MeMo: Towards Automatic Usability Evaluation of Spoken Dialogue Services by User Error Simulations

Sebastian Möller[1], Roman Englert[1], Klaus Engelbrecht[1], Verena Hafner[2], Anthony Jameson[3], Antti Oulasvirta[1], Alexander Raake[1], Norbert Reithinger[3]

[1] Deutsche Telekom Labs, TU Berlin, Germany; [2] DAI-Labor, TU Berlin, Germany; [3] DFKI, Germany
{sebastian.moeller|roman.englert|klaus-peter.engelbrecht|antti.oulasvirta|alexander.raake}
@telekom.de; verena.hafner@dai-labor.de; {jameson|norbert.reithinger}@dfki.de

Abstract

Proper usability evaluations of spoken dialogue systems are costly and cumbersome to carry out. In this paper, we present a new approach for facilitating usability evaluations which is based on user error simulations. The idea is to replace real users with simulations derived from empirical observations of users' erroneous behavior. The simulated errors must cover both system-driven errors (e.g., due to poor speech recognition) as well as conceptual errors and slips of the user, because neither alone is predictive of perceived usability. The simulation is integrated into a workbench which produces reports of typical and rare errors, and which allows usability ratings to be predicted. If successful, this workbench will help designers in making choices between system versions and lower testing costs at early phases of development. Challenges to the approach are discussed and solutions proposed.

Index Terms: spoken-dialogue system, evaluation, usability

1. Introduction

Spoken dialogue systems (SDSs) have reached a level of maturity sufficient for a number of realistic task-oriented applications. Still, interactions between user and system are often not sufficiently smooth and error-free. The sources of interaction problems are difficult to identify, because of the complex interrelationship between the modules of a spoken dialogue system.

The non-triviality of SDS design has raised the demand for efficient, valid and reliable assessment and evaluation methods. So far, systems are mainly evaluated by carrying out laboratory experiments with real users where *system performance* and *interaction behavior* are quantified by means of interaction parameters (e.g. task completion time, number of turns, word error rate, see e.g. [1]), and *usability* is measured in terms of user judgments on different quality aspects [2][3], or by usability heuristics [4]. Such studies are costly and require special expertise and resources [5].

In this paper, we present the underlying principles and the architecture of a new tool for *automatic* usability evaluation. The idea is to replace real users with *simulations* built based on empirical observations of real users. The following requirements apply for such a tool:

- *Easy to use* (requires less specialized skills and investment of time and effort from the developers);
- *Validity of simulation* (produces behavior sufficiently similar to that of real users); and
- *Systematic relationship of simulated behavior to usability judgments of real users* (gives realistic indications of how real users will perceive the usability of the system).

In what follows, we present related work, the basic principle of our approach, which is based on mental models (and thus carries the acronym MeMo), as well as some empirical evidence for the importance of mental model errors. In Section 2, we give an overview of the architecture of our usability evaluation tool, as well as of its composing modules. Finally, Section 3 discusses the advantages and limitations of the automatic usability evaluation approach.

1.1. Related work

Araki and Doshita [6] installed a mediator between a SDS and a simulated user, consisting basically of another SDS. The mediator introduced random noise into the communication channel; this was used to study the system's robustness against speech recognition errors. Similarly, López-Cózar et al. [7] proposed a rule-based user simulator which generates user prompts from a corpus of utterances previously collected in human-human dialogues. Others have also considered addressing psychological factors like cognitive strategies [8] and communicative and error recovery strategies of users [9].

In the broader field of human-computer interaction, there have been various attempts to employ simulations of users as a way of identifying usability problems or evaluating usability. If (as in the MeMo project) the focus is on non-expert users, a difficult part of this task is the creation of a simulation model that faithfully represents at least some aspects of the behavior of novice users. Some good results were obtained by [10], who used genetic algorithms to create deviations from an expert model that seemed typical of novice users. In the MeMo approach, some combination of generic strategies of novices and specific error types needs to be represented in the simulations; the development of such combinations is still an open issue.

Many approaches to the analysis and evaluation of users' performance on the basis of interaction logs have been developed [11]. With various methods, individual data events (such as errors) or patterns of events are identified in logs, and in some cases overall usability-related metrics are computed. A typical limitation to the possibilities for interpretation concerns lack of knowledge of the user's goals and methods. This limitation is less of a problem in the MeMo approach, since the logs in question concern the use of a system by a simulated user whose goals and methods are known to the analyst.

In the areas of robotics and artificial intelligence, using models based on real-world measurements for evaluation is common practice [12]. saves time and energy in experiments with mobile robots, and also avoids unnecessary experiments with animals [13]. Using this method for usability studies of SDSs is relatively new.

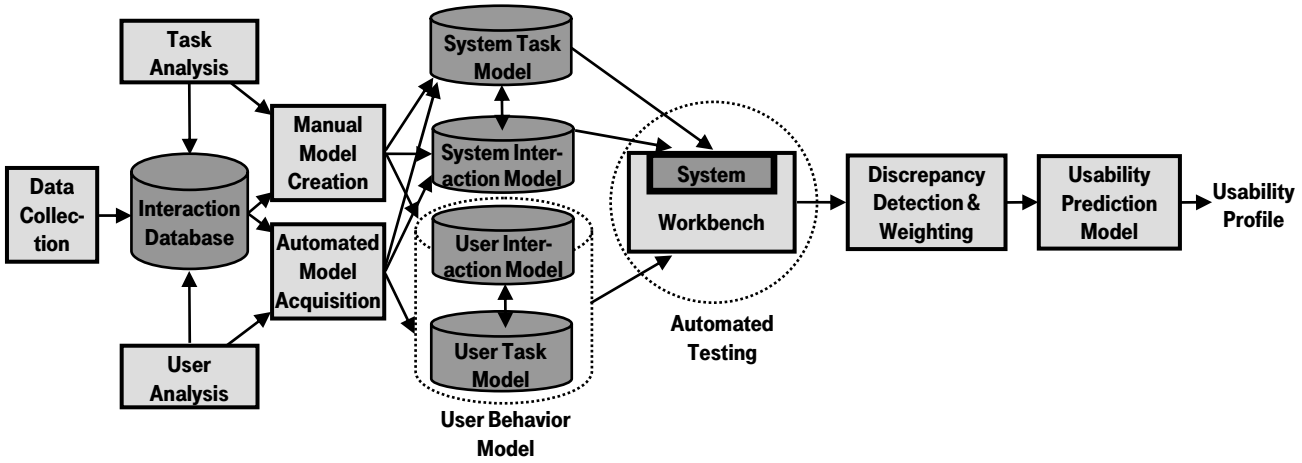

Figure 1 *Overall structure of the MeMo system.*

1.2. Our approach: Mental model errors

Our work builds existing work by considering the possibility to simulate conceptual errors (here called "mental model errors").

It is well known that interactive behavior is guided by so-called "mental models". This internal representation of the system and its running logic are used to describe, explain and predict the behavior of the system [14]. It has to be noted that a mental model is usually acquired on-the-fly, and that it is neither stable nor necessarily correct; the user may have ideas of what to do and how to do it, whereas the system is not necessarily capable of accomplishing that. Discrepancies between the user's mental model and the implementation of the system affect the smooth course of the interaction and may lead to interaction failures. Mental models cannot be simulated trivially with stable conceptual models, but we need to account for updating and learning through interaction.

To understand mental-model-related errors in spoken dialogue systems, we have analyzed transcripts of 24 users interacting with a SDS for controlling domestic devices such as lamps, blinds, TVs, video recorders, answering machines, etc. [15]. This system has been set up in the frame of the EC-funded IST project INSPIRE, see http://www.knowledge-speech.gr/inspire-project/. An error classification scheme has been constructed to cover the user errors observed in the transcripts:

1. No-input error (failure to provide input during the response interval or timeout of the system)
2. Vocabulary & grammar error (providing input which is not accepted due to the wording or the grammar, or due to insufficiencies of the speech recognition and speech understanding components)

Whereas these two types of errors are sufficiently covered by existing language- and task-based approaches, our current work attempts to address non-trivial conceptual errors of the following kinds:

3. State error (providing input which is not accepted by the system in the current state of the dialogue but which is valid in some other state)
4. Capability error (issuing a command or asking for information which is not supported by the system; e.g., asking to dim a lamp although it can be only turned on or off)
5. Modeling error (providing input which would be valid if the system represented the world in a different way; e.g., addressing devices according to their relative position although the system does not model that position)

Our preliminary analyses show that errors of the classes 3-5 are unexpectedly frequent and cover about 20-40% of all errors. The majority of errors are still related to class 2; thus speech recognition and understanding errors should also be incorporated as parts of the simulation.

2. System architecture

Figure 1 illustrates the structure of the MeMo tool. The left hand side depicts the creation of models underlying the simulation. User and system models provide input to the usability workbench – the core of the system. The workbench hosts the application system (i.e. the SDS) in the automatic testing cycle; it simulates interactions and generates loggings. The log-protocols serve the automatic detection of "errors", i.e. discrepancies between user and system models. Using appropriate weighting of the discrepancies, a prediction algorithm derives a usability profile from the protocols. This usability profile describes the impact of errors in terms of performance and quality indicators.

2.1. Generation of system and user models

Four types of models have to be generated. The *system task model* describes the tasks a user may perform with the help of the system. Despite their limitations, most state-of-the-art systems use an attribute-value description (slots) for internal task representation. Such a representation seems to be preferable also for the description of the system task model. The description may be directly derived from the system specification, using e.g. a graphical tool.

The *system interaction model* describes the flow of interaction which is coded in the system. Most commercial systems make use of finite state automata (FSA) or dialogue grammars for this purpose. Like the system task model, this model can directly be derived from a formal description of the system.

The *user task model* is a collection of potential user tasks with the system. The formal description of this model should be the same as the one of the system task model; in this way,

the detection of discrepancies between both models is largely facilitated. Its derivation may be based on:

- a general domain model, i.e. a model of objects and tasks conceivable for a specific domain; this may be derived by user studies or from reviews of existing systems and services in the domain and from listings of their capabilities;
- user studies carried out with a prototype version of the system, or with a similar system; and based on
- the system task model.

The user task model should reflect the frequency with which tasks are likely to be carried out. Such probabilities may be derived e.g. from user studies or marketing information. Both frequent and infrequent – but potentially error-prone – tasks should be modeled.

The *user interaction model* is a running simulation automatically generating user input. In general, user input will strongly depend on the system output in the previous turn. It is thus conceivable to base this model on the system interaction model: The latter determines a set of "correct" user inputs for each step of the dialogue. From these correct user inputs, errors are generated according to an error classification scheme:

- changing the user's vocabulary, e.g. using a synonym list
- changing the user's task (from the user's task model)
- transfer of responses which would be valid in other states of the dialogue, but which are not valid in the current state.

We have analyzed the six error classes listed in Section 1.2. in order to find out possibilities to generate them. State errors can be generated by producing utterances which would be valid in one system state but not in another; such a generation assumes that the simulation can access a system interaction model. Capability errors require the coding of "domain knowledge" and "domain expectations" towards a system, e.g., what humans typically expect of domestic devices. Modeling errors require an analysis and a formal representation of ways users refer to objects in an everyday interaction.

The list of all – correct and incorrect – user inputs is implemented into an executable user interaction model, e.g. in terms of a probabilistic FSA. This model generates paths through an interaction between the simulated user and the system with specified probabilities. The described procedures provide an "initial guess" for user behavior. This model may then be trained on the basis of collected interaction databases.

2.2. Workbench implementation

The workbench is a run-time-environment which takes system input, as it is generated by the actual system, and user input, as it is generated by the user behavior model, and generates logs of simulated interactions as an output. The logs are annotated by information from all models (system task and interaction models, user task and interaction models), providing *a priori* information for the following discrepancy detection and weighting modules. Interaction logs are generated for entire dialogues; they are clustered on a simulated-user-basis in order to generate errors typical for specific users or user groups, and can be used to learn adapted dialogue strategies [16].

2.3. Discrepancy detection and weighting

A basic assumption of our approach is that quality and usability depend on the consistency between the user's mental model and the model foreseen by the system developer. In case that the user behavior – or the one generated by the user behavior model – deviates from the one implemented in the system, discrepancies will occur which may impact the smooth flow of the dialogue. These discrepancies can be seen as "errors" in the dialogue flow, although it is not always possible to attribute the fault of the error to the user or to the system. Errors may be classified according to the criteria in Section 1.2.

During the development of the MeMo system, errors first have to be annotated and classified manually. The interaction database depicted on the left hand side of Figure 1 provides the necessary information for this purpose. Once the user interaction and task models have been established, errors can be generated artificially, as it is described in Section 2.1. Using the error generation, additional information gets available which facilitates an automatic classification of the errors from the interaction logs generated by the workbench.

The detected errors can be weighted according to their impact on the dialogue flow. For this purpose, observed consequences of the errors are annotated during the manual error annotation and classification process. The consequences may be formulated in terms of the consequences to progress in the dialogue: The dialogue may either progress (perhaps at a slower pace than possible), stagnate (e.g. when repetition or rephrasing is necessary), or regress (when the user has to re-iterate parts of the dialogue which have already been performed). The weighting will help to determine the impact of the errors on quality and usability.

2.4. Usability prediction

The usability prediction module provides a link between the instrumental annotation of the dialogue on the one hand, and the user opinion about quality and usability on the other. In this way, the severeness of dialogue problems can be weighted according to the perception of the user.

Input parameters to the algorithm are the error annotations and weightings provided by the previous module, as well as other parameters describing the interaction. Such parameters are described in detail in [5], and they have been recommended by ITU to support the evaluation of telephone-based spoken dialogue systems in [1]. For the sake of the MeMo system, only those parameters can be used which can be extracted in an instrumental way and which do not require an expert annotation. The target variables of the prediction model are subjective ratings given by users of the actual system, collected e.g. with the help of questionnaires [2][3]; these judgments may refer to general concepts like "overall quality" "usability" and "acceptability", or to more specific concepts like "perceived system understanding", "efficiency" or "cognitive demand" [15]. It is still unclear how these user judgments refer to "mental model errors". So far, linear regression models have been used for predicting "user satisfaction" mainly on the basis of interaction parameters, e.g. in the PARADISE framework [17]. We are currently investigating non-linear approaches as well and are expecting results soon.

In order to derive the prediction model, subjective interaction experiments have to be carried out. The interactions are logged and annotated according to the error classification scheme, and in parallel user judgments have to be collected. Based on this data, a model algorithm can be derived.

The output of MeMo is a usability profile. The profile should support developers in detecting weak parts of the system and in estimating their impact on usability. Our profile includes

- an estimation of the overall quality experienced by a "typical user",
- estimations of individual quality dimensions,
- the predicted frequency of errors of different types,
- the predicted consequences of the errors, as well as
- statistics of system interaction parameters observed during the simulation.

3. Discussion and current status

Our approach for automatic usability evaluation is based on "mental models", i.e. concepts users *might* have when they interact with SDSs. As long as no formal theories for mental model creation are available, the derivation of the user models necessary for the simulation has to be based on empirical data. Such data obviously reflect the characteristics of the user, the system, and the task. Consequently, the user model which is derived from this data, as well as the usability prediction algorithm, may be specific to the user, or to a certain group of users. We will assess the issue of user-dependency as soon as first data becomes available. Currently, we foresee that an adaptation of the user model will be necessary when moving from one system to another, but we are confident that the adaptation results in significantly less work than a new creation, or than carrying out formal user tests.

The workbench is currently being implemented at Deutsche Telekom Laboratories, in collaboration with DAI-Labor and DFKI. It will be based on interaction databases collected with two different systems: 1) The INSPIRE smart-home system (see Section 1.2.) and 2) a question-answering system based on semantic web technology, developed in the German BMBF-funded project SmartWeb.

We explicitly selected two systems which differ with respect to the type of task which can be resolved (domestic device operation vs. question-answering), the involved modalities (speech input and speech/visual output for INSPIRE vs. speech/stylus input and graphical/speech output for SmartWeb), as well as the usage environment (home vs. mobile). In this way, we hope that the error classification scheme will gain a certain degree of application-independence.

In order to use the MeMo tool with a new SDS, developers first have to specify the system task and interaction models. On the bases of these models, adapted versions of prototypical user interaction and task models will be derived using current approaches to dialogue learning. Integration of domain knowledge will be necessary to generate vocabulary errors and capability/modeling errors. Given all task and interaction models, the workbench could remain unchanged; error weighting may be adapted on the basis of additional user studies, in order to reflect the importance of errors in different domains. In the project, we will investigate how expensive the construction of these models is when switching to a new application domain.

Once its implementation is finished, it is expected that the tool will be valuable in early stages of service development. Developers will be able to optimize their systems without the involvement of human test subjects. The usability profile will provide quantitative indices for decisions on the market launch of new systems. Nevertheless, the tool is not meant to fully replace user evaluation – the automatic evaluation should still be complemented by user studies, but these studies can be carried out with an already-optimized system version, resulting in lower costs and shorter development cycles.

4. References

[1] ITU-T Suppl. 24 to P-Series Rec., *Parameters Describing the Interaction with Spoken Dialogue Systems*, International Telecommunication Union, Geneva, 2005.

[2] Hone, K. S. and Graham, R., "Towards a Tool for the Subjective Assessment of Speech System Interfaces (SASSI)", *Natural Language Engineering*, 6(3-4):287-303, 2000.

[3] ITU-T Rec. P.851, *Subjective Quality Evaluation of Telephone Services Based on Spoken Dialogue Systems*, International Telecommunication Union, Geneva, 2003.

[4] Nielsen, J., *Usability Engineering*, Morgan Kaufman, San Francisco CA, 1993.

[5] Möller, S., *Quality of Telephone-based Spoken Dialogue Systems*, Springer, New York NY, 2005.

[6] Araki, M. and Doshita, S., "Automatic Evaluation Environment for Spoken Dialogue Systems", in: *Proc. ECAI'96 Workshop, Lecture Notes in Artificial Intelligence No. 1236, 183-194,* Springer, Berlin, 1997.

[7] López-Cózar, R., de la Torre, A., Segura, J.C., and Rubio, A.J., "Assessment of Dialogue Systems by means of a New Simulation Technique", *Speech Communication*, 40: 387-407, 2003.

[8] Walker, M., "Experimentally Evaluating Communicative Strategies: The Effect of the Task", in: Proc. AAAI'94, 86-93, ACM Press, New York, NY, 1994.

[9] Carletta, J.C., "Risk Taking and Recovery in Task-Oriented Dialogue", *PhD Thesis*, University of Edinburgh, UK, 1992.

[10] Kasik, D. J. and George, H. G., "Toward Automatic Generation of Novice Test User Scripts." In *Proc. CHI '96,* ACM Press, New York, 244-251, 1996.

[11] Ivory, M.Y., and Hearst, M.A. "The State of the Art in Automating Usability Evaluation of User Interfaces", *ACM Computing Surveys,* 33(4): 470-516, 2001.

[12] Michel, O., "Webots: Symbiosis Between Virtual and Real Mobile Robots", in: J.-C. Heudin, (ed.), *Virtual Worlds, vol. 1434 of Lecture Notes in Artificial Intelligence*, Springer-Verlag, Berlin, 254-263, 1998.

[13] Hafner, V.V., Fend, M., König, P. and Körding, K.P., "Predicting Properties of the Rat Somatosensory System by Sparse Coding", *Neural Information Processing Letters and Reviews,* 4 (1): 11-18, 2004.

[14] Gentner, D. and Stevens, A. L. (Ed.), *Mental Models*, Lawrence Erlbaum Associates, Hillsdale NJ, 1983.

[15] Möller, S., Smeele, P., Boland, H., Krebber, J., "Evaluating Spoken Dialogue Systems According to De-Facto Standards: A Case Study", accepted for *Computer Speech and Language*, 2006.

[16] Alexandersson, J. and Reithinger, N. "Learning Dialogue Structures from a Corpus", in: *Proc EuroSpeech '97*, Rhodes, 2231-2235, 1997.

[17] Walker, M. A., Litman, D. J., Kamm, C. A. and Abella, A., "PARADISE: A Framework for Evaluating Spoken Dialogue Agents", in: *Proc. ACL/EACL 35th Meeting*, Madrid, 271-280, 1997.