

## ACT-R Meets Usability

### Or why Cognitive Modeling Is A Useful Tool To Evaluate The Usability Of Smartphone Applications

Nele Russwinkel  
nele.russwinkel@tu-berlin.de

Sabine Prezenski  
sabine.prezenski@zmms.tu-berlin.de

Dep. of cognitive Modeling in dynamic HMS  
TU Berlin  
Berlin, Germany

**Abstract**—The usability of two different versions (A and B) of a smartphone shopping list application is evaluated via user tests and cognitive modeling. The shopping list application allows users to select different products out of different stores. Version A has less menu-depth than version B. The results show that less product-search time is required for version B. This benefit of version B over A declines, as users become more experienced. Advantages of modeling approaches and disadvantages of empirical data are discussed. It is shown that cognitive modeling approaches with ACT-R are a powerful tool for model based usability testing.

**Keywords**-usability; cognitive modeling; ACT-R; mobile applications;

#### I. INTRODUCTION

Nowadays, smartphones and mobile applications are part of our everyday life. Application numbers are growing rapidly [1]. Successful applications obviously have a high usability. Evaluating usability with conventional usability testing requires time and money. Therefore, a pressing question is how the usability of applications can be guaranteed without costs exploding. In this paper, we will present our idea and first results of how cognitive modeling with ACT-R can serve as substitute for extensive usability testing. We will show how learning in Apps will proceed and also deal with the interesting question of version update (or switching) effects.

Cognitive architectures, such as ACT-R (Adaptive Control of Thought-Rational) [2] offer a computable platform that represents well established theories about human information processing. With cognitive architectures it is possible to simulate cognitive mechanisms and structures, such as visual perception or memory retrieval. These are organized in different modules and these modules communicate via their interfaces to the production system which are called buffers. ACT-R is a hybrid architecture, which means that it has symbolic (knowledge representations, such as chunks and rules called productions) and subsymbolic components (activation of chunks and utility of productions).

What exactly is usability? Standard ISO 9241-11 specifies usability as effectiveness, efficiency and satisfaction. Standard ISO-9241-110 describes general ergonomic principles for the design of dialogues between humans and information system, outlining seven import criteria (suitability for the task, suitability for learning, suitability for individualization, conformity with user expectations, self descriptiveness, controllability, and error tolerance).

Nielson's Usability heuristics describe ten general principles for interaction design, for example that consistency and standards should be applied [3]. There are also more specialized heuristics for mobile applications [4]. Developers should apply these heuristics when designing applications. Another, more technical way, to deal with usability is via pattern matching methods [5].

Other popular methods to assess usability of mobile applications are expert reviews or user data, which can be collected via questionnaires, qualitative methods (e.g., think aloud protocol) or usability tests. Particularly information about subjective satisfaction can only be obtained with qualitative measurements. Nevertheless quantitative user testing allows assessment of a wide range of usability criteria; e.g., task completion time as a measure for efficiency, the number of successful task completions as a measure for effectiveness, the number of and kind of mistakes give information about suitability for the task, conformity with user expectations, self descriptiveness, controllability and error tolerance. Suitability for learning can be measured via comparison of several runs [6].

In a review on different studies on usability of mobile applications R. Harrison et al. [7] stress, the importance of cognitive load of applications for successful usage. They also emphasize the difficulty of assessing cognitive load via heuristics or standards. Cognitive models can be powerful tools, when dealing with questions concerning cognitive load.

Cognitive models can serve as a substitute for (quantitative) user tests. User models build with ACT-R can simulate the interaction with a certain task. Cognitive modeling has two advantages over real user tests; first of all no human participants are needed when good and evaluated models exist and second, important information about underlying cognitive processes can be discovered.

Implications from these findings can then be used in designing further applications.

So far some first approaches were developed that combine modeling with usability. CogTool is a user interface prototyping tool, which is based on keystroke-level modeling and produces a simplified version of ACT-R code [8]. After the user manually compiled a storyboard, CogTool produces a cognitive model, which runs along the pathway as identified in the storyboard. CogTool then predicts how long a skilled user requires to complete the task [4]. CogTool has several limitations: It is not possible to let the model explore the interface since the model only runs along the ideal-pathway as defined by the storyboard. Therefore, information about potential user-errors or influence of workload cannot be achieved. MeMo- is a Usability Workbench for Rapid Product Development which can simulate user interactions with the system [9]. On the basis of task, possible solution pathways are searched by the model and deviations from these pathways are then generated; Different user groups (e.g., elderly users, novice users) are taken under consideration [9], which is clearly an advantage of MeMo over CogTool. Another advantage with MeMo persists in the potential of the model to produce errors. A clear disadvantage of MeMo arises due to the fact that it is not a cognitive modeling tool- important concepts about human cognition are not implemented.

In the following, we will first introduce a new tool that connects applications with a cognitive architecture to directly enable cognitive models to interact with an interface. Then we will describe our general approach. Afterwards, we will introduce the application we used and the usability study we conducted including the results. In the discussion section, the empirical results as well as the different modeling approaches are discussed for three main topics: Comparing app versions, whether learning occurs and switching effects between versions. Lastly, we draw conclusions and explain the further process of the approach.

## II. APPROACH

Our approach towards modeling the usability of interfaces differs from those described above: We developed a tool called „Hello Android” [10]. This tool enables a direct connection of the cognitive architecture with a Smartphone application via a TCP/IP protocol. In this vein, the user model can directly interact with the application, press buttons and in turn the model can perceive changes on the interface as well. The tool has many advantages for the modeler; first of all no mock-up version of the app or possible pathways need to be created, which saves a lot of time, compared to CogTool or MeMo. Secondly, we will model the application using the full possibility and functions of the ACT-R architectures, which allows investigating a great number of different aspects of how the app affects human information processing and individual differences (e.g., memory, experience or age). Thirdly, with our modeling approach we can evaluate processing time of an application as well as different kind of user mistakes. Main requirement for the usage of our approach are skills in modeling with ACT-R. The modeler just needs to know how

to write (or change) productions and have rudimentary knowledge of the subsymbolic part of ACT-R. No lisp-programming is needed.

## III. STUDY

Our study compares two slightly different versions of a shopping list application for Android. Both versions allow users to choose products out of either an alphabetically ordered list or via categorical search. The chosen products are then added to a list. Menu depth differs between the two versions: version A has one menu level more than version B. The first page of the application is the same for both versions: three buttons are presented: “overview”, “shops” and “my list”. For both versions, when selecting “overview” one gets a list of the alphabet. Three or two letters are always grouped together on one button, e.g., “ABC”, “DEF”.... Selecting one of those buttons then results in an alphabetical ordered list of the products. For example, when clicking on ABC, all possible products, with product names beginning from A to C appear in a list. A click on a small checkbox in the right of the product selects it. If you click on shops, then for both versions a list of seven shops (bakery, drugstore, deli, greengrocer, beverage store, stationery, and corner shop) appears. Each of these shops is represented by a button. For version B, selecting one of the shops results in an alphabetical ordered list of the products available in that particular shop. For example, by clicking on greengrocers all items that can be found in a greengrocers store are presented (apples, bananas, blueberries, cherries, etc.). A click on a small checkbox in the right of the product selects it as well. For version A, the shops each have seven subcategories. For example, when selecting greengrocers, one is presented with the subcategories exotic fruits, domestic fruits, tuber vegetables, herbs, seeds and nuts, mushrooms and salads. When selecting a subcategory, a list of products that can be found under this subcategory, appears. Again, a click on a small checkbox in the right of the product selects it. For both versions, selecting “My List” from page one results in a shopping list which comprises the selected products plus information about the store in which the products are available.

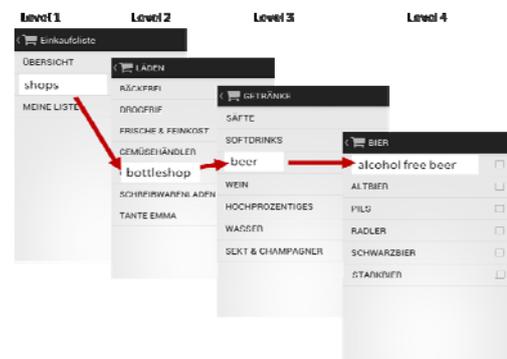


Figure 1: Version A of the shoppinglist application- Version B is similar, except that Level 3 “Getränke” is missing.

Sixteen voluntary student participants (seven male and nine female,  $\text{mean}_{\text{age}}=22$ ) took part in the study. After receiving standardized oral instructions participants were asked to select a list of products. For each trial, a product was read to participants by the investigator and participants had to find the product. He or she could select the product either via the “Overview” path or search the stores. After selecting a product, participants were asked to return to the first page and then the next trial started. After selecting eight products, participants were asked to read the shopping list (in order to assure learning of the store categories) and then the next block started, this time the items were the same but presented in a different sequence. After completing the second block, the investigator presented the participant the other version and the two blocks of trials were repeated. Half of the participants first worked with version B and half started with version A.

#### A. Results

In the following section, preliminary results are presented (see figure 1). This paper focuses on the mean trial time for the different blocks, which is defined as the time difference from when the participant leaves the start page until the item is selected. For participants of group “A first- B second“, the mean trial time of block 1 of version A is 10.67 sec and decreases approximately 4 seconds for block 2 (mean trial time 6.11 sec). After switching from version B “B second“, time decrease to 4.96 sec and reaches 4.43 sec for B second-expert (block 4). For participants of group “B first- A second” a trial in the first block has a mean duration of 8.74 sec and a trial in the second block a mean duration of 4.02 seconds. After participants switch to version A “A second“, time increase to 6.56 sec and decreases again to 4.42 seconds.

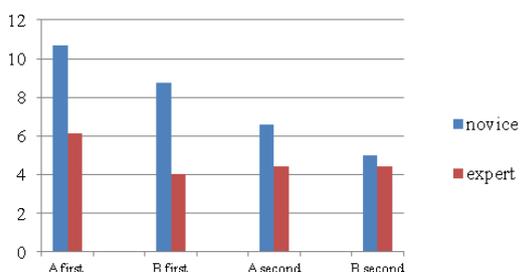


Figure 2: mean trial time for the different conditions

#### IV. DISCUSSION

The next passage explains how modeling captures usability aspects and depicts explanations for the effects found in the data.

##### B. Comparison of both versions:

*Empirical:* Version B is overall faster than version A, especially for novice users. The benefit of version B over A decreases and almost reaches nil, as block 4 for version A and version B (expert) show. Explanation: Obviously, more required clicks in version A are probably not the reason for the benefit of version B over A for novice users.

*Modeling:* The building of expectations-chunks takes longer for version A than for version B, because there are more interaction steps in A and therefore, more encoding of these steps is required. Furthermore, for version A more semantic knowledge (which shop holds which subcategory and which subcategory holds which product) is needed, but the knowledge of subcategory is unnecessary for version B. Version A also takes longer because the extra semantic knowledge needs more encoding in chunks for subsequent use. For both versions, knowledge is learned via trial and error. Useful pre knowledge is probably available (and provided in declarative memory) but especially for some categories pre knowledge is less obvious and those product-category pairs have to be learned. The first step in the model is to try to retrieve the information about categories, e.g., in which category a certain product might be found. If the retrieval is not successful the next step for the model is to search a different category. If this is successful the connection between product and category is encoded in declarative memory. An unsuccessful retrieval takes longer than a successful one and also requires more productions to proceed. The plus of productions takes a lot of time (each production takes about 50ms). Once the subject has acceptable knowledge about category membership, there are more successful than unsuccessful retrievals. As a consequence the mean trial duration decreases.

##### C. Does Learning occur?

*Empirical:* Our data shows a clear learning effect, as participants become more familiar with the application the mean trial duration decreases- there also seems to be a learning transfer from version A to version B.

*Modeling:* Production compilation is a useful ACT-R mechanism to model learning. In the beginning, for every interaction step a memory retrieval of the next processing step is required. After a few trials often used information is integrated in the productions. Trial duration decreases, since retrieval time is redundant and proceeding productions are integrated. Furthermore, retrieved expectancies can give detailed information were the next relevant button will be located. Therefore, eye- and finger-movements can be prepared early and initiated more quickly with practice. Because no additional information needs to be learned when switching from version A to B (note that version A includes all menu-structures of version B, but has more menu depth), the above mentioned learning processes are not disturbed and learning continues.

##### D. Are there switching effects?

*Empirical:* A switching effect occurs when participants familiar with version B change to version A. This becomes apparent in the increase of time from B first expert to A second. Nevertheless, participants using version A second still profit from version B, since A second is faster than A first.

*Modeling:* Switching from version B to A irritates the users, because they end up with a menu they did not expect and are not familiar with. In terms of the user model, this means they do not have instruction chunks that give the

information what is to do next. They have to go back and search for the back button and then learn the items that belong to the new categories. This takes time because it causes a number of additional productions to fire. But after a few trials, new category-product pairs are learned and the switching effect disappears. In the opposite case, users end up earlier with the final (more familiar) list that is already encoded in the expectancy chunks. They do not have to learn new category members and do not need to encode representations to declarative memory; therefore, fewer productions have to fire and mean trial duration is low.

## V. CONCLUSION

### *Conclusion over the usability of the two versions*

Both versions are suitable for users, but version B is slightly faster than version A. The benefit of version B decreases as user experience increases. Shallow menu structures are more convenient for novice users. Both versions of the application are easy to learn. Switching from version A to B has additional time cost in the beginning, whereas switching from B to A has not. We showed that user models can provide informed interpretations about the causes of usability, e.g., differences between versions can be explained through specific learning processes; a finding that is not possible with conventional usability tests.

### *Outlook*

In the near future, we will further investigate our data, with a stronger focus on potential user errors, data fitting and mobile context. Cognitive modeling of the usage of Smartphone applications with ACT-R is a promising approach for usability research. The goal of our research is to develop guidelines for ACT-R modelers describing the most relevant modeling concepts for usability of applications. These guidelines will raise the opportunity to quickly develop user models and improve and evaluate the usability of applications. As the number of new applications on the market further increases, cognitive modeling provides the solution for affordable and capacious usability testing.

## ACKNOWLEDGMENT

We would like to thank the students of Human Factors Engineering for their engagement in data collection and all members of our team for support. Special thanks to Lisa Doerr. Thanks also to Gabi Taenzer and Steffen Vaupel for the programming of the application.

## REFERENCES

- [1] J. Koetsier, *Google Play will hit a million apps in June* [Online]. Available from: <http://venturebeat.com/2013/01/04/google-play-will-hit-a-million-apps-in-2013-probably-sooner-than-the-ios-app-store/> 2013.01.04.
- [2] J. R. Anderson, *How Can the Human Mind Occur in the Physical Universe?*, New York: Oxford University Press, p. 304, 2007.
- [3] J. Nielsen, *Usability engineering*. Morgan Kaufmann Pub, 1994.
- [4] L. Kuparinen, J. Silvennoinen, and H. Isomäki, "Introducing Usability Heuristics for Mobile Map Applications," Proceedings of the 26th International Cartographic Conference, Dresden, Germany, 2013.
- [5] J. Engel, C. Herdin, and C. Maertin, "Exploiting HCI Pattern Collections for User Interface Generation," Proceedings of the 4<sup>th</sup> International Conference on Pervasive Patterns and Applications, (IARIA Proceedings) Nice, France, pp. 36-44, 2012.
- [6] D. Zhang and B. Adipat, "Challenges, Methodologies, and Issues in the Usability Testing of Mobile Applications," *International Journal of Human-Computer Interaction*, vol 18, pp. 293-308, September 2005, doi:10.1207/s15327590ijhc1803\_3.
- [7] R. Harrison, D. Flood, and D. Duce, "Usability of mobile applications: Literature review and rationale for a new usability model," *Journal of Interaction Science*, vol. 1, pp. 1-16, January 2013.
- [8] L. Teo, B. E. John, and P. Pirolli, "Towards a Tool for Predicting User Exploration," CHI '07 Extended Abstracts on Human Factors in Computing Systems. pp. 2687-2692, New York, NY, USA: ACM, 2007, doi:10.1145/1240866.1241063.
- [9] S. Möller, K.-P. Engelbrecht, and R. Schleicher, "Predicting the Quality and Usability of Spoken Dialogue Services," *Speech Commun.*, vol. 50, pp. 730-744, August/September 2008, doi:10.1016/j.specom.2008.03.001.
- [10] S. Linder, P. Büttner, S. Vaupel, G. Taenzer, and N. Rußwinkel, "Towards an Efficient Evaluation of the Usability of Android Apps by Cognitive Models," *Kognitive Systeme*, Universität Duisburg-Essen, in press.