

# Observing the Wizard: In Search of a generic Interface for Wizard of Oz Studies

Stephan Schlögl  
Trinity College Dublin  
Ireland  
schlogls@cs.tcd.ie

Gavin Doherty  
Trinity College Dublin  
Ireland  
Gavin.Doherty@cs.tcd.ie

Nikiforos Karamanis  
Trinity College Dublin  
Ireland  
Nikiforos.Karamanis@cs.tcd.ie

Anne Schneider  
Trinity College Dublin  
Ireland  
schneia@cs.tcd.ie

Saturnino Luz  
Trinity College Dublin  
Ireland  
Saturnino.Luz@cs.tcd.ie

## ABSTRACT

Early prototyping is important for developing high-quality software. Wizard of Oz (WOZ) techniques are particularly useful for early stage evaluations of applications using Language Technology Components (LTC). However, an effective interface to support the task of the wizard is crucial. While several wizard interfaces have been built to date, most of them were designed for specific experiments. The more general issue of supporting the sometimes highly demanding cognitive task of the wizard has remained largely unexplored in the HCI literature. In this paper we report on work in progress that explores the wizard task and aims to define a generic wizard user interface layout that can be used in different experimental settings using LTCs. Informed by the literature different scenarios for WOZ experiments were identified. Based on a first WOZ experiment in which different wizards were observed and their behaviour was analysed, insight into the tasks of a wizard was gained. The results of this experiment were combined with sketching to identify high-level concepts that can be applied in a generic wizard interface.

## Categories and Subject Descriptors

H.5.2 [User Interfaces]: Ergonomics; H.5.2 [User Interfaces]: Graphical user interfaces (GUI); H.5.2 [User Interfaces]: Natural Language; H.5.2 [User Interfaces]: User-centered design.

## General Terms

Design, Human Factors.

## Keywords

Wizard of Oz; User Evaluation; Language Technology; Sketching.

## 1. INTRODUCTION

Wizard of Oz (WOZ) is a prototyping method that uses a human wizard to mimic some of the functionality of a system in an exper-

imental setting. We report on work in progress on a WOZ prototyping framework, which aims at the generic creation of wizard interfaces for different scenarios testing Language Technology Components (LTCs). Our main ambition is to support the highly demanding cognitive task of the wizard [15]. The use of LTCs has significantly increased in recent years as their performance has improved. Components used in applications comprise Automatic Speech Recognition (ASR), Speech Synthesis (SS) and Machine Translation (MT). Examples include speech-based interaction in cars to keep the driver's attention on the road and the usage of web-based translation tools such as GOOGLE TRANSLATE and YAHOO! BABEL FISH to understand text written in a foreign language. 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 [15]. One way of supporting the wizard is to provide her with an appropriate interface that helps to fulfil her task efficiently, so that human intervention is imperceptible to the user interacting with the prototype.

In this paper we discuss first insights into the kind of problems a wizard faces, and how certain design choices might help to solve them. We begin with a study of the literature that reports on the usage of WOZ. Subsequently we present four different scenarios that helped to define the design space for a common wizard interface. A prototype for one of the scenarios was built and a study was conducted in which several wizards were observed and their behaviour analysed. Finally, we show how results from this study were combined with sketching to identify generic interface concepts that can be applied to all the presented scenarios.

## 2. RELATED WORK

Wizard of Oz has been used as a design method for almost 40 years, beginning with Erdman & Neal [7] who tested the concept of a self-service airline ticket kiosk. Dahlbäck et al. [5] highlight the use of WOZ as an interaction design technique. Its application primarily serves the purpose of informing the design of a not yet existing technology component. Several researchers have pointed to the importance of early user studies informing the design and development of computer systems (e.g. [9], [2]). Based on [13], for instance, the quality of the final system depends to a degree on the number of participants that are tested before its implementation, and the ability of the collector to use the gathered information to inform the system design. In the case of LTCs WOZ studies are an efficient way of achieving this. They facilitate the development of realistic and rich dialogue models and support the designer in producing a more natural interaction. As opposed to assuming a certain dialogue flow, WOZ experiments can be used to explore the dialogue space in more detail. In recent years WOZ has been used in various projects to gain early feedback on user behaviour. Rajman et al. [14] used it to test multi-modal access to a media data base and Karpov et al. [11] as an analysis instrument for their spoken dialogue system. Furthermore, it was used to simulate a virtual doorman [12] and test a web based companion [1]. Aspects that were explored comprise dialogue models, multimodal information retrieval, human agent interaction and emotion. So far most of the wizard interfaces were built for one set of experiments only. Less effort has been spent on understanding the role of the wizard and on designing interfaces to support it.

## 3. SCENARIOS

Personas and scenarios are widely used techniques in interaction design [4]. 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 this method as a way of gathering requirements for an interface from a wizard point of view. Inspired by situations exploited in previous research (cf. [17], [3], [8]) we defined four different experimental scenarios (see Figure 1), which reflect attempts to combine various LTCs (ASR, MT, SS) in multilingual settings. With those scenarios in mind we designed different interface layouts that would support the wizard side in various experiments.

<b>Scenario 1: A multilingual conversation</b>
A smartphone application used to translate between a Japanese taxi driver and an English speaking business traveller.
<b>Scenario 2: Location based storytelling</b>
A travel device that tells stories to tourists triggered by their current location.
<b>Scenario 3: The multilingual instructor</b>
A multilingual 'Avatar' that talks a user through the setup process of a video game console.
<b>Scenario 4: The adaptive helpdesk</b>
An multilingual InfoPoint terminal that advises customers about the best internet connection that fulfils their specific requirements.

Figure 1: Scenarios inspired by situations in previous WOZ studies.

## 4. A FIRST PROTOTYPE

In order to get a hands-on impression of the task of the wizard a first prototype of a WOZ tool was built. We elaborated on scenario 4 (see Figure 1) to simulate the interaction between a German speaking customer and a system recommending appropriate Internet con-

nection bundles. To do this, we first defined a preliminary dialogue and tested it for accuracy and completeness by using a chat tool and simple copy and paste mechanisms. Based on this we then developed a wizard interface to support the 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, three different sets of pre-translated dialogue utterances were created. The first set was translated by a German native speaker, the second via GOOGLE TRANSLATE and for the third we used the SYSTRAN machine translation engine.

### 4.1 Interface Layout

The interface of the prototype (see Figure 2) was split into two areas. The 'Dialogue flow' was shown on the left side of the interface (see Figure 3), 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'. The wizard had to choose between these utterances to respond to the test subject and progress through the dialogue flow. When an utterance was chosen that would lead to the next stage in the dialogue flow the display of utterances was updated automatically to show a new set of utterances appropriate for the stage of the dialogue. The wizard could also switch between dialogue stages manually by clicking on the links under 'Dialogue flow'. Utterances aiming to help the wizard recover from misunderstandings, could be chosen from an orange box in the 'Not understood' section of the screen.

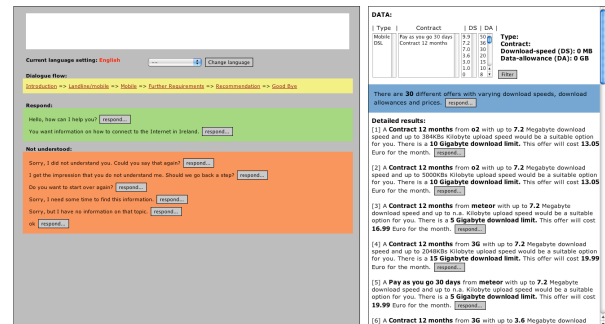


Figure 2: The wizard interface that was tested.

All utterances were displayed in English. The current utterance, which was sent to the test subject, was displayed in German in a white box on top of this area. A drop-down menu allowed defining the set of pre-translations (i.e. native German, Google translate, or Systran). The display was updated whenever a new utterance was sent, i.e. there was no record of the history of interactions between the wizard and the test subject.

The right side of the interface was dedicated to the domain data (see Figure 4), i.e. information on current offers of Internet connection as suggested in scenario 4. This area of the screen was therefore showing utterances recommending different offers. Based on the flow of the dialogue those were filtered automatically so that the wizard had fewer offers to choose from as the dialogue progressed. For situations in which a test subject would change her requirements, filters on the top of this area were implemented to allow the wizard to change the display of offers manually without going back in the dialogue.

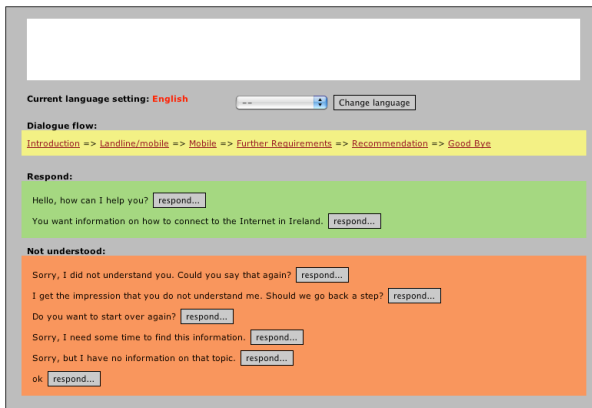


Figure 3: The left area of the wizard interface showed dialogue-related information.

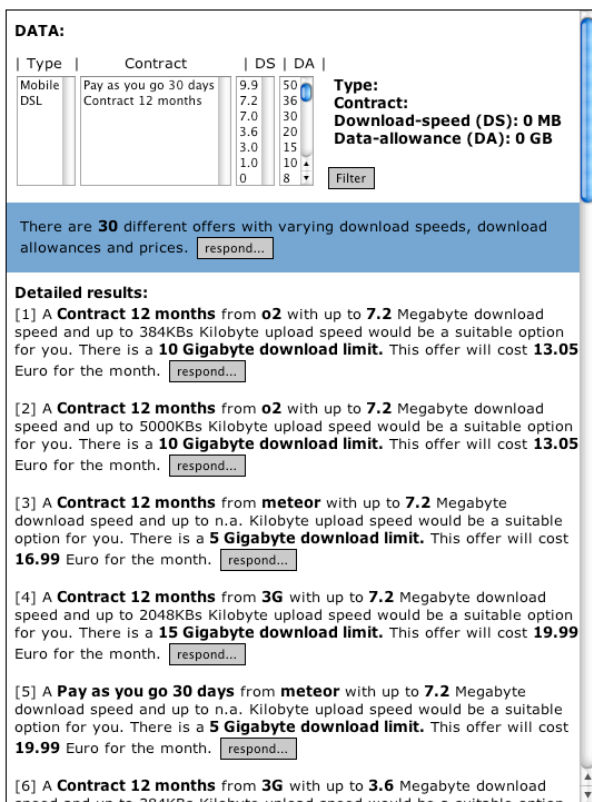


Figure 4: The right area of the wizard interface showed domain-specific data.

## 5. EVALUATION

A usability study was conducted in which different wizards were using the prototype. This was supplemented with observations on how the developer himself was using the tool. Important insight into what kind of problems a wizard is experiencing at runtime was gained.

### 5.1 Usability Study

A formal usability study was conducted. Driven by scenario 4, several wizards were observed using the prototype. The developer of

the interface (who is a native German speaker) acted as a customer.

#### 5.1.1 Test Setting

Following the scenario we asked 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 of the task and allowed to explore the wizard interface for about five minutes. Then, the wizard interacted with the customer. The goal of the test was to evaluate the usability of the wizard interface as well as the completeness and accuracy of the dialogue utterances it supports.

#### 5.1.2 Method

A formal usability test approach [6] was chosen to analyze 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 highly 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.

#### 5.1.3 Participants

Two members of our research team and two external people acted as a wizard. Different levels of familiarity with the tool were apparent. 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.

## 5.2 Set-up

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



Figure 5: 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'.

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 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.

### 5.3 Observations

In addition to testing the interface with different wizards, we also conducted extensive observations of how the developer himself uses the tool. This took place during another experiment, which explored how MT quality impacts on the interaction from the point of view of the customer. 12 German native speakers acted as customers for this experiment following instructions on two different tasks. In the first task customers were searching for a mobile Internet connection without compulsory contract duration. In the second task they were interested in a landline (DSL) connection offering a high download speed and download allowance. The set-up of these sessions were the same as in the usability test, except for the fact that the developer always acted as wizard and the customers were not told that they were interacting with a human. The aim of these sessions from the point of view of interface development was to observe an experienced wizard in action and supplement the data collected from these observations with those of the usability test.

## 6. RESULTS

Based on the results of the usability tests and the observation sessions we were able to identify several issues regarding the usability of the wizard interface as well as the wizard task. Generally those findings can be subdivided into interface insights, dialogue insights and general recommendations for testing wizard interfaces.

### 6.1 Interface Insights

The identified problems with the interface were caused by its general layout, the representation of the dialogue flow and domain data as well as by the used terminology. In addition timing issues exacerbated the task of the wizard.

#### 6.1.1 General Interface Layout

The first main issue observed was that most wizards (an exception being the developer of the wizard interface) did not understand why the wizard interface was separated into different areas. Especially, wizards had difficulties to switch their attention from one area 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.

#### 6.1.2 Representation of the Dialogue Flow and Finding Responses

Wizards had problems to control the dialogue and find suitable utterances since they were not familiar with the set of utterances before the interaction started. Furthermore, the representation of the dialogue flow and the current state were unclear. The feedback utterance *ok* which was intended as a filling response for instance 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 *Are you satisfied?* but the utterance which was in the set to fulfil this purpose was *Is there anything else I can do for you?*. Similarly another wizard was searching for a specific utterance that would summarize the amount of remaining offers. Such an utterance was available, but it seems that the blue background colour for this utterance, which was meant to highlight it (see Figure 4), was rather hindering it. In addition what seemed to be unclear is whether there was any structure within a particular state in 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.).

#### 6.1.3 Representation of the Domain Data

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, wherefore the chosen parameters were not applied. Furthermore, the only way to clear already set filters was to log out and login again. A function to switch back to un-filtered results was missing. It must also be pointed out that even the developer of the interface had problems operating the filters at runtime, which again highlights the high pressure a wizard is experiencing during an experiment.

#### 6.1.4 Terminology Issues

A third 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. A second 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 subject and not the language of the wizard interface. One aspect that could have been problematic for wizards is that they need 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 showing up in the way it was sent to the customer (i.e. in machine translated German) in the textbox on top of the wizard interfaces. Yet wizards did not perceive this 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.

#### 6.1.5 Timing Issues

Finally the missing information about a subject's status of attention seemed to be a problem. One wizard, for instance, accidentally pressed a response button too early without waiting for the subject's response to the previous utterance. As it turned out, the subject had not responded because she was not finished reading the utterance. For the wizard, however, it felt as if the subject was waiting for the next utterance. There seems to be a need for notifying the wizard when a customer has processed a response. The problem of these acknowledgement tokens or backchannels has also been discussed

by [10]. Also it was mentioned that wizards had to guess when a test subject would be available so they could start the dialogue.

## 6.2 Dialogue Insights

In this section we discuss problems with the dialogue structure. Some of these issues relate 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 dialogue itself, illustrating the types of issue with the design of the specific dialogue that the WOZ technique is intended to uncover.

A main aspect that was observed is that it was hard for wizards to use the confirmation sentences that are commonly used in interactive voice response systems. It was mentioned that naive 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. In the exploratory study described here, a wizard is per se neither a customer care agent, nor a dialogue system. Therefore a more detailed test script for the wizard was recommended. Alternatively one might argue that a well-designed interface would need to actively enforce confirmation behaviour. It was also highlighted that if such a strategy needs 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 found to be not the case for this dialogue. If those utterances were missing, it was however shown that wizards tend to go on without confirming.

Another reported issue was that utterances that are used together should be grouped together and utterances after which the wizard does not wait for a customer response should be combined. These problems mainly occurred when a wizard needed to combine utterances from the right area of the interface (i.e. domain data) with utterances from the left area of the interfaces (i.e. general dialogue utterances). 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 a long time to find them.

Looking at the richness of the dialogue it was thought 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, it was asked why one would want to tell people how many offers there are left. 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 problematic for the dialogue flow.

Finally it was pointed out that a test subject might not know the difference between mobile and landline/DSL Internet. Similarly the difference between a contract and a pay-as-you-go option 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.

## 6.3 General Recommendations for Testing Wizard Interfaces

Two general recommendations emerged after analysing these initial experiments. First it was found to be difficult and tedious for the observer to map observed behaviour to the dialogue progress. As a result we provided the observer with a script as well as an optimal

dialogue flow. This made it easier for him to follow the happenings and spot deviations. The second aspect we found to be important for testing wizard interfaces is the type of people acting as wizards. That is, in order to gain consistent results wizards should have a certain amount of domain knowledge (i.e. in this example customer care agents of an Internet provider could be used).

## 7. DISCUSSION AND GENERIC FINDINGS

This first round of usability tests highlighted several problems with our prototype. Some of which can be categorized as basic usability problems that can be fixed relatively easy for a given scenario (e.g. terminology). Other, more generic problems, however, require further investigation. In the following we want to focus on the later category and discuss three problems we identified that could, if solved, contribute to the quality of a generic wizard interface layout.

### 7.1 Control over the Dialogue

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. Future explorations of how to improve the general layout of a wizard interfaces 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 need to be found that 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.

### 7.2 Feedback from a Test Subject

From a dialogue perspective certain timing issues seem salient. For example, it was shown that 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. Such shows that for a wizard interface it might be important to notify the wizard about the processing status of a test subject. It should, however, be mentioned that the situation of not knowing a test subject's status of attention reflects the possibilities of a real dialogue system more accurately. Giving the wizard too much information might 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 needs to be found.

### 7.3 Slot-filling Metaphor

From a rather functional point the main problem of our interface was the handling of filters, which can be seen as filling slots to extract the relevant domain data. That is, it was not understood by the wizards that those slots were mainly filled by the dialogue flow and only needed to be adapted (i.e. filter needed to be adjusted) in cases where the test subject would change her opinion throughout the dialogue. Doing so was meant to help the wizard to choose between fewer options as the dialogue progressed. From a wizard point of view, however, this behaviour was not followed. It seemed that since a filter possibility was given, wizards wanted to use it; they wanted to be in control. A solution for this problem might be a dialogue that is designed in a way so that no additional filtering is needed. An alternative way of representing the slots might be a different solution. Generally it seemed, however, that any offered functionality that would pull the wizard out of her direct interaction with a test subject, needs to be clear and very intuitive to use. Opaque interface behaviour would only confuse a wizard and influence her performance.

The next section of this paper will report on efforts undertaken to tackle the first of those generic problems identified - namely a generic interface layout that supports the wizard in controlling the dialogue flow.

## 8. A NEW INTERFACE LAYOUT

Our experiment clearly showed that a wizard needs to be able to follow and control the dialogue flow. Trying to tackle this problem we looked at the other scenarios (see Figure 1) in order to find generic interface concepts that can be applied in different WOZ settings. To do so we used paper prototypes and sketches as a means of quick design and comparison of ideas.

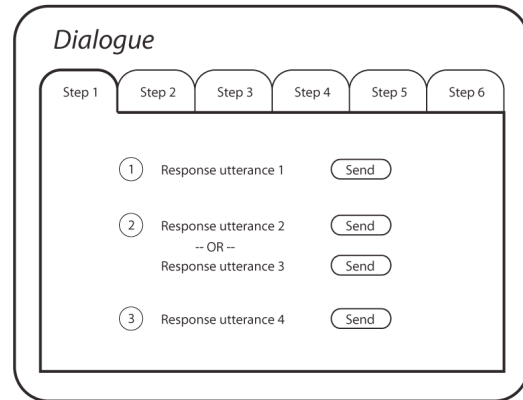
### 8.1 Sketching

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 [18]. Schön [16] 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 designers understanding of the situation. We used sketches to compare different interface layouts that would guide the wizard through the dialogue. Paper-based interfaces for our four scenarios (see Figure 1) were created. We were able to identify three main layout concepts that seemed to be applicable to all scenarios. These are discussed in the following.

#### 8.1.1 Dialogue

All of the layouts we designed had in common that the main area of the screen was dedicated to the running dialogue. That is, 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 amount 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 that can be applied to all the scenarios. Using a tab structure we were able to split the dialogue into different stages but at the same time maintain the overall structure which helps the

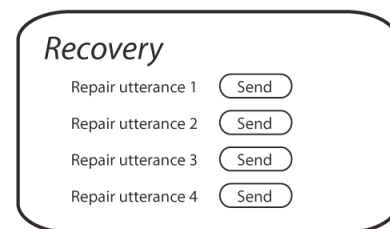
wizard to keep track of the dialogue progress and allows to quickly switch between different dialogue stages (see Figure 6).



**Figure 6:** A tab structure representing the dialogue and its stages may help the wizard to choose from a predefined set of response utterances.

#### 8.1.2 Recovery

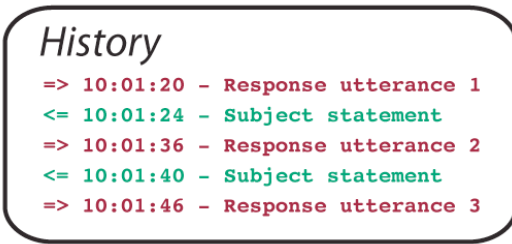
The second important concept we identified, as being relevant for an intuitive to use wizard interface is to support the common problem of error recovery. Since there is a probability of misunderstanding in any kind of dialogue situation, a wizard needs to be offered a way of recovery. Recovery might be gained through sending the last utterance once more or by using predefined repair utterances that would invoke a test subject 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 (see Figure 7).



**Figure 7:** A separate area for repair utterances should allow the wizard to act spontaneously.

#### 8.1.3 History

The third aspect we thought of being 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 already used and possibly what a test subject has replied. Also statements should be easily distinguishable. That is, a wizard should be able to see what came from a test subject and what she acting as a wizard sent. As one way of tackling this problem we propose the use of symbols and different colours (see Figure 8).



**Figure 8:** A history using different symbols and colours to differentiate between wizard and subject statements might support the wizard's task.

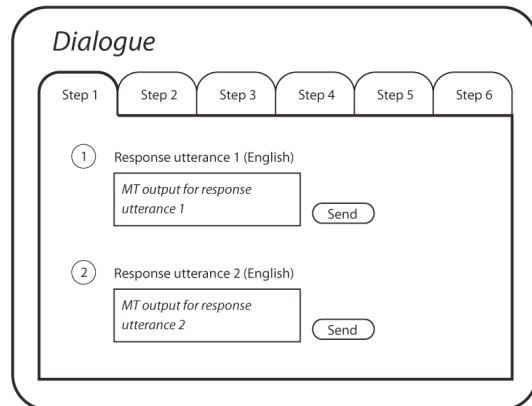
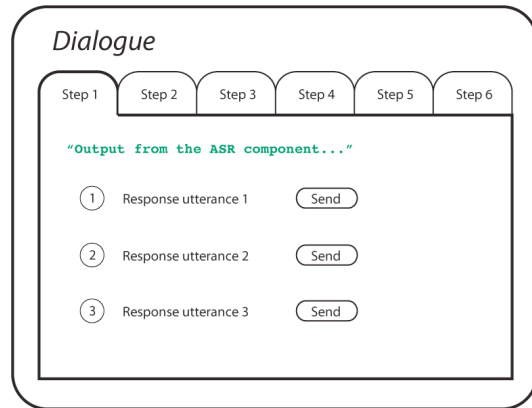
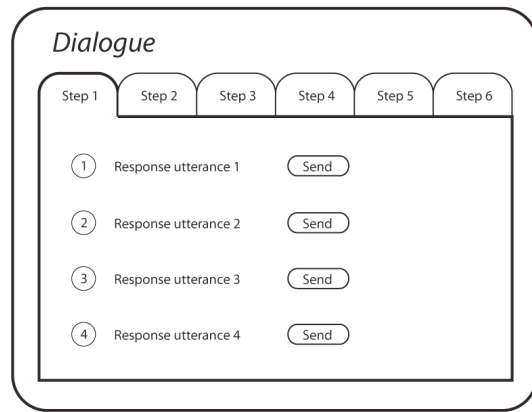
## 8.2 Level of Simulation

As discussed earlier, the main usage of WOZ is to create low-fidelity prototypes of applications using LTCs. In those prototypes the function of the wizard is mainly to simulate a not yet existing LTC. In an alternative experiment approach a wizard could, however, correct the output of a component or choose from output streams (e.g. different MT engines) having different quality levels. Changing the role of the wizard from a simulator to somebody who is rather correcting technology has an effect on the wizard interface.

In order to identify those influences we went back to our sketches and looked at how certain LTCs would change the wizard interface layout. 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 on the experiment setting, require a wizard to be able to choose from a set of utterances produced by different MT engines, or even allow her to correct them. Editable text boxes holding on-the-fly MT results could be a solution here. If SS 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 SS 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 adapted our generic interface layout for the different uses of LTCs. We kept the general layout concepts we identified earlier (i.e. dialogue, repair and history) but changed certain interface elements within them.

The outcome of this additional round of sketching is shown in Figure 9 were 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. A wizard would simply listen to what a subject says and then choose from a selection of possible responses. In the second version ASR is used for which the 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 a German-speaking subject is saying. Response utterances that are only available in English would be translated at runtime via a MT component. The interface



**Figure 9:** Different LTCs change the interface layout. Using the first layout the wizard would listen to a test subject and choose from a selection of response utterances. With the second layout the wizard would not hear the test subject but rather see the result of an ASR module. The last layout shows the interface when MT is used to translate English utterances into German. In this case the wizard would be able to correct translations before sending them.

would, however, allow the wizard to post-edit a translation and correct possible mistakes before sending it to the test subject.

## 9. CONCLUSION

Results from a study investigating the task of the wizard in WOZ experiments were reported. Based on the literature and a predefined scenario a first prototype of a WOZ tool was built. Different wizards using the tool in an experimental setting were observed and their behaviour was analysed. Drawing from the results of this experiment sketching was used to identify generic interface concepts that can be used to create a wizard interface supporting different experimental settings. First abstract screen layouts of such an interface were presented.

## 10. ACKNOWLEDGEMENTS

This research is supported by the Science Foundation Ireland (Grant 07/CE/I1142) as part of the Centre for Next Generation Localisation ([www.cngl.ie](http://www.cngl.ie)) at Trinity College Dublin. Furthermore we want to thank Dr. Ielka van der Sluis for her valuable comments and feedback.

## 11. REFERENCES

- [1] J. Bradley, O. Mival, and D. Benyon. Wizard of oz experiments for companions. In *Proceedings of BCS HCI*, pages 313–317, 2009.
- [2] B. Buxton. *Sketching user experiences*. Morgan Kaufman, 2007.
- [3] C. L. Chen and T. V. Raman. AxsJAX: a talking translation bot using google im: bringing web-2.0 applications to life. In *Proceedings of W4A*, pages 54–56, 2008.
- [4] A. Cooper. *The Inmates are running the asylum*. Sams Publishing, 2004.
- [5] N. Dahlbäck, A. Jönsson, and L. Ahrenberg. Wizard of oz studies: why and how. In *Proceedings of IUI*, pages 193–200, 1993.
- [6] J. S. Dumas and J. C. Redish. *A Practical Guide to Usability Testing*. Intellect, 1999.
- [7] R. L. Erdmann and A. S. Neal. Laboratory vs. field experimentation in human factors: an evaluation of an experimental self-service airline ticket vendor. *Human Factors*, 13:521–531, 1971.
- [8] P. Geutner, F. Steffens, and D. Manstetten. Design of the VICO spoken dialogue system: Evaluation of user expectations by wizard-of-oz experiments. In *Proceedings of LREC*, 2002.
- [9] J. D. Gould and C. Lewis. Designing for usability: key principles and what designers think. *Communications of ACM*, 28(3):300–311, 1985.
- [10] D. Jurafsky and J. H. Martin. *Speech and Language Processing*. Prentice Hall, second edition, 2008.
- [11] A. Karpov, A. Ronzhin, and A. Leontyeva. A semi-automatic wizard of oz technique for let'sfly spoken dialogue system. In *Lecture Notes in Computer Science: Text, Speech and Dialogue*. Springer, 2008.
- [12] K. Mäkelä, E.-P. Salonen, M. Turunen, J. Hakulinen, and R. Raisamo. Conducting a wizard of oz experiment on a ubiquitous computing system doorman. In *Proceedings of IPNMD Workshop*, pages 115–119, 2001.
- [13] W. C. Ogden and P. Bernick. Using natural language interfaces. In M. Helander, editor, *Handbook of Human-Computer Interaction*. Elsevier Science Publishers B. V., 1988.
- [14] M. Rajman, M. Ailomaa, A. Lisowska, M. Melchiar, and S. Armstron. Extending the wizard of oz methodology for language-enabled multimodal systems. In *Proceedings of LREC*, pages 2531–2536, 2006.
- [15] D. Salber and J. Coutaz. A wizard of oz platform for the study of multimodal systems. In *Proceedings of INTERACT and CHI*, pages 95–96, 1993.
- [16] D. A. Schön. *The reflective practitioner: how professionals think in action*. Arena, 1991.
- [17] S. Stüker, C. Zong, J. Reichert, W. Cao, M. Kolss, G. Xie, K. Peterson, P. Ding, V. Arranz, J. Yu, and A. Waibel. Speech-to-speech translation services for the olympic games 2008. In S. Renals, S. Bengio, and J. G. Fiscus, editors, *Proceedings of MLMI*, pages 297–308, 2006.
- [18] M. Suwa and B. Tversky. What architects see in their sketches: implications for design tools. In *Proceedings of the CHI*, pages 191–192, 1996.