peissner et al. [2001] argue that a wizard needs to know exactly the dialogue flow, especially the permanent available functions and prompts, in order to be able to manage the complex interface. Furthermore, deep knowledge of the grammar including permissible variations is needed, for which extensive training will be indispensable in order to gain the required speed and competence in the decision process. We believe that a combination of both a supportive wizard interface as well as appropriate wizard training are necessary, in order to achieve a reliable simulation.

Additional Functions

While a usable wizard interface is important for running an effective simulation, other features might help during the preparation and analysis phase. Several of the wizard tools found in the literature differentiate between the 'designing' and the 'running' stage of an experiment. Some also offer functions for analysing the collected data. Davis et al.'s [2007] SKETCHWIZARD for example, enables designers to build and run simulations of pen-based user interfaces. They also introduced a safe area where an experimenter can prepare and train interaction turns (=pen

strokes) without updating a participant's view. Also Hundhausen et al. [2007] differentiate between design mode, edit screen and run mode, when describing WOZ PRO, their tool for prototyping pen-based interactions. Klemmer at al.'s [2000] SUEDE, on the other hand, supports analysis by letting researchers look at an already conducted experiment in the form of a click-able HTML document.

In addition to supporting the different phases of a WOZ experiment it seems furthermore important to allow for easy extension and reuse of a tool. For example, Munteanu and Boldea [2000] report that while in their study automatic speech recognition, semantic analysis, dialogue management, and natural language generation were only simulated, provisions have been made for them to be easily included as they become available in the future. Similar adaptability was shown in OZLAB which has been used in a variety of different studies (e.g. Pettersson [2002]; Pettersson and Siponen [2002]; Molin [2004]).

Also the support of certain standards as well as proprietary formats for collecting, importing, exporting and defining data seems crucial, especially if one needs to work with other third party applications. Smeddinick et al.'s [2010] QUICKWOZ for example allows for controlling the animations performed by their 3D characters through XML, which creates a possible link between the WOZ tool and other applications. Dow et al's [2005c] DART, on the other hand, stores the time-synchronized data produced by a WOZ session in ADOBE DIRECTOR[5] casts (collections of Lingo scripts, text files, and other media), which later can be re-imported into other applications capable of handling those formats.

Another form of support can be found in automatically creating wizard interfaces based on a designed interaction scenario. Dow et al. [2005c] address this feature by offering the generation of an interface consisting of buttons and the corresponding list of possible actions. Similarly, Rajman et al.'s [2006] EPFL dialogue platform automatically builds interfaces based on a pre-defined application model and Smeddinick et al.'s [2010] QUICKWOZ automatically creates individual buttons for the most frequently used animations based on an expression list.

In order to allow for this sort of automation a WOZ tool would need to offer a high degree of flexibility. Usually this sort of adaptability can only be achieved through a highly modular architecture, where it is possible for different parts to be turned on and off without influencing the stability of the overall application. An architecture like this also allows for better localisation and customisation, so that for example the language for Text-to-Speech can be switched by simply exchanging a module. Adaptability, however, should not end at the wizard side of an experiment. Customisable client (ie. participant) interfaces for conducting comparative studies also offer more flexibility in terms of possible interaction scenarios. Pettersson and Siponen [2002] for example explain that in OZLAB pre-written sentences, like any other graphical object, can easily be made visible for a participant. Studies with alternate modalities are therefore easy to conduct without investing much time into adapting the client interface.

Generally summarising the characteristics a WOZ tool should possess, Salber and Coutaz [1993a] list flexibility, resuability, genericity and extensibility. Flexibility concerns not only the

[5] http://www.adobe.com/products/director/ [Accessed: July 25^{th} 2012]

Technical set-up	Define the physical set-up of the experiment
	Select set-up appropriate hardware
	Select and install appropriate software
	Test technical set-up
The wizard interface	Support action selection
	Offer customization possibilities
	Run wizard training sessions
Additional functions	Support design, conduct and analysis of experiments
	Support different formats for import, export and modification of data
	Support automatic generation of wizard and client interfaces
	Allow for flexibility, re-usability, genericity, and extensibility

Table 4.3 – Summary of tool-based recommendations for WOZ found in the literature.

adaptability to a certain experiment set-up, but also the possibility to employ several wizards and support the different roles they can take on. Reusability, on the other hand, comprises the characteristic that it should be possible to re-use the WOZ tool in other (similar) experiments, as opposed to spending time and resources in building a new application every time the method needs to be applied. This also tries to tackle the problem of WOZ applications being throw-away tools as highlighted by Dow et al. [2005c]. With genericity Salber and Coutaz [1993a] further refer to the aspect of supporting WOZ experimentation at different levels. They give the example of a handwriting recogniser where WOZ could first be used to study the integration of handwriting in an interface, before in a second stage, a working recognition engine could be plugged in and the wizard be used only to correct and confirm results. Finally, extensibility relates to the already discussed aspect of allowing for future extensions of a tool. Probable changes might involve the integration of, by then, working technology components or the addition of new input and output modalities. Also adaptation to support new hardware form factors should be possible.

After having explored requirements for general WOZ tool support we now move on to further deepen our understanding of the area through prototyping and tool development activities. Here it needs to be highlighted that, while the following description tries to be progressive, the actual tool construction happened as part of an iterative development process, which constantly switched between implementation, evaluation and analysis tasks.

4.3.3 Learning from Low-fidelity Prototyping

In order to obtain first hands-on experience with the task of the wizard, we started working on the implementation of a prototypical WOZ tool. To do so we elaborated on Scenario 4 (cf. Table 4.2) for which we had to support the simulation of an interaction between a German speaking customer and a system recommending appropriate Internet connection bundles. To

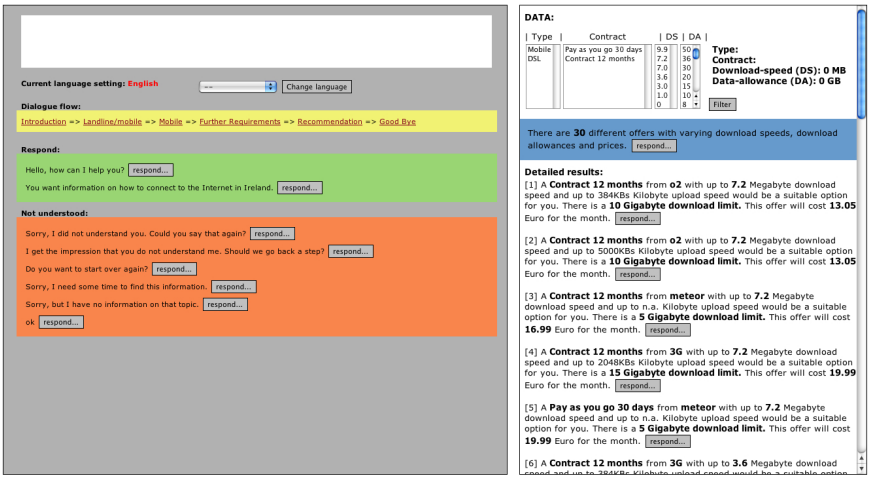

Figure 4.1 – An initial wizard interface.

do this, we first defined a preliminary dialogue and tested it for accuracy and completeness using a chat tool and simple copy and paste mechanisms. Based on this we then developed a wizard interface that supported this dialogue. PHP, HTML and CSS were used to create the interface that ran within an APACHE web-server and used a MYSQL database to store and retrieve dialogue utterances, domain and log data. To support interaction in the customer's native language, we further created three different sets of pre-translated dialogue utterances. The first set was translated by a German native speaker, the second via GOOGLE TRANSLATE[6] and for the third we used the SYSTRAN[7] machine translation engine.

Initial Interface Layout

The resulting interface of this initial prototype (cf. Figure 4.1) was split into two areas. The 'Dialogue flow' was shown on the left side of the screen, highlighted by a yellow background colour. The utterances to be used in a particular stage of the dialogue flow were represented in a green box in the middle of the screen under the label 'Respond'. During an experiment the wizard had to choose between these utterances to respond to a test participant and progress through the dialogue flow. When an utterance was chosen that would lead to the next stage in the dialogue flow, the display was updated automatically to show a new set of utterances appropriate for the new dialogue stage. The wizard could also switch between dialogue stages manually by clicking on the respective links under 'Dialogue flow'. Utterances aiming to help the wizard recover from misunderstandings were situated in an orange box in the 'Not understood' section of the screen.

For this specific scenario all utterances were displayed in English. Only the current utterance sent to a test participant, was in German, shown in a white box on top of this area. A drop-down menu allowed for defining the set of pre-translations to be used (i.e. native German,

[6] http://translate.google.com/ [Accessed: July 16th 2012]
[7] http://www.systran.co.uk/ [Accessed: September 19th 2012]

Google translate, or Systran). The display was updated whenever a new utterance was sent, i.e. at this stage there was no record of the history of interactions between the wizard and a test participant.

The right side of the interface was dedicated to the domain data, i.e. information on current offers of Internet connection bundles as suggested by Scenario 4 (cf. Table 4.2). Based on the flow of the dialogue, potential offers were filtered automatically so that the wizard had fewer options to choose from as the dialogue progresses. For situations in which a test participant changed her requirements, filters at the top of this area were available that permitted the wizard to change the display manually without going back in the dialogue.

Evaluation Method

In order to evaluate the usability of this first prototype we conducted a small-scale usability study with four different wizards. Following the scenario we asked the wizards to help a customer retrieve information. The customer was a German speaker whose task was to search for different ways of connecting to the Internet in Ireland. The system we intended to simulate was supposed to understand German spoken input and give back German text output, using MT in both cases. A wizard was given a one-page description (cf. Appendix B) of the task and was allowed to explore the wizard interface for about five minutes. Then, the wizard interacted with the customer. The goal of this test was to evaluate the usability of the wizard interface as well as the completeness and accuracy of the dialogue utterances it supports.

A formal usability test approach (Dumas and Redish [1999]) was chosen to analyse the prototype. Actions of the wizard were logged, and the wizard's screen captured. The speech of the customer was recorded as well as comments made by the wizard. The wizard's facial expressions were video recorded. A third person observing and taking notes was sitting next to the wizard. Since the wizard's task was thought to be cognitively demanding, thinking aloud was not actively requested. Yet, the wizard sometimes made spontaneous comments. A retrospective analysis of certain actions of the wizard was conducted after the task.

In total four people, two members of our research team and two external people, acted as a wizard. Different levels of familiarity with the tool were apparent. The four people included the person who designed the dialogue, and therefore knew about the dialogue structure, a colleague who was not directly involved in the experiment design but knew about its domain, and two other researchers outside our team who had no previous experience in this role and were not familiar with the test setting. Participants took part in the experiment voluntarily and were not paid.

The person acting as the customer was sitting in front of a 15 inch computer screen. He could use the mouse to start the test, and to check the English source of responses by clicking a button on his screen (see Figure 4.2). His main mode for interacting with the wizard, however, was through a microphone. The interface he was looking at was web-based and running in full

Figure 4.2 – The interface that was used to display response utterances to subjects playing the roles of customers. If the German translation was incomprehensible they could see the English source by clicking the button labelled 'Englischer Originaltext'.

screen mode within the FIREFOX[8] web-browser. A laptop computer connected to the screen was placed in a desk drawer. SKYPE[9] was used to transfer the speech input to the wizard.

The wizard sat in a different room together with an observer who took notes of her actions. A 13-inch laptop was used to run the FIREFOX web-browser that showed the wizard interface, and the SCREENFLOW[10] screen casting software, which recorded the wizard's behaviour, comments, screen and mouse actions as well as the customer's speech. In addition the wizard's face was captured using a camera integrated in the screen of the laptop. In order to better hear the customer's voice, headphones were used.

Looking at the results of this usability test we were able to identify several issues regarding the intuitiveness of the initial wizard interface as well as some challenges of the wizard task. Generally these findings can be subdivided into interface and dialogue insights, insights connected to this specific wizard task, and more generic insights that are possibly transferable to other WOZ set-ups.

Interface and Dialogue Insights

The first main issue observed was that most wizards did not understand why the wizard interface was separated into different areas, and therefore had difficulties switching their attention from one side of the screen to the other. In addition minor interface problems were identified. For example, several wizards complained that the wizard interface did not fully fit the screen and they therefore frequently had to scroll. Such interface alignment issues could impact on the performance of the wizard, given the real-time nature of the task.

Furthermore, wizards had problems controlling the dialogue and finding suitable utterances since they were not familiar with the set of utterances before the interaction started. Also, the representation of the dialogue flow and the current state were unclear. Furthermore, the feedback utterance *"OK"* which was intended as a 'filler' response was not found easily. In general, wizards complained that they could not find the right response or had a different phrase in mind when searching. For example, one wizard was searching for a response utterance like

[8] http://www.mozilla.org/en-US/firefox/new/ [Accessed: September 19th 2012]
[9] http://www.skype.com/intl/en/home [Accessed: September 19th 2012]
[10] http://www.telestream.net/screen-flow/ [Accessed: August 13th 2012]

"Are you satisfied?", but the utterance that was in the set to fulfil this purpose was *"Is there anything else I can do for you?"*. In addition, what seemed unclear is whether there was any structure within a particular state of the dialogue flow (i.e. whether all sentences had to be used before moving to the next state, in which order they were supposed to be used etc.).

Another apparent problem was how to deal with domain data, i.e. offers of Internet providers to be recommended to a customer. Per default Internet offers were sorted by 'PRICE' with the cheapest offer on top. Wizards were, however, missing the possibility to sort them by 'SPEED' or 'DOWNLOAD ALLOWANCE'. It was also observed that wizards forgot to press the filter button when they used the filter option for the first time, in which case chosen parameters were not applied.

Another problem concerned the use of domain-specific terminology. Some of the terms used for interface elements were felt to be ambiguous. For example, the meaning of 'Dialogue Flow' being the heading of an interface element was not clearly understood. Also abbreviations like 'DS' (= download speed) and 'DA' (= download allowance) were confusing. Further, it was not understood that 'DSL' means landline and that 'no contract' would be equivalent to a so-called 'pay-as-you-go' option. Another aspect regarding language was mentioned for the language drop-down field. Here it was not clear that changing the language would only change the language of the response utterances that are sent to a test participant and not the language of the wizard interface. One aspect that could have been problematic for wizards is that they needed to deal with two different languages at the same time. That is, while wizards were listening to the customer speaking in German, they had to choose responses that were displayed in English. After sending an utterance, however, it was displayed in the way it was sent to the customer (i.e. in machine translated German) in the text-box at the top of the wizard interface. Even though retrospectively we thought that this would be difficult for wizards to deal with, wizards did not actually perceive it as confusing. It turned out that they were actually not reading the translated text that was sent to the customer, they only used the appearance of the text as a sign that their utterance was sent.

Task Insights

In addition to interface and dialogue insights, the usability tests also highlighted some problems with the actual wizard task. Some of these issues related to the methodology (particularly, the training and instructions given to wizards), some to the wizard interface (support for maintaining the dialogue flow), and some to the specifics of the dialogue itself; thus illustrating the types of human-machine design issues that the WOZ technique is intended to uncover.

One particular problem that was observed, was that it was hard for wizards to use the confirmation sentences that are commonly used in interactive voice response systems. It was mentioned earlier that in-experienced subjects acting as a wizard need to be given guidance and training on how to behave, given the nature of the end-user task, and the goals of the experiment. The exploratory study described here clearly confirms this argument. As wizards

were neither customer care agents, nor specialised in mimicking a dialogue system, a more detailed test script would have been necessary. Alternatively one might argue that a well-designed interface would need to actively enforce confirmation behaviour. However, if such a strategy is to be employed the dialogue model must be checked for consistency. That is, confirmation utterances for all parts of the dialogue need to be available, which was not the case for this dialogue. If these utterances were missing, it was found that wizards tend to go on without confirming. In general it was noticed that if utterances were not properly situated they were either not used at all (e.g. *"These are the options you have:"*) or it took wizards a long time to find them.

Looking at the richness of the dialogue, wizards reported that more flexibility is needed when it comes to the 'further requirements' section. One wizard, for instance, pointed out that negotiations about the price were missing within the dialogue structure. Some other utterances, however, were perceived as inappropriate or ambiguous. For example, several wizards wondered why one would want to tell people how many offers were left, and also an utterance priming for download speed vs. price (i.e. *"Do you, for example, prefer a lower price to the download speed? Or is price not an issue?"*) was found to be confusing and hindered the dialogue flow. Finally it was pointed out that a test participant might not know the difference between mobile and landline/DSL Internet. Similarly the difference between a contract and a 'pay-as-you-go' option (as already highlighted in the interface insights) might not be obvious. It was therefore recommended, to add utterances that would give more information, based on which, the subject could make his/her decision. Even though these comments are very specific to this experimental setting they clearly demonstrate the benefits of using WOZ as an instrument for designing human-machine dialogues.

Generic Findings

This first round of usability tests highlighted several problems with our prototype. Some of which can be categorized as basic usability problems, which can be corrected relatively easily for a given scenario (e.g. terminology). Other, more generic problems, however, require further investigation. Following, we want to focus on the later category and discuss three problems we identified that have the potential, if solved, to influence generic WOZ tool support.

The most important problem we were able to identify is that of the wizard being able to follow and control the dialogue flow at any point. With our prototype it happened that wizards hardly knew where they were within the dialogue and hence had problems finding the right response utterances. Hence, additional explorations of how to improve the general layout of the wizard interface need to be conducted. The main issue that needs to be solved here is the amount of information that is displayed to a wizard at any given time during an experiment. Solutions would have to reduce information overload in order to reduce the cognitive load (e.g. only display relevant utterances to the wizard) but at the same time keep the overall structure of an experiment visible, so that a wizard does not get lost within a dialogue. In general subtle

ways of guiding the wizard throughout an experiment but at the same time keeping her in control, are desirable.

From a dialogue perspective our usability tests highlighted an interesting timing issue. That is, in some cases a wizard would use two subsequent utterances without waiting for a response from the test subject. It was noticed that sometimes when the wizard did not receive a response from the test subject she would send an *"I can't understand you"* utterance after 10 to 12 seconds as to check on the test subjects status. This timing seemed to be consistent among all participants and therefore shows that for a wizard interface it can be important to notify the wizard about the processing status of a test participant. However, it should be mentioned that the situation of not knowing a test participant's status of attention, more accurately reflects a real dialogue system. Giving the wizard too much information may lead to an unrealistic representation of a future system, which would further lead to a biased experiment outcome. It seems that a balance between the accurate representation of a future system's functionality and the support given to a wizard mimicking this functionality has to be found.

A final generic problem of our interface was the handling of filters. The aim of theses filters was to reduce the amount of data a wizard had to deal with. During the course of a dialogue they were selected automatically and only needed to be adjusted if the test participant changed requirements. From a wizard's point of view, however, this behaviour was not followed. It seemed that since a filter option was provided, wizards wanted to use it; they wanted to be in control. A solution for this problem may be a dialogue that is designed in a way so that no additional filtering is needed. Alternatively, these slots could be represented in a different way. In general, however, it seemed that any offered functionality that would divert the wizard from her direct interaction with a test subject, needs to be clear and very intuitive to use. Opaque interface behaviour would only confuse the wizard and influence her performance.

In the following section we report on how we used a simple sketching exercise to tackle one of the generic problems identified, namely, how to create a generic interface layout that supports the wizard in controlling the dialogue flow. Implementing the results of this exercise and testing their effectiveness is then part of the second stage of our research agenda, which will be the focus of the next chapter of this thesis.

4.3.4 Learning from Sketching

Our usability tests clearly showed that a wizard needs to be able to follow and control the dialogue flow. In trying to tackle this problem, we were not only interested in a solution that would help the one scenarios we built our initial prototype for, but we were also interested in searching for a generic interface concept that could be applied in a range of different settings (cf. Table 4.2). To do so we used paper prototypes and sketches as a means of quickly designing and comparing ideas.

Paper prototyping and sketching are easy and cheap methods to gain early feedback on design. They help to develop ideas and discuss different possible solutions. Through sketches,

designers act like architects, when they find new relations and features whilst viewing their own sketch (Suwa and Tversky [1996]). Schön [1984] argues that the designer reflects-in-action. Thereby it is not just the actual drawing act, but the way in which the situation talks back to the designer and lets her change initial thoughts. This intrinsic process of drawing and at the same time communicating with the sketch, defines the designer's understanding of the situation. In our case we used sketches to compare different layouts of possible generic wizard interfaces. We created paper-based interfaces for each of the four scenarios defined earlier (cf. Table 4.2). Going through this process we were able to identify three main layout concepts that seemed to be applicable. These concepts are discussed in the following sections.

Dialogue

All of the layouts we designed had in common that the main area of the screen was dedicated to the running dialogue. In our understanding, the wizard's task mainly consists of choosing from a predefined set of dialogue utterances. Supporting this selection process would significantly simplify the role of the wizard. In order to do so our goal was to decrease the number of possible utterances to choose from. Comparing different designs we found that subdividing the dialogue into smaller steps seems a valid solution for this problem; one that could be applied to all the scenarios we designed for. Using a tab structure we were able to split the dialogue into different stages but at the same time we maintained the overall structure, which may help the wizard to keep track of the dialogue progress and also allows for quickly switching between different dialogue stages (cf. Figure 4.3).

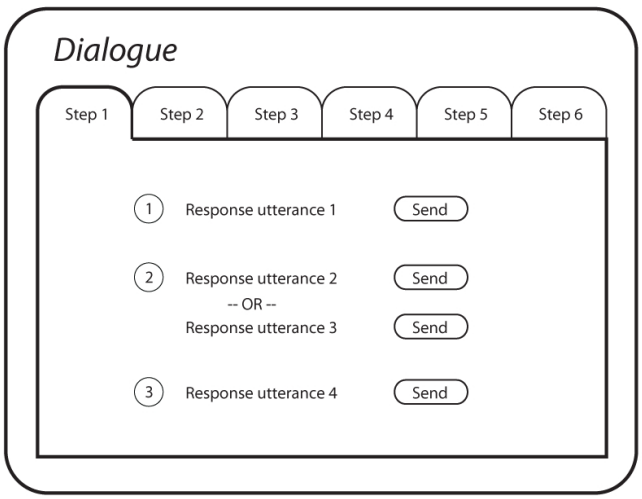

Figure 4.3 – A tab structure representing the dialogue and its stages may help the wizard to choose from a predefined set of response utterances.

Recovery

The second important concept we identified as being relevant for an intuitive to use wizard interface, is the support of the common problem of error recovery. Since there is a probability of misunderstanding in any kind of dialogue situation, a wizard needs to be offered a way of recovery. Recovery might be gained through re-sending the last utterance or by using predefined repair utterances that would invoke a test participant to repeat or specify her last statement (e.g. *"Sorry, I did not understand you. Could you please repeat."*). The problem of predefined repair utterances is, however, that they do not occur at a certain point in time. Rather a wizard needs to be able to use them spontaneously within the running dialogue. As a generic solution for this, we propose a separate area for repair utterances that would be constantly visible to the wizard independent of the dialogue stage (cf. Figure 4.4).

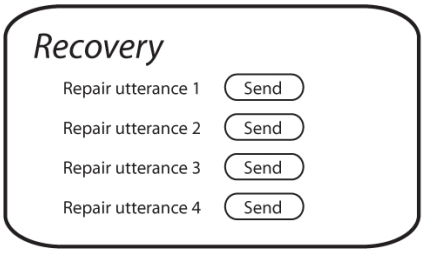

Figure 4.4 – A seperate area for repair utterances should permit the wizard to act spontaneously.

History

Finally, the third aspect we found to be helpful for showing the progress of the dialogue flow is to introduce a history function. The idea here is that a wizard might want to see what utterances she has already used and possibly what a test participant has replied. Also, incoming and outgoing utterances should be easily distinguishable. That is, a wizard should be able to see what came from a participant and what she herself, acting as a wizard, has sent. As one way of tackling this problem we propose the use of symbols and different colours to differentiate between wizard's and test participant's utterances (see Figure 4.5).

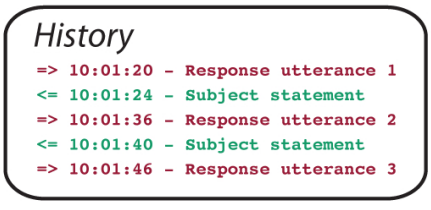

Figure 4.5 – A history using different symbols and colours to differentiate between wizard and subject statements might support the wizard's task.

4.4 Summary

Having completed this initial phase of building a prototype and evaluating its usability, and having reflected upon the results of these tests with the described sketching exercise, leading to concrete concepts for a generic wizard interface, we conclude this primary stage of our research program. Summing up, we started our exploration route with an extensive discussion of existing tools and what we can learn from them. Then we moved on to identifying global recommendations for running WOZ experiments, and finally we created an initial prototype of a WOZ tool supporting a single test scenario. The tool was evaluated using a small-scale usability study and results were reflected by a set of sketches aiming for the design of a better wizard interface.

All this serves as the basis for the next stage in our research agenda which looks more closely at generic WOZ tool support. To do so, the next chapter of this thesis analyses the WOZ method from two different perspectives. First, we re-examine the literature and look more closely at the user (i.e. wizard) side of the experimentation. As part of this, we report on the results of an interview study that aimed at identifying some of the wizard challenges that are not usually found in publications. From there we move on to the system perspective of WOZ, and look more closely at technological aspects of tool support. The goal of this whole process is to move away from the here described prototypical WOZ tool for a single scenario, to a generic platform for WOZ experimentation that is employable for a variety of different experiments.

Chapter 5

Generic Support for Wizard of Oz Experimentation

"Get an oil-can and oil my joints. They are rusted so badly that I cannot move them at all; (...)"
– The Tin Man

This chapter analyses generic support for WOZ. Doing so, we look at two different perspectives of WOZ experimentation. We start at the user perspective that analyses people's interests in employing the method (cf. 5.1), and then move on to the system perspective which aims at defining wizard and technology dependent requirements (cf. 5.2). The insights gained from this analysis forms the necessary foundation for offering generic tool support.

5.1 User Requirements

By looking at the user perspective we are interested in the distinct goals and interests researchers follow when running WOZ experiments. Reflecting on what we had seen in the literature and other reported examples, we start with a categorization of potential users and their motives for employing the method. In a next step, we expand on this theoretical analysis and discuss the results of an interview study in which we contacted researchers from both academia and industry and asked them about their experience with WOZ.

5.1.1 Potentially Interested Parties

As can be seen from the variety of application areas described earlier, many different types of scenarios can be explored using the WOZ technique. However, it takes some time and experience to obtain the skills needed for designing and running WOZ experiments, and so it is worth considering who our users might be. Looking at the literature and professionals in the field, we can find a dedicated set of parties that potentially show an interest in employing the

method. In the following section we look at them in more detail and highlight their distinct motivation for using WOZ.

Users of Wizard of Oz

From a design perspective, students studying Human-Computer Interaction (HCI) and Interaction Design (ID) will generally be introduced to WOZ, yet only a small proportion of these will actually experience the method when compared to exercises based on the use of paper prototypes. One reason for this lack of practical usage might be that in order to be applicable in a HCI teaching context, any approach would have to have a low logistical and technical overhead to enable students to quickly design and carry out evaluations.

Experienced interaction designers are another obvious user group. Our own interviews with developers of systems based on Interactive Voice Response (IVR) suggest that WOZ is used within product development. However, the opinion was expressed that the limited time-scale typically available for interaction design activities, especially within smaller projects, often impedes or limits the application of the method. Therefore, it seems that more exploratory uses, such as those represented by the HCI research literature, may be a more sensible starting point, when it comes to understanding users coming from the area of voice-interface design and development.

Another distinct category of users are people working in computational linguistics as they usually have a strong interest in gathering language corpora. Such corpora are vital resources both for scholarly work and for the development of LTCs. For example, the WOZ method can be used to collect an initial language corpus upon which LTCs are trained and improved (Lamel [1998]). Collecting context-specific and language-specific corpora helps to expand the reach of existing technologies. The desired output from an experiment in this setting is typically the input supplied by the non-wizard user, whether it is typed text, speech, or multi-modal input (for example speech or gestures).

Finally, people involved in the development of these technology components may also be interested in WOZ, as it permits them to evaluate the performance of their products in a real-world setting, for example within a specific application context. Using WOZ, those technologies can be tested in more realistic, task-focussed evaluations (Note: Many existing LTC evaluations are otherwise based on context-free standardised benchmarks), without the need to construct a fully working system around them (which is usually not the focus of work for component developers). The benchmark of such an experiment might be the word error rate for the recognition of application-specific utterances, rather than the design of the dialogue itself or other aspects of the task performance.

Resulting Fields of Application

Looking at these different user groups and their interest in employing the method, one can find three main application areas for WOZ experimentation: Firstly, interaction design, where the

User	Interaction Design	Component Evaluation	Corpus Gathering
ID & HCI Students	x	-	-
ID & HCI Professionals	x	x	-
NLP Researchers	-	x	x
Developers	-	x	x

Table 5.1 – Potential users of WOZ and their interests in the method.

flexibility to explore a range of different types of scenarios is key, but which might make use of LTCs as part of delivering an authentic experience. Secondly, component evaluation, testing the quality of existing technology, which requires that (at least partially) working components be integrated, and finally, corpus gathering, which may or may not require the integration of working components. Table 5.1 summarises the identified users for WOZ and highlights which of those application fields are potentially interesting for them.

5.1.2 Talking to Experts

In order to further increase our understanding of the different user groups and their distinct usage scenarios for WOZ, we conducted an interview study with researchers from industry and academia. In total five professional voice interface designers and 25 academics who had recently published relevant work in the area (i.e. mostly within the last five years) were approached via email and asked for a phone interview. Positive responses from three of the designers and 17 researchers (seven of them working in NLP, five in HCI, and five in the area of multi-modal interaction) led to a total of 20 interviews, each of which lasted between 17 and 30 minutes. While all of the interviewees were actively involved in at least one WOZ study, 13 of them indicated that they had used WOZ in a variety of experiments. Interviews were semi-structured (cf. Appendix A), and interviewees were asked about their motivation for using the method, the challenges they had to overcome when doing so, and the tools they had employed. The recordings were fully transcribed and analysed, going through an open coding process first, and then focused on some of the core aspects of WOZ prototyping. While this analysis method has its roots in Grounded Theory (Glaser and Strauss [1967]), we would like to highlight that it was not our goal to build such a holistic theory from data. Similar to Muller and Kogan [2010], we used this approach as a guideline for conducting a more structured analysis of the transcribed interviews. The results are discussed in the following section, and summarised in Table 5.2.

Reason for using WOZ

Exploring new design ideas before they are implemented was cited as a reason for using WOZ by the majority (13) of interviewees. A typical statement to that effect was S10: "*So we had this idea of building this multi-lingual translation system, but we were not very sure, so we wanted to do a WOZ simulation in which we placed a tri-lingual person in the middle.*". The method

Reason for using WOZ	Exploring new design ideas
	Collecting an initial dataset
	Comparing specific design solutions
	Evaluating technology
Challenges to overcome	Delays caused by aspects of the wizard task
	Make participants believe that they are interacting with a system
	Simulate consistent system behaviour
	Simulate erroneous or suboptimal system performance
	Recruit participants
	High experiment costs
	Ethical issue of deceiving participants
Methodological aspects	Subject to improvisation
	Simulation as limitation
	Elasticity of results
Tools employed	Proprietary tools supporting a very specific experiment setting

Table 5.2 – Summarised results of 20 phone interviews conducted with interviewees from industry and academia who have experience with WOZ experimentation

was found to be especially useful as a means of obtaining early feedback on a proposed design direction (S03: "*So whenever you already have, you know, a draft design and you want to show it to customers, or you just want to show it to an initial set of users in order to design it in the right direction.*") or a low-fidelity proof of concept study (S10: "*Yeah this is what, this was just a proof of concept*"). In addition interviewees, especially from the NLP domain, stressed the value of WOZ for collecting data (stated by 8 interviewees) (S12: "*You generate data without having a dialogue system, and you create from this small dataset, you create simulated environments, and in that simulated environments you can train dialogue strategies.*") in order to explore dialogue strategies (S09: "*Yes the research was focused on the dialogue between the participants, the communication, what they would say.*"). Two researchers specifically highlighted its qualities for comparing specific design solutions (S14: "*The biggest point is to save time in developing the actual technology, to make it possible to test out alternatives without over-committing to one of them early on.*"), in which case the possibility for quickly putting together different design proposals is perceived as a key property of the method (S03: "*Like if you were thinking about a A or B design you can quickly put both together and then ride through you know half a dozen people and see which of the two designs seems to work better.*"). Finally, two others mentioned that they had used it to evaluate some of their technology components (S15: "*We performed WOZ experiments three times to evaluate our dialogue system.*").

Challenges to overcome

In terms of problems interviewees were facing, it seems that delays coming from the wizard constitute a main challenge, specifically mentioned by 9 of the 20 interviewees (S07: "*The delay seemed to be the biggest problem.*"). These delays were attributed to a number of different aspects of the wizard task, such as the underestimated amount of time required for flawless typing (S09: "*All they needed to do is type in a message and type enter.*"), information overload (S04: "*Moreover the problem was that theoretically I should only look at the non-verbal behaviour and the acoustic information.*") or may be rooted in a lack of wizard training (S19: "*No, no specific training at all, we made some pilots.*"). Another particular challenge was found in 'hiding the wizard' (mentioned by 7 interviewees) so that people would believe that they are in fact interacting with a piece of technology rather than a human being (S11: "*You have to make sure that users really don't feel that there is someone staying in the other room.*"). This requirement for realism of the simulated functionality is not restricted to giving the user the impression that they are interacting with a real system but it also involves reflecting the complexity of the underlying technology in a way that conforms to the designer's expectations (S20: "*The more complicated the technology the bigger the challenge of making the simulation reflect what might really happen.*"). Also in connection with this issue, some interviewees highlighted the challenge of maintaining consistent wizard behaviour (mentioned by 7 interviewees), and remarked on how important it is to simulate similar interactions (S16: "*I mean it has to be consistent. It has to give the same answer all the time.*"). This seems especially true for the quality and validity of the responses given to users, as variable wizard actions can lead to confusion for a test participant (S18: "*If you are not consistent then the user will be very confused by what they are seeing and they might give you feedback on something, on, well they will give you feedback and you, it will be hard for you to know whether or not they are responding or I should say which version of the interface they are responding to.*").

Finally, two interviewees mentioned that the simulation of errors or suboptimal system performance helps in the testing of error-recovery routines and in conveying realistic system behaviour (S03: "*So you start becoming better at mimicking a real system so from time to time you would throw in an error or a misrecognition or something that would basically make the participant to try to recover .*"). Consequently, system-driven probabilistic error routines may be seen as a way to reduce a wizard's workload while at the same time increasing the validity of the simulated system (S20: "*So I think that idea of we are going to just have the wizard do a very well defined task, so that they can focus on just doing that right and let the system simulate the errors and simulate the delays and simulate all that stuff.*"). However, support for this sort of functionality is not generally available.

Other challenges that were mentioned include the recruitment of participants (S09: "*And it took quite a long time to find participants, because it was very important that the participants, that both participants come.*"), the high cost of experimentation (S15: "*No special challenges, but, WoZ experiments cost too much. It is a big problem for us.*"), as well as the ethical issue

of deceiving participants (S19: "*Yeah. You have to, you have to handle that with care. So there is some ethical, ethical considerations also to be made.*"). While recruitment is a common experimental problem, it can be argued that web-based approaches (such as those employed in remote usability testing) have the advantage of widening the potential pool of participants. Modern tools should therefore be based on this sort of web technology. With regard to cost, reducing the amount of technical effort required for constructing experiments, and reducing the logistical overhead of running and analysing experiments would be separate dimensions. The former highlighting a challenge of WOZ experimentation, the latter being applicable to all types of user studies. Finally, ethics is an important issue for WOZ as a methodology, as many experiments will involve deceiving participants. Hence, ensuring participants are debriefed appropriately is a vital aspect of the method, without which a scientific application may not be valid.

Methodological Aspects

In addition to distinct challenges the interviews also provided some insight into methodological aspects of WOZ experimentation. We could see that interviewees generally regard the method as low-fidelity in terms of the work put into creating the product to be tested. Due to the overhead that comes with the creation of the simulation environment as well as the involvement of real users, however, the experiment costs are often perceived as disproportional when compared with similar exploration methods such as sketching and wire-framing. Hence, in order to decrease those costs, WOZ experiments are usually subject to improvisation and ongoing adaptation, which can cause inconsistent results. In general researchers highlighted that a common pitfall of WOZ is an insufficiently defined research question, a problem that is often rooted in this potential variability of the experiment (S13: "*It was really difficult to pin one question down and information getting back that answers one specific research question* "). Also, since it is unrealistic to control for all the possible actions of the human wizard when reacting to often unique contextual circumstances, it is easy to be drawn away from a defined path of exploration (S19: "*So, it is easy to just continue and not keep on the small red thread ... to keep on track really*"). In trying to get as close to the real thing as possible, researchers see in the simulation the biggest limitation of the method (S17: "*I mean, it, it is a simulation, that is the main, the main limitation.*"). While on the one hand it permits them to quickly compare different possible solutions without technical adjustments, on the other hand it lacks the strict accuracy an experimental setting usually provides (S3: "*There are other factors in play that basically don't get replicated because it is too hard*"). As such, WOZ is often seen as a low-fidelity exploration method with a high price-tag whose results are very elastic and hence require a great amount of analysis and interpretation (S14: "*Well, that is interesting, because we did not really see any trends necessarily in the logging data mostly because there is a lot of noise. It's hard to really know what the trends would have been.*").

Tools employed

When asked about the tools they used to conduct their WOZ experiments, all of the interviewees stated that they had employed self-developed programs. Even though they mostly found that the implementation time for those solutions was feasible, the effort often appeared disproportionately high given that WOZ is often regarded as a low-fidelity prototyping method (S12: "*I think to develop a stable version of that, took us one person month at least.*"). Likewise, several interviewees expressed an interest in a more general WOZ prototyping tool, that could be adapted to their research interest in a flexible manner (S09 "*Yeah, yeah it is very cool this idea, if it is researchers like myself ... that we can just manipulate, make it our own ... fit it to our own research ... and not having to develop a system on our own every time*").

Similar demands have emerged from within our own research environment. As part of an extensive research program on development and application of language technologies, we were increasingly facing the problem of how to test technology components with real users, without the overhead of creating an application environment for each case. The analysis of the literature earlier, complemented by the interview study presented here, supports the conclusion that there is a need for generic tool support which has not been addressed or explored sufficiently, and that such tool support should pay particular attention to experimentation involving real or simulated language technology components. The following section therefore aims at identifying a set of additional requirements this sort of tool support would need to meet.

5.2 System Requirements

While the previous analysis was focusing on the users of WOZ and their motivation for employing the method, this section takes this insight and turns it into relevant system requirements. We approach this task by following two different pathways. On the one hand, we look more closely at the role of the wizard and the various needs the different aspects of this task create. On the other hand, we focus on the technology at hand, what possibilities it offers as well as what restrictions it bears. Looking at these two rather generic angles, the goal is to expand upon those requirements which only refer to very concrete experiment scenarios.

5.2.1 Role-dependent Requirements Gathering

In order to methodologically define requirements for a generic WOZ tool, we first focus on the wizard task and how it can be supported. We start by summarising the aspects that were mentioned during our interview study, and then go on and look more closely at the various tasks a wizard could potentially accomplish, and how they might influence the set of requirements that need to be supported. Doing so, we discuss the different wizard roles which, if effectively supported, can also significantly influence both the architecture of an entire tool as well as the layout of a supportive wizard interface.

General Requirements for Wizard of Oz Tool Support

Having started with the different groups of users potentially interested in employing the WOZ method and their distinct application scenarios, we now move on and look more closely at the requirements for tool support, what existing tools offer and where support should be improved. We can generally categorize requirements for WOZ experiments into functional and non-functional factors (summarised in Table 5.3).

Functional requirements	Support both structured and flexible interactions
	Support component integration with human intervention
	Support experimental data capture and export for analysis
Non-functional requirements	Reduce overhead in experiment construction and installation
	Reduce cognitive burden on Wizard during experimentation

Table 5.3 – General requirements for supporting WOZ experimentation.

Firstly, from a functional point of view a WOZ tool needs to provide support for running tightly controlled experiments as well as more exploratory studies. Even within tightly controlled experiments some flexibility for dealing with the unexpected may be useful. Features that would allow for more structured interactions include possibilities for creating, selecting and grouping responses, and the availability of filters and similar aids that help a wizard retrieve information.

Secondly, when we consider scenarios where different existing technologies are combined, there is a potential for this interconnection to increase failure. A representative example would be the analysis of Speech-to-Speech translation where the output of an ASR component is used as input for machine translation (MT) and its output then again fed to the Text-to-Speech synthesiser (TTS). Whereas humans might very well tolerate small mistakes coming from single components, technology is less forgiving. That is, while humans using contextual information might handle small speech recognition errors, they can lead to problems when forwarded to a translation service. Supporting the function of a human in the loop who acts as an enhancement rather than a replacement for the technology can hence be seen as crucial requirement for comprehensive WOZ support, as this allows for exploring these kind of dependency problems in more detail.

Thirdly, tracking mechanisms and data exports need to be available in order to analyse user behaviour. In addition, being able to gather data on wizard task performance and how it changes depending on the experiment setting, and over the course of an experiment, can be seen as a feature that could make this prototyping method more robust. Existing problems with this sort of data logging were explicitly mentioned in our interview study (S12: "*Actually the logging is another challenge which ... what happened to us is that we lost data*"), and therefore clearly highlight its importance and the pending need for improvement.

On the other hand, from a non-functional point of view we see that currently the require-

ments for installing multiple software components and configuring the network (to support the connection between the user and the Wizard) quickly increases the amount of time and resources needed for running WOZ, and therefore diminishes its value as a low-fidelity prototyping method. A further complication is that technology components are often platform-specific. Hence, reducing this cost of setting up, designing and running experiments would make the method more attractive and accessible to researchers and designers of all fields.

Finally, another non-functional aspect that currently poses significant challenges for WOZ experiments, is the workload of the human wizard while running evaluations (Salber and Coutaz [1993a]). Correct timing, consistency and general machine-like behaviour directly influences the quality and representativeness of an experiment, and therefore, if not controlled, can influence obtained evaluation results. Support could come from visible instructions and reminders that might help the wizard achieve consistency, or from highly customizable wizard interfaces. Yet more research on those elements is needed in order to evaluate their effectiveness. Furthermore, if we allow for changing the wizard's role from replacing to enhancing technology, as outlined earlier, additional support for that role might be required.

Next we look more closely at the task of the wizard and its different roles and analyse how they may change the requirements for necessary tool support.

The Task of the Wizard and its Design Space

While previously we saw that DM tools and pure WOZ tools both incorporate useful features (cf. 4.1), neither type of tool provides a full range of support for the use of WOZ as a low-fidelity prototyping method. In order to build more appropriate instruments we need to better understand the task of the wizard as well as the design space for WOZ.

Looking at these aspects in more detail we took the example of language technologies and started our exploration with the consideration of available components and how they might be integrated. Researchers have reported on WOZ evaluations in the area of Human-Machine dialogue as well as computer supported Human-Human dialogue. The latter is especially relevant to machine translation where technology aims to build a bridge between people who do not share a common language (Bederson et al. [2010]). From a more component-based view, WOZ has been used to simulate ASR, MT, Natural Language Understanding, and Natural Language Generation as well as TTS. Only rarely has it been used to enhance existing technology. Examples include the adaption of a storyline (Dow et al. [2010]), the mimicking of social behaviour (Deruyter et al. [2005]), or the annotation of language (Janarthanam and Lemon [2009]). In general, however, the task of the wizard is to replace a single component or a combination of several components. In most cases the language understanding as well as the output generation part are simulated, supplemented by one or more other components. Therefore we might generalise that standard dialogue management (i.e. the integration of Natural Language Understanding (NLU) and Natural Language Generation (NLG)), is the main task a wizard needs to deal with. From a Natural Language Processing (NLP) point of view, dialogue

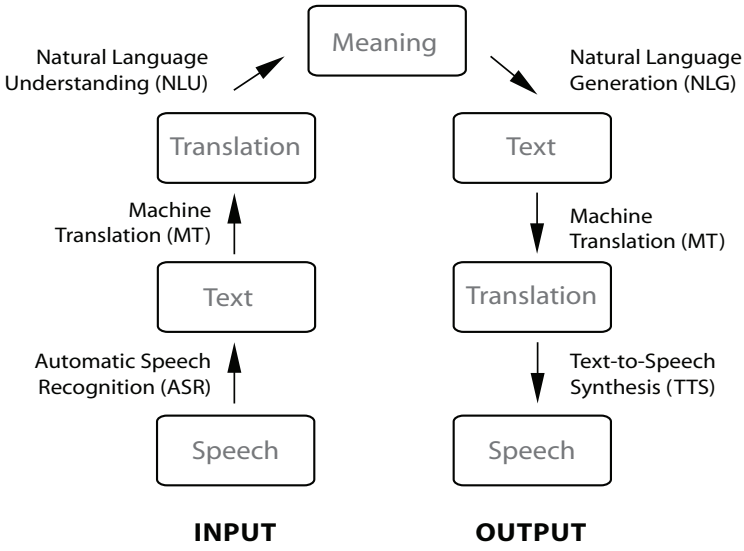

Figure 5.1 – The Interaction Pipeline of Language Technology.

management constitutes the center piece of a pipeline architecture that starts on the input side with Automatic Speech Recognition (ASR) and ends at the output side with Text-to-Speech Synthesis (TTS). In between we might further find Machine Translation (MT) on both sides (see Fig. 5.1). In a non-speech scenario ASR and TTS can be replaced by other, text-based, input and output modalities, which eventually leads to a total of 16 different task settings a wizard possibly has to deal with when running WOZ experiments (see Table 5.4).

In the most basic form the interaction on both sides is based on text (Case 1) and the wizard's task is limited to managing the dialogue. An application scenario for this pure form of WOZ can be found in prototyping a chatbot or a natural language user interface (e.g. Kelley [1984]). Adding an ASR component on the input side changes the task of the wizard from interpreting text input to interpreting speech (Case 2). Even though the difference here seems rather small it can lead to an increase in cognitive load for the wizard as spoken text cannot be revisited later on, which further might lead to performance problems especially when dialogue partners use long sentences. An example for this form of interaction can be found in prototyping dictation software (e.g. Gould et al. [1983]). The complexity of the wizard task increases even more in cases where an additional translation component is to be simulated. The simulation here could happen from speech input, which needs to be first processed and then translated (Case 3). In this case the task of the wizard can be compared to somebody simultaneously translating from one language into another (e.g. Stüker et al. [2006]). Alternatively, the recognition could be handled by a working ASR component or excluded for text-based input, which reduces the task of the wizard from translating from speech to translating from text, either on the input (Case 4) or on the output side (Case 5). Application examples for this type of task include a multi-lingual chat (e.g. Chen and Raman [2008]) or a Text-to-Text translation

system (e.g. Bederson et al. [2010]). Looking at the output side, we see a similar combination of possible components. For example prototyping a Text-to-Speech function of a booking system would require a wizard to simulate TTS from input text (Case 6) (e.g. Yang et al. [2000]), and in cases where the system needs to be multi-lingual an additional translation task can be found either on the input (Case 7) or on the output side (Case 8) (e.g. Gould et al. [1987]). The highest degree of complexity exists in situations where the wizard task comprises the complete interaction pipeline as highlighted in Case 9. Even though this is possible, it seems unlikely that a WOZ setting would require MT on both the input as well as the output side. For the same reason Cases 10, 11 and 12 seem less plausible. More realistic, however, would be the case of simulating a Speech-to-Speech translation system in which MT is used only on one side of the pipeline (Cases 13 and 14) (e.g. Krause [1996]; Kikui et al. [2003]). Taking away the multi-lingual aspect, simulating an IVR system would reduce the wizard's task to understanding speech input and producing appropriate speech output, either directly (perhaps using some sort of distortion device) or indirectly by choosing from a set of pre-recorded utterances (Case 15). Application areas for this sort of prototype include in-car navigation (e.g. Geutner et al. [2002]) as well as transactions such as booking tickets (e.g. Lamel [1998]; Karpov et al. [2008]). Finally, multi-lingual information retrieval using speech (Case 16) would require the wizard to first process a spoken request in one language and then provide appropriate information from multi-lingual sources (e.g. Schneider et al. [2010]).

The configurations detailed above provide a broad coverage of WOZ scenarios involving LTCs. However, we note that using WOZ for simulating multi-modal interaction dramatically increases its application area and at the same time places even higher demands on the wizard (Salber and Coutaz [1993b]). Here the aspect of processing information coming from different input channels and aligning the respective output has been the focus of several recent research attempts (e.g. Melichar and Cenek [2006]; Lee and Billinghurst [2008]; Serrano and Nigay [2010]).

Role-dependent Interface Changes

As discussed earlier, we do see the main usage for WOZ in creating low-fidelity prototypes of applications using LTCs. In those prototypes, the function of the wizard is usually to simulate a not yet existing component. If we go back to the above discussed roles, however, a wizard might also be employed to correct the output of a component or alternatively choose from various output streams (e.g. different MT engines) having different quality levels. Changing this role has also an effect on the wizard interface.

In order to explore those influences, we go back to our earlier sketches (cf. Section 4.3.4) and look at how the use of certain LTCs as well as changing roles of the wizard might change the layout of our main dialogue interface. If ASR was used, for example, the wizard interface would need to present the recognised text to the wizard. A text field situated on top of the dialogue utterances could be employed for that. The use of MT, on the other hand, may, depending

	Input		Processing			Output		
Case	Text	ASR	MT	DM	MT	TTS	Text	Example
1	x	-	-	x	-	-	x	Natural-Language Interfaces
2	-	x	-	x	-	-	x	Speech Recognition
3	-	x	x	x	-	-	x	Text-based Feedback
4	x	-	x	x	-	-	x	Text-to-Text Translation
5	x	-	-	x	x	-	x	Text-to-Text Translation
6	x	-	-	x	-	x	-	Speech-output
7	x	-	-	x	x	x	-	Multi-lingual Text-to-Speech
8	x	-	x	x	-	x	-	Multi-lingual Text-to-Speech
9	-	x	x	x	x	x	-	Less plausible
10	x	-	x	x	x	-	x	Less plausible
11	-	x	x	x	x	-	x	Less plausible
12	x	-	x	x	x	x	-	Less plausible
13	-	x	x	x	-	x	-	Speech-to-Speech Translation
14	-	x	-	x	x	x	-	Speech-to-Speech Translation
15	-	x	-	x	-	x	-	In-Car Voice Control
16	-	x	-	x	x	-	x	Multi-lingual Inf. Retrieval

Table 5.4 – Design Space for WOZ studies with LTCs and associated application examples. For the input stage we find text for cases where user input comes from a physical input modality such as a keyboard or a touchscreen and ASR where the user would speak to the system. For the processing stage we find, depending on the tested scenario, MT either before, after or on both sides of the DM. Finally, at the output stage we find either spoken output provided by a TTS or text-based output in case a visual interaction modality is simulated.

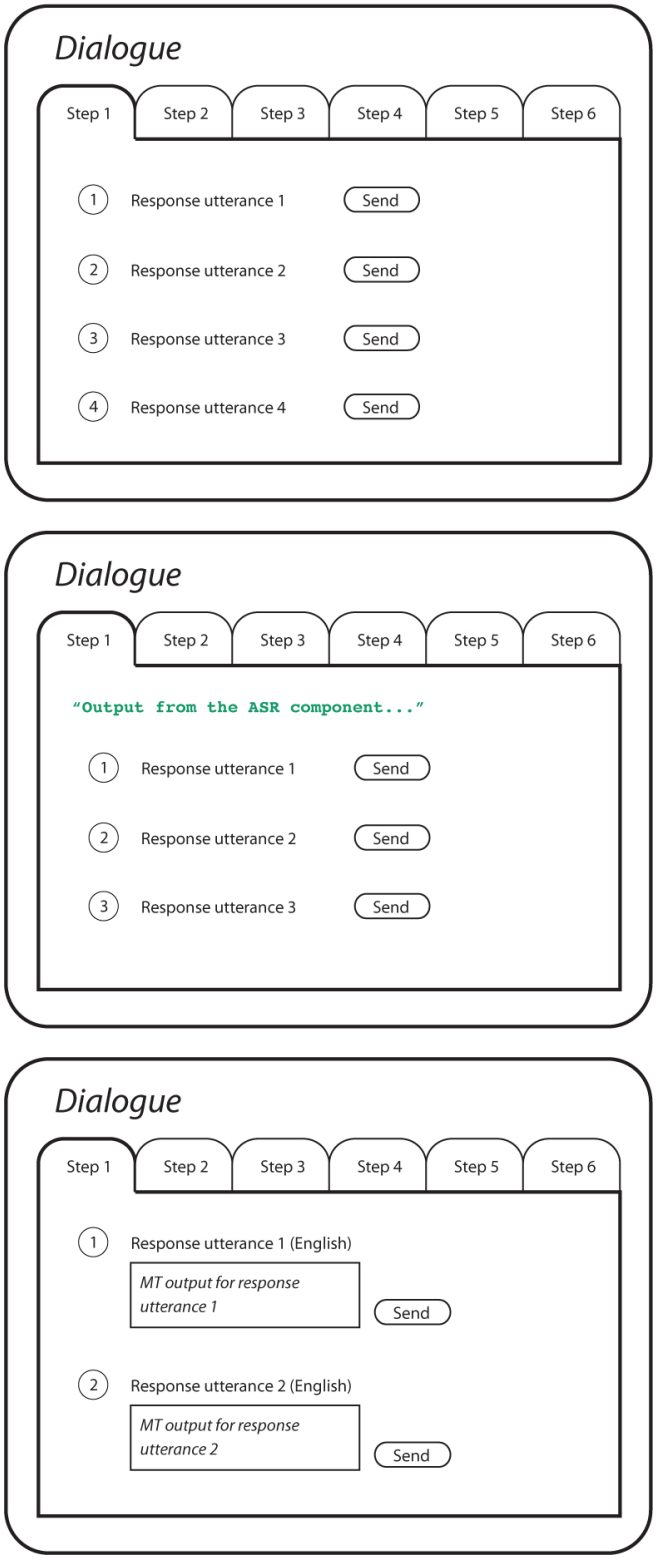

Figure 5.2 – Using different LTCs changes the layout of the wizard interface.

on the experiment setting, require a wizard to be able to choose from a set of utterances produced by different MT engines, or even permit her to correct them. Editable text boxes holding

on-the-fly MT results could be a solution here. If TTS is part of a system to be tested there could be a component integrated that would take text (e.g. pre-defined utterances or free text) as input and 'speak' it to a test subject. Alternatively a wizard might need to choose from a set of pre-recorded utterances to be played or even read out the utterances using some kind of distortion mechanism. In all cases different interface elements would be needed. Pre-recorded utterances could be started by the press of a button. A working TTS module, on the other hand could accept any kind of text input and in the case where the wizards voice mimics the system output, the text could be properly displayed to the wizard so that it would be easy for her to read.

Looking at those aspects we can expand our generic interface layout for the different uses of LTCs. We keep the general dialogue concept identified earlier but change certain interface elements within it. The outcome of this additional round of sketching is shown in Figure 5.2, where two different versions of an interface using a certain LTC are compared to an interface where no LTC is used. In the first version, no LTC is employed and so a wizard would simply listen to what a test participant says and then choose from a selection of possible responses. In the second version, ASR is activated for which its result is displayed to the wizard who then again chooses from a set of possible responses. In the last version we thought of MT being used to support a multilingual experiment setting. Here a bilingual wizard would listen to what for example a German-speaking subject is saying. Response utterances that are only available in English would then be translated at runtime via an integrated MT component. The interface would, however, enable the wizard to post-edit translations and correct possible mistakes before sending them to a test participant.

5.2.2 Technology-dependent Requirements Gathering

After having analysed user- and role-depended requirements, we also need to think about the technological aspects of WOZ support. To do so we first look at technologies used in existing tools and then discuss a general tool architecture that aims to provide the necessary flexibility to cover the various uses cases and WOZ set-ups presented in earlier sections, as well as supports the different roles a wizard might need to perform when running an experiment (cf. 5.4). Furthermore, we discuss how this type of architecture may be supported by current (and future) technologies and services.

Technology Platforms of Existing Tools

If we look at some of the existing WOZ tools we find a variety of different technology platforms that have been used to create wizard interfaces. From a typological perspective, we can generally differentiate between two kinds of applications. On the one hand, we find a small number of tools such as SUEDE and the CSLU toolkit that work as stand-alone applications where the simulation is achieved by a single computer. This type of application does, however, provide only limited interaction options for a test participant, for which experiments are usually

based on pure speech input. On the other hand, we find network-based client/server architectures where separate interfaces for both the wizard as well as the test participant are available. In addition to different tool typologies developers employed various technology platforms (i.e. programming languages) in order to build the relevant interfaces. While for stand-alone applications the TCL/TK platform[1] or JAVA[2] seem to be the favourite choices for networked solutions, we also find VISUAL C#[3] and ADOBE DIRECTOR[4] solutions. Table 5.5 shows a variety of different WOZ tools and the the technology platform that was used to implement them.

Tool	Type	Technology Platform	Reference
CSLU Toolkit	Standalone	Tcl/Tk	Sutton et al. [1998]
R. B.'s WOZ tool	Standalone	Tcl/Tk	Online[5]
NEIMO	Client/Server	Appletalk	Salber and Coutaz [1993a]
Polonius	Client/Server	ROS Framework	Lu et al. [2011]
SUEDE	Standalone	Java	Klemmer et al. [2000]
MDWOZ	Client/Server	Java	Munteanu and Boldea [2000]
WOZ Pro	Client/Server	Java	Hundhausen et al. [2007]
BrickRoad	Client/Server	Java	Liu and Li [2007]
Topiary	Client/Server	Java	Li et al. [2004]
EPFL Dial. Pl.	Client/Server	Java	Rajman et al. [2006]
SketchWizard	Client/Server	Visual C#	Davis et al. [2007]
QuickWoZ	Client/Server	Visual C#	Smeddinck et al. [2010]
Domer	Client/Server	Visual C#	Villano et al. [2011]
Ozlab	Client/Server	Adobe Director	Pettersson and Siponen [2002]
Dart	Client/Server	Adobe Director	MacIntyre et al. [2004]
Olympus	Client/Server	Web Technologies	Bohus et al. [2007]
Jaspis	Client/Server	Web Technologies	Turunen and Hakulinen [2000]

Table 5.5 – Existing WOZ Tools and the technology platform that was used to implement them.

As can be seen from the table, JAVA and VISUAL C# seem to be the most popular platforms for implementation whereas ADOBE DIRECTOR is predominantly used for supporting media-heavy experiments. Furthermore, we can see that most of the listed client/server tools follow a classic thick client model, which requires the installation of dedicated software components on all of the used experiment computers. A different model is used by OLYMPUS and JASPIS, which employ open-source web technologies to offer various language-related services. Both, however, do not provide dedicated graphical interfaces a wizard could use to interact with a test participant.

Another tool aspect that seems important, especially for applications that are based on proprietary software platforms, is their support for different file formats. Import and export

[1] http://www.tcl.tk [Accessed: August 14^{th} 2012]
[2] http://www.oracle.com/technetwork/java/javase/downloads/index.html [Accessed: August 14^{th} 2012]
[3] http://msdn.microsoft.com/en-us/vstudio/hh388566.aspx [Accessed: August 14^{th} 2012]
[4] http://www.adobe.com/products/director/ [Accessed: August 14^{th} 2012]
[5] http://www.softdoc.de/woz/index.html [Accessed: April 7^{th} 2012]

of data is crucial, not only in cases where a chosen tool does not support all the experiment tasks that need to be accomplished, but also in situations where third party products may allow for a more efficient generation of relevant source and design files. From the tools we could test, we can say that most of them do offer import and export feature at least to some extent. A positive example can be found in SUEDE, which supports designing a dialogue, running it and afterwards looking at the results in analysis mode. Also, its experiments are saved in an XML format which makes it easier to work with them in third party applications. Similar functionality is offered by RICHARD BREUER'S WOZ TOOL. While here we do not find an analysis mode, the software supports various export formats and also lets designers save tested dialogues, including their grammar, in VoiceXML format. The CSLU TOOLKIT allows for saving the trial-logs in two different text formats and DART, as already explained earlier, uses the ADOBE DIRECTOR format. So overall, when it comes to the export of generated data and log files existing tools offer relevant functionality. The import of data structures, however, seems more complicated. Generally import functionalities are available with all the tools we tested. However, in all cases special formatting is required, which makes it more difficult to use alternative software tools to design experiments. While for small experiments the design functionalities offered by the tools are usually sufficient, more extensive research interests can require complex and versatile designs for which alternative tools often provide better design support.

A final aspect of interoperability can be found in the possible integration of external products and services during runtime. Also here we find some support with existing tools, although the offered functionality is rather limited. The CSLU TOOLKIT, for example, allows for changing the talking head that is integrated and also offers access and integration of web-based information resources. RICHARD BREUER'S WOZ TOOL lets the user access media that is stored on the file system but does not offer any additional integration functionalities, and SUEDE can be seen as an entirely closed environment. Client/Server tools are generally more open to expansion. OLYMPUS, for example, is an aggregation of different modules offering different services and so it also offers various interfaces for expansion. Similarly, the XML-based architecture of the JASPIS dialogue manager makes it easy to integrate the software with other services. In both cases, however, the necessary upfront engagement with the tool in order to understand its features and Application Programming Interfaces (API) can be time consuming. A standard format for integrating components into WOZ experiments is currently missing.

In summary, combining the flexibility of open web technologies with the interface quality as well as the import and export functionalities of some of the tools discussed earlier, we see a potential for a new category of application. Open access would allow for different researchers and designers to use the tool, to improve it, and eventually adopt it to their very own requirement. Using web technologies, on the other hand, would significantly increase the potential user base, as most designers are capable of building web-based tools and interfaces and furthermore expand on the possibilities for testing interaction scenarios beyond the traditional computer workstation setting. However, for such a tool to be employable in a variety of set-

tings we first need to define an architecture that on the one hand covers a range of possible use cases and on the other hand allows for a flexible integration of external services. Only if the architecture enables researchers to develop their own components and integrate them using standard APIs, such a tool would go beyond what is currently available.

A Comprehensive Tool Architecture

Looking at the earlier outlined design space, it appears that a tool that aims to more comprehensively support the application of the WOZ method would need to offer a way of combining existing technologies in a more flexible manner. Some of those technologies might be fully working, others, however, might still be in an early development stage, and would rely on a wizard to raise their quality to an acceptable level. To make this possible, a software architecture is required which supports a flexible use of technology. Ideally, this would provide a modular, 'pluggable' framework that allows for components to be integrated or replaced easily. From an architectural point of view, we need to define each of the components, the different stages of development they can be in, and their relationships to each other.

As regards the different task responsibilities a wizard can take on within the interaction pipeline, one can define several different modes technology components can be in. A component can be relevant for a given setting i.e. it is needed and therefore needs to be represented in some form (e.g. ASR in a hands-busy-eyes-busy situation), or it is irrelevant for which it needs to have the possibility to be turned off (e.g. MT in a monolingual setting). In the case where a technology is needed, one can further distinguish between three different states. In the best circumstances the technology is of production quality and therefore can be used in a black-box manner producing results either for the wizard or a test participant. On the other hand, if the performance of a component is not sufficient, a wizard's task can be to enhance its quality. This type of scenario is particularly useful when the goal of an experiment is to investigate the improvement in quality that is needed for a technology to be acceptable, and therefore requires some sort of correction mode. Finally, in a setting where a component is needed but not available, it is usually the task of the wizard to completely simulate the missing functionality.

In summary, a comprehensive WOZ tool should enable a wizard to complement existing technology on a continuum by permitting her to simulate and correct technology, before finally using it as a black-box. Likewise Dow et al. [2005c] argue that a wizard might first take on the role of a 'controller' who simulates technology. Then, in a second stage act as a 'moderator' who approves technology output, before finally moving on to being a 'supervisor' who only overrides output in cases where it is really needed.

By looking at these different modes and carrying them on to the LTC level it is possible to further deduce a set of rules that handle the relationship between consecutive technology components. The first rule defines a fully working component as a black-box for which it can be preceded as well as followed by components in any state. If a component is simulated by the

	Input		Processing			Output	
Example	Text	ASR	MT	DM	MT	TTS	Text
Kelley [1984]	ON	OFF	OFF	ON	OFF	OFF	COR
Bederson et al. [2010]	ON	OFF	OFF	SIM	SIM	OFF	SIM
Gould et al. [1983]	OFF	SIM	OFF	SIM	OFF	OFF	SIM
Geutner et al. [2002]	OFF	SIM	OFF	SIM	OFF	ON	OFF
Schneider et al. [2010]	OFF	SIM	OFF	SIM	ON	ON	OFF
Karpov et al. [2008]	OFF	COR	OFF	ON	ON	ON	OFF

ON . . . The technology component is relevant for the given use case. A working solution is available.

OFF . . . The technology component is not relevant for the given use case and therefore not considered.

SIM . . . The technology component is relevant for the given use case. No solution is available so that the wizard has to simulated the component.

COR . . . The technology component is relevant for the given use case. A solution is available but does not produce satisfactory results. The wizard augments the technology by changing or overriding component output where necessary.

Table 5.6 – Some examples from the literature showing possible component/state combinations.

wizard, however, it needs to be followed by a working component. In cases where two or more consecutive components need to be simulated, they merge into one single task for the wizard (e.g. simulated ASR followed by simulated MT). Also when a corrected component follows a simulated one, both components merge into a simulation task for the wizard, as it seems defective to first simulate input for a component and then correct its output. Similarly, when one or more simulations follow a correction, all merge into an integrated simulation. Finally, a component can only be in correction mode when either its preceding component is fully working or when it receives its input directly from a test participant. Table 5.6 illustrates some of the possible component-state combinations and the related task of the wizard. Integrating those rules into a software architecture should allow for a more flexible use of technology when running WOZ experiments.

Wizard of Oz and the Web

An increasing number of traditional software applications are now offered in a web-based form, and the applications available are becoming more complex. The almost ubiquitous availability of high-speed internet has been an important factor, but also recent advances in web

technologies have been critical in supporting this transition from locally installed software to cloud-based web applications. While some of the WOZ experiment environments presented in the literature were built to some extent using web technologies (e.g. Turunen and Hakulinen [2000]), the majority are based on conventional software tools. The lack of simple support for web-based speech input and output has been a major obstacle, leading to the use of locally installed software, with associated installation effort, software dependencies and compatibility problems. Recent advances in web technologies, however, provide better support for dealing with speech. Modern web browsers are able to process audio and video data in real time and without the need for additional plug-ins. Upcoming web standards (i.e. the forthcoming HTML5 standard[6]) go further by giving access to computer hardware through the browser. These standards open up new possibilities for WOZ experimentation. We are now able to integrate speech input and output into web-based platforms, which significantly reduces the set-up requirements for an experiment environment. Furthermore, by using web services it is possible to build flexible tool architectures, such as the one just presented, in a way which allows for components to be integrated and replaced easily and on-the-fly.

As well as removing problems associated with installation, there is also a benefit in terms of interoperability with other platforms i.e. it is easy to integrate WOZ experiments into existing web-based software environments. For example, if a new interaction modality for a web-based help system needs to be tested, a WOZ client can quickly be added to an already existing interface. From the point of view of the wizard, it is further possible to add additional information channels such as video of the user or location data, which allows for the evaluation of not only speech but also multi-modal interaction. Finally, the possibility of running WOZ experiments on different platforms with different form factors (e.g. smartphones, tablets, media centres) represents another significant advantage that web-based solutions have over traditional software.

5.3 Integrating Requirements

If we look at the previously outlined design space for WOZ it seems that, (1) including the different configurations described in Table 5.4, (2) allowing for a flexible integration of different technology components as shown in Table 5.6, and (3) offering all interactions via web interfaces as discussed in Section 5.2.2, constitute the main features of an advanced WOZ tool, where a human wizard can work together with (imperfect) technologies. In order to evaluate this assumption and furthermore explore additional aspects of wizard support the next chapter of this thesis reports on a series of evaluations, looking at wizard workload, wizard consistency, and experiment construction. These evaluations were carried out with and alongside the development of a generic software platform for WOZ experimentation.

As presented earlier, we initially used a web-based technology probe to obtain first hands-on experience with WOZ prototyping. Results of a usability test with this tool were discussed

[6] http://www.w3.org/TR/2012/WD-html5-20120329/ [Accessed: April 7^{th} 2012]

in section 4.3.3. While this analysis provided useful feedback on the intuitiveness of our initial interface, the small time-frame of those tests did not allow for exploring realistic wizard behaviour and its change over time. Hence, before using our analysis results to construct a better, more generic tool, we employed our prototype for some more time, and conducted an in-depth study with a single wizard in a realistic experiment setting carrying out 11 experiments. The goal of this study was to look at wizard workload and identify some of those aspects of the wizard task that may remain significantly challenging throughout the course of a number of experiments. Next, we used those results and combined them with the findings presented in this chapter in order to start building a WOZ prototyping platform. Again, an initial evaluation with a small number of wizards was used to identify general usability issues before the platform was eventually employed to more thoroughly explore challenges of running as well as constructing WOZ experiments. To do so we conducted a study in which one wizard interacted with 17 users, giving us the possibility to identify a number of potentially problematic aspects with respect to wizard consistency over time, and then carried out an evaluation of the experiment construction process in which 10 experts and 51 non-experts used a slightly improved version of the prototyping platform to design language-based WOZ experiments. Table 5.7 exemplifies the different prototypes that were used to run those studies, while additional information will be provided in the following chapter.

Initial Technology Probe	Exploring Wizard Workload (cf. Section 6.2)
WOZ Prototyping Platform	Constructing a Generic Platform for WOZ (cf. Section 6.3) Analysing Wizard Performance (cf. Section 6.4)
Updated WOZ Platform	Supporting Experiment Construction (cf. Section 6.5)

Table 5.7 – The tool prototypes used to explore different aspects of Wizard of Oz experimentation.

5.4 Summary

Based on Chapter 4, which was looking at existing tool support and what can be learned from previous experiments, this chapter has focused on generic aspects of WOZ experimentation. We looked at potential users and their interest in the method. Furthermore the results of an interview study were discussed which identified exploration of new design ideas as the major use case for WOZ, highlighted a variety of challenges wizards are facing when planing and conducting experiments, and showed that researchers usually build their own wizard tools. Moving on to the system perspective, we presented a comprehensive analysis of the wizard task, its roles, and their possible influence on the layout of a potential wizard interface. We furthermore discussed technological aspects of existing tools and described how a comprehensive and flexible tool architecture in combination with modern web technologies can be used to offer more generic tool support. In the next chapter we use the gained insights to move from the initial prototype described in Section 4.3.3 to a generic platform for WOZ experimentation.

Along this road we further report on a series of evaluations that were specifically targeted at exploring the task of the wizard as well as the creation of WOZ experiments.

Chapter 6

A Platform for Wizard of Oz Experimentation

"Please make yourself comfortable while I go to the door
of the Throne Room and tell Oz you are here"
– A Soldier

This chapter describes our approach of integrating previously identified requirements with the results obtained from a series of evaluations looking at various aspects of the wizard task. We start with an exploration of wizard workload (cf. 6.2) and then describe a generic platform for WOZ experimentation (cf. 6.3), which, after being evaluated, is furthermore used to explore wizard performance over time (cf. 6.4) as well as the experiment construction process (cf. 6.5).

6.1 A Multi-component Research Approach

Expanding on the initial findings presented in previous chapters, the following sections of this thesis constitute the empirical core of our research agenda. If we reflect on our methodology discussed in Chapter 3 we may argue that Chapter 4 and Chapter 5 were mainly used to build a conceptual framework for WOZ prototyping. Such was achieved through an extensive study of the literature followed by some low-fidelity prototyping and user evaluations, which led to the extraction of basic requirements for WOZ experimentation. Building on that we then used an interview study as well as a structured system analysis to further refine user demands and bring them to a more generic level. From that perspective these earlier parts were investigating the first of our research questions, whose main concern was the nature and application area of WOZ and the types of scenarios in which it can be used as a prototyping method (cf. Section 1.4).

The following sections of this thesis now aim at using this insight to explore various additional aspects of WOZ prototyping. Herein a strong focus lies on the wizard task and how it can be supported through a new WOZ prototyping platform. Using both experimentation

and observation, as defined by SDM, we drive the development and evaluation process of this platform; which in turn is again used as a tool for exploration. This approach is used to tackle our two remaining research questions. That is, on the one hand it aims at identifying features and mechanisms that can be used to ease the burden of the wizard task, and on the other hand it explores the design and conduct of natural language based WOZ experiments. To some extent this is achieved by investigating the actions of wizards while running a WOZ experiment (with our platform) and constantly identifying additional software features that might help them do so. In addition it is, however, also necessary to look at WOZ experimentation from a more holistic point of view and include the experiment design and construction phase. The following sections therefore represent an integrated research process consisting of several components. One component looks at generic WOZ tool support that addresses mainly our second research question, which is concerned with the features to be implemented into a prototyping environment. From an SDM point of view this falls into the stage of prototype building and evaluation. With respect to our last research question, which is concerned with the change of wizard behaviour over time and the challenges of experiment construction we, however, require a more experiment-driven analysis approach. Such is achieved through one research component that specifically looks at wizard performance and wizard consistency, and a final research component that focuses on the experiment construction process and how it can be supported.

In summary we can therefore argue that through this combination of different research components we are able to generate insight into WOZ prototyping not only from a technological but also from a methodological perspective.

6.2 Exploring Wizard Workload

In order to explore wizard workload, how somebody gets accustomed to the task, and what aspects remain challenging even after a considerable number of conducted experiments, we employed our previously described prototype in a more realistic WOZ setting. We used an in-house study related to machine translation that was conducted by a researcher in a computational linguistics laboratory. This one person was observed acting as a wizard over 11 sessions with different test participants. The scenario was similar to the one we already used for our usability tests i.e. we simulated a speech-based interaction between a German speaking customer and a system recommending appropriate Internet connection bundles. Test customers were international students who were told that they would be interacting with a prototype of a new adviser system, which would understand spoken input. They were asked to complete two tasks with the system. First they needed to obtain information on an offer for pre-paid Internet, and after that they were asked to inquire about a land-line contract. They were told that the system could understand spoken input but would reply via text output on the screen. None of the test customers knew that they were interacting with a human until after the test was completed. Participation in the study was voluntary and compensated with a EUR 10 book

voucher.

As for the gathering of data we employed a similar set-up as already used in our initial study (cf. Section 4.3.3). That is, SKYPE[1] was used to create and audio connection between a test-participant and the wizard (Note: Only the test-participant's speech was transferred. The wizard's microphone was put to mute.). The actions of the wizard were logged (i.e. all the utterances the wizard sent were time-stamped and stored in a database), and her screen was captured. The speech of test-participants was recorded as well as comments made by the wizard. Also, the wizard's facial expressions were video recorded (Note: Recordings were used to identify times of confusion but no structured analysis of the video-material was conducted) and retrospective analyses of certain actions of the wizard were conducted after the task.

While next we report on the main findings of this second round of experimentation with our initial prototype, it needs to be highlighted that due to the iterative development methodology we had been following, where implementation, evaluation and analysis underlie very short cycles and the borders between those phases were often blurred, it was not always possible to assign each of the insights to the one evaluation cycle where it was discovered. Hence, we understand the following discussions rather as a continuation of our tool-driven exploration started earlier on (cf. Section 4.3).

6.2.1 General Results

Taking what we had discussed earlier and combining it with the observations and retrospective analysis of these 11 wizard sessions we were able to gain additional insights with respect to the composition of a wizard interface as well as practical difficulties associated with being the wizard (interpretation of pauses by the test-participants, deciding when to initiate a repair dialogue, etc.). Three earlier identified factors influencing the interface layout were confirmed by this study. The first factor is the scenario that needs to be supported. That is, in situations where there is a clearly defined dialogue structure the interface may act as a step-by-step tool. However, in settings where the interaction between a person and a system is rather loose and dynamic (as it was the case in our test scenario), a wizard simulating the system needs more flexibility. The second aspect we found influential for the interface layout is the role a wizard is playing. The wizard interface will appear differently when a wizard is simulating the behaviour of an LTC, compared to a situation where an actual LTC is in place and the wizard is only correcting its output. In the second case the interface needs to offer some kind of edit element whereas in the first case no interface element is needed at all. The third factor identified is the modality through which a test participant interacts with the system. For instance, a situation in which a participant would interact with a system by speech only is treated differently to a situation in which speech is coupled with one or more other input modalities. Thus, input modalities can be active, like text input or gestures, or passive as for instance location parameters. Both cases have a different effect on the wizard interface.

[1] http://www.skype.com/intl/en/home [Accessed: September 19^{th} 2012]

Looking at the wizard's point of view, the task of handling domain data, timing, and the use of confirmation utterances were earlier identified as being particularly challenging. That is, influenced by the general layout of our wizard interface, wizards had difficulties switching their attention from one area of the screen to another. The problem was observed mainly at the end of an experiment where they needed to select responses from the main dialogue flow as well as from the domain data area, which led to confusion and delays. Furthermore the dialogue flow itself caused some difficulties. As wizards were not very familiar with the prepared dialogue utterances most of them had problems finding the right responses or they had a different response in mind when searching. Here one could argue that wizards should receive sufficient training or at least be given more time to acquaint themselves with the utterances before running a test. If we compare these findings with the result of the 11 experiment sessions our 'experienced' wizard conducted, we however see that in stress situations even an expert may struggle. That is, although our researcher was involved in the experiment construction process and therefore familiar with all the utterances, he did at times experience problems finding the appropriate response. This raises an interesting question with respect to methodology, namely who should act as the wizard. From the literature (e.g. Bevacqua et al. [2010b]; Bradley et al. [2009]; Strauß et al. [2006]), it seems that the experiment designer will generally act as the wizard. While this could potentially be a source of bias, the experience of the evaluations would suggest that using other people as wizards is a strategy which will encounter difficulty in practice without considerable familiarisation and training.

More specifically looking at the wizard task, our evaluations (four usability tests evaluating the wizard interface and this single WOZ study looking at wizard performance over time) identified three aspects, which require further investigation. Firstly, it was challenging for wizards to deal with domain data. Both the four wizards in the initial usability study, as well as the one person who acted in this role as part of the in-depth study with 11 test participants had problems. That is, wizards had difficulties finding the information demanded by customers. This specific problem of information retrieval under time pressure was foreseeable and so the prototype design tried to offer support by providing a filter function and automatically adapting those filters based on the dialogue progress. However, the latter seemed to confuse wizards, as it was observed several times that they manually changed filter values even though there was no need for doing so. Several wizards mentioned during the post-test interviews that they felt lost and that by manually adapting the filters they tried to regain control.

The second interesting aspect of the wizard task was an issue participants had with the timing. Since the customer interaction with the prototype was in a speech-in-text-out format, it was difficult for a wizard to estimate the time a customer would need to read a response utterance. This problem was observed several times, when a wizard assumed a problem with a sent response and therefore sent an additional utterance in order to check on a customer's status. Usually, however, it was not a problem with the sent utterance, but the customer was simply not finished reading. The log files of the 11 sessions with one single wizard highlight that this issue remains over time i.e. it seems independent of a wizard's experience, showing

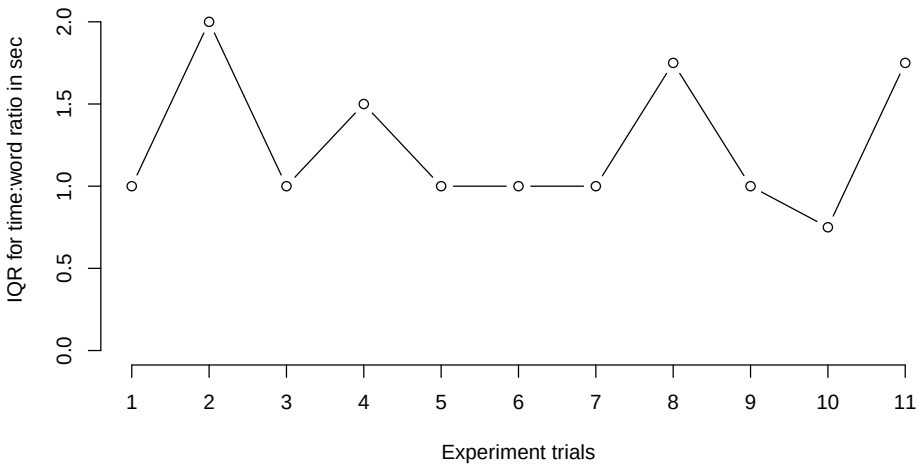

Figure 6.1 – A wizard's variations (IQR values) of estimating a test customer's reading speed over the course of 11 experiment trials.

variations of the wizard's estimations between one and two seconds for any of the trial runs. Fig. 6.1 shows the interquartile range (IQR) per trial of the time our 'expert' wizard estimated a test customer would need to read a word (Note: An IQR of zero would indicate consistent estimations).

An analogous problem would be knowing when a pre-recorded or synthesised speech utterance has finished output. This shows that a lack of status information can influence the interaction and therefore reduce the reliability of the produced experiment result. The problem of these acknowledgement tokens or back-channels has also been discussed by Jurafsky and Martin [2008].

A third issue observed concerned the use of confirmation sentences, which are commonly used in speech-operated systems. The confirmation sentence tries to resemble the direct feedback normally given by GUIs, which is otherwise missing in a speech-based interaction paradigm. As WOZ tries to prototype realistic technology behaviour, it is important that this sort of feedback is mimicked. Our first evaluation, however, showed that it is relatively easy for a wizard to forget to send a confirmation utterance. This behaviour, which was observed with both the wizards in our usability tests and the experienced one acting in 11 test sessions, is quite natural, as humans would not usually confirm every sentence of a dialogue partner. Yet, from a system perspective such characteristics are necessary in order to maintain the fidelity of the experience. Therefore, one might argue that a well-designed WOZ interface would need to actively enforce confirmation behaviour.

6.2.2 Designing a New Wizard Interface Layout

Summarising all our evaluation results collected up to this point, we were able to define four common concepts that may consistently be applied in a wizard interface layout. Firstly, a wizard interface should have a dialogue representation area that serves as the main interaction channel between the wizard and a test participant. Second, based on our experiment as well as the literature there is a need for a representation of the dialogue history. In other words, a wizard should be able to see what has happened previously in the dialogue. A third commonality between different WOZ experiments is that they all have a built-in way of dealing with errors. An interface area that is dedicated to dialogue repair strategies may therefore help controlling the interaction. Finally, in cases where the dialogue progress is based on some kind of slot filling mechanism, the status of those slots needs to be visible to the wizard. Hidden automatisms can lead to confusion and therefore should only be used with experienced wizards. Taking those concepts into account, the following section describe our efforts of constructing a better, more generic tool for WOZ experimentation.

6.3 Constructing a Generic Platform for Wizard of Oz

Informed by these evaluation results we left our initial prototype and started building a new application platform. The goal here was not only to tackle the so far discovered problems, but also to move away from a tool that supports only one specific WOZ setting (i.e. the described customer-machine dialogue recommending Internet connection bundles) to something that would support the generic application of WOZ. To do so we implemented the software architecture described in Section 5.2.2 and furthermore integrated several LTCs, i.e. one ASR component, one TTS component and two different MT components, via web services. Similar to the wizard interface that was used with our initial prototype this new tool also offered an interface based on a staged dialogue structure (cf. Figure 6.3). However, several additional features were integrated in order to address some of the problems identified.

6.3.1 Platform Architecture

The new WOZ platform was implemented using the GOOGLE WEB TOOLKIT[2] which supports the construction of web interfaces using the JAVA[3] programming language. The core of the platform is represented by a shared database that is responsible for exchanging input and output data between wizards and test participants. It stores both static user data as well as dynamic logging data. A single table is used as a shared event stream. Wizard and client (i.e. test participant) interfaces are based on a modular software architecture which consists of three different types of classes. View classes are responsible for the visual layout and therefore allow for future variations and adaptations of the interfaces without influencing the underlying logic

[2] https://developers.google.com/web-toolkit/ [Accessed: October 12^{th} 2012]
[3] http://www.java.com/en/ [Accessed: October 12^{th} 2012]

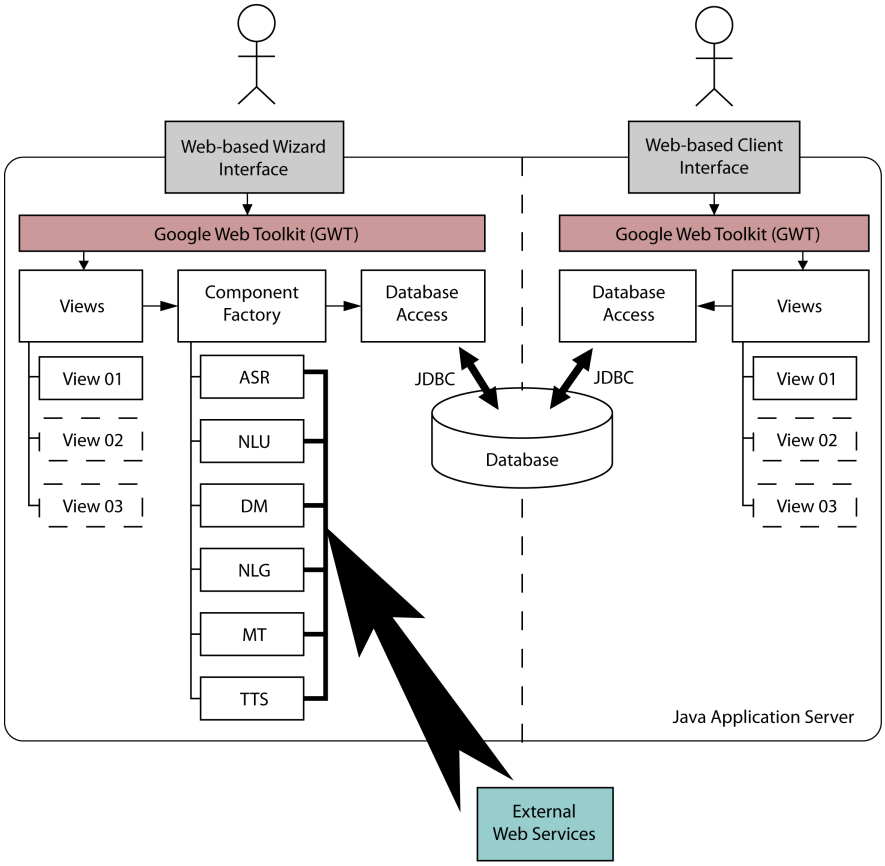

Figure 6.2 – Architecture of a new WOZ platform.

of the platform. Component classes are used to encapsulate various language technologies along the LTC pipeline (cf. 5.1). This not only permits the development of components as separate entities but also allows for hosting them externally and access them through standardised web services. Finally, database access classes act as proxies providing methods for storing and retrieving data. Figure 6.2 illustrates the architecture of this new WOZ platform.

6.3.2 Interface Elements

First the dialogue structure of this new WOZ platform was implemented using a tab-layout. This layout had the goal of helping wizards distinguish the different dialogue stages and allow them to more easily track dialogue progress (cf. Figure 6.4). Second, the area for recovery utterances was converted into a more general place-holder for frequently used utterances (cf. Figure 6.5). Third, we added editing functions permitting a wizard to add, edit and delete utterances as well as easily move them between different dialogue stages or mark them as frequently used (cf. Figure 6.6). Furthermore, in order to provide more freedom when interacting with a test participant, the interface offered the possibility to include a chat-style text input field. Whereas in most WOZ experiments this kind of free interaction should be avoided, in some situations the possibility of exploring a design space more freely can be desired (cf. Figure 6.7).

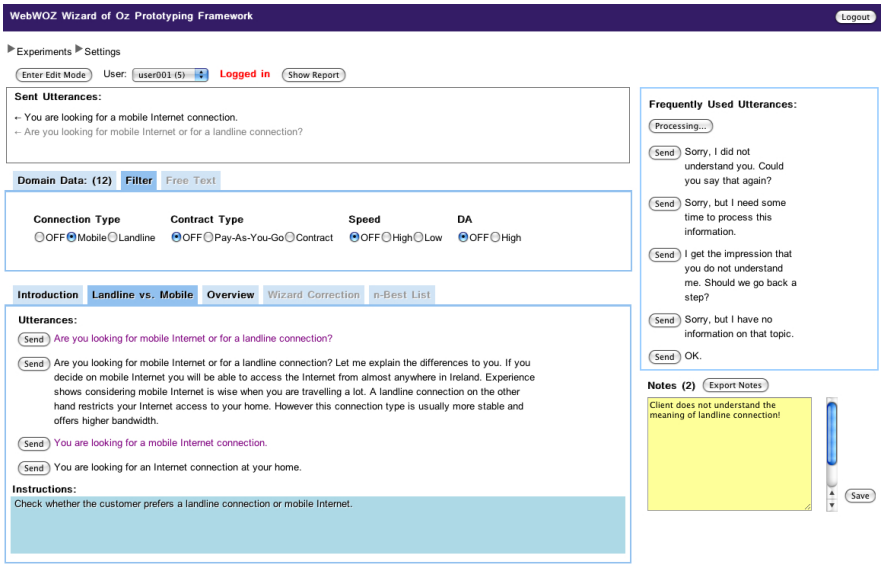

Figure 6.3 – A wizard interface based on a generic application framework.

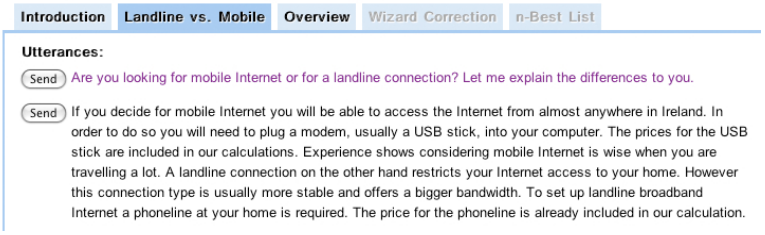

Figure 6.4 – A tab-layout to make it easier to distinguish different dialogue stages.

With a fourth feature, we tried to tackle the problem of retrieving domain specific information. A configurable filter mechanism for domain-data-based utterances was implemented that permitted wizards to specify filters as well as filter values themselves, and therefore aimed at giving them better control over the domain data (cf. Figure 6.8). Furthermore, a history element was introduced which held all the utterances sent to a test participant. Utterances were listed in chronological order and preceded by an arrow to the left, marking them as outgoing; the most recent sentence being highlighted in a different colour. In cases where the ASR component was used, its input was also displayed in the history, preceded by an arrow to the right, marking it as incoming (cf. Figure 6.9). Finally, in order to alleviate timing problems, a notification system was implemented, aiming at informing a wizard of a test participant's availability (cf. Figure 6.10). Also, areas for taking time-stamped notes (cf. Figure 6.11) and fields for reminders and instructions (cf. Figure 6.12) were added, with the aim of improving the consistency of wizard behaviour.

6.3.3 Integration of Language Technology Components

In terms of language technologies this new WOZ platform enabled the wizard to choose from a set of pre-configured language components to be used in an experiment. It was possible to

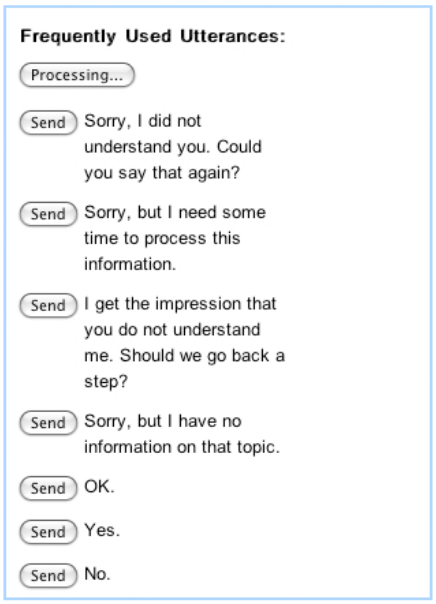

Figure 6.5 – Area for frequently used utterances.

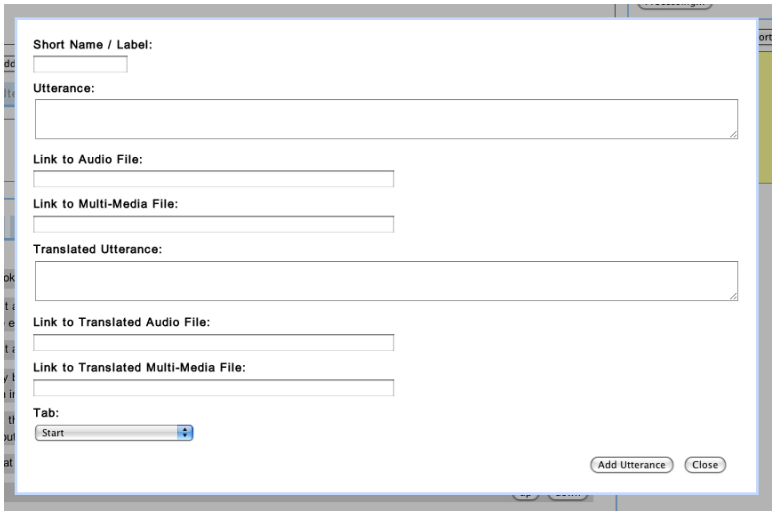

Figure 6.6 – Function to add utterances.

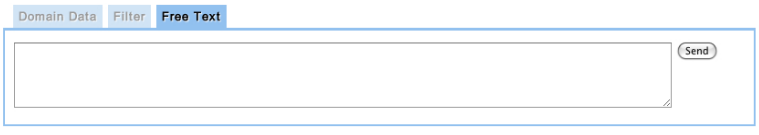

Figure 6.7 – Chat function to provide more freedom when interacting with test participants.

Figure 6.8 – Configurable filter mechanism for domain-data-based utterances.

Figure 6.9 – History showing already sent and received utterances.

Figure 6.10 – Notification system informing a wizard of a test participant's availability.

choose which components to turn on and which to turn off. Additional component settings that were supported include the language for ASR and TTS as well as the language pair, i.e. the source and the target language, for an MT component. If an experiment setting required a more controlled set-up in which MT or TTS (or both) needed to be held consistent, the prototype further offered the possibility to add pre-defined translations as well as pre-recorded audio.

In cases where the wizard was to play an augmentation role, he/she could choose whether a component should be treated as productive for which the output would immediately be sent to a test participant, or whether a component was unreliable and would require for the output to be post-edited. Two modes of augmentation were supported. In N-best list mode the wizard could choose from a list of possible outputs (given the used technology component supported this feature). This mode was particularly useful in situations where a component did produce understandable results, but the output was ambiguous. For cases where the component quality is considerably flawed, we also provided a correction mode in which a wizard was able to fully edit the output before it was sent. Both modes were supported for ASR as well as MT components.

6.3.4 Evaluation of Features

Using this advanced prototype a new round of usability tests was conducted. The goal was to test whether the implemented set of features would generally be understood and help the wizard task. For this a member of our research team took on the role of the customer (for matters of comparison we used the same customer-machine scenario as for the initial evaluations described in Section 6.2) interacting with three different wizards. Again, none of the test wizards had acted in this role before. They had unlimited time to explore the new wizard interface

Figure 6.11 – Area for time-stamped notes.

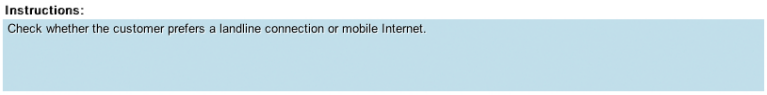

Figure 6.12 – Field for wizard instructions.

and ask questions. A test facilitator was present throughout the whole test helping with severe problems and observing the interaction with the WOZ platform. A retrospective interview was used to investigate further difficulties encountered during the experiment.

The results of this usability test conducted with the new tool suggest that the concepts introduced by the modified wizard interface were helpful. The tab layout representing the dialogue structure was found to be useful as it separated the dialogue into digestible steps while at the same time it enabled the wizard to keep an eye on the dialogue as a whole. Even though the wizards did not design the dialogue themselves, none of them was lost or had problems navigating. Also the concept of having a dedicated area for frequently used utterances was quickly understood and several wizards highlighted utterances they would like to put there. The handling of domain data, which led to some confusion with our initial interface, was now less problematic. Wizards had to adapt the filters manually which sometimes led to stress towards the end of a session, as they forgot to do so throughout the experiment; however, when asked if they would have preferred for the filters to be adapted automatically, all stated their preference for the manual approach. Finally, the integrated notification mechanism that showed a test participant's current state of availability was useful but did not go far enough. As the implemented version only highlighted whether somebody was logged in or not, it was not possible to deduce whether a participant was reading, thinking or simply waiting for a response.

6.4 Analysing Wizard Performance

In addition to evaluating and verifying the features of our new prototyping platform we were also interested in further exploring aspects of the wizard task. Hence, a new round of evaluation was focusing on wizard performance over time, particularly looking at task adaptation, consistency and potential changes of wizard behaviour. To do so we observed two different WOZ studies (both had dedicated research interests and therefore qualified for objective analysis). While the first study (Study A) was again motivated by a researcher in the computational linguistics research group, the second one (Study B) was conducted in collaboration with a second research team from a different university.

6.4.1 Evaluation Method

Two different WOZ studies were observed. In both cases a single wizard was interacting with 17 and 13 test participants, respectively, over the span of several weeks. While in the first study we specifically aimed at evaluating our new WOZ tool and how it is used over time (i.e. wizard performance), the second study provided additional data on wizard behaviour and

eventually made it possible to conduct a meta-analysis on wizard consistency. Below we briefly summarize the two studies before discussing their distinct results. At this point we would like to note that our analysis aimed at understanding the wizard side of an experiment. Readers who are interested in the results of these studies from a participant's perspective are advised to consult the referred publications.

6.4.2 Study A

The first study was building upon the results of our initial experiment (cf. Sections 4.3.3 and 6.2) and aimed at extending them into the spoken language domain. That is, this time the researcher employed text as well as synthesised speech output, which permitted us to compare those two modes with respect to wizard consistency on sending utterances. The technical set-up was similar to the one described in Section 6.2 using our WOZ platform alongside an active Skype session to transfer utterances and speech between the wizard and her test-participants.

From an experimental point of view the person who acted as the wizard was interested in understanding the influence machine translated utterances have on synthesised speech output and how this changes the user experience when interacting with a system. For this very specific experiment setting it was therefore important to control quality for both MT and TTS. Hence, utterances were again pre-translated into German and consequently synthesised. The 17 test participants who took on the role of test customers were all native speakers of German. They were told that they would be interacting with a system that understands spoken input and asked to solve two information retrieval tasks similar to those used in the previous study. In one of the tasks the system would communicate with them via speech output, in the second, it would produce text on a screen. After the experiment participants were informed that a human wizard operated the system. Additional details and results of this study exploring the influence MT has on TTS can be found in Schneider [2013].

6.4.3 Study B

The second study employed WOZ to collect a corpus of realistic dialogue utterances for an online language pronunciation trainer. For this purpose our research partners from a different institution had already developed a working prototype of a system that could analyse a test-participant's pronunciation of an English sentence and highlight which words or parts of a sentence were mispronounced. Linking this analysis to actual textual feedback was, however, not supported at the time. The study therefore used a human wizard to produce real-time textual feedback based on the results of the pronunciation analysis. Our WOZ tool was used to implement the study. Different text elements were prepared so that they could be assembled to flexibly form a feedback sentence (a function that we first had to implement which gave us an additional use case to include in our tool architecture). The wizard was able to compose the sentence and fill in the specifics or alternatively create a response completely from scratch. On the client side the pronunciation system was integrated with the WOZ client interface (i.e.

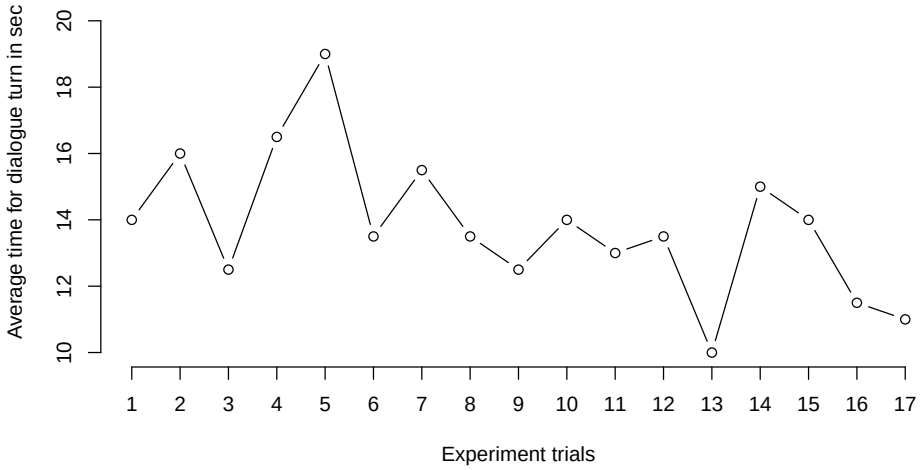

Figure 6.13 – Average times for a dialogue turn in the different trial runs of Study A including the time a test customer needed to read or listen to a sent utterance and respond.

it was running in a separated frame integrated in the web-based pronunciation training system). Again, Skype was used to transfer the spoken input from a test-participant to the wizard. Feedback sent from the wizard was then displayed in a text box situated in the bottom of the screen. A member of the research team, who was also involved in developing the pronunciation analysis system, acted as a wizard. One trial run was conducted to test the set-up after which 12 test-participants were recruited to train their pronunciation (i.e. 13 experiments in total). Additional details and results of this study with respect to the pronunciation training aspect can be found in Cabral et al. [2012].

6.4.4 Results

Looking at the results of the first study (i.e. Study A) we can see a certain adaptation to our new wizard interface as well as to the task. Over the span of 17 experiment sessions there was a trend towards decreased selection time for response utterances as well as a faster filtering of domain data which resulted in quicker dialogue turns (cf. Figure 6.13). Furthermore the wizard reported on relying on the notification mechanism, which was particularly useful in one case where the experiment set-up experienced some connection problems. On the other hand the study also exhibited a certain humanisation of the responses over time. For example, it was observed that during the first experiment sessions the wizard used confirmation utterances to acknowledge customer input. Over time this behaviour, however, changed and the use of 'OK' as a way of confirmation increased. Figure 6.14 shows the trend of those two parameters throughout the course of the 17 experiments trials. Looking at the two tested modes (i.e. text and speech) separately we furthermore find that in speech-based interaction the wizard seemed

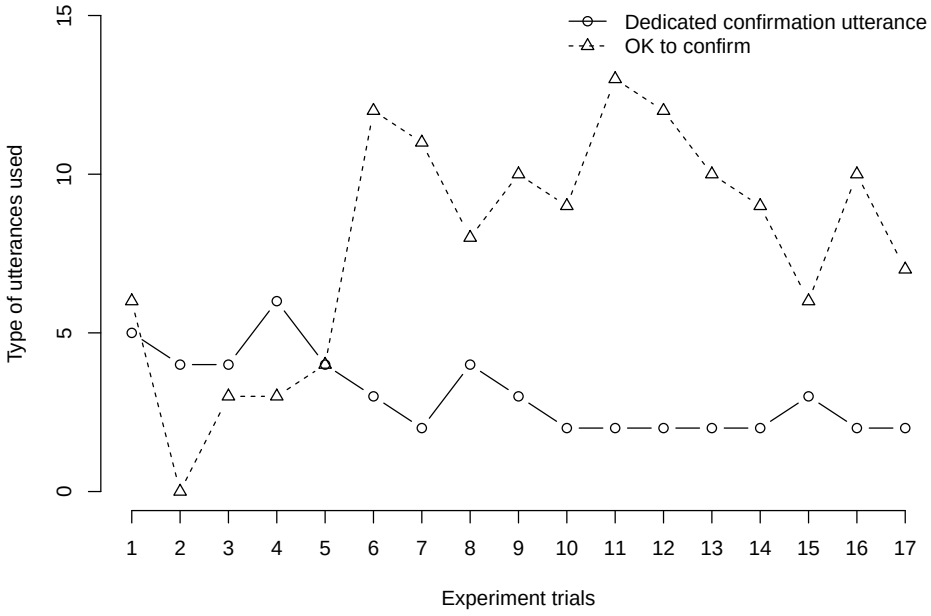

Figure 6.14 – Types of utterances used to confirm customer (i.e. test participant) input over the course of 17 experiment trials.

to utilize 'OK' more often (max=8, median=4) and confirmation utterances slightly less often (max=3, median=1) than in their text-based counterparts (max=6|4, median=3|1).

The challenge of consistently simulating machine-like behaviour was also apparent in cases where the wizard responded to input that seemed unlikely to be understood by a machine. In general, estimating this threshold of what can realistically be processed by a system and what should instead trigger an error recovery routine, can be seen as a major challenge when applying the method, particularly if consistency needs to be achieved over the span of many experiment sessions. Peissner et al. [2001] recommend the use of probabilistic errors to simulate more realistic behaviour. In our case, we observed that the wizard tried to solve this issue by always using the same set of utterances to start as well as to end a dialogue (including simulated recognition errors). For the main part of the conversation, however, this usage of a strict set of utterances was not applicable.

Comparing some of these results from Study A with the second study (i.e. Study B) we find an interesting difference of wizard behaviour. During the first trials of Study B the wizard's response time was stable, even slightly decreasing (not significant), which points to better wizard performance due to task and interface habituation, similar to what was already observed in Study A. After some time, however, it started increasing again (cf. Figure 6.15 line 'IQR of the measured response times'). With respect to this result it is important to point out that unlike Study A, this second study was less structured and depended more on a test participant's performance. That is, in Study A the dialogue and its possible system utterances (i.e. the

utterances the wizard was able to use) were entirely pre-defined. Study B, however, enabled the wizard to interact with a participant in a chat-like fashion. While some of the utterances were still pre-defined and sent by clicking a button, we implemented the possibility to give context specific feedback, i.e. feedback that could be adapted to the unpredictable performance of a test participant pronouncing a word or sentence in English. To do so, a wizard was able to choose a pre-defined utterance and change it so that the feedback fits the result produced by the pronunciation analysis. In cases where changing a pre-defined utterance was too cumbersome, the wizard could alternatively type the complete feedback and send it on.

If we look at how the wizard used this feature over time we see that, except for the second trial where the test participant had problems operating the client interface and the wizard therefore needed to send numerous clarification utterances, the number of utterances processed in this ways stayed more ore less consistent (mean=12.54, mode=11, median=11) throughout the course of all 13 experiment trials. Analysing the content of the generated utterances more closely, however, we see that the average number of words that were used started increasing with the seventh trial (cf. Figure 6.15 line 'Average number of words produced in an utterance'). While we are unsure why the wizard changed his feedback style after the sixth session (Note: As the experiment was used to develop a corpus of feedback utterances the goal of the wizard could have been to test an increased number of possibilities), the increasing number of typed words is likely the cause for this increase in response time. The connection between those two parameters becomes even more apparent if we calculate the 90^{th} percentile of the measured response time (Note: We use the 90^{th} percentile in order to clean the data from the expectationally long response times that sometimes happened due to unforeseen participant actions; e.g. a participant took a long time to respond to a wizard response) and compare it with the average number of words produced in an utterance (cf. Figure 6.15 lines '90^{th} percentile of the measured response time' and 'Average number of words produced in an utterance'). A Pearson significance test showed a highly significant positive linear correlation between these two variables; r(11)=0.9469, p=9.336e-07.

What we see here is an example that shows how one can successfully adapt to the wizard task (Note: This was the first time our wizard was engaged in this role) as well as to its operation interface, leading to measurable performance improvements over time. Yet, we also see that this type of familiarisation can lead to changing wizard behaviour. Producing longer utterances, as we observed in our example, points to a certain humanisation of wizard responses that comes with growing experience. From a participant's perspective longer and more complex utterances lets a system appear more intelligent and therefore can create a false impression of the product's potential. In cases where the perceived system performance changes throughout the duration of an interaction this might furthermore lead to frustration.

If we look at the variation of produced utterance lengths within a trial we do see that the average standard deviation in the first six trials is slightly lower (mean SD=3.12) than in the subsequent ones (mean SD=4.23), indicating a greater lack of consistency. However, this figure is highly influenced by the actual performance of individual test participants and so after having

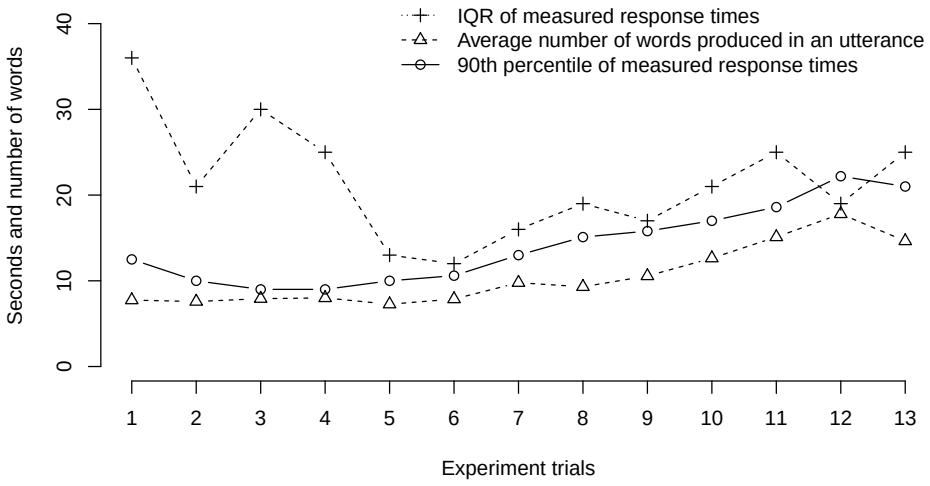

Figure 6.15 – IQR of the measured response time per trial, average number of words produced in an utterance per trial, and 90^{th} percentile of the measured response time per trial. While initially the response time decreases, more words per utterance reverses the effect later on.

analysed the relevant content in more detail we believe that participants did not experience any change in system behaviour. Yet between experiment trials we see a difference where the first instances resembled a less 'sophisticated' system than the later ones. The results of the questionnaires that were given to participants after they had finished a test suggest that they did not experience a difference, as the feedback quality was constantly rated with 4 or 5 on a five point Likert item ranging from 1 'not enough' to 5 'very good'.

However, while for a single test participant the intelligence of a system (beyond a certain threshold) might not matter, it can very well influence the results of a study and therefore highlights an important challenge of simulation.

Following, another consistency challenge, namely the time a wizard gives test participants to read a text utterance, is discussed as part of a meta-analysis conducted over the observed WOZ experiments.

6.4.5 Meta-analysis of Wizard Consistency

Varying wizard consistency was highlighted as a challenge for WOZ experimentation and so looking at the above described studies and including data from our initial round of experimentation (the in-house WOZ study with 11 test participants described in Section 6.2) we were interested in exploring an issue that was already discovered earlier on (cf. Section 6.2.1), namely the consistency with which a wizard estimates a test-participant's reading speed in order to send the next system utterance. This is an issue that is very specific to the (mainly) text-based output of these WOZ experiments but relates to the rather stressful task of the wizard. Gen-

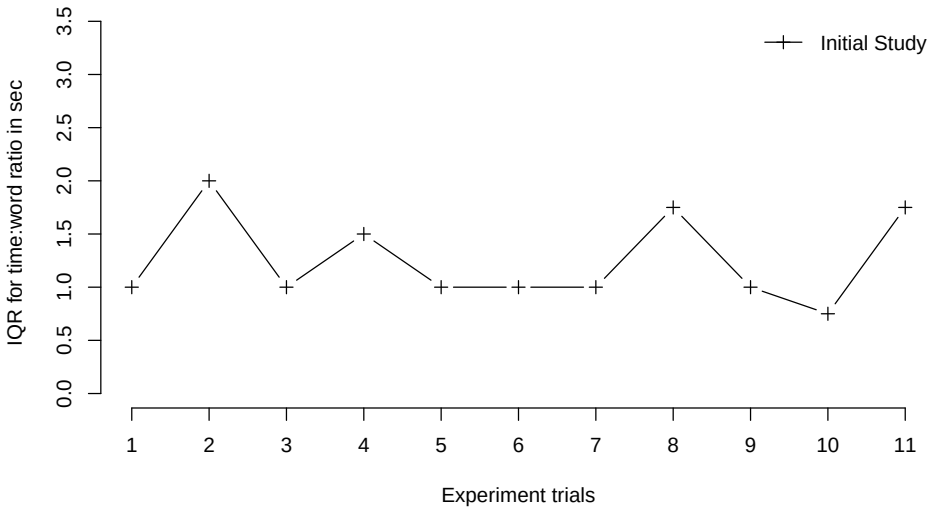

Figure 6.16 – Interquartile Range (IQR) Values for Text-based Interaction in Initial Study.

erally, a wizard first listens and interprets a test-participant's spoken input, and then searches for an appropriate response utterance to reply (in multi-modal settings several input and output channels might be simulated). In a sense this task can be seen as an information retrieval problem under time pressure in which a majority of unsuitable responses act as distractors for finding the one suitable response. To tackle this problem utterances are often limited to one or two sentences. Short utterances are easier to distinguish from each other and at the same time allow for a higher degree of flexibility. Therefore, a wizard can 'feed' them to a test-participant bit by bit and they can be re-used for different responses. The consequence is, however, that a wizard might need to send several utterances in a row without receiving a test-participant's input in between. Here it is up to the wizard to estimate the time a test-participant needs to read and process one utterance so as to decide when to send the next one.

Analysing the log files of our initial WOZ study (11 experiments) we were able to identify 156 instances in which the wizard had to estimate a test-participant's reading speed. In the above described Study A (17 experiments) there were 117 instances (i.e. looking solely at text-based interaction) and Study B (13 experiments) contained 218. Next we took the time a wizard waited before sending a follow-up utterance and divided it by the number of words that were used in the preceding utterance. In an optimal setting this time/word ratio would be consistent during the course of one study trial and furthermore stable throughout the whole study. Taking the Interquartile Range (IQR) of those ratios for each trial provides a measurement for in-trial consistency. If the IQR is zero it can be assumed that the wizard acted consistently throughout the trial, any number above zero constitutes variability in her estimations of a test-participant's reading speed. Figures 6.16, 6.17 and 6.18 show the IQR values for all three WOZ studies and their changes over time.

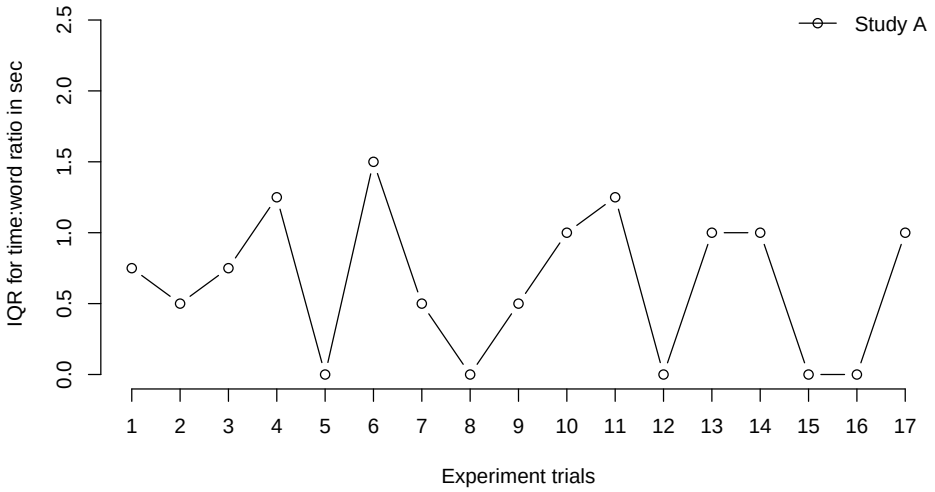

Figure 6.17 – Interquartile Range (IQR) Values for Text-based Interaction in Study A.

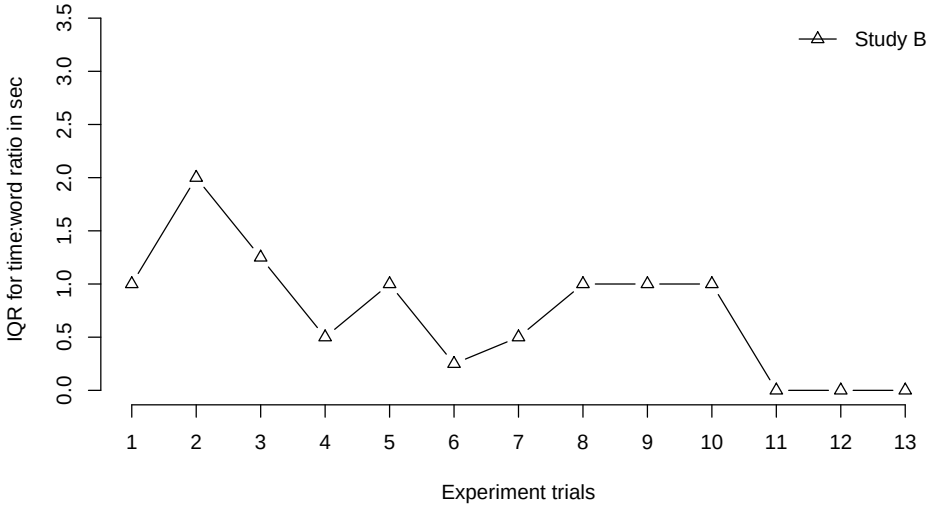

Figure 6.18 – Interquartile Range (IQR) Values for Text-based Interaction in Study B.

At this point it needs to be highlighted that the wizards in those three studies were not actively asked to estimate a participant's reading speed. Nor were those studies specifically designed for exploring this aspect of wizard behaviour. We simply report here on a meta-analysis conducted in order to better understand the challenges that were involved. The results of this analysis show that a wizard's natural actions (i.e. their decisions on when to send the next utterance) vary significantly between experiments, and also within an experiment, wizards were inconsistent in their estimation of when a test-participant would have finished reading one utterance and therefore could be confronted with the next one. In the initial study (cf. Section 6.2) the time/word ratio varied in each experiment between 0.5 and 2 seconds. In Study A the wizard was consistent in 5 experiments out of 17 (i.e. IQR = 0.00) and in Study B only the last 3 experiments showed no variation. One could infer that, since consistent behaviour on this matter was not explicitly required from the wizards, those results simply reflect a faulty experimental design. Yet, one of the wizards explained that she was reading a sent utterance slowly in her mind before she sent the next one, which shows a certain awareness of the problem. Furthermore, it could be argued that consistent behaviour generally depends on experience, for which a wizard needs to receive sufficient training before being able to run a study with real test-participants. However, based on our interview series with 20 researchers who have experience with WOZ experimentation (cf. Section 5.1.2) we found that the time spent on wizard training is often less than 30 minutes. Only two of them reported that a wizard received extensive training before interacting with test-participants. Furthermore, our data shows that even with appropriate wizard experience, the estimation of a test-participant's reading speed remains a challenge. The wizard in Study B seemed to benefit from the experience gained over the course of 13 trials, which led to consistent behaviour at the end. The wizards in the initial study and Study A, however, did not show significant improvements over time. Hence it can be argued that additional wizard support on this matter may be needed, even in cases where the wizard has received an appropriate level of training.

On a more critical point, one could argue that letting a wizard first collect all the relevant utterances before sending them on would eliminate the problem of estimating reading speed altogether. However, this would likely result in an increased overall response time, potentially influencing experiment results. In addition, a lot of experimental settings, particularly in the mobile domain, need to deal with limited screen space where sending a collection of several utterances is often not suitable. Finally, from a more general perspective, it may happen that a test participant is not responding in which case a system (i.e. the wizard) has to initiate a check-up routine. Consistently simulating this time-window, which necessarily needs to vary with the amount of text a participant has to process, requires similar support. Hence, even though this issue is a very specific issue of simulating text-based interaction, it clearly shows a potential pitfall when employing the WOZ method; one that designers and researchers should know about in order to take the appropriate precautions when designing and conducting experiments.

To further reflect on this aspect we looked at speech-based interaction. As already pointed out earlier, in Study A one of the two tasks a test-participant had to complete was using syn-

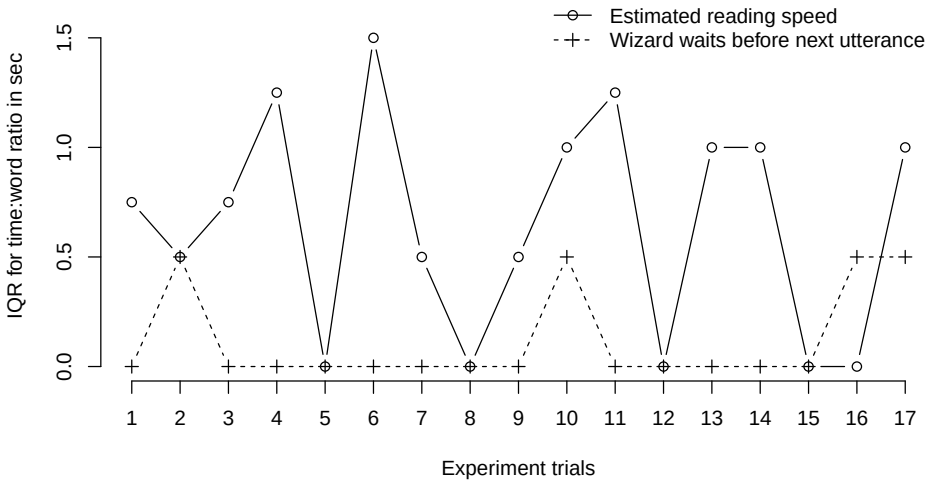

Figure 6.19 – A wizard's differences in consistency (IQR value) when dealing with speech-based utterances (continuous line) compared to dealing with text-based ones (dotted line).

thesised output. The wizard could hear the output and therefore was able to send the follow-up utterance as soon as the previous recording was finished, as opposed to estimating the time a participant needed to read it. We again calculated the IQR in order to contrast it with the values produced with the text-based outputs. Figure 6.19 compares the IQR values for Study A's speech-based output with the ones from the text-based mode. As can be seen from the results, the time/word consistency in speech-based interaction is significantly better than it is in its text-based counterpart (one-tailed paired Student's t-test: t(16)= 3.9105, p=0.0006). In fact, in 13 out of 17 trials a wizard's response time in relation to the preceding utterance can be seen as entirely consistent. The remaining 4 trials show a variation of only 0.5 seconds, which might be explained by additional time needed to search for a follow-up utterance.

Based on these results it seems that supporting the wizard with an additional audio (or possibly video) channel can be seen as an important awareness information, which we would recommend using by default, unless the experiment format is such that there is a concern it could lead to overly 'intelligent' behaviour of the simulated system. In cases where no audio/video output is part of the simulation, alternative functionalities, like for example timers, could be used in order to help the wizard estimate timing consistently.

With these results we concluded our analysis of wizard performance and consistency. Additional evaluations may be conducted as soon as other realistic WOZ experiment settings can be explored. The remainder of this chapter focuses on another set of evaluations that were conducted in order to look at the construction process of WOZ experiments and whether the editing features implemented with our prototyping platform would support researchers in designing their own studies.

6.5 Supporting Experiment Construction

While the previous evaluation rounds were mainly focusing on running WOZ experiments and the challenges a wizard is confronted with when acting under time pressure, this final set of evaluations looks at designing an experiment. Earlier, having a low overhead on experiment construction was identified as an important requirement for WOZ tool support. Therefore our prototyping platform introduced the possibility to quickly create and configure a new experiment as well as add, edit and delete dialogue utterances. Unrestricted interaction (if desired) is supported by the use of a chat-box, and generic filtering allows for quickly browsing different sorts of domain data. The purpose of integrating these features was to give wizards the opportunity to design WOZ experiments themselves without the need for any programming experience; similarly to Content Management Systems (CMS) regularly used for maintaining websites. Yet, while the purpose of a CMS system is usually the easy maintenance of content, it needs to be pointed out that the goal of our WOZ platform is more far-reaching. That is, we wanted to reduce the complexity of setting up and configuring an entire experiment environment. Features to that effect include the automatic generation of client (i.e. test-participant) interfaces as well as the easy configuration of integrated language components. Also, we considered all the steps of designing, running and analysing experiments and therefore included functions that would allow for the export of experiment data and the possibility to import experiment set-ups from files created by third party software programs. Hence, even though our tool is comparable to a CMS system for websites, we want to clearly highlight that its functionality addresses a problem that goes beyond the pure maintenance of content.

6.5.1 Evaluation Method

After having corrected some minor interface and stability issues with the platform, we started our evaluations with a proof of concept activity, looking at the coverage of the design space. The goal was to test whether each of the plausible use cases identified earlier (cf. Table 5.4) could be realised without adapting the tool.

In a next step we wanted to test whether also other people (i.e. potential wizards) would be able to design experiments with our platform. To do so we conducted two sets of studies. First we recruited what we considered expert wizards i.e. researchers from the NLP area and asked them to design two different WOZ experiments. Second, in order to test whether our tool would also allow for none-experts to create WOZ experiments, we ran another study with participants from outside the NLP domain.

6.5.2 Covering the Design Space

The goal of this task was to test whether we were able to implement a variety of different WOZ set-ups covering the main LTC configurations highlighted in Table 5.4. Instances integrating experiments with several component combinations were created, achieving one of the goals

identified earlier as being important for comprehensive WOZ tools support. While it was not our goal to actually run all of these experiments (i.e. we were mainly interested in whether we could build them), this task was crucial to evaluate the range of possible use cases our prototyping platform would be able to support. To do so previous literature was used as a source for realistic product ideas. Elaborating on these ideas we consequently built a total of 8 different WOZ experiments, all of which were using a combination of at least two different language technologies (i.e. ASR, MT, TTS). Both the experiment structures as well as the relevant text utterances were created. Following we describe the products/services for which this WOZ experiments were implemented in some more detail. It needs to be highlighted, however, that while all of them were fully implemented, only MYSPEECH and INTERNETADVISOR were eventually subject to experimentation, employing a setting where a wizard interacts with a number of different test-participants.

DialogueTranslator

DIALOGUETRANSLATOR constitutes a multi-modal translation system whose goal is to translates a person's spoken input on the fly from one language into another, allowing for a fluent dialogue. Technologies that are employed in the WOZ experiment set-up comprise Automatic Speech Recognition, Machine Translation as well as Text-to-Speech Synthesis. The tasks of the wizard for this scenario reaches from full interpretation of the input and consequently producing the output, to correcting the input by either directly changing the result before it is sent on or choosing from an N-best list of possible options. Corrections can be applied either to the recognition results before they are sent to the MT module or to the translation result before it is sent to the receiver.

MySpeech

MYSPEECH is a system that permits people to improve their pronunciation of a foreign language. The system records a language learner's input, analyses the pronunciation of the words and then gives appropriate feedback based on the analysis results. Whereas currently this feedback is mainly presented as a graph, a future version of the system will support a more textual notation. The WOZ method is used to test the user experience for this kind of descriptive feedback and to collect a preliminary corpus of possible feedback utterances. To do so a Speech-to-Text WOZ setting is applied in which the wizard interprets the analysis results and gives accompanying textual feedback. An external ASR system and the chat-feature of the prototyping platform are used. The setting is comparable to a chat scenario in which one party speaks and the other one answers via text-chat.

PhotoPal

PHOTOPAL is a system that incorporates an Embodied Conversational Agent (ECA) integrated with your photo library. It aims to add a social aspect to the process of sorting and organising

your photos by offering a multi-lingual 'Companion' who is interested in the memories and experiences you connect with those pictures. A future system will incorporate data from previous conversations as well as data from your online activities that can be linked back to your photos. Based on this information it tries to generate an active and entertaining dialogue. During the development stage WOZ is used to simulate the 'intelligence' of the system. The wizard uses a free-text field coupled with canned utterances to interact with a test participant. An animation and Text-to-Speech Synthesis are used to present the wizard's statements. The MT module is activated in order to support interactions in different languages.

WorldNews

WORLDNEWS is an RSS reader application for smart-phones that translates news into different languages and reads them aloud using Text-to-Speech technology. The system is especially useful in mobile settings where it works like a digital audio player. Simulating this functionality WOZ can be used to obtain early customer feedback. At the beginning the wizard acts as an interpreter, translating news items from one language into another. In a second stage the wizard's task shifts from translating to post-editing, initially correcting translations and later on choosing the most appropriate ones from an N-best list of options.

MultiTrans

MULTITRANS is an application that automatically transcribes spoken words into written text. In addition it supports cross-language transcription, which offers direct transcription into foreign languages. That is, a user can speak in one language and the software produces the equivalent text in another language. WOZ is used in different stages throughout the development of this application. First the wizard simulates speech recognition, later on machine translation. For both tasks the wizard either produces the entire output or augments results coming from the relevant technology components utilising the correction or N-best list feature.

MultiChat

MULTICHAT is a multi-lingual chat program that supports automatic translation of text into a variety of different languages. In addition it can be used to train language understanding by offering a sub-title feature including the relevant synthesised speech output. WOZ helps to obtain early user feedback. A wizard is used to manually correct MT output.

VoiceSearch

VOICESEARCH is a plug-in for web browsers that offers voice-based internet surfing. Modern speech recognition technology is used to translate speech into a search request. Moreover, the request is automatically translated into different languages in order to increase the amount of possible answers. During the development a human wizard is used to override ASR output

where it is needed, testing the level of recognition quality that is required in order to offer a satisfying user experience.

InternetAdvisor

INTERNETADVISOR is an interactive multi-lingual information terminal, which recommends appropriate Internet connections bundles to customers in their native language. As such it understands spoken input in different languages and gives back recommendations via text as well as voice output. Testing the scenario a bi-lingual wizard is used to choose appropriate responses from a set of prepared dialogue utterances.

Implementing those scenarios demonstrated that our current WOZ prototyping platform and its employed wizard interface allows for the experimentation with different combinations of technology components. Furthermore we have shown that different platforms reaching from computer terminals to smart phones and tablet devices are supported. Even though only two of the described settings were subject to real experimentation, we believe that the others are similarly realistic and could be explored without demanding additional integration work. Next we continue our analysis of the construction process by describing the results of an evaluation where other researchers were asked to build WOZ experiments with our prototyping platform.

6.5.3 Testing Expert Wizards

In order to further evaluate the experiment creation process of our WOZ platform we conducted a study with 10 researchers working with language technologies. None of them had participated in any of the earlier studies; five of them had experience with WOZ. They were given a short introduction to the prototyping platform and a written manual they could refer to (cf. Appendix D). Following registration with the platform they had to carry out two design tasks (cf. Appendix C). For the first task they were given a set of utterances used in a phone banking application. They had to add these utterances to a new experiment, arrange them in a useful way, and add any utterances they thought were missing. The second task was to design a pizza ordering system. This time participants were not given any utterances and therefore had to come up with their own designs. They were asked to complete a questionnaire demanding subjective ratings of task difficulty and task satisfaction after each of the two tasks, and a post-test questionnaire looking at the overall usability of the platform at the end of the test. The overall goal of this evaluation was to see whether participants would be able to add, edit and arrange utterances without having first participated in a real training session. Furthermore we were interested in whether they would make use of two genuine features of our platform, namely the tabbed dialogue structure and the dedicated area for frequently used utterances.

Success Rate and Complexity of Designs

Overall participants did not experience any particular difficulties in achieving the given tasks. All were able to complete the first task within the given time frame of 30 minutes and could present their design for the second another 30 minutes later. If we look at the produced dialogues in more detail we see that for the first task two participants did not create all of the 16 utterances they were given (i.e. P02 missed one utterance and P08 left out four) and one participant added to them (i.e. P07 created 22 instead of 16 utterances). The additions were caused by three single utterances that were each split into two separate ones, one additional question, one additional confirmation utterance and one additional advice utterance.

For the second task we see that participants used the previous example as a guideline. On average they produced roughly the same number of utterances (mean=16.20, median=15) and an average of 11.51 (median=10.50) words per utterance in Task 1 compared to 10.74 (median=10.00) words per utterance in Task 2 lets furthermore suggests that the produced dialogues exhibit about the same level of complexity. However, as could be expected from the more individualistic task setting, the variance between test participants was significantly higher in the second task (Task 1: SD of number of on average produced utterances=2.60, SD of number of on average produced words per utterance=1.12; Task 2: SD of number of on average produced utterances=6.97, SD of number of on average produced words per utterance=3.64). They had to create their own designs, for which some of them showed a better performance and more creativity than others. Overall, however, the results of the study show that our 'experienced' participants were capable of handling both tasks successfully without additional upfront training.

General Feedback

In order to obtain some general wizard feedback we used post-task questionnaires to measure task difficulty as well as task satisfaction (cf. Appendix C). For the first a 7 point Likert scale running from 1 *very difficult* to 7 *very easy* was employed. The results show that the tasks were perceived as easy to complete leading to means of 6.2 for both the first (mode=6, median=6, SD=0.4) as well as for the second task (mode=7, median=6.5, SD=0.9). To measure task satisfaction we used three questions (i.e. 1. Satisfied with the easiness of completion, 2. Satisfied with the amount of time it took, and 3. Satisfied with the supporting information) and again employed a 7 point Likert scale, this time running from 1 *very satisfied* to 7 *not satisfied at all*. Also here the feedback was overall positive with means of 2.3/2.5/2.3 (mode=3|2|1, median=2.5|2|2.5, SD=0.8|1|1.5) for the first and 1.8|2.1|2.1 (mode=2|2|1, median=2|2|2, SD=0.4|0.9|1.1) for the second task.

In addition we asked participants about their interactions with the system and what types of features were missing. Looking at those results we see that at least some of them (i.e. 2 out of 10) would have liked a more direct way to connect utterances and hence enforce a sequential order. Most of them, however, liked the freedom of independent utterances. Additional features

that were recommended include drag-and-drop support to arrange utterances as well as instant feedback on when utterances are successfully stored in the system. Finally, an import function to add utterances from third party applications and more intuitive labels to reduce ambiguity were requested. Overall, however, the results of the evaluation indicate that participants liked our platform and that they were able to use it easily without having received extensive preceding training. The System Usability Scale (SUS) score (cf. Brooke [1996]) of 77/100 produced by the post-test usability questionnaire (cf. Appendix C) seems to confirm these results (Note: A SUS score of 68 and beyond should be considered as above average[4]).

Use of Features

In addition to obtaining subjective feedback about the usability of our tool based on questionnaires we were also interested in whether participants would intuitively use some of the features we had integrated. One aspect here was the possibility to create a structured dialogue, i.e. using tabs to organize a dialogue in different stages. Looking at the log-files we found rather strong differences in people's preferences on that matter. While on average a participant created 3.71 (median=4) tabs per task the results highlight a spread going from 1 tab only to creating 8 (SD=2.63). A one-tailed paired Student's t-test showed no significant difference between the two tasks (one participant was excluded for this analysis as he did not complete the second task); t(8)=-0.9578, p=0.8169. However, it seemed that people structured the dialogue more in the second task (median=3 for the first task vs. median=4 for the second task) when they had to come up with their own selection of utterances, i.e. when they had more freedom to design the interaction. As participants were not asked to actually run any experiments with their created dialogues, we are unable to tell whether they would have performed some fine-tuning after conducting a number of trial-runs, however we believe over time the number of tabs would have harmonized.

A second aspect we were interested in was the use of frequently used utterances. The use of this feature shows us whether participants actually differentiate between utterances that are dedicated to a specific dialogue stage and those that could be used several times throughout an interaction. Also here the log-file revealed a rather varied behaviour. On average participants defined 4.86 utterances as frequently used (median=3) but an overall range of up to 28 frequently used utterances points to rather great personal differences. When asked about their understanding of the concept participants were in favour of the idea. However, we believe that without actually running experiments it was difficult for them to identify those utterances that are unlikely to be associated with only a single dialogue stage, which probably influenced the use of this feature. Furthermore, it needs to be highlighted that the limited amount of time (i.e. 30 minutes for designing a complete dialogue) most likely reduced the total number of utterances created. A more intense engagement with the design process would probably have resulted in more complex solutions, and consequently led to the inclusion of utterances that are

[4]http://www.measuringusability.com/sus.php [Accessed: March 11^{th} 2013]

less obvious.

Nevertheless, this first round of testing the creation stage showed that potential wizard users were able to successfully design experiments with our WOZ platform. They used different dialogue steps and understood the concept of frequently used utterances. We believe that actually running experiments would probably lead to various adaptations both in the structure of the dialogues as well as the number of utterances used. However, a test scenario where both the design as well as the conduct of an experiment is evaluated, would require significantly more time from our participants. In addition, as in such an experiment design they would act as wizards, we would need to provide them with (dummy) test participants, which seems not only due to the number of necessary people but also due to the lack of control we would have over their actions, unrealistic to achieve. Furthermore, design adaptation and an increased usage of tabs and frequently used utterances might only be observed over time, which would require an even greater number of test runs. Such an exhaustive evaluation is unfortunately out of the scope of this research agenda and hence we decided to keep design phase and experiment phase separated and rather increase the diversity of people that might be confronted with WOZ experimentation. Therefore, while for this evaluation we had mainly recruited potential wizards who are familiar with the WOZ method, we conducted another round of tests for which the goal was to expand the potential user base into other areas.

6.5.4 Testing Naive Wizards

While at least some researchers in NLP and HCI might be familiar with WOZ prototyping (cf. Section 5.1), people working in other fields are rarely exposed to the method. However, as described earlier, various application areas can benefit from human simulation, and especially with the advent of language technology being used in devices of everyday use we require distinct contributions from different disciplines to offer intuitive and novel design solutions. Hence, one goal of our platform is to reduce this entry barrier and make WOZ prototyping accessible to people outside the field of NLP research.

In order to test whether our approach of integrating CMS-like features into a WOZ platform makes it easy enough to be used by novices, we conducted tests with 51 student participants. Students' backgrounds included Computer Science, Information Systems as well as HCI. No NLP people were recruited this time. Participants were confronted with the same two task as their predecessors and also completed the same questionnaires evaluating task difficulty, task satisfaction and tool usability. In addition to obtaining feedback from naive wizards this was also a performance test for our WOZ platform as we were running several sessions in parallel (i.e. the study consists of three on-line sessions all of which were dealing with 8 o 19 participants at the same time).

Even though one could argue that using student participants might lead to somewhat distorted study results, as they may feel the need for responding rather positively, it can nonetheless be seen as a generally accepted procedure, performed by a number of experimental studies

presented in the literature (e.g Dahlbäck and Jönsson [1989], Hill [1993], Liu and Khooshabeh [2003], Dow et al. [2010], etc.). In order to further reduce potential side-effects we chose students from external institutions and from courses we were not actively involved in. Furthermore we did not have any authority over participants, as we were only coming to class for this one study, and where not involved in the evaluations of any course works or exams. Despite these precautions we are, however, aware that a potential for 'too positive' answers still remains and therefore needs to be taken into account. Nevertheless, given that we were not looking for a strict 1:1 comparison between expert wizards and naive ones but rather sought to obtain initial feedback, we believe that the following results can be seen as a valid indicator for judging the quality of our approach to making WOZ prototyping more accessible.

Success Rate and Complexity of Results

Looking at the produced results we had to exclude 12 of the 51 participants as they either did not given us the necessary consent to use their data or did not submit any designs. From the remaining 39 participants, 28 were able to finish the first task and work on second within the given time-frame. If we compare the results to what the 'expert' wizards had achieved in the previous study, we see that in terms of the complexity of the produced designs the two groups of wizards are fairly close in their performance. In the first task our 39 students produced on average 12.36 utterances (median=15.00, SD=6.47) with an average words/utterance rate of 10.36 (median=10.00). The 28 students who managed to work on the second task additionally created on average 14.39 utterances (median=12.50, SD=7.93) with an average words/utterance rate of 10.85 (median=9.00). While this shows that the complexity of designs varies more with non-experts as it does with more experienced wizards the difference, especially in the second task, is unexpectedly low. Figure 6.20 and 6.21 show how the two groups perform in comparison.

We therefore reason that, while there is still a certain entry barrier to overcome (some participants did not submit any design solutions), non-expert wizards are generally able to use our tool to design WOZ experiments for natural language-based human-machine dialogues. They experience slightly more problems and they also might require more time, but their final results can be compared to designs produced by more experienced NLP researchers.

General Feedback and Features

Looking at the subjective feedback we received from the student participants we see an increase in perceived task difficulty for the first (mean=4.3, mode=6, median=4, SD=1.7) as well as for the second task (mean=4.8, mode=6, median=5, SD=1.6). Also, people were significantly less satisfied with them completing the tasks (Task 1: mean=3.5, mode=3, median=3, SD=1.7; Task 2: mean=2.8, mode=2, median=2, SD=1.6), the time it took them to do so (Task 1: mean=3.1, mode=2, median=2, SD=1.7; Task 2: mean=3.1, mode=2, median=2.5, SD=1.8) as well as the support information they were provided with (Task 1: mean=4, mode=3, median=4,

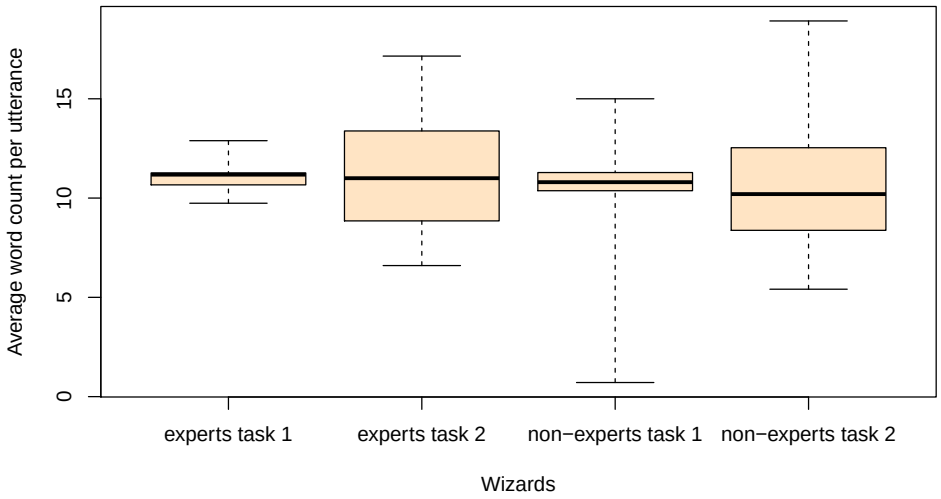

Figure 6.20 – Comparing experts and non-experts on the complexity (average words per utterance) of created utterances.

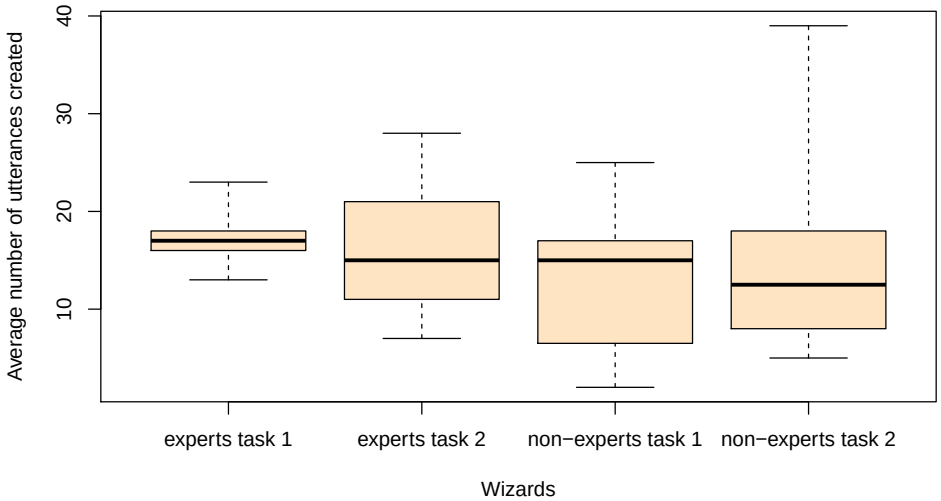

Figure 6.21 – Comparing experts and non-experts on the number of created utterances.

SD=1.9; Task 2: mean=3.6, mode=5, median=3, SD=1.7). The same result is reflected by the questionnaire assessing the overall tool usability which showed an average SUS score of 58/100.

While from these figures we can see that subjectively those people had more difficulties with the tasks and our WOZ platform than the more WOZ aware researchers we tested earlier, the log files of the experiment also show some positive aspects. That is, the majority of the participants (28 out of 51) were able to fully complete the first task, and consequently started working on the second, more open-ended one. They produced 401 utterances (mean=13, median=12, SD=8) and made use of tabs and frequently used utterances, even though compared to NLP researchers they structured the dialogues significantly less (mean=1.60 tabs, median=1 tab) and defined less utterances as frequent (mean=1.49 utterances, median=1 utterance).

As for the lower satisfaction levels it seems reasonable that being confronted with a novel task (none of the participants had any previous experience in designing a natural-language based human-system dialogue) without first having received adequate training would always result in lower scores. A different outcome on this subject would be rather surprising and even point to a faulty experiment design as it seems unrealistic that novices would solve these tasks with the same easiness as researchers who are generally more accustomed to the WOZ method. While one could argue that participants should have received sufficient training before being tested, it seems unrealistic to achieve a comparable level of expertise within the time frame offered by the given user studies. Even if we had doubled the time for a session it would have been unlikely that researchers would have felt significantly different about their satisfactory level, given that it would always have resulted in them solving a rather unfamiliar type of task. Furthermore, the main goal of this study was to see the difference between novices and more experienced wizards and consequently to evaluate whether appropriate tool support would be able to narrow this gap. The results suggest that our WOZ platform enabled non-expert wizards to design WOZ experiments. They might not have been satisfied with the result and better support in terms of documentation and tool tweaking may be necessary, but nevertheless to a great extent they were able to solve their given tasks.

Looking at the feedback given with respect to the tool we find similarities to what was already highlighted earlier. That is, participants would like to have better ways of connecting utterances as well as better integration with other applications and prototyping tools. A challenge was also found in the terminology used to label buttons and describe functions. While with more experienced participants this is mainly a matter of resolving ambiguity, non-experts require additional explanations as they are simply not familiar with the employed language. Here one could argue that the language to be used for labelling and explaining system functions needs to reflect the language of the user. However, in cases where one seeks to expand the potential user group, such is often difficult to achieve. For example, while the term ‘utterance’ is common in NLP and probably the best choice for naming a language fragment in a human-system dialogue, users without a certain amount of linguistic background might not be familiar with this description. In this case it is recommended that the tool documentation

should lower the entry-barrier and provide the relevant information as well as a directory of used terms. Additionally, help texts should be shown in order to give direct feedback about system functions. Both arrangements had, at least to some extent, been implemented with our WOZ platform and its documentation before running the test. However, additional tests should have been conducted as to evaluate their effectiveness.

Finally, a feature that was mainly requested by novice wizards can be found in an interactive walk-through tutorial. These integrated step-by-step instructions seem especially helpful at the beginning, as they not only introduce users to the employed terms but also explain the basic concepts of an application. Several of our participants mentioned that they did not know how to start their tasks and at the beginning only found their way through trial-and error. Even though research tells us that the majority of people, especially non-experts, prefer to learn by exploration rather than by instructions (cf. Trudel and Payne [1995]), a short tutorial or at least an optional introduction video can be useful and often provides the necessary set of basic information for somebody to get started. Nevertheless, even without interactive help our novice wizards were able to find their way around and could solve, at least to some extend, the required tasks.

Hence, in summary the results of this evaluation show that generally the design of WOZ experiments does require a certain amount of knowledge and understanding of the method, however, even without dedicated training people were able to produce workable solutions. Easy access to relevant tools such as our WOZ platform can therefore be seen as an enabler which, given better documentation and possibly some upfront training, could potentially be used to open up the design space of WOZ prototyping to people outside the field of language technologies.

6.6 Summary

This chapter discussed two important aspects of WOZ tool support. Firstly, we reported on a series of template experiments which illustrated that using our WOZ prototyping platform we can achieve good coverage of the design space for language-based WOZ experiments. As part of this we also described a series of evaluations that were conducted alongside the implementation of this platform. The fact that real experiments were successfully carried out using the tool is a significant validation of the approach. These evaluations also allowed us to conduct an analysis of wizard workload and a study that investigated wizard performance over time. The results of these two study formats show that wizard performance varies over time where more experience leads to faster response times but also to a certain humanisation of a wizard's actions. Consistency, however, is difficult to achieve even for experienced wizards, for which relevant tool support should be provided. These findings complemented previously identified requirements for WOZ tool support and consequently helped shape the design of our generic wizard interface.

The second main aspect of this chapter related an investigation of the construction process

of WOZ experiments. We have analysed whether both experienced as well as naive wizards manage to build realistic WOZ experiments with our tool, and whether more experience automatically leads to better, mores sophisticated design solutions. The results of this analysis show that our platform enables both types of users to design WOZ experiments without having received any upfront training. Our findings furthermore suggest that the difference between those wizards who have experience with natural language processing and the WOZ method and those who are new to this domain is unexpectedly small, and that both types of users produce similar dialogue designs if they are given a tool with a low entry-barrier. In summary, we thus believe that such a WOZ prototyping platform can make a contribution towards increasing the potential user base for WOZ as a low-fidelity prototyping and design method for human-machine dialogues.

Chapter 7

Conclusion

"If you had known their power you could have gone back to your Aunt Em the very first day you came to this country."
– Glinda the Good Witch of the South

WOZ is a valuable prototyping method for designing interactive technologies. Particularly for systems that involve some sort of natural language processing, the kind of feedback that can be gained helps to shape design. However, the dependency on a human wizard makes the method susceptible to errors.

In order to explore WOZ prototyping and some of its challenges this thesis employed a Systems Development Methodology expanded by methods from User-Centred Design. This combinatory approach permitted us to evaluate the technical aspects of our work and drive the engineering process as well as to generate new knowledge with respect to the employment of our tool and the use of its features in different experimental settings. Different methods generated different types of results.

First, an analysis of previous work produced a list of recommendations for conducting WOZ experiments and identified a number of basic requirements for tool support. Building on that an interview study highlighted additional methodological challenges before an analysis of the wizard task and its different roles informed the design of a more generic tool architecture. Finally, a prototypical instantiation of this architecture provided validation of the approach, and generated insight into technological restrictions. Throughout this process a set of evaluations was used to expand the body of knowledge on WOZ prototyping, specifically on the wizard task and its accompanying challenges of planning, designing and running WOZ studies.

The overall goal of this thesis was to better understand WOZ from a wizard's point of view. We analysed people's motivation for using the method and looked at the sort of problems they experience when doing so. In addition to that we looked at WOZ tool support an how it can be improved. Previous work has shown that existing WOZ tools are often built for specific experiment settings and rarely re-used. Hence our goal was to construct a generic

WOZ prototyping platform that can be used to create and conduct experiments in a variety of settings. The realisation of this platform was achieved in close accordance with what we could learn from the literature and from talking to researchers involved in WOZ experimentation. In addition we took into account the wizard task and the various forms it can take on. After having first explored a single scenario our initial tool prototype was expanded to establish coverage of the design space for language-based WOZ experimentation. Analysing our platform we showed that potential wizards were able to quickly and successfully build experiments without extensive training. A set of additional evaluations conducted with this platform generated usage data from motivated wizards performing real experiments. Those not only provided initial validation of our approach to WOZ prototyping and the architecture underlying our platform, but also generate new insights into aspects of the wizard task. As well as informing system development, they have permitted us to explore WOZ tool support at a more general level, and have highlighted different challenges such as workload and consistency problems. In the following we summarise the results of this thesis and discuss future work.

7.1 Wizard of Oz Prototyping

The initial part of this thesis was mainly concerned with building up a general understanding of the field of WOZ prototyping, both from an experimental point of view as well as from motivational perspective. Starting with an in-depth literature analysis we provided a structured overview on WOZ prototyping and the type of application scenarios it had been used for in the past. The history of the method was discussed and the motivations of its initial users highlighted.

It was also shown how gradually researchers/designers have adopted WOZ and adjusted it to fit their very specific evaluation interests. Previously conducted WOZ studies in a range of categories were considered, with the aim of identifying various differences between application fields, both from a methodological perspective as well as from a motivational one. It was highlighted that in some cases, such as in the majority of NLP related studies, human simulation is used as a quantitative evaluation method, where a structured and highly controlled experiment setting is required in order to obtain valid results. Statistical significance in these use cases is important as usually various machine learning algorithms will be based on the interaction data. However, we have also discussed examples of a different class of research, which increasingly focuses on user experience and other social aspects of human-machine interaction, such as acceptability and user preference. In theses cases WOZ is predominantly employed as a qualitative research instrument whose goal is to explore potential new interaction spaces and initiate discussions. Both classes of research generate different demands on WOZ tool support and our goal was to create a framework for a range of possible solutions.

While the literature analysis tried to consider the entire field of WOZ prototyping, the rest of the thesis has mainly focused on language technology applications. This decision was supported by the fact that the simulation of language-based interaction remains the major ap-

plication area for WOZ. An analysis of papers published in the ACM as well as the IEEE digital library showed that roughly two thirds of the work on WOZ is related to the field of language technology. Our analysis furthermore showed that emerging areas such as multi-modal interaction or human-robot interaction usually feature some form of language modality, and that this aspect of a study often requires the greatest amount of simulation as existing language technology is simply not robust or versatile enough to be used without extensive tuning and optimisation efforts. Despite the language-based focus of this work we, however, strongly believe that the work presented here, which was gained from both the literature and the evaluations, is generally applicable to the entire field of WOZ prototyping.

7.2 Wizard of Oz Tool Support

A strong motivation for this thesis was a lack of generic WOZ tool support. After reviewing a set of available tools it was found that none of the obvious applications would be suitable for our initial study design described in section 6.2. A significant part of the already mentioned literature study was therefore concerned with the identification of potential software programs that would allow for the creation and conduct of this type of WOZ experiment. As such the research involved the analysis and comparison of existing prototyping tools, looking at their features and assessing their support for different technology components. In doing so it was highlighted that existing tools provide insufficient support for this form of prototyping and that a better understanding of the method and more flexibility is needed in order to allow for a more widespread application. Based on this insight a concept for better tool support was provided and a possible implementation in form of a web-based WOZ prototyping platform was created.

Following a systematic analysis of the design space and its influence on the wizard task, we presented several requirements which seem crucial to improve future tool support. These included functional requirements such as the possibility of conducting highly controlled as well as exploratory WOZ experiments, and non-functional requirements such as an intuitive wizard interface. Also, a tool architecture was described that would allow for a flexible combination of technology components and support the different roles a wizard may take on as part of an experiment.

Building upon this analysis we used a Systems Development Methodology expanded by methods from User-Centred Design to build a novel WOZ prototyping platform. The development process was user-driven and iteratively evaluated. The resulting prototyping platform differs from existing WOZ tools in its support for different types of application scenarios that go beyond the frontiers of a single experiment. To achieve this we integrated several features enabling the platform to adapt to various experimental settings and wizard tasks.

First, to reduce the overhead in experiment construction and software installation as well as to increase the flexibility in terms of supporting both highly structured and more exploratory experimentation, we implemented a web-based wizard interface offering CMS-like editing functionalities. Second, a flexible integration of language technologies via web-services has been

achieved and it was further outlined how this architecture would support a possible augmentation role of the wizard. Third, several data logging and export mechanisms were integrated supporting the analysis of experiment results.

The wizard interface for this WOZ prototyping platform was subject to optimisation throughout the whole development process. Feedback generated by small-scale usability tests as well as data gathered from a series of evaluations focusing on wizard behaviour over time, led to various changes of interface elements and to the integration of new features. The final design therefore reflects our efforts of creating a generic tool interface that would support a variety of different wizard roles without introducing unnecessary complexity.

Even though the main application area for the constructed WOZ platform lies in language-based WOZ experimentation it needs to be highlighted that its modular nature may allow for dealing with various types of interaction (given that the relevant modules are built), and its web-based implementation may be used to run experiments on a variety of different devices with different form factors ranging from traditional computers to smart-phones, tablets and other forms of internet-enabled smart devices.

7.3 The Task of the Wizard

Another strong motivation of this thesis was to specifically investigate the task of the wizard and the challenges a person in this role is facing. In doing so the goal was on the one hand to identify supportive factors that could be integrated into our WOZ prototyping platform and on the other hand to increase our understanding of WOZ as an evaluation method. In terms of the former, user tests were conducted where wizards were observed creating as well as running WOZ experiments. Small-scale usability studies with 3-5 participating wizards, each of which was running one experiment, were used to evaluate the intuitiveness of this task. Experienced difficulties influenced the design of the wizard interface and led to features such as the tabbed dialogue structure or the area for frequently used utterances.

User studies were also conducted to investigate the task of constructing an experiment. Recruited participants included both, more advanced wizards and non-experts in the field of WOZ prototyping. The results of these evaluations suggest that the CMS-like editing features that were integrated into our WOZ prototyping platform are workable and easy to learn. For example, it was shown that even non-expert wizards can construct WOZ experiments quickly, which is critical if a technique comparable to ‘sketching’ is to be supported. Furthermore it was demonstrated that with this platform a comprehensive coverage of the design space for language-based WOZ experimentation can be achieved; supporting the complete list of scenarios outlined in Table 5.4.

Finally, three different studies investigated wizards over the span of several experiment sessions. The goal was to identify performance variations and their possible causes as well as behavioural changes and their potential motivations. The results of these evaluations suggest that wizard behaviour can be influenced in several different ways. On the one hand through

learning effects where, despite increasing a wizard's performance, familiarisation may lead to undesirable human-like behaviour. On the other hand, through insufficient awareness, which was identified as a cause for inconsistent wizard actions. We presented results that show that additional acoustical information provided by an audio (or possibly video) channel can improve a wizard's performance. However, we also highlight that such support might lead to overly 'intelligent' system behaviour in which case alternative, more specific solutions may sometimes be preferable (e.g. a timer function). Looking more closely at inconsistent wizard performance we also presented a meta-study analysing a wizard's ability of estimating a test participant's reading speed. Results suggest that it is difficult to keep this timing consistent.

In summary, this thesis has shown that more generic support for WOZ experimentation is possible. Some aspects of the wizard task, such as varying wizard behaviour, may, however, remain challenging even if appropriate tool support is provided. A combination of wizard training and awareness features may help control some of them. However, researchers/designers need to know about these challenges so that they can take the appropriate precautions when designing, conducting and running experiments. The presented results also suggest that WOZ support needs to be flexible. Tools should allow for a variety of different settings, while at the same time help the wizard control as many confounding variables as possible. A useful combination of flexibility and control can be seen as the key challenges for designing these WOZ tools whereupon more research looking at wizards at work, such as the one presented here, may help in achieving the right balance.

7.4 Challenges and Future Work

Focusing on the wizard, the research described in this thesis has investigated the use of a prototyping method from a researchers/designers perspective. A main challenge of this approach was its dependency on researchers and their specific interests in answering dedicated research questions. In order to explore 'real' WOZ experimentation it was important to identify interested parties who had an intrinsic motivation for using the WOZ method over the period of an entire experimental study (i.e. in this case 11, 13 and 17 experiment sessions), and also agreed to being observed and analysed themselves. Through this it was possible to explore wizard behaviour over time and identify effects of task habituation. However, while the conducted long-term evaluations constituted an ideal setting for evaluating the WOZ method, they also posed significant challenges. In particular, the requirement for realistic research interests from third parties, is logistically difficult to meet.

This brings up the question of how design and research methods and their tools are most efficiently evaluated. The research presented in this thesis aimed to produce valid results by mixing different methods of exploration. Being conducted alongside the development of a tool, experimental evaluations were supplemented by small-scale usability tests and the preceding literature review was augmented by results from an interview study. While the resulting WOZ

prototyping platform reflects these findings, it needs to be highlighted that alternative research approaches might have led to similar insight. For example, instead of conducting new experimental evaluations, log-files from previous WOZ experiments could have been analysed. Also, dedicated experiments comparing different aspects of tool support could have been conducted in order to better quantify the effectiveness of certain features.

However, while we take potential criticism of our chosen research approach seriously, we believe that the presented work does provide a valuable contribution and new insight to the domain of WOZ prototyping, and so we plan to continue this route of exploration. In doing so, future work will further look at the performance of wizards over time and how this might lead to additional requirements with regards to customizing and adjusting our WOZ prototyping platform. Furthermore we want to expand the variety of experiment settings we can support. A first step in this direction has already been undertaken by preparing the tool to be used by a different group in a European research project. Some interface adaptations were necessary to support the distinct requirement of this new setting, which has provided us with an additional use case to be integrated into the existing tool architecture.

Future work will also more deeply examine consistency and performance issues of the wizard task and how they can be addressed as well as explore the use of the WOZ platform for HCI teaching. As part of this we plan to evaluate some features that are already integrated but have not undergone any assessment so far (e.g. notes and instructions). Investigating these aspects will require an extensive program of experimentation with multiple wizards running their own experiments, and as part of this we plan to make the system available to the wider HCI community, for both teaching and research purposes. As a first step the current version of the platform has been published under the Apache License (Version 2.0) and made available for download[1]. From a technical perspective an extension of the covered design space towards an inclusion of multi-modal aspects is planned. Also here first steps have already been undertaken by integrating video output. With some creativity, we believe, a number of richer multi-modal scenarios could be explored using this tool. Likewise, the advent of mobile web browsers opens up a number of interesting possibilities, particularly in the context of Speech-to-Speech translation (cf. Stüker et al. [2006]).

Another interesting direction for further exploration is the provision of feedback to wizards on their own consistency, along metrics such as response time and spread of utterances used. This might also play a role in training and piloting, permitting the wizard to decide at what point she has practised the dialogue enough and is ready to commence the full experiment. Also, different aspects of Dialogue Management (DM) may be explored. So far we have treated DM as an integrated wizard task. However, its different components such as language understanding, output generation, and information retrieval constitute a set of additional perspectives that are worthy of substantial further exploration. Finally, more effort needs to go into investigating how wizards best handle domain data. In our experiments this data was relatively small and easy to navigate. Focused studies are required to explore how more complex data structures

[1] Download link: https://github.com/stephanschloegl/WebWOZ

influence a wizard's workload and how this might impact the outcome of an experiment.

In summary, future work will continue to investigate WOZ from a wizard's perspective. It will build on the findings of this research an expand them through additional experiments and user studies that analyse wizards in various settings, handling different tasks, and using different interaction modalities. Special attention will also be given to improving the presented WOZ prototyping platform. To do so the system has been installed in a new environment in the author's current institute where it will be used in at least two different research projects. In a next step it is planned to adapt the tool so that it supports additional experimental settings and application scenarios. Although we are aware of the fact that the great difference between the interests of individual researchers may eventually impede the design of a truly generic WOZ tool, we strongly believe that our platform can be a helpful starting point for a variety of researchers/designers who wish to use WOZ as an evaluation method.

Appendix A

Talking to Experts: Interview Guide

Study-related Questions

- What were the challenges when running your WOZ study?
- How much time did you spent preparing the study?
- Did you conduct any trial runs?
- How did you treat the results of the study?

Wizard-related Questions

- Who was acting as the wizard?
- How were/was the wizard(s) recruited?
- What was the wizard(s) background?
- Did you conduct any wizard training?
- How high was the wizard's work load?
- Did you measure wizard consistency?
- Did you encounter any problems related to the wizard task?

General Questions

- What advice can you give to people using WOZ

Appendix B

Running Experiments: Test Script

The Test Scenario

In this test you are asked to act as an intelligent computer system that is supposed to understand spoken input and give back text-based responses. More concrete, you are acting as a salesperson that is interacting with another person, your customer, through a web-based interface. The customer does not know that she interacts with a human, but thinks that a computer system is the actual dialogue partner she is talking to. In order to make this look real you are asked to use a restricted language. The interface will provide you with text-based responses to choose from. You just press the response button of one of those responses and the customer will see it on her screen.

One specific setting of this test is, that the customer is interacting with you in a different language then English. Therefore, the English responses you choose will be translated into another language (in this case German) before they are displayed on the customer's screen. For you, however, the responses are always shown in English. You can see what is displayed to the customer by looking at the text box in the top left area of the interface.

The Customer's Goal

The customer wants to get information on which internet connection she should choose wherefore she contacts a computer system that is supposed to understand her spoken natural language input and displays text-based responses on the screen. She starts the dialogue by clicking a start button, which consequently initializes a greeting by the system. You can see this greeting too, which informs you that the customer has started the dialogue and therefore will soon star talking to you (=the system).

Your Task

Your task is to act like a computer system that helps the customer to choose from different offers. In order do so several issues need to be clarified: First, after changing some settings (cf.

The wizard's task step-by-step), you need to find out whether the customer wants a land-line (=DSL) or a mobile internet connection. The next thing to clarify is whether she would like to sign up for a contract or prefers a pay-as-you-go option (=no contract). Here it is important to know that in this test a contract has a minimum length of 12 month. Thus, if a customer stays less than 12 month in Ireland, pay-as-you go is the option she should go for since there is no compulsory commitment.

As soon as those main aspects (connection and contract type) are specified, requirements like download-speed (=DS) and data-allowance (=DA) might be important. The interface has a filter functionality that can be used to sort possible offers by download-speed (=DS in MBit) and data-allowance (=DA in MByte). Results from which you can choose are then always ordered by price putting the cheapest first.

Your Task Step-by-Step

1. Login to the system using the user and password provided.
2. Set the language to Systran translation.
3. Start interacting with the test subject by using the respond buttons. Note that you can mostly use the responses from top to bottom. This will help and guide you throughout the dialog. Sometimes, however, you need to choose between different responses! Also, in order to mimic a spoken language system you should confirm what the customer has said. You will find appropriate responses.
4. The interface also includes clarification questions in case some aspects of the ongoing dialogue are not clear.
5. Finally recommend a suitable internet connection and say good bye.

Important

This test is testing WebWOZ – a web-bases application. It is not testing you or your abilities to act as a computer system. If something is not working or at some point you do not know what to do, it is not your fault but rather a problem with the interface. Tell us what is wrong or why you have difficulties in a given situation, so that we can fix this and thus constantly improve the usability of the software.

Also, you are free to withdraw from this test at any time without giving a reason. If you have any questions please feel free to ask us.

Appendix C

Constructing Experiments: Tasks and Questions

Task 1

You are asked to design a Wizard of Oz experiment in order to test the applicability of a novel speech controlled banking application. To do so you first need to come up with an initial dialogue model i.e. you need to think about the human-computer dialogue and design the required system utterances (i.e. all the system prompts) for the experiment, which are then used by a wizard to interact with a test participant. Maybe think about your Internet banking application as an example for possible functionalities. There is no need to implement all of the functions but in essence your application should support two different scenarios. On the one hand it should permit new customers to ask general questions, open an account, and forward them to a bank representative. On the other hand it should offer basic functionalities to existing customers. Examples here include checking your balance and transferring money.

Some aspects you should keep in mind when designing the dialogue:

- A wizard needs to be able to confirm input coming from a test participant, so offer appropriate confirmation utterances.
- A wizard needs to be able to inform a test participant in case more time is needed to obtain a demanded piece of information. Thus integrate utterances like for example "Sorry, but I need some time to retrieve this information".
- Try to support the wizard in being consistent. Aspects that you should think of include dialogue structure and the wording of utterances.

Task 2

Your previous Wizard of Oz experiment was successful and so the company hired you for a follow-up project. This time they want you to be even more creative. The application for

which you need to design a dialogue is one that permits you to order pizza. As a little help you find an example for a pizza menu on the next page. However, feel free to come up with your very own creation!

Some aspects you should keep in mind when designing the dialogue:

- A wizard needs to be able to confirm input coming from a test participant, so offer appropriate confirmation utterances.
- A wizard needs to be able to inform a test participant in case more time is needed to obtain a demanded piece of information. Thus integrate utterances like for example "Sorry, but I need some time to retrieve this information".
- Try to support the wizard in being consistent. Aspects that you should think of include dialogue structure and the wording of utterances.

Questions after each Task

Please answer the following questions regarding the task you just completed:

- Overall, I am satisfied with the ease of completing this task.
  Strongly agree ◯ ◯ ◯ ◯ ◯ ◯ ◯ ◯ Strongly disagree
- Overall, I am satisfied with the amount of time it took to complete this task.
  Strongly agree ◯ ◯ ◯ ◯ ◯ ◯ ◯ ◯ Strongly disagree
- Overall, I am satisfied with the provided support information when completing this task.
  Strongly agree ◯ ◯ ◯ ◯ ◯ ◯ ◯ ◯ Strongly disagree
- Overall, this task was?
  Very difficult ◯ ◯ ◯ ◯ ◯ ◯ ◯ ◯ Very easy

Questions after the Experiment

Please answer the following questions regarding the system (WebWOZ) you just used:

Questions regarding the tool

- I think that I would like to use this system frequently.
  Strongly agree ◯ ◯ ◯ ◯ ◯ ◯ ◯ ◯ Strongly disagree
- I found the system unnecessarily complex.
  Strongly agree ◯ ◯ ◯ ◯ ◯ ◯ ◯ ◯ Strongly disagree
- I thought the system was easy to use.
  Strongly agree ◯ ◯ ◯ ◯ ◯ ◯ ◯ ◯ Strongly disagree

- I think that I would need the support of a technical person to be able to use this system.
  Strongly agree ◯ ◯ ◯ ◯ ◯ ◯ ◯ ◯ Strongly disagree
- I found the various functions in this system were well integrated.
  Strongly agree ◯ ◯ ◯ ◯ ◯ ◯ ◯ ◯ Strongly disagree
- I thought there was too much inconsistency in this system.
  Strongly agree ◯ ◯ ◯ ◯ ◯ ◯ ◯ ◯ Strongly disagree
- I would imagine that most people would learn to use this system very quickly.
  Strongly agree ◯ ◯ ◯ ◯ ◯ ◯ ◯ ◯ Strongly disagree
- I found the system very cumbersome to use.
  Strongly agree ◯ ◯ ◯ ◯ ◯ ◯ ◯ ◯ Strongly disagree
- I felt very confident using the system.
  Strongly agree ◯ ◯ ◯ ◯ ◯ ◯ ◯ ◯ Strongly disagree
- I needed to learn a lot of things before I could get going with this system.
  Strongly agree ◯ ◯ ◯ ◯ ◯ ◯ ◯ ◯ Strongly disagree

Comments

- These are problems I had using WebWOZ:
- These are features I would like to have added to WebWOZ:
- Here are some final recommendations:

What did I learn today?

- What is Wizard of Oz?
- For which typical application areas would you use Wizard of Oz?
- What are the benefits of Wizard of Oz?
- What are the limitations of Wizard of Oz?
- What do I need to run a Wizard of Oz experiment?
- What do I need to be aware of when designing and running a Wizard of Oz experiment?

Appendix D

The WebWOZ Prototyping Platform: Manual

Wizard of Oz (WOZ) is an evaluation method frequently used in dialogue design and user experience research. Its name stems from the famous novel 'The Wonderful Wizard of Oz' (Baum [1900]) in which a formidable wizards plays all kind of tricks to deceive his opponents and make them believe in his power. Like the wizard in the novel, a person running a WOZ experiment cheats by telling test participants that they would interact with a computer system. In practice, however, they interact with a human who mostly sits in a different room and uses a 'wizard interface' as a communication channel. By doing so it is possible to explore ideas and concepts for technology before investing a considerable amount of money and resources building them. The WOZ method has been around for more than 40 years. Its first application was reported by Erdmann and Neal [1971] who tested the concept of a self-service airline ticket kiosk before Gould et al. [1983] used it to explore their idea of the 'The Listing Typewriter'. From the very beginning WOZ was mainly used to explore speech-based interaction. Even though over time different exploration areas including multi-modal interaction (e.g. Salber and Coutaz [1993a]) were added, the primary WOZ scenario is still around spoken dialogue. In order to run a WOZ experiment one needs an interaction interface for test participants and an additional interface that is operated by the 'wizard'. Both interfaces are connected so that the wizard can influence the output on a participant's screen. In addition it is mostly the case that a participant's voice is transmitted through separate audio channel.

Even though the principle set-up of WOZ experiments is consistent, most of the interfaces built are used for one specific experiment only. The goal of WebWOZ is to deal with this issue and provide a tool for designers and researchers that lets them explore interactions beyond the frontier of one specific experiment. Rather they are able to easily create different experiments in order to test a variety of scenarios. In addition it is meant to extend the classic WOZ paradigm by offering a plug-able architecture that allows for the incorporation of language technology components as part of an experimental setting. The following will lead you through the set-up of a typical WOZ experiment and describe the different features of WebWOZ and how they are

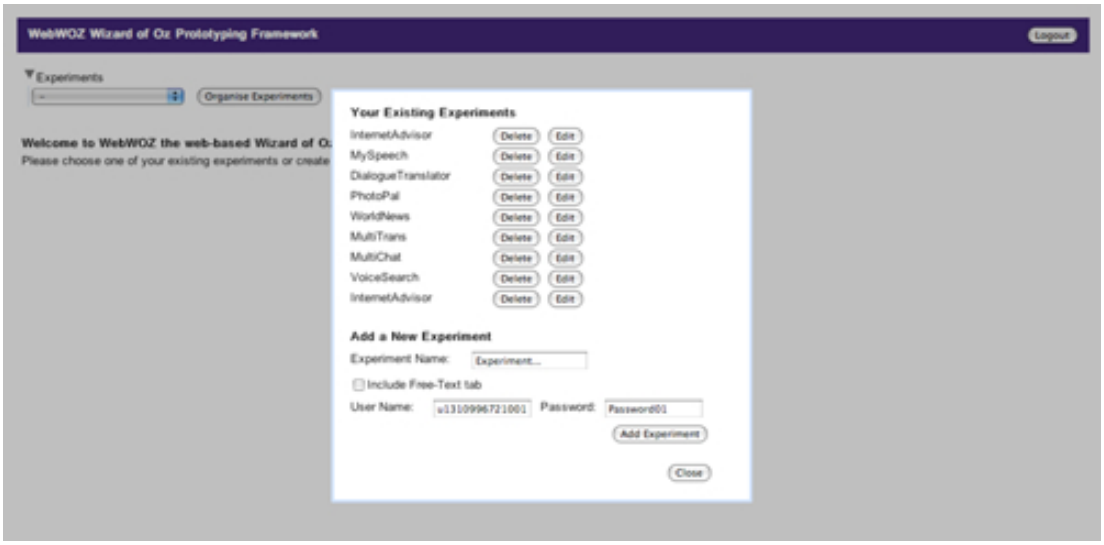

Figure D.1 – Add and Edit WebWOZ Experiments.

used.

Creating a New Experiment

After you are logged in to WebWOZ as a wizard you will see a drop-down menu that holds all your experiments as well as a button that lets you organise them. If you are logged in with a new wizard account the drop-down menu will be empty. *Organise Experiments* permits you to 'Add' a 'New Experiment' as well as to 'Edit' and 'Delete' existing ones (cf. Figure D.1). To add a new experiment you need to focus on the bottom part of the dialogue and enter an 'Experiment Name' and a 'User Name' as well as a 'Password' for the primary test participant you want to interact with. When running the experiment the test participant then needs to go to the WebWOZ client interface and login with this user name and password. Additional test participants can be added after the experiment is created.

The check-box *Include Free-text tab* permits you to add a chat-like interaction channel to the experiment that offers more flexibility for the wizard. In order to create a realistic experiment set-up a wizard is typically obliged to choose from a pre-defined set of utterances. In certain settings, however, it can be desirable to allow for more flexibility. Therefore, by adding the free-text option, the wizard is not bound to the pre-defined utterances but is rather able to interact freely.

After clicking *Add Experiment* the new experiment is shown under 'Your Existing Experiments'. Here if you click on *Edit* you are able to change the experiment name as well as the free-text option and add additional test participants. Via *Delete* you can delete an experiment. If you close the organise experiments dialogue you will find the newly created experiment(s) in the drop-down menu.

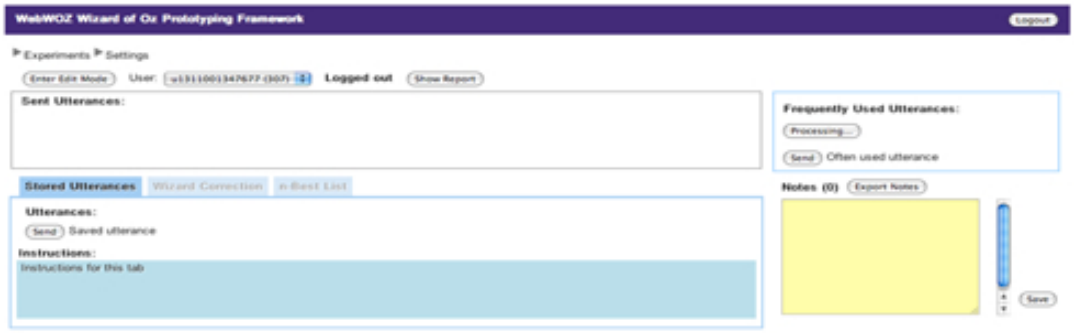

Figure D.2 – Default Wizard Interface.

The Wizard Screen

After you choose a newly created experiment from the drop-down menu you arrive at the default wizard interface (cf. Figure D.1). The interface is separated into different areas. On the very top there are two menus. One that permits you to switch to other Experiments as well as edit them and a second holding experiment related settings. Both are flip-down menus highlighted by the little arrow next to hem. They appear if you click on them and disappear if you click again. Right below the menus you find a button *Enter Edit Mode* and a drop-down menu that lets you choose with which test participant you are interacting. Next to the drop-down menu you are informed about the test participant's status i.e. whether he or she is logged in, and a button *Show Report* which holds the log file for the complete interaction with the selected test participant.

Below you find an area called 'Sent Utterances' that displays a history of everything that is sent to a test participant throughout an active session. In case the experiment makes use of participant input this area additionally holds either text-input coming directly from the participant or the results of a used speech recognition module.

Next to the history there is an area dedicated to 'Frequently Used Utterances'. Here the designer can add utterances that are used relatively often throughout an experiment and therefore can hardly be added to one dialogue stage only. In addition the area holds the *Processing...* button that can be used to notify a test participant that input is processed, which gives the wizard some time to search for the right response utterance or to formulate an appropriate one.

The area below the history is dedicated to the running dialogue and therefore the main interaction area for the wizard. Tabs are used to allow for structuring the dialogue. By default there is one activated tab called 'Stored Utterances' and two deactivated tabs, one called 'Wizard Correction' and a second one called 'N-best List'. The latter two will be activated in cases where the wizard's task is to enhance the output of a language technology component rather than to mimicking its functionality for which they take on a special role. The first tab, however, represents a standard tab that is used to organise a dialogue by representing a certain dialogue stage. It holds pre-defined utterances dedicated to this stage. Additional tabs can be added so that it is possible to group utterances based on their occurrence throughout the dialogue. Doing

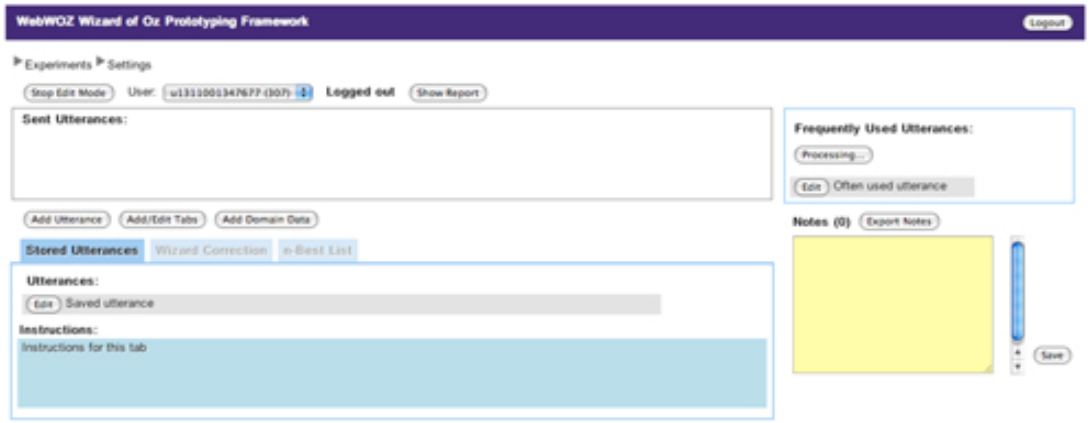

Figure D.3 – WebWOZ Edit Mode.

so helps to easy the task of the wizard and decrease search time. In addition the designer can define instructions for a tab, a.k.a. a dialogue stage, which is meant to help the wizard in being consistent and following a certain protocol when using utterances.

Finally the bottom right corner of the interface allows for taking 'Notes'. Every note is time stamped and added below, accessible via scrolling. In addition it is saved to a log file, which allows for keeping track of incidents happening during an experiment. The log file can be exported by clicking on *Export Notes*.

Add and Edit Utterances

In a usual WOZ experiment a wizard communicates with test participant by choosing from a pre-defined set of utterances. In order to add those utterances to an experiment please click on *Enter Edit Mode*. In edit mode an additional row of buttons is visible that lets you *Add*, *Edit* and *Delete* Utterances as well as Tabs (cf. Figure D.3). When handling utterances WebWOZ distinguishes between two types. First 'regular utterances' for dialogue related interactions with a test participant. Second, for cases where the wizard mimics some sort of knowledge base and therefore needs to send utterances based on a certain set of domain data, WebWOZ integrates the concept of 'domain utterances'. Domain utterances have the advantage that they are displayed in a separate area on top of the standard dialogue flow wherefore they are accessible without switching dialogue stages. In addition it is possible to add filter tags to them so that a wizard easily finds the right response by manipulating those filter options. The use of domain utterances will be discussed in more detail later in this document. For the moment, however, the focus lies on regular dialogue related utterances.

In order to add a regular dialogue utterance please click on *Add Utterance*. The following dialogue permits you to add a new utterance (cf. Figure D.4). 'Short Name / Label' lets you create a short description of the utterance. 'Utterance' holds the actual utterance. With 'Link to Audio File' and 'Link to Video File' it is possible to connect the utterance with a prepared

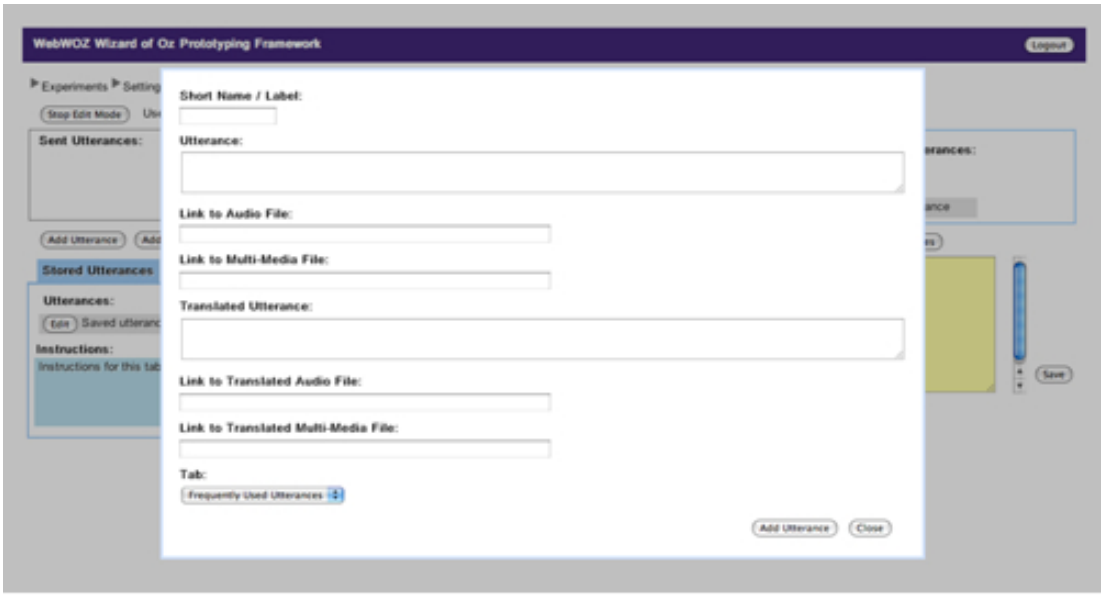

Figure D.4 – Add Utterance.

audio and video file, respectively. The files need to be located on an accessible server and the given link needs to be absolute (e.g. http:///link-to-file/filename.mp3). In terms of file formats it needs to be highlighted that currently its support depends greatly on the web browser that is used on the participant's side. However, since WebWOZ is based on HTML 5, whose standardization is ongoing, the support of different media file types should be solved in the near future.

In case the designed experiment is based purely on text or makes use of an on-the-fly text-to-speech synthesis the fields holding the links to audio and video files can remain empty. In order to support multi-lingual experiments WebWOZ further permits you to specify a translation for your utterance, including its connection to a relevant media file. Also here, if no translation is needed or an on-the-fly machine translation service is used, those fields can remain empty. Finally, the tab to which the utterance should be added, can be selected from a drop-down menu at the bottom of the dialogue. With a new experiment only two options are available. On the one hand 'Frequently Used Utterances', which describes the area next to the history, and on the other hand 'Stored Utterances' standing for the first tab in the dialogue area. The following section will show you how you can add additional tabs or edit existing ones, so that it is possible to give more structure to the dialogue. For the moment you can just select one of the two existing options and click on *Add Utterance*, so that the utterance is added to the selected tab. In order to edit an utterance you need to click on the *Edit* button next to it. It will open a dialogue which lets you change all its attributes as well as permits you to delete it. In order to use an utterance, i.e. send it to a test participant, you need to exit edit mode by clicking on *Stop Edit Mode*.

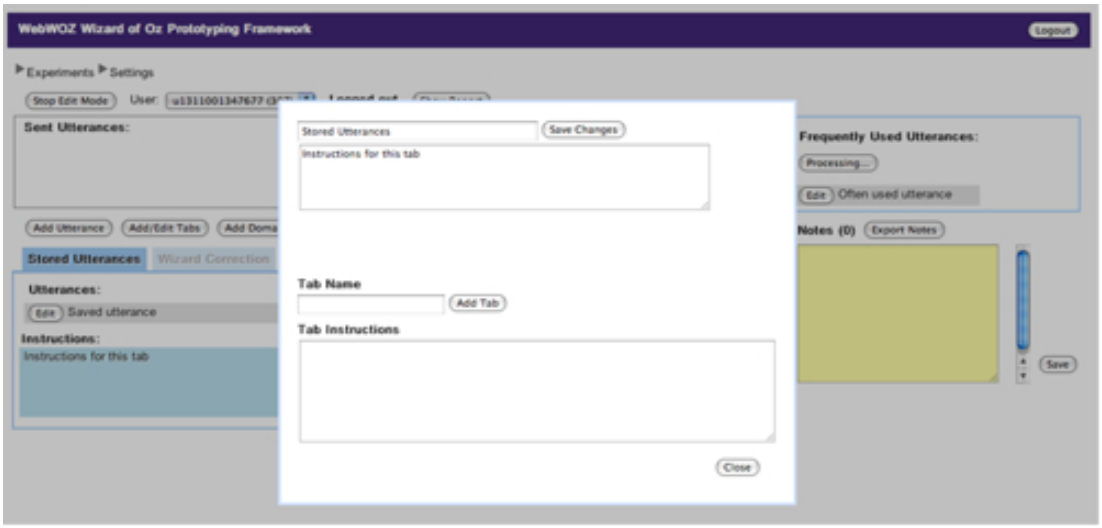

Figure D.5 – Add Tabs.

Add and Edit Tabs

In order to organise utterances WebWOZ permits you add tabs. A new tab can be created entering the edit mode and clicking on *Add / Edit Tabs*. The following dialogue lets you edit and delete existing tabs as well as add new tabs (cf. Figure D.5). In terms of editing you are able to change the name of the tab and add some instructions for the wizard, which will be displayed in the bottom. If the instructions field is empty, the box usually holding instructions is hidden, giving more space to utterances. With exception of the first tab all other tabs that are created can also be deleted, given they do not contain any utterances.

If you want to delete a tab that contains utterances, you first need to delete those utterances or move them to a different tab. In order to add a tab enter a 'Tab Name' and optional 'Tab Instructions' and click on *Add Tab*. If you close the dialogue you will see that the tab was added to the dialogue structure. Now utterances can be added to this tab or moved to it from others.

Frequently Used Utterances

Frequently used utterances are utterances that a wizard needs to use on an irregular basis and which therefore are hard to add to only one specific dialogue tab. They are shown in a separate area on the screen (cf. Figure D.6). Typically those utterances include ways of error recovery like "Sorry, but I did not understand you. Could you please repeat?" or gap fillers like "Please hold!". Those utterances are treated like general utterances and can therefore be added in edit mode using the *Add Utterance* button and edited using the *Edit* button next to them. In addition the area for frequently used utterances holds a button called *Processing...*. A wizard can use this button to notify a participant that the computer is processing the last request. On the participants screen this will display the utterance "Processing..." followed by an additional dot

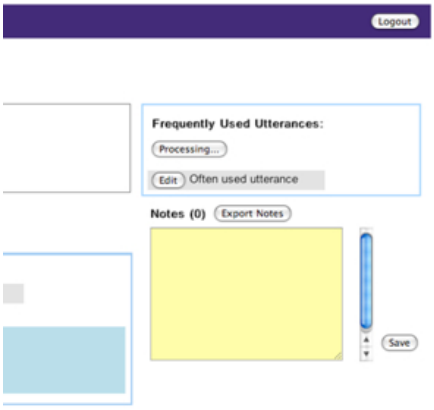

Figure D.6 – Frequently Used Utterances.

Figure D.7 – Taking Notes.

every second to signalise an ongoing working process. Doing so gives the wizard some time to search for the right utterance and at the same time shows the test participant that the system is responsive. As soon as the wizard sends the next utterance the Processing... is reset so that the next time it is used it starts again with three dots at the end.

Taking Notes

WOZ is an evaluation method and therefore one often wants to make notes throughout an experiment and later revisit them in order to better understand a given situation. In order to support this process WebWOZ offers a notes functionality that allows for the wizard to write a note within the wizard interface. Notes are time stamped and saved to a database so that they can be exported later on (cf. Figure D.7).

Settings

The settings are accessible via the flip-down menu on top of the wizard screen. Here it is possible to configure the interaction pipeline between the wizard and the test participants (cf. Figure D.8).

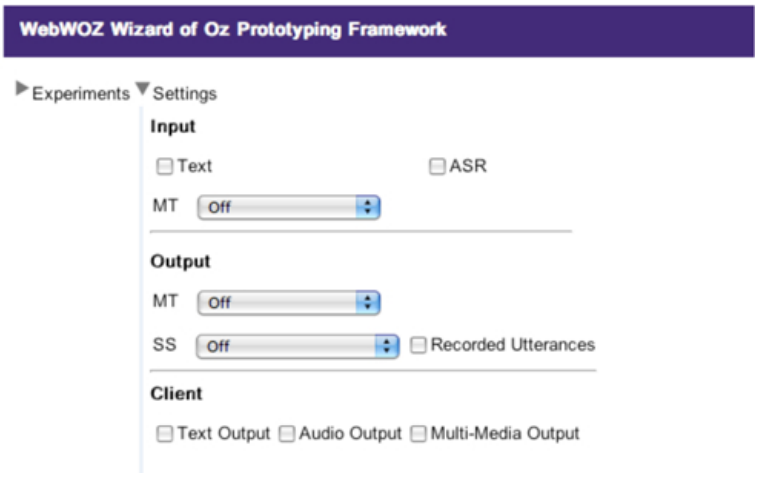

Figure D.8 – Settings.

Input

Input defines how the test participant interacts with the system. If 'Text box' is checked the participant will be able to use a chat-like input window to interact with the system. ASR activates speech recognition for the participant. When activated the recognition language can be defined and furthermore it is possible to activate a 'Correction Mode' which is used in experiments where the wizard rather enhances the output of language technology component than replaces it as a whole (e.g. the wizard corrects the output of the speech recognizer). Correction includes 'Wizard Correction' which lets the wizard amend what is recognized and 'N-best List' which permits to choose from a list of possible recognition results. Depending on which option is selected, the output arrives at the dedicated tab within the dialogue structure.

Both, text input and speech recognition can be activated at the same time, even though the use of a single modality might be more likely. In cases where the experiment tries to explore a multi-lingual setting it is further possible to push the client input, whether text based or recognized by the speech recognizer, through an on-the-fly machine translation (MT) system before it arrives at the wizard. For this purpose WebWOZ incorporates two external MT systems. If a MT system on the input side is selected it is possible to define source as well as target language. Also here a 'Correction Mode' can be activated, which works the same way as already explained in connection with the ASR option. However, a correction can only happen at the last stage of an input. Hence, if an MT system is used the correction mode for ASR, if set, will be deactivated. On the other hand, if a correction mode for ASR is activated, it is not possible to use MT on the input side.

Even though WebWOZ supports text as well as speech recognized input by participants, it is often the case with WOZ experiments, that advanced input modalities are evaluated. For example a speech recognizer that goes beyond the rather limited functionality offered by the one used in WebWOZ, or alternatively some sort of multi-modal interaction. In this case it is possible that neither the Text nor the ASR box is checked for which the experiment set-up

needs to cater for an alternative method of transferring input from a participant to the wizard. In most cases this can be achieved via some sort of Voice-Over-IP system in combination with live video streams.

Output

The Output settings concern everything that goes from the wizard to the test participant. Similar to the input side it is here possible to send the utterance first through an MT system before it arrives at the participant. In addition it is possible to send it to an on-the-fly text-to-speech synthesizer (TTS). WebWOZ integrates two different text-to-speech systems. Both, however, have still experimental status for which their functionality can not be guaranteed. Even though on-the-fly MT as well as TTS is supported, in some WOZ experiments a researcher might prefer a more controlled set-up. Therefore it is possible to use pre-recorded utterances, either based on audio or based on multi-media files, both stored on an external server. So if the Recorded Utterances box is checked, neither TTS nor MT is available on the output side. Rather the pre-recorded files that are connected to an utterance (cf. Add and Edit Utterances) will be played to a participant.

Client

Finally, the Client settings define what actually arrives at a test participant. Options consist of 'Text Output', which is basically the utterance in pure text form, 'Audio Output' either from an on-the-fly TTS or pre-recorded, and 'Multi-Media Output' coming from a connected multi-media file or a special purpose web-service producing on-the-fly multi-media content. Options can be used in different combination. Note, however, that if none of the options is selected, nothing will arrive at the participant.

Domain Utterances

As already mentioned earlier in some WOZ experiments the wizard mimics a certain knowledge. For example an information system whose final recommendation depends on different parameters. In order to help the wizard run such an experiment, WebWOZ incorporates the concept of 'domain utterances'. Domain utterances are special utterances that serve the purpose of giving a recommendation. In order to arrive at a final recommendation it is possible to add different filters and values to domain utterances. Important here is that all the domain utterances of an experiment share the same filters. Therefore, filter values need to be set for all of them.

Domain utterances can be created in edit mode by clicking on *Add Domain Data*. The following dialogue is similar to the one discussed for adding regular utterances. One difference, however, is that it does not allow for defining a specific tab the utterance should go into. That is because domain utterances are situated in a separate area on top of the actual dialogue structure.

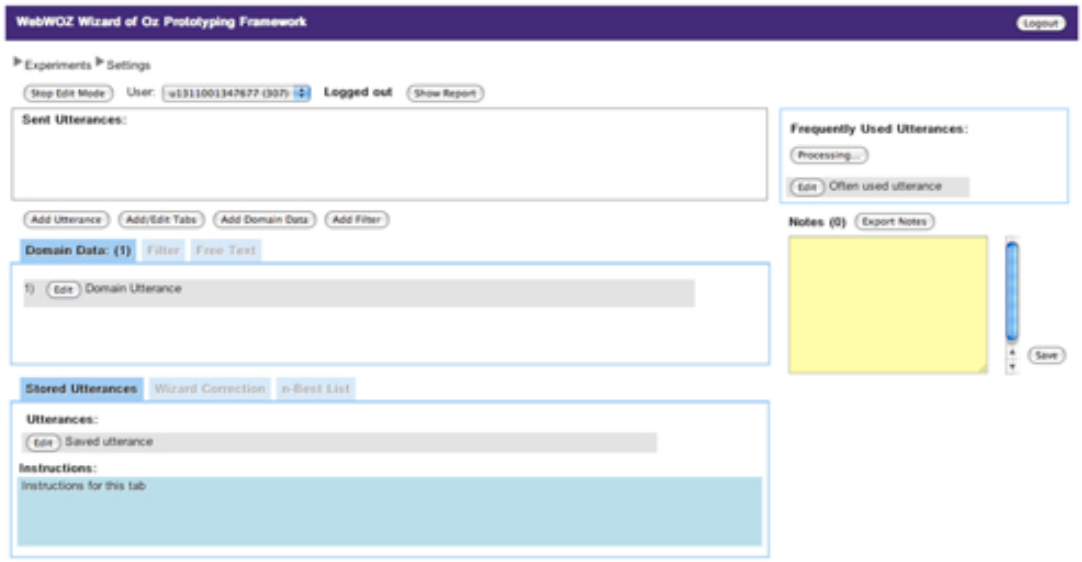

Figure D.9 – Area for Domain Utterances.

After adding a domain utterance this additional area appears. Also, an additional button to add filters is now visible (cf. Figure D.9). The new area consists of several tabs which are active or inactive, depending on the experiments settings. In addition to domain data it also holds filters for domain utterances and the option to use free text to interact with a test subject. The free text option can be activated in the general experiment settings via *Organise Experiments* which can be found in the experiment's flip-down menu. Filters can be added through clicking on the *Add Filter* button. If non of these elements is used within an experiment, the whole area disappears.

Adding Filters

In order to make it easier for wizards to search through domain utterances WebWOZ permits to add different filters to them. A filter can be added in edit mode by clicking on *Add Filter* (cf. Figure D.10). Note, however, that at least one domain utterance needs to be created for this button to be visible. A 'Filter Name' needs to be specified as well as a 'Default Value' which is added to all the existing domain utterances as well as to the ones that are created afterwards. In addition to the default value additional filter values can be specified. In theory it is possible to define unlimited values as well as it is possible to define unlimited filters. However, the limited screen space might make it difficult to handle them.

After adding a filter it can be deleted as well as edited by clicking on *Edit* next to it. Also, now it is possible to set the filter values for existing domain utterances by clicking on *Edit* next them. The following dialogue will then have a drop-down menu for all the filters that were created holding the different filter values (cf. Figure D.11).

Without changes a domain utterance always uses the 'Default Value' for filters. By adapting those values it is however possible to help the wizard searching for domain utterances. So is

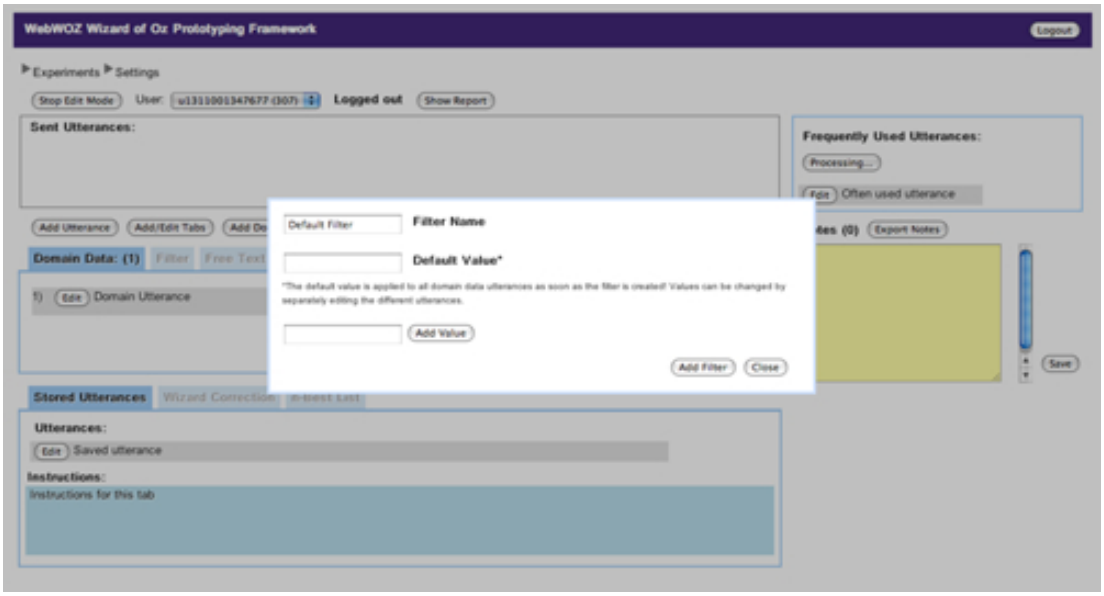

Figure D.10 – Add Filters.

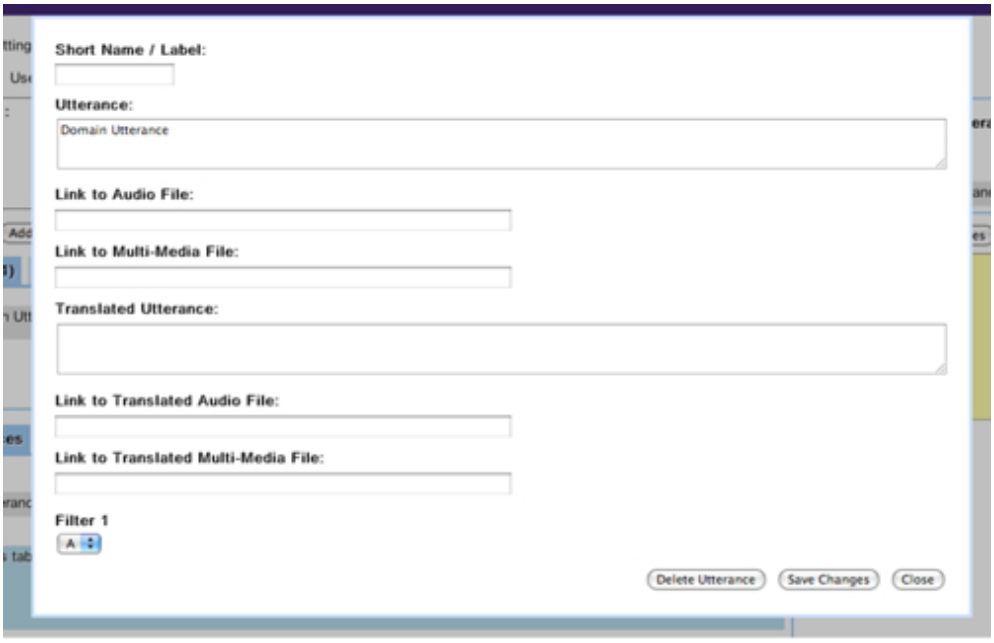

Figure D.11 – Setting Filter Values for Domain Utterances.

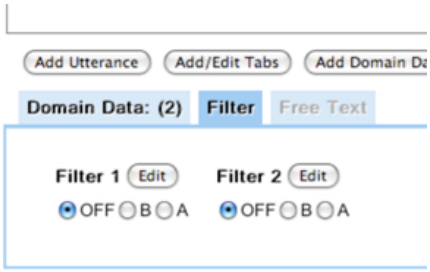

Figure D.12 – No Filter Values Selected.

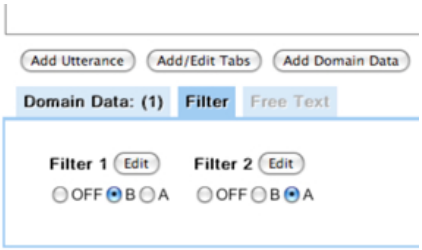

Figure D.13 – Reduced Number of Domain Utterances through filtering.

it possible to reduce the number of domain utterances by adapting the filters (cf. Figure D.12 & D.13).

Free Text

In order have more flexibility when interacting with a test participant an experiment designer can activate the free text option. This can be done through selecting the Include 'Free-Text tab' option in the general experiment settings. In order to so one needs to click on the *Organise Experiments* button which can be found under the experiment's flip-down menu. Next to any of 'Your Existing Experiments' an *Edit* button can be found which leads to the dialogue for changing the general experiment settings (cf. Figure D.14 & D.15). Here it is possible to change the name of the experiment, add additional test participants and adapt their login parameter, as well as set the free text option. If the box next to 'Include Free-Text tab' is checked and the changes saved, the dedicated tab in the wizard interface is activated (cf. Figure D.16). Now, it is possible for a wizard to interact freely with a test participant. However, this sort of interaction is only advised for experiments that make use of on-the-fly services for TTS and MT or focus on a pure text based interaction, as no media files or translations can be prepared.

With the 'Free Text' option an additional function of WebWOZ is activated. As an experiment sometimes requires the adaptation or concatenation of prepared utterances it is possible to use the *Free Text* button to send an utterance to the free text field instead of sending it directly to a test participant. There it can be amended or another utterance can be added before it is sent on (cf. Figure D.17).

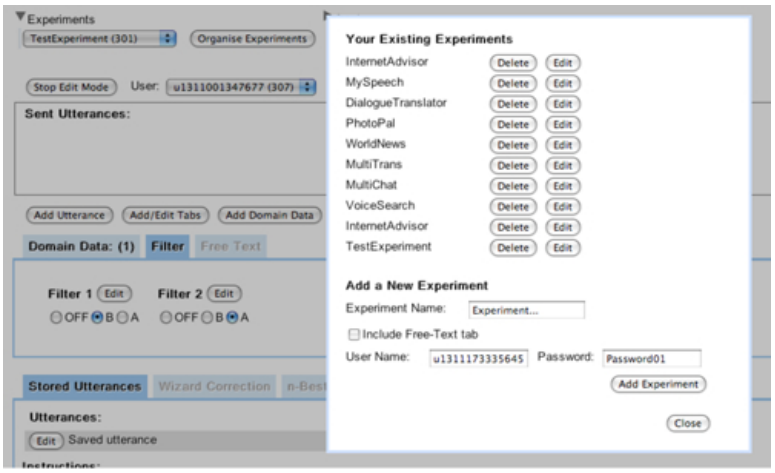

Figure D.14 – Edit Experiment Settings.

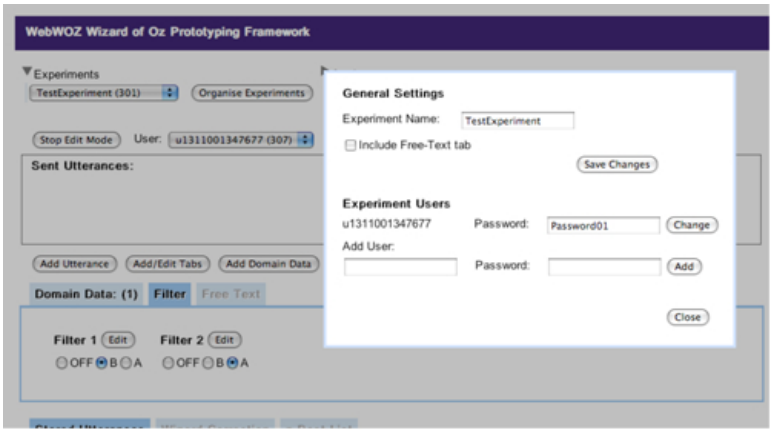

Figure D.15 – General Experiment Settings.

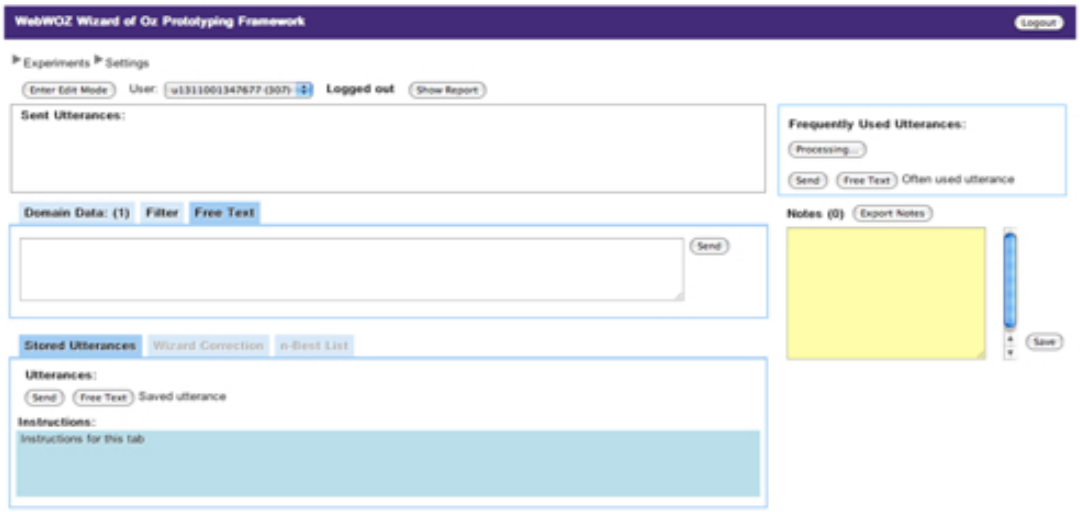

Figure D.16 – Activated Free Text Interaction.

Figure D.17 – Sending Utterances to the Free Text Field.

Bibliography

Akers, D. Wizard of oz for participatory design: Inventing a gestural interface for 3d selection of neural pathway estimates. In *Proceedings of CHI*, pages 454–459, 2006a.

Akers, D. CINCH: A cooperatively designed marking interface for 3d pathway selection. In *Proceedings of UIST*, pages 33–42, 2006b.

Aleksy, M.; Stieger, B., and Fantana, N. Utilizing mock-ups in the development of distributed information systems for semantic data federations. In *Proceedings of CISIS*, pages 307–312, 2010.

Ammicht, E.; Gorin, A. L., and Alonso, T. Knowledge collection for natural language spoken dialogue systems. In *Proceedings of Eurospeech*, 1999.

Andersson, G.; Höök, K.; Mourão, D.; Paiva, A., and Costa, M. Using a wizard of oz study to inform the design of sentoy. In *Proceedings of DIS*, 2002.

Argyris, C.; Putnam, R., and Smith, D. M. *Action Science: Concepts, Methods, and Skills for Research and Intervention*. Jossey Bass, 1985.

Bailey, B. P.; Biehl, J. T.; Cook, D. J., and Metcalf, H. E. Adapting paper prototyping for designing user interfaces for multiple display environments. *Personal and Ubiquitous Computing*, 12:269–277, 2008.

Bailey, K. D. *Methods of social research*. The Free Press, 1982.

Balbo, S.; Coutaz, J., and Salber, D. Towards automatic evaluation of multimodal user interfaces. In *Proceedings of IUI*, pages 201–208, 1993.

Bälter, O.; Engwall, O.; Öster, A.-M., and Kjellström, H. Wizard-of-oz test of ARTUR - a computer-based speech training system with articulation correction. In *Proceedings of ASSETS*, pages 36–43, 2005.

Barrett, R.; Kandogan, E.; Maglio, P. P.; Haber, E. M.; Takayama, L. A., and Prabaker, M. Field studies of computer system administrators: Analysis of system management tools and practises. In *Proceedings of CSCW*, pages 388–395, 2004.

Baum, L. F. *The Wonderful Wizard of Oz*. George M. Hill, 1900.

Bederson, B. B.; Hu, C., and Resnik, P. Translation by iterative collaboration between monolingual users. In *Proceedings of Graphics Interface*, pages 39–46, 2010.

Benzmüller, C.; Fiedler, A.; Gabsdil, M.; Horacek, H.; Kruijff-Korbayova, I.; Pinka, M.; Siekmann, J.; Tsovaltzi, D.; Vo, B. Q., and Wolska, M. A wizard-of-oz experiment for tutorial dialogues in mathematics. In *Supplementary Proceedings of AIED (Vol. VIII: Advanced Technologies for Mathematics Education)*, pages 471–481, 2003.

Bernhaupt, R.; Jenisch, S.; Keyser, Y., and Will, M. Capture the flag: Simulating a location-based mobile game using the wizard-of-oz method. In *Proceedings of ACE*, pages 228–229, 2007.

Bernsen, N. O. and Dybkjaer, H. Knowledge acquisition for a constrained speech system using woz. In *Proceedings of EACL*, page 467, 1993.

Bernsen, N. O.; Dybkjaer, H., and Dybkjaer, L. Wizard of Oz Prototyping: When and how? *Working Papers in Cognitive Science and HCI*, pages 1–13, 1994.

Bertenstam, J.; Beskow, J.; Blomberg, M.; Carlson, R.; Elenius, K.; Granström, B.; Gustafson, J.; Hunnicutt, S.; Högberg, J.; Lindell, R.; Neovius, L.; Nord, L.; Serpa-Leitao, A., and Ström, N. The waxholm system - a progress report. In *Proceedingso of ESCA*, pages 281–284, 1995.

Bertomeu, N.; Uszkoreit, H.; Frank, A.; Krieger, H.-U., and Jöorg, B. Contextual phenomena and thematic relations in database qa dialogues : results from a wizard-of-oz experiment. In *Proceedings of NAACL-HLT*, number June, pages 1–8, 2006.

Bevacqua, E.; Hyniewska, S. J., and Pelachaud, C. Positive influence of smile backchannels in ecas. In *Proceedings of AAMAS*, 2010a.

Bevacqua, E.; Hyniewska, S. J., and Pelachaud, C. Evaluation of a virtual listener' s smiling behavior. In *Proceedings of CASA*, 2010b.

Beyer, H. and Holtzblatt, K. *Contextual Design: Defining Customer-Centered Systems*. Morgan Kaufmann, 1998.

Biesta, G. J. J. *Pragmatism and educational research*. Rowman and Littlefield, 2003.

Blake, S. P. *Managing for responsive research and development*. W. H. Freeman and Company, 1978.

Bobrow, D. G.; Kaplan, R. M.; Kay, M.; Norman, D. A., and H.and WinogradThompson, T. GUS, a frame-driven dialog system. *Artificial Intelligence*, 8(2):155–173, 1977.

Bohus, D. and Rudnicky, A. I. RavenClaw: Dialog management using hierarchical task decomposition and an expectation agenda. In *Proceedings of ICSLP*, 2003.

Bohus, D. and Rudnicky, A. I. Sorry, i didn't catch that! – an investigation of non-understanding errors and recovery strategies. In *Proceedings of SigDial*, 2005.

Bohus, D.; Raux, A.; Harris, T. K.; Eskenazi, M., and Rudnicky, A. I. Olympus: An open-source framework for conversational spoken language interface research. In *Proceedings of NAACL-HLT*, pages 32–39, 2007.

Bradley, J.; Mival, O., and Benyon, D. Wizard of oz experiments for companions. In *Proceedings of BCS-HCI*, pages 313–317, 2009.

Bradley, J.; Benyon, D.; Mival, O., and Webb, N. Wizard of oz experiments and companion dialogues. In *Proceedings of BCS-HCI*, pages 117–123, 2010.

Braun, I. and Rummel, N. Facilitating learning from computer-supported collaborative inquiry: the challenge of directing learners' interactions to useful ends. *Research and Practice in Technology Enhanced Learning*, 5(3):205–242, 2010.

Brooke, J. SUS - a quick and dirty usability scale. Technical report, 1996.

Cabral, J. P.; Kane, M.; Ahmed, Z.; Abou-Zleikha, M.; Székely, /'E.; Zahra, A.; Ogbureke, K.; Cahill, P.; Carson-Berndsen, J., and Schlögl, S. Rapidly testing the interaction model of a pronunciation training system via wizard-of-oz. In *Proceedings of LREC*, 2012.

Carbonell, J. G. Discourse pragmatics and ellipsis resolution in task-oriented natural language interfaces. In *Proceedings of ACL*, pages 164–168, 1983.

Caroll, J. M. Scenarios and design cognition. In *Proceedings IEEE Joint International Conference on Requirements Engineering*, 2002.

Carroll, J. and Aaronson, A. Learning by doing with simulated intelligent help. *Communications of the ACM*, 31(9):1064–1079, 1988.

Cenek, P.; Melichar, M., and Rajman, M. A framework for rapid multimodal application design. In *Proceedings of TSD*, pages 393–403, 2005.

Chandler, C. D.; Lo, G., and Sinha, A. K. Multimodal theater: Extending low fidelity paper prototyping to multimodal applications. In *Proceedings of CHI*, pages 874–875, 2002.

Chen, Charles L. and Raman, T. V. AxsJAX: A talking translation bot using google im: Bringing web-2.0 applications to life. In *Proceedings of W4A*, pages 54–56, 2008.

Chen, Y.; Naveed, A., and Porzel, R. Behavior and preference in minimal personality: A study on embodied conversational agents. In *Proceedings of ICMI*, 2010.

Cheng, H.; Bratt, H.; Mishra, R.; Shriberg, E.; Upson, S.; Chen, J.; Weng, F.; Peters, S.; Cavedon, L., and Niekrasz, J. A wizard of oz framework for collecting spoken human-computer dialogs. In *Proceedings of ICSLP*, 2004.

Chignel, M. H. A taxonomy of user interface terminology. *SIGCHI Bulletin*, 21(4):27–34, 1990.

Clemmensen, T. Four approaches to user modelling - a qualitative research interview study of hci professionals' practice. *Interacting with Computers*, (16):799–829, 2004.

Consolvo, S.; Harrison, B.; Smith, I.; Chen, M.; Everitt, K.; Froehlich, J., and Landay, J. A. Conducting in situ evaluations for and with ubiquitous computing technologies. *International Journal of Human-Computer Interaction*, 22(1):107–122, 2007.

Cooper, A. *The Innmates are running the asylum*. Sams Publishing, 2004.

Corradini, A. and Cohen, P. R. On the relationships among speech, gestures , and object manipulation in virtual environments: Initial evidence. In *Proceedings fo CLASS Workshop on Natural, Intelligent, and Effective Interaction in Multimodal Dialog Systems*, 2002.

Coutaz, J.; Salber, D.; Carraux, E., and Portolan, N. NEIMO, a multiworkstation usability lab for observing a multiworkstation usability and analyzing multimodal interaction. In *Proceedings of CHI*, pages 402–403, 1996.

Coyle, D.; Doherty, G.; Matthews, M., and Sharry, J. Computers in talk-based mental health interventions. *Interacting with Computers*, 19:545–562, 2007.

Dahlbäck, N. and Jönsson, A. Empirical studies of discourse representations for natural language interfaces. In *Proceedings of EACL*, pages 291–298, 1989.

Dahlbäck, N. and Jönsson, A. An empirically based computationally tractable dialogue model. In *Proceedings of COGSCI*, 1992.

Dahlbäck, N.; Jöonsson, A., and Ahrenberg, L. Wizard of oz studies - why and how. In *Proceedings of IUI*, pages 193–200, 1993.

Davies, J. T. *The scientific approach*. The Academic Press, 1973.

Davis, J. Active help found beneficial in wizard of oz study. *Information and Software Technology*, 40:93–103, 1998.

Davis, R. C.; Saponas, S. T.; Shilman, M., and Landay, J. A. SketchWizard: Wizard of oz prototyping of pen-based user interfaces. In *Proceedings of UIST*, pages 119–128, 2007.

De Marconnay, P.; Crowley, J. L., and Salber, D. Visual interpretation of faces in the NEIMO multi-modal test-bed. In *Proceedings of IJCAI. Workshop Looking at People: Recognition and Interpretation of Human Action*, 1993.

Dearman, D.; Hawkey, K., and Inkpen, K. M. Effect of location-awareness on rendezvous behaviour. In *Proceedings of CHI*, pages 1929–1932, 2005.

Delaborde, A.; Tahon, M.; Barras, C., and Devillers, L. A wizard-of-oz game for collecting emotional audio data in a children-robot interaction. In *Proceedings of AFFINE*, 2009.

Deruyter, B.; Saini, P.; Markopoulos, P., and Vanbreemen, A. Assessing the effects of building social intelligence in a robotic interface for the home. *Interacting with Computers*, 17(5): 522–541, 2005.

Dow, S.; Lee, J.; Oezbek, C.; Macintyre, B.; Bolter, J. D., and Gandy, M. Wizard of oz interfaces for mixed reality applications. In *Proceedings of CHI*, pages 1339–1342, 2005a.

Dow, S.; Lee, J.; Oezbek, C.; Macintyre, B.; Bolter, J. D., and Gandy, M. Exploring spatial narratives and mixed reality experiences in oakland cementery. In *Proceedings of ACM ACE*, pages 51–60, 2005b.

Dow, S.; Macintyre, B.; Lee, J.; Oezbek, C.; Bolter, J. D., and Gandy, M. Wizard of oz support throughout an iterative design process. *IEEE Pervasive Computing*, 4(4):18–26, 2005c.

Dow, S.; Mehta, M.; Macintyre, B., and Mateas, M. Eliza meets the wizard-of-oz: Blending machine and human control of embodied characters. In *Proceedings of CHI*, pages 547–556, 2010.

Dumas, J. S. and Redish, J. C. *A Practical Guide to Usability Testing*. Intellect, 1999.

Dybkjaer, H.; Bernsen, N. O., and Dybkjaer, L. Wizard-of-oz and the trade-off between naturalness and recogniser constraints. In *Proceedings of Eurospeech*, 1993.

Edlund, J.; Gustafson, J.; Heldner, M., and Hjalmarsson, A. Towards human-like spoken dialogue systems. *Speech Communication*, 50(8-9):630–645, 2008.

Erdmann, R. L. and Neal, A. S. Laboratory vs. field experimentation in human factors: An evaluation of an experimental self-service airline ticket vendor. *Human Factors*, 13:521–531, 1971.

Fabbrizio, G.; Tur, G., and Hakkani-Tür, D. Automated wizard-of-oz for spoken dialogue systems. In *Proceedings of ICSLP*, pages 1857–1860, 2005.

Ferreira, P.; Sanches, P., and Höök, K. License to chill! how to empower users to cope with stress. In *Proceedings of NordiCHI*, pages 18–22, 2008.

Fiedler, A. and Gabsdil, M. Supporting progressive refinement of wizard-of-oz experiments. In *Proceedings of ITS*, pages 62–69, 2002.

Fraser, N. M. and Gilbert, N. G. Simulating speech systems. *Computer Speech and Language*, 5:81–99, 1991.

Fügen, C.; Kolss, M.; Bernreuther, D.; Paulik, M.; Stücker, S.; Vogel, S., and Waibel, A. Open domain speech recognition and translation: lecture notes & speeches. In *Proceedings of ICASSP*, 2006.

Gandhe, S. and Traum, David R. First steps towards dialogue modelling from an un-annotated human-human corpus. In *Proceedings of IJCAI*, 2007.

Geutner, P.; Steffens, F., and Manstetten, D. Design of the VICO spoken dialogue system: Evaluation of user expectations by wizard-of-oz experiments. In *Proceedings of LREC*, 2002.

Glaser, B. G. The constant comparative method of qualitative analysis. *Social Problems*, 4 (12):445, 1965.

Glaser, B. G. *Theoretical Sensitivity: Advances in the methodology of Grounded Theory*. Sociology Press, 1978.

Glaser, B. G. and Strauss, A. L. *The discovery of grounded theory*. Aldine, 1967.

Glaserfeld, E. *Constructivism in education*. Pergamon Press, 1989.

Goguen, J. and Linde, C. Techniques for requirements elicitation. In *IEEE International Symposium on Requirements Engineering (RE'93)*, pages 152–164, 1993.

Goldstein, M.; Bretan, I.; Sallnas, E.-L., and Bjork, H. Navigational abilities in audial voice-controlled dialogue structures. *Behaviour & Information Technology*, 18(2):83–95, 1999.

Good, M. D.; Whiteside, J. A.; Wixon, D. R., and Jones, S. J. Building a user-derived interface. *Communications of the ACM*, 27(10):1032–1043, 1984.

Gorin, A. L.; Riccardi, G., and Wright, J. H. How may i help you? *Speech Communication*, 23(1-2):113–127, 1997.

Gould, J. D. and Lewis, C. Designing for usability: Key principles and what designers think. *Communications of the ACM*, 28(3):300–311, 1985.

Gould, J. D.; Conti, J., and Hovanyecz, T. Composing letters with a simulated listening typewriter. *Communications of the ACM*, 26(4):295–308, 1983.

Gould, J. D.; Boies, S. J.; Levy, S.; Richards, J. T., and Schoonard, J. The 1984 olympic message system: a test of behavioral principles of system design. *Communications of the ACM*, 30(9):758–769, 1987.

Graesser, A. C.; VanLehn, K.; Rose, C. P.; Jordan, P. W., and Harter, D. Intelligent tutoring systems with conversational dialogue. *AI Magazine*, 22(4):39–51, 2001.

Greenbaum, J. and Kyng, M. *Design At Work - Cooperative design of Computer Systems*. Lawrence Erlbaum, 1991.

Guindon, R. User request help from advisory systems with simple and restricted language: Effects of real-time constraints and limited shared context. *Human-Computer Interaction*, 6 (1):47–75, 1991.

Guindon, R.; Schuldberg, K., and Conner, J. Grammatical and ungrammatical structures in user-adviser dialogues: Evidence for sufficiency of restricted languages in natural language interfaces to advisory systems. In *Proceedings of ACL*, pages 41–44, 1987.

Guo, G.; Stan, Z. L., and Kapluk, C. Face recognition by support vector machines. In *Proceedings IEEE International Conference on Automatic Face and Gesture Recognition*, pages 96–201, 2000.

Gustafson, J.; Bell, L.; Beskow, J.; Boye, J.; Carlson, R.; Edlund, J.; Granström, B.; House, D., and Wirén, M. Adapt – a multimodal conversational dialogue system in an apartment domain. In *Proceedings of ICSLP*, pages 134–137, 2000.

Gustafson, J.; Boye, J.; Fredriksson, M.; Johanneson, L., and Königsmann, J. Providing computer game characters with conversational abilities. In *Proceedings of IVA*, 2005.

Hajdinjak, M. and Mihelic, F. The Wizard of Oz System for Weather Information Retrieval. In *Proceedings of TSD*, pages 400–405, 2003.

Hasan, H. Information systems development as a research method. *Australasian Journal of Information Systems*, 11(1):4–13, 2003.

Hauptmann, A. G. Speech and gestures for graphic image manipulation. In *Proceedings of CHI*, pages 241–245, 1989.

Hemmeryckx-Deleersnijder, B. and Thorne, J. M. Awareness and conversational context-sharing to enrich tv-based communication. *ACM Computers in Entertainment*, 6(1):1–13, 2008.

Henderson, V.; Lee, S.; Brashear, H.; Hamilton, H.; Starner, T., and Hamilton, S. Development of an american sign language game for deaf children atlanta area school for the deaf. In *Proceedings of IDC*, pages 70–79, 2005.

Heron, J. *Cooperative Inquiry: Research into the human condition*. Sage, 1996.

Hill, W. C. A wizard of oz study of advice giving and following. *Human-Computer Interaction*, 8(1):57–81, 1993.

Hill, W. C. and Miller, J. R. Justified advice: a semi-naturalistic study of advisory strategies. In *Proceedings of CHI*, pages 185–190, 1988.

Howell, M.; Love, S., and Turner, M. The impact of interface metaphor and context of use on the usability of a speech-based mobile city guide service. *Behaviour & Information Technology*, 24(1):67–78, 2005.

Höysniemi, J.; Hämäläinen, P., and Turkki, L. Wizard of oz prototyping of computer vision based action games for children. In *Proceeding of IDC*, pages 27–34, 2004.

Hudson, S. E.; Fogarty, J.; Atkeson, C. G.; Avrahami, D.; Forlizzi, J.; Kiesler, S.; Lee, J. C., and Yang, J. Predicting human interruptibility with sensors: A wizard of oz feasibility study. In *Proceedings of CHI*, pages 257–264, 2003.

Hundhausen, C.; Balkar, A.; Nuur, M., and Trent, S. WOZ Pro: A pen-based low fidelity prototyping environment to support wizard of oz studies. In *Proceedings of CHI*, pages 2453–2458, 2007.

Janarthanam, S. and Lemon, O. A wizard-of-oz environment to study referring expression generation in a situated spoken dialogue task. In *Proceedings of ENLG*, pages 94–97, 2009.

Jondahl, S. and Mørch, A. Simulating pedagogical agents in a virtual learning environment. In *Proceedings of CSCL*, pages 531–532, 2002.

Jönsson, A. and Dahlbäck, N. Talking to a computer is not like talking to your best friend. In *Proceedings of SCAI*, pages 297–307, 1988.

Jurafsky, D. and Martin, J. H. *Speech and Language Processing*. Prentice Hall, second edition, 2008.

Karpov, A.; Ronzhin, A., and Leontyeva, A. A semi-automatic wizard of oz technique for Let'sFly spoken dialogue system. In *Lecture Notes in Computer Science: Text, Speech and Dialogue*, pages 585–592. Springer, 2008.

Kelley, J. F. An empirical methodology for writing user-friendly natural language computer applications. In *Proceedings of CHI*, pages 193–196, 1983.

Kelley, J. F. An iterative design methodology for user-friendly natural language office information applications. *ACM Transactions on Information Systems*, 2(1):26–41, 1984.

Keskin, C.; Balci, K.; Aran, O.; Sankur, B., and Akarun, L. A multimodal 3D healthcare communication system. In *3D Conference*, 2007.

Kieffer, S.; Coyette, A., and Vanderdonckt, J. User interface design by sketching: A complexity analysis of widget representations. In *Proceedings of EICS*, pages 57–66, 2010.

Kikui, G.; Sumita, E.; Takezawa, T., and Yamamoto, S. Creating corpora for speech-to-speech translation. In *Proceedings of EUROSPEECH*, pages 381–384, 2003.

Kitamura, Y. and Tsujimoto, H. An initial evaluation of multiple animated information. In *Proceedings of AAMAS*, pages 1032–1033, 2003.

Klemmer, S. R.; Sinha, A. K.; Chen, J.; Landay, J. A.; Aboobaker, N., and Wang, A. SUEDE: A wizard of oz prototyping tool for speech user interfaces. In *Proceedings of UIST*, pages 1–10, 2000.

Koulouri, T. and Lauria, S. Exploring miscommunication and collaborative behaviour in human-robot interaction. In *Proceedings of SIGdial*, pages 111–119, 2009.

Krause, D. Using an interpretation system - some observations in hidden operator simulations of VERBMOBIL. In *Proceedings of ECAI*, pages 41–54, 1996.

Krebber, J.; Möller, S.; Pegam, R.; Jekosch, U.; Melichar, M., and Rajman, M. Wizard-of-Oz tests for a dialog system in smart homes System Evaluation Set-Up Tests with the INSPIRE WoZ tool. In *Proceedings of CFA/DAGA*, pages 1149–1150, 2004.

Krüger, A.; Aslan, I., and Zimmer, H. The effects of mobile pedestrian navigation systems on the concurrent acquisition of route and survey knowledge. In *Proceedings of MobileHCI*, pages 446–450, 2004.

Lamel, L. Spoken language dialog system development and evaluation at LIMSI. In *Proceedings of ISSD*, pages 9–17, 1998.

Lamel, L.; Rosset, S.; Gauvain, J. L., and Bennacef, S. The LIMSI arise system for train travel information. In *Proceedings of ICASSP*, pages 501–504, 1999.

Landauer, T. K. Psychology as a mother of invention. In *Proceedings of CHI and GI*, pages 333–335, 1987.

Lathrop, B.; Cheng, H.; Weng, F.; Mishra, R.; Chen, J.; Bratt, H.; Cavedon, L.; Bergmann, C.; Hand-Bender, T.; Pon-Barry, H.; Bei, B.; Raya, M., and Shriberg, L. A wizard of oz framework for collecting spoken human-computer dialogs: an experiment procedure for the design and testing of natural language in-vehicle technology. In *Proceedings of 12th International Congress on Intelligent Transportation Systems, San Francisco*, 2004.

Lee, M. and Billinghurst, M. A wizard of oz study for an ar multimodal interface. In *Proceedings of ICMI*, pages 249–256, 2008.

Lee, S.; Henderson, V.; Hamilton, H.; Starner, T.; Brashear, H., and Hamilton, S. A gesture-based american sign language game for deaf children. In *Proceedings of CHI*, pages 1589–1592, 2005.

Lee, S. Y.; Mott, B. W., and Lester, J. C. Investigating director agents' decision making in interactive narrative: A wizard-of-oz study. In *Proceedings of INT3*, 2010.

Lewin, K. Action reserach and minority problems. *Journal of Social Issues*, 2(4):34–46, 1946.

Li, Y.; Hong, J. I., and Landay, J. A. Topiary: A tool for prototyping location-enhanced applications. In *Proceedings of UIST*, pages 217–226, 2004.

Li, Y.; Welbourne, E., and Landay, J. A. Design and experimental analysis of continuous location tracking techniques for wizard of oz testing. In *Proceedings of CHI*, pages 1019–1022, 2006.

Li, Y.; Hong, J. I., and Landay, J. A. Design challenges and principles for wizard of oz testing of location-enhanced applications. *IEEE Pervasive Computing*, 6(2):70–75, 2007.

Li, Y.; Cao, X.; Everitt, K.; Dixon, M., and Landay, J. A. Framewire: a tool for automatically extracting interaction logic from paper prototyping tests. In *Proceedings of CHI*, pages 503–512, 2010.

Life, A.; Salter, I.; Temem, J. N.; Bernard, F.; Rosset, S.; Bennacef, S., and Lamel, L. Data collection for the mask kiosk: Woz vs prototype system. In *Proceedings of ICSLP*, pages 1672–1675, 1996.

Lisowska, A.; Rajman, M., and Bui, T. H. ARCHIVUS: A system for accessing the content of recorded multimodal meetings. In *Proceedings of MLMI*, pages 291 – 304, 2005.

Litman, D. J. and Pan, S. Predicting and adapting to poor speech recognition in a spoken dialogue system. In *Proceedings of AAAI*, pages 100–111, 2000.

Liu, A. L. and Li, Y. BrickRoad: A light-weight tool for spontaneous design of location-enhanced applications. In *Proceedings of CHI*, pages 295–298, 2007.

Liu, A. L.; Hile, H.; Kautz, H.; Borriello, G.; Brown, P. A.; Harniss, M., and Johnson, K. Indoor wayfinding: Developing a functional interface for individuals with cognitive impairments. In *Proceedings of ASSETS*, pages 95–102, 2006.

Liu, L. and Khooshabeh, P. Paper or interactive? a study of prototyping techniques for ubiquitous computing environments. In *Proceedings of CHI*, pages 1030–1031, 2003.

Liu, X.; Rieser, V., and Lemon, O. A wizard-of-oz interface to study information. presentation strategies for spoken dialogue systems. In *Proceedings of IWSDS*, 2009.

Lu, D. V.; Smart, W. D., and Park, M. Polonius: A wizard of oz interface for hri experiments. In *Proceedings of HRI*, pages 197–198, 2011.

Lyons, K. and Skeels, C. Providing Support for Mobile Calendaring Conversations: A Wizard of Oz Evaluation of Dual – Purpose Speech. In *Proceedings of MobileHCI*, pages 243–246, 2005.

MacIntyre, B.; Gandy, M.; Dow, S., and Bolter, J. D. DART: A toolkit for rapid design exploration of augmented reality experiences. In *Proceedings of UIST*, pages 197–206, 2004.

Mai, N. T. T.; Hai, T. T. T., and Son, N. V. Wizard of oz for designing hand gesture vocabulary in human-robot interaction. In *Proceedings of KSE*, pages 232–238, 2011.

Mäkelä, K.; Salonen, E.-P.; Turunen, M.; Hakulinen, J., and Raisamo, R. Conducting a wizard of oz experiment on a ubiquitous computing system doorman. In *Proceedings of IPNMD*, pages 115–119, 2001.

Markus, L.; Majchrzak, A., and Gasser, L. A design theory for systems that support emergent knowledge processes. *MIS Quarterly*, 26(3):179–212, 2002.

Maulsby, D.; Greenberg, S., and Mander, R. Prototyping an intelligent agent through wizard of oz. In *Proceedings of CHI*, pages 277–284, 1993.

Mavrikis, M. and Gutierrez-Santos, S. Not all wizards are from oz: Iterative design of intelligent learning environments by communication capacity tapering. *Computers & Education*, 54(3):641–651, 2010.

Mavrikis, M.; Dragon, T., and Mclaren, B. M. The design of wizard-of-oz studies to support students ' learning to learn together. In *Proceedings of ITS*, 2012.

McGuire, R. M.; Hernandez-Rebollar, J.; Starner, T.; Henderson, V.; Brashear, H., and Ross, D. S. Towards a one-way american sign language translator. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, pages 2–7, 2004.

Meguro, T.; Minami, Y.; Higashinaka, R., and Dohsaka, K. Wizard of oz evaluation of listening-oriented dialogue control using pomdp. In *Proceedings of IEEE ASRU*, pages 318–323, 2011.

Melichar, M. and Cenek, P. From vocal to multimodal dialogue management. In *Proceedings of ICMI*, pages 59–67, 2006.

Molin, L. Wizard-of-oz prototyping for cooperative interaction design of graphical user interfaces. In *Proceedings of NordiCHI*, pages 425–428, 2004.

Morgan, D. L. *Focus Groups as Qualitative Research*. Sage Publications Inc., 1997.

Muller, M. J. and Kogan, S. *Grounded Theory Method in HCI and CSCW*. 2010.

Munteanu, C. and Boldea, M. MDWOZ: A wizard of oz environment for dialog systems development. In *Proceedings of LREC*, 2000.

Nardi, B. *Context and Consciousness: Activity Theory and Human-Computer Interaction*. MIT Press, 1995.

Nunamaker, J. F.; Minder, C., and Purdin, T. D. M. System development in information systems research. *Journal of Management Information Systems*, 7(3):89–106, 1991.

Otto, M.; Friesen, R., and Dietmar, R. Message oriented middleware for flexible wizard of oz experiments in hci. In *Lecture Notes in Computer Science*, pages 121–130. 2011.

Oviatt, S. L.; Cohen, P. R.; Fong, M., and Frank, M. A rapid semi-automatic simulation technique for investigating interactive speech and handwriting. In *Proceedings of ICSLP*, pages 1351–1354, 1992.

Paay, J.; Kjeldskov, J.; Christensen, A.; Ibsen, A.; Jensen, D.; Nielsen, G., and Vutborg, R. Location-based storytelling in the urban environment. In *Proceedings of OZCHI*, pages 122–129, 2008.

Paiva, A.; Andersson, G.; Ho, K., and Martinho, C. SenToy in FantasyA: Designing an affective sympathetic interface to a computer game. *Personal and Ubiquitous Computing*, 6(5-6): 378–389, 2002.

Paul, M. Overview of the iwslt 2009 evaluation campaign. In *Proceedings of IWSLT*, 2009.

Peissner, M.; Heidmann, F., and Ziegler, J. Simulating recognition errors in speech user interface prototyping. In *Proceedings of HCI International*, pages 233–237, 2001.

Pettersson, J. S. Visualising interactive graphics design for testing with users. *Digital Creativity*, 13(3):144–156, 2002.

Pettersson, J. S. and Siponen, J. Ozlab – a simple demonstration tool for prototyping interactivity. In *Proceedings of NordiCHI*, pages 295–296, 2002.

Piemot, P. P.; Felciano, R. M.; Stancel, R.; Marsh, J., and Yvon, M. Designing the PenPal: Blending hardware and software in a user-interface for children. In *Proceedings of ACM CHI*, pages 511–518, 1995.

Pilkington, R. M. Question-answering for intelligent on-line help: The process of intelligent responding. *Cognitive Science*, 16(4):455–489, 1992.

Pirker, H.; Loderer, G., and Trost, H. Thus spoke the user to the wizard. In *Proceedings of Eurospeech*, pages 1171–1174, 1999.

Porzel, R. How computers (should) talk to humans. In *Proceedings of Workshop on 'How People talk to Computers, Robots and other Artificial Communication Partners*, pages 7–37, 2006.

Qvarfordt, P.; Jönsson, A., and Dahlbäck, N. The role of spoken feedback in experiencing multimodal interfaces as human-like. In *Proceedings of ICMI*, pages 250–257, 2003.

Rajman, M.; Ailomaa, M.; Lisowska, A.; Melchiar, M., and Armstron, S. Extending the wizard of oz methodology for language-enabled multimodal systems. In *Proceedings of LREC*, pages 2531–2536, 2006.

Raux, A.; Bohus, D.; Langner, B.; Black, A. W., and Eskenazi, M. Doing research on a deployed spoken dialogue system: One year of Let's Go! experience. In *Proceedings of INTERSPEECH*, pages 65–68, 2006.

Read, J. C.; Mazzone, E., and Höysniemi, J. Wizard of oz evaluations with children – deception and discovery. In *Proceedings of IDC*, 2005.

Rice, M. and Alm, N. Sociable TV: Exploring user-led interaction design for older adults. In *Proceedings of the European Conference on Interactive Television*, pages 126–135, 2007.

Rieser, V. and Lemon, O. Learning effective multimodal dialogue strategies from wizard-of-oz data: Bootstrapping and evaluation. In *Proceedings of ACL*, pages 638–646, 2008a.

Rieser, V. and Lemon, O. Simulation-based learning of optimal multimodal presentation strategies from wizard-of-oz data. In *Proceedings of AISB*, 2008b.

Rieser, V. and Lemon, O. Learning human multimodal dialogue strategies. *Natural Language Engineering*, 16(1):3–21, 2010.

Rieser, V.; Lemon, O., and Liu, X. Optimising information presentation for spoken dialogue systems. In *Proceedings of ACL*, pages 1009–1018, 2010.

Rizzo, P.; Lee, H.; Shaw, E.; Johnson, W. L.; Wang, N., and Mayer, R. E. A semi-automated wizard of oz interface for modeling tutorial strategies. In *Proceedings of UM*, pages 174–178, 2005.

Saint-Aimé, S.; Grandgeorge, M.; Le-Pévédic, B., and Duhaut, D. Evaluation of emi interaction with non-disabled children in nursery school using wizard of oz technique. In *Proceedings of IEEE Robio*, pages 1147–1153, 2011.

Salber, D. and Coutaz, J. A wizard of Oz platform for the study of multimodal systems. In *Proceedings of INTERACT and CHI*, pages 95–96, 1993a.

Salber, D. and Coutaz, J. Applying the wizard of oz technique to the study of multimodal systems. In *Proceedings of EWHCI*, pages 219–230, 1993b.

Saulnier, P.; Sharlin, E., and Greenberg, S. Exploring interruption in HRI using wizard of oz. In *Proceedings of HRI*, pages 125–126, 2010.

Scherer, S. and Schwenker, F. Emotion recognition from speech using multi-classifier systems and rbf-ensembles. *Studies in Computational Intelligence*, 83:49–70, 2008.

Scherer, S. and Strauß, P.-M. A flexible wizard of oz environment for rapid prototyping. In *Proceedings of LREC*, pages 958–961, 2008.

Schieben, A.; Heesen, M.; Schindler, J.; Kelsch, J., and Flemisch, F. The theater-system technique: Agile designing and testing of system behavior and interaction, applied to highly automated vehicles. In *Proceedings of AutomotiveUI*, pages 43–46, 2009.

Schneider, A. H. *Intrinsic and Extrinsic Component Evaluation in Interactive Multilingual Speech Applications*. PhD thesis, Trinity College, University of Dublin, 2013.

Schneider, A. H.; van der Sluis, I., and Luz, S. Comparing intrinsic and extrinsic evaluation of mt output in a dialogue system. In *Proceedings of IWSLT*, pages 329–336, 2010.

Schön, D. A. *The reflective practitioner: how professionals think in action*. Basic Books, 1984.

Schuler, D. and Namioka, A. *Participatory Design: Principles and Practices*. Lawrence Erlbaum, 1993.

Sefelin, R.; Bechinie, M.; Müller, R.; Seibert-Giller, V.; Messner, P., and Tscheligi, M. Landmarks: Yes, but which? five methods to select optimal landmarks for a landmark- and speech-based guiding system. In *Proceedings of MobileHCI*, pages 287–290, 2005.

Séquin, C. H. Rapid prototyping: a 3d visualization tool takes on sculpture and mathematical forms. *Communications of the ACM*, 48(6):66–73, 2005.

Serrano, M. and Nigay, L. Temporal aspects of care-based multimodal fusion: From a fusion mechanism to composition components and woz components. In *Proceedings of ICMI*, pages 177–184, 2009.

Serrano, M. and Nigay, L. A wizard of oz component-based approach for rapidly prototyping and testing input multimodal interfaces. *Journal of Multimodal User Interfaces*, 3(3):215–225, 2010.

Shiomi, M.; Kanda, T.; Koizumi, S.; Ishiguro, H., and Hagita, N. Group attention control for communication robots with wizard of oz approach. In *Proceedings of HRI*, pages 121–128, 2007.

Silvervarg, A. and Jönsson, A. Towards a conversational pedagogical agent capable of affecting attitudes and self-efficacy. In *Proceedings of the Second Workshop on Natural Language Processing in Support of Learning: Metrics, Feedback and Connectivity*, 2010.

Skantze, G. Exploring human error recovery strategies: Implications for spoken dialogue systems. *Speech Communication*, 45(3):325–341, 2005.

Skantze, G. and Hjalmarsson, A. Towards incremental speech generation in dialogue systems. In *Proceedings of SIGdial*, pages 1–8, 2010.

Skantze, G.; House, D., and Edlund, J. User responses to prosodic variation in fragmentary grounding utterances in dialog. In *Proceedings of ICSLP*, pages 2002–2005, 2006.

Smeddinck, J.; Wajda, K.; Naveed, A.; Touma, L.; Chen, Y.; Hasan, M. A.; Latif, M. W., and Porzel, R. QuickWoZ: A multi-purpose wizard-of-oz framework for experiments with embodied conversational agents. In *Proceedings of IUI*, pages 427–428, 2010.

Soegaard, M. *Prototyping*. Retrieved 21st June 2012 from http://www.interaction-design.org/encyclopedia/prototyping.html, 2010.

Somers, H. Language engineering and the pathway to healthcare: A user-oriented view. In *NAACL Workshop on Medical Speech Translation*, pages 32–39, 2006.

Starlander, M. Using a wizard of oz as a baseline to determine which system architecture is the best for a spoken language translation system. In *Proceedings of Nodalida*, pages 161–164, 2007.

Steinfeld, A. and Scassellati, B. The oz of wizard : Simulating the human for interaction research. In *Proceedings of HRI*, pages 101–107, 2009.

Stenton, P. and Whittaker, S. User studies and the design of natural language systems. In *Proceedings of EACL*, pages 116–123, 1989.

Strauss, A. *Qualitative analysis for social scientists*. Cambridge University Press, 1987.

Strauß, P.-M.; Hoffmann, H.; Minker, W.; Neumann, H.; Palm, G.; Scherer, S.; Schwenker, F.; Traue, H. C.; Walter, W., and Weidenbacher, U. Wizard-of-oz data collection for perception and interaction in multi-user environments. In *Proceedings of LREC*, pages 2014–2017, 2006.

Strauß, P.-M.; Hoffmann, H.; Minker, W.; Neumann, H.; Palm, G.; Scherer, S.; Traue, H. C., and Weidenbacher, U. The pit corpus of german multi-party dialogues. In *Proceedings of LREC*, pages 2442–2445, 2008.

Stücker, S.; Fügen, C.; Kraft, F., and Wölfel, M. The ISL 2007 english speech transcription system for european parliament speeches. In *Proceedings of INTERSPEECH*, 2007.

Stüker, S.; Zong, C.; Reichert, J.; Cao, W.; Kolss, M.; Xie, G.; Peterson, K.; Ding, P.; Arranz, V.; Yu, J., and Waibel, A. Speech-to-speech translation services for the olympic games 2008. In *Proceedings of MLMI*, pages 297–308, 2006.

Sutton, S.; Cole, R.; de Vielliers, J.; Schalkwyk, J.; Vermeulen, P.; Macon, M.; Yan, Y.; Kaiser, E.; Rundle, B.; Shobaki, K.; Hosom, P.; Kain, A.; Wouters, J.; Massaro, D., and Cohen, M. Universal speech tools: The CSLU toolkit, 1998.

Suwa, M. and Tversky, B. What architects see in their sketches: implications for design tools. In *Proceedings of CHI*, pages 191–192, 1996.

Takayama, L.; Leung, L.; Jiang, X., and Hong, J. I. You're getting warmer ! how proximity information affects search behavior in physical spaces. In *Proceedings of CHI*, pages 1028–1029, 2003.

Thomas, G. and James, D. Reinventing grounded theory: some questions about theory, ground and discovery. *British Educational Research Journal*, 6(32):767–795, 2006.

Trudel, C.-A. and Payne, S. J. Reflection and goal management in exploratory learning. *International Journal of Human-Computer Studies*, 42(3):307–339, 1995.

Tsovaltzi, D.; Rummel, N.; Pinkwart, N.; Harrer, A.; Scheuer, O.; Braun, I., and McLaren, B. M. Cochemex: Supporting conceptual chemistry learning via computer-mediated collaboration scripts. In *Proceedings of EC-TEL*, pages 437–448, 2008.

Turunen, M. and Hakulinen, J. Jaspis- a framework for multilingual adaptive speech applications. In *Proceedings of ICSLP*, pages 719–722, 2000.

Turunen, M.; Hakulinen, J.; Salonen, E.-P.; Kainulainen, A., and Helin, L. Spoken and multimodal bus timetable systems: Design , development and evaluation. In *Proceedings of SPECOM*, pages 389–392, 2005.

Villano, M.; Crowell, C. R.; Wier, K.; Tang, K.; Thomas, B.; Shea, N.; Schmitt, L. M., and Diehl, J. J. DOMER: A wizard of oz interface for using interactive robots to scaffold social skills for children with autism spectrum disorders. In *Proceedings of HRI*, pages 279–280, 2011.

Voida, S.; Podlaseck, M.; Kjeldsen, R., and Pinhanez, C. A study on the manipulation of 2d objects in a projector/camera – based augmented reality environment. In *Proceeedings of CHI*, pages 611–620, 2005.

Wahlster, W. *Verbmobil: foundations of speech-to-speech translation*. Springer, 2000.

Walker, M. A.; Rudnicky, A.; Prasad, R.; Aberdeen, J.; Bratt, E. O.; Garofolo, J.; Hastie, H.; Le, A.; Pellom, B.; Potamianos, A.; Passoneau, R.; Roukos, S.; Sanders, G.; Seneff, S., and Stallard, D. DARPA communicator: cross-system results for the 2001 evaluation. In *Proceedings of ICSLP*, pages 269–272, 2002.

Walter, S.; Hrabal, D.; Scheck, A.; Kessler, H.; Bertrand, G.; Nothdurft, F.; Minker, W., and Traue, H. Individual emotional profiles in wizard-of-oz- experiments. In *Proceedings of PETRA*, 2010.

Wenger, E. *Communities of Practice: Learning, Meaning, and Identity*. Cambridge University Press, 1998.

White, K. F. and Lutters, W. G. Behind the curtain: Lessons learned from a wizard of oz field experiment. *SIGGROUP Bulletin*, 24(3):129–135, 2003.

Whittaker, S.; Walker, M. A., and Moore, J. D. Fish or fowl: A wizard of oz evaluation of dialogue strategies in the restaurant domain. In *Proceedings of LREC*, 2002.

Williams, J. D. and Young, S. Characterizing Task-Oriented Dialog using a Simulated ASR Channel. In *Proceedings of ICSLP*, pages 185–188, 2004.

Winterboer, A. and Moore, J. D. Evaluating information presentation strategies for spoken recommendations. In *Proceedings of RecSys*, pages 157–160, 2007.

Wirén, M.; Eklund, R.; Engberg, F., and Westermark, J. Experiences of an in-service wizard-of-oz data collection for the deployment of a call-routing application. In *Proceedings of NAACL-HLT*, pages 56–63, 2007.

Wooffitt, R.; Fraser, N. M.; Gilbert, N. G., and McGlashan, S. *Humans, Computers and Wizards*. Routledge, 1997.

Yang, Y.; Okamoto, M., and Ishida, T. Applying wizard of oz method to learning interface agent. *IEICE Trans. Fundamentals*, E00-A(1), 2000.

Zobl, M.; Geiger, M.; Bengler, K., and Lang, M. A usability study on hand gesture controlled operation of in-car devices. In *Proceedings of International Conference on Human Computer Interaction*, pages 166–168, 2001.