Combining Cognitive ACT-R Models with Usability Testing Reveals Users Mental Model while Shopping with a Smartphone Application

Sabine Prezenski
sabine.prezenski@tu-berlin.de

Nele Russwinkel
nele.russwinkel@tu-berlin.de

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

***Abstract*—The usability of two different versions of a smartphone shopping list application for Android is evaluated via user tests and cognitive modeling. The mobile application enables users to compose a shopping list by selecting items out of different stores and product categories. The two versions of the linear hierarchical application differ in menu depth. Two empirical studies compare novice and expert product search time. The first study focuses on efficiency, suitability for learning, mental load and mental models. The second study supplements the findings of the first study and investigates varying expectations between products. An ACT-R based cognitive modeling approach provides in depth explanations for the effects found in the empirical study. The study shows that for expert users, product search with a 3 layer or a 2 layer version is equally efficient, due to the same amount of mental load. Expert and novice user rely on different strategies when searching for items- novice users need to access their general knowledge frequently, experts use their mental model of pathways leading to the items. The suitability of the mental model of users, explains why version updates that introduce a new layer produce longer product search times - and those reducing the number of layers do not.**

Keywords-cognitive modeling; ACT-R; usability; smartphone; application; menu; mental load; mental model.

I. INTRODUCTION

Nowadays, life without mobile applications and smartphones is hard to imagine. New evaluation methods for mobile applications are needed [1] because the market for is growing rapidly [2]. For an application to be successful, high usability is compulsory. Conventional usability testing is time and money consuming. Therefore, a pressing question is how the usability of applications can be guaranteed, without costs exploding. On the long run, cognitive models can serve as a substitute for usability testing. The following paper is a step towards this goal.

The current paper investigates menu depth in a real-world setting with a new smartphone application. In contrast to this, most studies concerning menu depth use artificial labels and tasks in a laboratory setting.

Our work demonstrates that the most important factor for menu design is not reducing the number of clicks, but users' mental processes. An important finding is that the best number of levels of menu hierarchy may differ from case to case, but is one that maintains users' mental load to a minimum. Well designed applications address users' mental models of these applications. The fact that mental models of users are not static, but evolve as they develop from novices to experts is a further topic of this paper. The learning processes while handling a new application is studied.

In the current work, cognitive modeling is used to explore users' mental processes. We will demonstrate how ACT-R based cognitive modeling explains results obtained in empirical usability studies. It is shown, that cognitive modeling with ACT-R has the potential to replace traditional user tests to a certain extent, but also help to understand the underlying mental mechanisms in this kind of human-machine interaction.

The empirical part consists of two studies on two different versions of a shopping list application. The Android application allows users to select products out of a categorized hierarchical list or via an alphabetical product overview. With the application, users can compose a shopping list. The two versions differ in menu depth. Although both studies concern the same application, design and purpose of the studies differ.

The first study allows a conclusion on the overall usability of the application, due to the fact that all navigating possibilities, the app provides, are allowed. The sample size of the study permits statistical testing to compare the versions, too. In the first study, an ACT-R modeling approach is introduced. The second study supplements the first study. It restricts functionality of the application in order to substantiate the model assumptions about mental models of users from the first study. Furthermore, the second study enables to conduct learning curves and to investigate different expectations on product affiliated categories.

In both studies, users repeatedly search for the same products. Therefore, novice and expert users can be studied and the suitability for learning of the application can be evaluated.

The modeling part further addresses how mental models of novice and expert users develop as users become more experienced with an application. It also investigates how version updates of software challenge users' mental model. This study also unfolds the relationship between menu hierarchy and cognitive load.

II. THEORY

A. Usability

Standard ISO 9241-11 specifies usability as effectiveness, efficiency and satisfaction. General ergonomic principles for the design of dialogues between humans and information systems are specified in standard ISO-9241-110, which outlines seven 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 are another way of specifying usability; they describe ten general principles for interaction design, for example that consistency and standards should be applied for successful applications [3].

1) Measurement of Usability: There are various methods to judge the usability of mobile applications; user focused assessment makes an important distinction between expert reviews and user data. A more engineering centered approach is, e.g., the method of pattern matching [4], which allows designers to assess certain usability problems, without interacting with users. There are different approaches to evaluate user data; either via qualitative methods (e.g., think aloud protocol), questionnaires or user tests. Particularly, information about subjective satisfaction can only be obtained with qualitative measurements, e.g., questionnaires or interviews. Nevertheless, high correlations between subjective satisfaction and quantitative measurements of usability are expectable [5] [6]. Therefore, assuring effectiveness and efficiency is important for achieving subjective satisfaction. 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 in a given time as a measure for effectiveness. The number and kind of mistakes give information about suitability of the application for the task, about conformity with user expectations, about self descriptiveness, controllability and error tolerance. Suitability for learning can be measured via comparison of several runs. Furthermore, contrasting inexperienced users (so called novice user) and very experienced users (experts) provides an interesting insight into the question if experience with an application provides a benefit for users. A reduction in time on task is expected for expert users, since they have developed a mental plan for the handling of an application. One should be aware of the fact, that not only performance in terms of time on task improves with experience, but that the structure of individuals knowledge (their mental model) changes as well [7].

The problems with user tests and questionnaires are similar to those of general psychological test; various testing aspects (such as reactance, conformity and other motivational issues) influence the outcome. Besides psychological testing effects, user tests are expensive and time consuming. Individuals need to be recruited and tested. Therefore, alternative methods that do not require user testing would be of value. Furthermore, methods that state precise concepts, which can then be transferred to other applications, are eligible. Cognitive Modeling fulfills the requirements mentioned above and is further a method to assess usability. Cognitive modeling is a helpful approach to learn more about cognitive processes during the interaction with applications. In addition, cognitive modeling offers the opportunity to explore the structure of users' knowledge. In the future, predictive cognitive models can serve as a substitute for user tests.

User tests can assess the most important aspects of usability, such as effectiveness and suitability for learning. Quantitative measurements can be replaced by cognitive models. In addition, cognitive models offer explanations about mental processes influencing usability, a benefit that goes beyond the scope of simple user tests.

2) Usability and Smartphones: In the field of mobile applications, special challenges for usability testing exist. Especially, aspects of mobile context, limited connectivity, small and varying display size and aspects concerning data entry methods should be accounted for [8]. In a review on different studies on usability of mobile applications, Harrison et al. [9] stress the importance of mental load of applications for successful usage.

Mental load is defined as the mental cost required fulfilling a task [10]. Mental (or cognitive) load is a multidimensional concept, with subjective, objective and psycho-physiological components and therefore difficult to measure [11]. The PACMAD (People At the Centre of Mobile Application Development) usability model for mobile devices includes mental load into the ISO definition and further incorporates the user, the task and the (more mobile) context [10].

It is highly questionable if mental load, a multidimensional and crucial concept for mobile usability, is assessable with user test. User tests can assess the most important aspects of usability, such as effectiveness and suitability for learning. Quantitative measurements can be replaced by cognitive models. In addition, cognitive models offer explanations about mental processes influencing usability, a benefit that goes beyond the scope of simple user tests.

B. Modeling and Usability

It is stated that mental load is impossible to asses via heuristics or standards [9]. Hence, a different approach is needed. On the other hand, assessing cognitive load with cognitive models is possible and already carried out [11] [12].

CogTool [13] and MeMo [14] are tools that allow user modeling of smartphone applications and websites and provide insights about usability problems.

CogTool is a user interface prototyping tool, which produces a simplified version of ACT-R [15] code; it is based on keystroke-level modeling [13]. KLM divides tasks as into different kind of actions (e.g., keystrokes, pointing) and mental processes, which are represented through mental operators [16]. A specific amount of time is assumed for

each action and each mental operator. Total task time is composed of the sum of these. In order to produce a cognitive model with CogTool, one has to manually click together a storyboard. The model then runs along the pathway as described in the storyboard. CogTool predicts how long a skilled user will take for the specified task [13]. CogTool has limitations, for example, it is not possible for the model to explore the interface since the model only runs along the ideal-pathway (as defined by the storyboard). As a result of this, information about potential user errors or the influence of workload cannot be achieved. Furthermore, information about learning or the difference between expert and novice users cannot be uncovered using CogTool.

MeMo is a Usability Workbench for Rapid Product Development, which can simulate user interactions with the system [14]. On the basis of tasks, 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 [14], which is clearly an advantage of MeMo over CogTool. Another advantage of MeMo is that the model can produce errors, just as real users would do. A distinct disadvantage of MeMo is that it is not a cognitive modeling tool- important concepts about human cognition such as learning are not implemented. Therefore, the validity of the conclusions attained with MeMo is questionable.

Besides the introduced ready-to-use tools, there are numerous modeling approaches that uncover how different cognitive aspect or design factors affect usability. These modeling approaches attempt to describe and predict coherence between usability influencing aspects. Results obtained from the modeling approaches introduced below, can be used to derive general advice for designers.

1) Mathematical Models: Mathematical models are developed to predict measurements of usability, such as selection time as a function of external factors [16]. For example, the relation between item position and target search time in a linear menu can be described as a predictive mathematical model [17]. Other mathematical models focus on how factors such as menu size, target position and practice influence usability factors [16]. Such numerical models provide straightforward advice to designers. But on the downside, they are not helpful in identifying why these relations exist and give no information about learning and workload influence. One does not gain insight on how or why ongoing cognitive processes influence usability.

2) Cognitive Models: Therefore, to reveal the causes of differences in usability performance measurements, cognitive models should be consulted. Just as mathematical models, they provide numerical predictions. But cognitive models simulate the interaction with an application in the way users would interact with the application. Specific cognitive processes such as attention, perception and memory are incorporated in these kinds of models and can serve as an explanation for differences in usability findings. The best way to build scientifically grounded cognitive models is to use cognitive architectures.

a) Cognitive Architectures and ACT-R: Cognitive architectures 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. EPIC [18] and ACT-R [15] are two architectures used for modeling aspects of human computer interaction. This paper focuses on an ACT-R model of user interaction. To understand the modelling approach descirbed later it is helpful to know about the core mechanisms of ACT-R [15]. ACT-R is a hybrid architecture, which means that it has symbolic (knowledge representations such as chunks and rules called productions) and sub symbolic components (activation of chunks and utility of productions). The symbolic part consists of different modules and their interfaces (called buffers), with which these modules communicate with the production system. Only one element can be stored in each buffer at a given time. Similar to the brain, ACT-R distinguishes different areas called modules, which process certain classes of information. For instance, the declarative memory module, can store information in units called chunks. These chunks can be retrieved, which means that a chunk, which matches the given criteria is put into the according buffer and can be processed further by the production system. New chunks are constructed in the imaginal module. Other modules process visual or auditory information. There are also output modules such as vocal and motor modules. These are just some of the available modules. Furthermore, the sub symbolic components of the architecture are important. If some chunks are retrieved and used more often than other chunks, these chunks are given a higher activation level. This activation level determines how quickly a chunk can be retrieved or if it can be retrieved at all. Information that is not often used will decay over time and at some level will be forgotten and hence cannot be retrieved. The structure of chunks is characterized by different slots (or attributes) that can be filled with information. Category membership is represented in slots; this allows building semantic networks. Furthermore, new chunks can be learned during a task. The production system persists of rules defined by an “if” and “then” part. If the cognitive system with its modules and chunks in the buffers meet the conditions of the rule, the rule can be selected. In this case, the action part is executed. The production systems enables to initiatechanges to the chunks or to send requests to the modules (e.g., to the motor module “press mouse button”). If particular rules are more useful than others they receive a higher utility level and will be preferred to others. Also, reward can be given to productions if they lead to a goal, which also influences the utility level. It is possible to enable a process called production compilation. Here productions can be combined if they precede each other often or if identical chunks are

frequently retrieved in similar situations. This way the model will become faster just as human behavior improves as a task is done multiple times.

b) ACT-Droid: In the modeling approach presented, a new tool, ACT-Droid [19] is implemented, which has outstanding benefits to other approaches. Instead of replicating mobile applications or websites as mock-ups or reprogramming aspects of the application for the model, ACT-Droid allows developing user models that directly interact with Android smartphone applications. The ACT-Droid tool enables a direct connection of the cognitive architecture with an Android smartphone application via TCP/IP protocol. With this tool the modeling process becomes more convenient and much faster. In general one has to define a simlple interface version of the application in Lisp, with which the model can then interact with. When using ACT-Droid the ACT-R model can directly interact with the original Android application. The user model can interact with buttons and perceive changes on the interface.

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, models interacting with the application, can implement the full possibility and functions of the ACT-R architectures, which allows investigating a great number of different aspects of how applications affects human information processing and individual differences (e.g,. memory, learning, experience or age). Thirdly, with this modeling approach processing time as well as different kind of user mistakes can be evaluated.

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 sub symbolic part of ACT-R. No lisp-programming is needed. Thus, ACT-Droid makes modeling with ACT-R much less complicated and more straightforward.

c) Modeling of Smartphone Applications: Applications for smartphones are small programs. Most applications are very specific for their field and are hence built to solve limited tasks. The limited scope of applications and the fact that successful applications have a simple and consistent design, make them profound for cognitive modeling. Developing a user model able to interact with the application is an accomplishable modelling task and can help uncover difficulties in the application that negatively influence usability. Some factors influencing the usability of smartphone applications, especially mental load or aspects concerning mental models are especially eligible to be evaluated with cognitive models.

Building up a mental model of an app the user normally orients oneself towards the menue structure.

C. Menus

An important research question concerning usability is how a menu structure should be designed in order to offer the best opportunity for navigating an application [20]–[23]. Menus help users find the right information. Different types of menus exist, e.g., square menus, pie menus, linear menus, hierarchical menus [23]. This paper focuses on linear hierarchical menus. Research on menu structures and design has revealed many important factors contributing to successful usability of menus. The following findings are derived from strongly controlled laboratory studies or studies dealing with either desktop menus or website menus. Consequently, it is questionable if these findings can be transferred to real-life smartphone applications.

Zhang states that clear and consistent labeling, predictability, a minimum of interaction steps, but also the avoidance of long list in a menu structure are import factors, contributing to the usability of menus [8].

Nilsons [17] identifies important factors that influence item selection time, such as menu length, item placemen and menu organization. Shorter menus are beneficial; e.g., users are faster in selecting and searching items from shorter than from longer menus [24]. When it comes to menu organization, organization of items in a menu is more beneficial than random placement of items [25]. Semantic and alphabetic organizations are two typical ways of organizing items. The target position of an item on a menu has a strong influence on item search time. Targets positioned on the top of a menu are found faster than those positioned in the middle [26]. Targets positioned on the bottom of the menu are likewise easier to be found [16]. The more users are familiar with a menu, the less time they take for finding an item, this is known as practice effect [20] [27]. If the target item is included in the menu, scanning is quicker, than if the target is not included in the menu [16]. Target items are found faster, if the item label is strongly associated to the target item [21].

a) Menu Hierarchies: Menu hierarchies are another factor that influences the usability. Lee and MacGregor [28] point out that two main factors influence search time for an item in a hierarchical menu; the number of pages (or levels) that have to be accessed and the time required to select alternatives from pages. The required time is directly dependent on the number of alternatives per page. For smartphones, finding an adequate depth and breath for the menu hierarchy is especially important. The small display size and scrolling time, make it even more important to provide users with and easy accessible and transparent menu. Some design experts recommend, that menus of mobile phones should rather be narrow and deep than shallow and broad [29]. Others state that adding more hierarchy levels (especially for menus with more items) is advisable, but that hierarchies with more than three levels should be avoided [30]. In General, findings in the literature are conflicting [31]. Another aspect influencing hierarchies

is scrolling. Cockburn and Gutwin propose a mathematical model linking the relationship between menu- scrolling and hierarchy on desktop computers [31]. When investigating an adequate ration between menu depth and items, instead of discussing the numbers of items or menu level, one can also focus on the question if users mental model matches the model of such a menu [23] [30]. Since measuring mental models is doubtful with classical methods and ACT-Droid provides the possibility to model application, this essential topic of menu hierarchies should be studied with cognitive modeling.

b) Menu Structures: Currently, most modeling studies on menu design focus mainly on visual processing [16] [24] [26] [32] [33] for evaluating menu design and usability aspects. According to an EPIC model from 1997 [32], eye movement pattern are a 50/50 mixture of sequential top-to-bottom and randomly searching. When serial top-to-bottom search is executed, the users' eye moves down the menu with constant distance in each saccade. Because of parallel examination of multiple items, the eyes regularly pass the target item by one saccade. Motor movements do not occur before the target is located. An ACT-R model from the same year [33] on the other hand predicts, that eye movements are exclusively top-to-bottom, and that the distance of each saccade varies. The eyes never pass the target items and that motor movement follows the saccades and happens before the target is located. Succeeding studies [24] [26] that used an eye tracking experiment and an ACT-R / PM model found that the first eye fixation is almost always towards the top of a menu and that most of the time visual search is top-to bottom. Search is rarely random. Some items are skipped and then backtracked. These models, with a focus of visual encoding, provide explanations for effects, such as the preferred item position. Mathematical models have a strong focus on different visual search strategies as well. Serial search and direcfted search are two modeled visual search strategies [16]. Serial search labels, top-to-bottom search and directed search describe search as determined by focusing to the assumed location of the target. Mathematical models also suggest that visual search of novice and expert users follows different functions [20]. These models describe the fact that as learning proceeds performance increases, which is explained through remembering the position of visual items [20]. As a conclusion, most modeling studies dealing with aspects of menu design, focus on visual and motor processing and few studies compare expert and novice performance [20]. Even though empirical studies indicate that menu-structures should incorporate the mental model of potential users and claim that mental models changes as users become more familiar with the menu structure [25] [21], as far as the authors are aware, no cognitive modeling studies focusing on this exists. This paper will present how cognitive model can address user's mental models of menu structure of applications.

III. METHODS

A. Purpose of this study

The study concept is designed to investigate menu hierarchies of an application. Two versions of an application, differing in menu depth, are compared using concepts derived from cognitive modeling. One version has two subcategories with the disadvantages of more required clicks; the other has only one level of sub-categories and therefore requires fewer clicks. In this paper, we develop modeling concepts for different aspects of usability. Aspects, such as efficiency, suitability for learning and the development of mental models are measured. A combination of empirical data and cognitive modeling approaches is presented.

We propose that, when it comes to menu structures, it is important that a menu should be designed to fit the cognitive capabilities of humans. If designers focus on reducing clicks, they might miss the turning point, that less clicks are associated with more cognitive load (e.g., memory load). In general, care should be taken in the design of applications, so that the principles of optimal human information processing are met. It is commonly agreed, that human knowledge is represented in form of a semantic network [34]. Within this network, categories are associated with subcategories and retrieval of subcategories succeeds best and faster when the category representations are addressed. In the study we will assess how well an application meets the mental model that users have about the application.

Novice (first interaction with a new version) and expert (second interaction with a new version) behavior will be compared, so information about the evolving mental model of users can be obtained. Furthermore, the suitability for learning can also be measured. Most studies investigating menu designs are strongly controlled and hence artificial laboratory studies [16] [26]. Quite the contrary is the case for the current study, which is conducted with a smartphone application. Furthermore, a very realistic task is utilized. Test persons are asked to select products from a shopping list application which provides them with a ready to use shopping list. This paper presents a combination of an empirical study of the usability of an Android shopping list application with cognitive modeling approaches. Cognitive modeling of the user behavior incorporates the full ACT-R architecture. Although visual processing of menus is modeled, this study focuses on the development of an adequate mental model and learning processes as users would do it.

B. The Application

Both versions of shopping list application are designed for Android. The application allows users to select products out of either an alphabetically ordered list or via categorical search (see Fig. 1). The chosen products are then added to a list. The difference between the two versions is menu depth: The three layer version (*3L*) has one more menu level than the two layer version (*2L*). The first page of the application is the same for both versions: three buttons are presented: "overview", "shops" and "my list". For both versions, after

selecting "overview" the list of the alphabet appears. 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. A click on a small checkbox in the right of the product selects it. If users click on shops, the *categorical pathway* is accessed. For both versions, clicking on shops results in a list of seven shops (bakery, drugstore, deli, greengrocer, beverage store, stationery, and corner shop). Each of these shops is represented by a button. For the *2L version*, 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.) and are selected by a click on the checkbox. For the *3L* version, the shops have seven subcategories, each. 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 and can be selected via the checkbox. 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. Different product pathways for alcohol free beer. The orange path is the alphabetical pathway. The green and blue paths are the categorical pathway. The green pathway is the pathway of the *3L* app, the blue of the *2L* app.

C. Procedure

The first study is designed in order to investigate if the user interaction with the two versions differs on a statistical significant level. It further allows a conclusion on the overall usability of the application- namely on efficiency, effectiveness and suitability for learning. To ensure conditions close to real-life interaction, all navigating possibilities the application provides are allowed. An ACT-R modeling approach concerning the evolvement of user's mental model and the influence of cognitive load is also presented. The second study supplements the first study. The functionality of the application is restricted, only the categorical pathway is allowed for product search. This restriction is necessary for two reasons. First, it substantiates assumptions about mental models of users derived from the first study. Second, the restriction allows investigating learning curves. With help of the learning curves differences between products can be uncovered. In both studies participants were asked to find products, using both versions of the shopping list application. The application was presented on a Google Nexus 4 smartphone, running with Android 4.1.2. Each product was read to the participant by the experimenter and then the participant was asked to find the product and select it. In the first study participants were free to choose the pathway, which led them to the products. They were instructed to use the different possibilities the app provided, so they could either find the products via the *alphabetical* or via the *categorical* pathway. In the second study, participants were asked to select products merely using the *categorical* pathway. 26 student participants (12 male and 14 female, age_{mean}= 23) participated in the first study and 17 student participants (6 male and 11 female, age_{mean}= 26) participated in the second study. After receiving standardized oral instructions participants were instructed 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. After selecting a product, participants were asked to return to the first page and then the next trial started.

TABLE I. DESIGN OF STUDY 1

order of versions	*3L* (new)	*3L* (expert)	*2L* (new)	*2L* (expert)
***3L* first, *2L* second**	Block 1	Block 2	Block 3	Block 4
***2L* first, *3L* second**	Block 3	Block 4	Block 1	Block 2

Enforcing the participants to always return to the first page (e.g., starting point) was necessary for reasons of experimental control. After selecting eight or nine products, participants were asked to read the shopping list (in order to assure learning of the store categories). For the next block, the items were identical 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. For the first study half of the participants first worked with the *2L version* and the other participants began with the *3L version* (see Table I). In the second study all participants first worked with the *2L version* and then switched to the *3L version* (see Table V).

IV. RESULTS

A. Study 1

1) Hypothes:

a) The first study investigated how product search time is influenced by menu depth, expertise and version specific expectancies, e.g., users' mental model of the respective application. Product search time is an indication of efficiency.

b) The main difference between both *versions* of the application is menu depth. We expected overall product search time to be longer for the three layer version than for the shallower two layer version.

c) As participants become more familiar with the application, we expected product search time to decrease as experience increases. Otherwise, the application would not be *suitable for learning*.

d) Finally, we were interested in how *version specific expectations*, gained with one version of the application can influences performance in product search with the other application. We propose that learning transfers from one version to the other will occur.

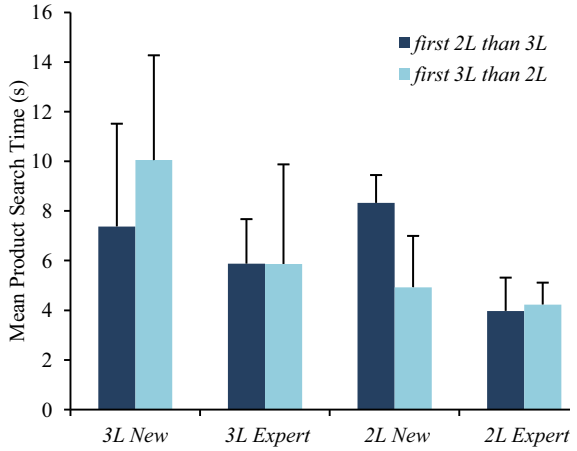

Figure 2. Mean trial time of study 1.

2) Descriptive results: All products could be found with both versions by all groups of participants, therefore, effectiveness of the application is given. Fig. 2 shows the mean trial time and standard deviations for the different conditions. The green bars represent the group *2L first*, *3L second* and the blue bars the group *3L first*, *2L second*.

For participants of group *3L first*, *2L* second, the mean trial time of block 1 (*3L* new) is 10.048 seconds and decreases approximately 4 seconds for block 2 (*3L* expert) (mean trial time 5.861 seconds). After switching to the *2L* version (*2L new*) time decrease to 4.928 seconds (block 3) and reaches 4.229 seconds for *2L expert* (block 4). For participants of group *2L first*, *3L second* a trial in the first block (*2L new*) has a mean duration of 8.322 seconds and a trial in the second block (*2L expert*) a mean duration of 3.971 seconds. After participants switch to *3L* version (block 3, *3L new*) time increase to 7.376 seconds and decreases again to 5.875 seconds (block 4, *3L expert*).

3) Statistical analysis and results: To investigate how product search time is influenced by menu depth, expertise and version specific expectancies a 2x2x2 ANOVA was conducted.

The following factors were considered: the factor *order of the versions* with the two steps *"3L first, 2L second"* and *"first 2L than 3L";* the repeated measurement factor *version* with the two steps *"3L version"* and *"2L version"* and the repeated measurement factor *expertise* with the two steps *"new"* and *"expert"*.

The overall result of Levene test for sphericity was not significant therefore, the overall distribution of the error variance is equal in all groups and an ANOVA could be performed.

The ANOVA reveals a significant main effect of factor *version* with $F(1,24)=12.527$, $p<0.005$ and a medium effect size (partial $\eta 2=0.343$). Descriptive results indicate that the *2L version* is overall faster than the *3L version*, indicating that shallower menu depth, results in less search time. This effect is labeled *version effect.* Another significant main effect is found for the factor *expertise* $F(1,24)=29.625$, $p<0.001$ and a medium to large effect size (partial $\eta 2 =0.552$). Descriptive results show, that performance in the new conditions is slower than in the expert conditions, which is a clear indication that learning occurs. This effect is labeled *experience effect.*

The interaction between *version* and *order of the versions* is also significant $F(1,24) =7.076$, $p<0.05$, with a medium effect size (partial $\eta 2=0.228$). The interaction between *version, novelty* and *order of the versions* is further significant, with $F(1,24) =13.661$, $p<0.001$ and a medium effect size (partial $\eta 2=0.363$).

TABLE II. DESCRIPTIVE STATISTICS OF STUDY 1

		Mean	Std. Deviation	N
3L new	*2L* first, *3L* second	10.048	4.137	13
	3L first, *2L* second	7.376	4.220	13
	total	8.712	4.315	26
3L expert	*2L* first, *3L* second	5.861	1.793	13
	3L first, *2L* second	5.875	4.013	13
	total	5.868	3.045	26
2L new	*2L* first, *3L* second	4.928	1.122	13
	3L first, *2L* second	8.322	2.063	13
	total	6.625	2.375	26
2L expert	*2L* first, *3L* second	4.229	1.340	13
	3L first, *2L* second	3.971	.879	13
	total	4.100	1.118	26

Our data show a *version effect* (the *2L version* is overall faster than the *3L*), an overall *experience effect* (main effect of expertise) and an interaction between all three factors, which we label *version specific expectation effect,* and which might be related to users expectancies. These expectancies might be influenced by users exposure to different version.

4) Post-hoc Tests: To uncover the origin of the significant effects, post-hoc t-tests were computed. Differences within groups can be discovered with paired sample tests and differences between the groups with independent sample t-test. Note that the alpha level was set to 0.01 (instead of 0.05) in order to counteract alpha-error accumulation.

The paired sample test reveals the following interesting effects: a *learning through experience effect,* a *transfer effect* and a *switching effect.* When the corresponding version is presented first, for both versions, a statistical significant *learning through experience effect* is revealed by comparing the new and the expert condition. For *2L version first*, the difference between new and expert is highly significant ($t(12)=6.940$, $p<0.001$) as is the difference between new and expert for *3L version first*, with $t(12)=3.590$, $p<0.005$. The comparison between new and expert condition with the same version, but presented as second version revealed no significant effects. So for both versions performance in the third and fourth run does not improve significantly. Nevertheless, the improvement between new users and expert users is a clear indication that both versions of the application are suitable for learning.

There are significant *transfer effects* in the group *3L first, 2L second*; as there is a significant improvement between *3LN and 2LN*, with $t(12)=5.221$, $p<0.001$as well as from *3LN to 2LE* ($t(12)= 5.098$, $p<0.001$ and from *3LE to 2LE* (($t(12)=3.591$, $p<0.01$).

TABLE III. POST-HOC COMPARISON WITHIN SUBJECTS

order	conditions	t	df	sig. (2-tailed)	effectsize (dz)
2L first, *3L* second	*2LN*vs. *2LE***	6.940	12	0.000	1.924
	2LE vs. *3LN**	- 3.273	12	0.007	0.907
	2LN vs. *3LE*	1.956	12	0.074	0.542
	2LE vs. *3LE*	-1.699	12	0.115	0.471
	3LN vs. *3LE*	1.272	12	0.227	0.352
	2LN vs. *3LN*	0.795	12	0.442	0.220
3L first, *2L* second	*3LN* vs. *2LN***	*5.221*	12	0.000	1.448
	3LN vs. *2LE***	5.098	12	0.000	1.414
	3LE vs. *2LE**	3.591	12	0.004	0.995
	3LN vs. *3LE**	3.590	12	0.004	0.995
	3LE vs. *2LN*	1.537	12	0.150	0.426
	2LN vs. *2LE*	1.343	12	0.204	0.372

Note that *2L* and *3L* connote in *2L* version and *3L* version and *E* in expert and *N* in new.

For the group *2L first, 3L second* a significant *switching effect*, e.g., a drop in performance between *2LE and 3LN* is revealed (t (12) = -3.273, p<0.01).

The independent sample t-test compares conditions that do not comprise of the same users. For novice users, interacting the very first time with this application, descriptive results indicate a general advantage for the *2L version*. Nevertheless, on a statistical level, for first time users, both versions are equally difficult (*3LN1 vs. 2LN2*, t (24) =1.346, p=0.2). For expert users, without experience from a different version, the *3L version* is significantly slower, than the *2L version* (*3LE1* vs. *2LE2*, t (24) = 3.412, p<0.005).

by the data, a conceptual ACT-R model was developed that does the same task as the participants.

The model was written to search for products just as human participants do. For simplicity, only product search via the *categorical* pathway is modeled. As first step encoding of the requested product is required. Please note that this paper does not focus on visual-motor processing, and no eye tracking data is collected, we will present merely the core concept of how the models mental model changes. Nevertheless, supplementing the model with visual processes is unproblematic. This is done through goal buffer.

TABLE IV. POST-HOC COMPARISON BETWEEN SUBJECTS

conditions	t	df	sig. (2-tailed)	effectsize(ρ)
2LE2 vs. *2LN1*	-6.000**	24	0.000	2.353
2LN2 vs. *2LN1*	-5.211**	24	0.000	2.043
3LN1 vs. *2LE2*	5.181**	24	0.000	2.032
3LE1 vs. *2LE2*	3.412*	24	0.002	1.338
3LE1 vs. *2LN2*	-3.247*	24	0.003	1.273
3LN1 vs. *3LE2*	2.611	24	0.015	1.023
2LE2 vs. *3LN1*	-2.563	24	0.017	1.005
2LN2 vs. *2LE1*	2.421	24	0.023	0.949
2LN2 vs. *3LN1*	-2.021	24	0.055	0.792
3LN1 vs. *3LN2*	1.631	24	0.116	0.639
2LE2 vs. *3LE1*	-1.403	24	0.173	0.550
3LN1 vs. *2LN2*	1.346	24	0.191	0.528
3LE1 vs. *3LN2*	-1.192	24	0.245	0.467
2LN2 vs. *3LE2*	-0.819	24	0.421	0.321
2LE2 vs. *2LE1*	0.580	24	0.567	0.227
3LE1 vs. *3LE2*	-0.012	24	0.991	0.004

Note that *2L* and *3L* connote *2L* version and *3L* version. *E* means expert and *N* means new. The numbers 1 and 2 correspond to the group. Number 1 stands for the group *3L* first, *2L* second and 2 for group *2L* first, *3L* second. For example *2LE2* vs. *2LN1* is the comparison between the *2L* version expert, where the *2L* version is presented as first version (group 2) versus the *2L new* version, when the *2L* version is presented second (group 1).

But as users become more experienced with the application in general (e.g., comparing *2L expert second* with *3L expert second*) both version do not differ on a statistical level (*2LE2* vs. *3LE1*, n.s.). Table IV presents the results of a two-way t-test between the two groups. In conclusion, statistical differences between both versions are found, but for real novice users and very experiences users, both versions do not differ on a statistical level. Therefore, efficiency is the same for both versions.

5) The ACT-R Modell: In order to get a deeper understanding about the causes and mechanisms indicated

Hence, the goal buffer contains the information about what product is required. The next step represents an attempt to retrieve information from declarative memory about the *version specific category membership* of the required product.

This knowledge is represented as a chunk and consists of all vital information for finding the product in the application. Hence, *version specific category membership chunks* contain all information necessary to navigate the application. These chunks represent the mental model. The slots of the chunks contain the words leading to the product (see Fig. 3). For models new to the application, *version*

specific category membership chunks are nonexistent and for this reason the retrieval is unsuccessful. Therefore, such *version specific category membership chunks* have to be build via the general semantic knowledge. The general semantic knowledge depicts world knowledge and includes *association chunks* between products and shops and between products and subcategories. This general semantic knowledge (*association chunks*) is utilized for searching the requested product. If the retrieval of a *version specific category membership* chunk is unsuccessful, a different product search strategy is selected. Word for word, each term represented on screen, is read. For each word the attempt to retrieve an *association chunk* between the desired product and the current word is made. For example if the requested product is *alcohol free beer* and the word *bakery* is read, an attempt to retrieve a chunk that holds an association between *bakery* and *alcohol free beer* is made. If such a chunk cannot be retrieved, the next word is read, for example *beverage store* and again the attempt to retrieve a chunk that holds an association between *alcohol free beer* and *beverage store is made.* If such a chunk can be retrieved, the word is selected and a *version specific category-membership chunk* is build.

In summary, navigating the application is modeled via two types of chunks- *association chunks* and *version specific category membership chunks*. *Association chunks* represent world knowledge and contain association between different words and can be retrieved from declarative memory without prior exposure to the application. *Version specific category membership chunks* represent the mental models of the pathways leading to the products. They are built in the models imaginal buffer, during exposure with the application and can only be retrieved if the product was encountered before.

6) Discussion of the effects: Is a shallower menu-structure beneficial?

a) Empirical: The *version effect* discovered in the ANOVA indicates that the *2L version* is overall faster than the *3L version.* Post-hoc tests show no statistical difference, neither for real novice users, nor for very experienced users. On a descriptive level, as users become more familiar with the application, the benefit of a shallower version is less relevant, since the difference *2L expert* vs. *3L expert* is less than the difference between *3L new* vs. *2L new*.

P2
```
alcohol-free-beer-P2
isa product-path
first                bottleshop

final                alcohol-free-beer
```

P3
```
alcohol-free-beer-P3
isa product-path
first                bottleshop
second               beer
final                alcohol-free-beer
```

A 1
```
beer-alcohol-free-beer
isa assosiation
target               alcohol-free-beer
associate            beer
```

A 2
```
bottleshop-alcohol-free-beer
isa assosiation
target               alcohol-free-beer
associate            bottleshop
```

Figure 3. An example of different chunk-types used in the model. P2 and P3 are the version specific category membership chunks for the 2L and the 3L version, respectively. A1 and A2 are the association chunks.

The building of these chunks takes place in the imaginal buffer. Fig. 3 represents the different chunks used in the model for the product *alcohol free beer*. After the word is selected, the next page of the application opens and the process of reading and searching for *association chunks* is repeated. Furthermore, if retrievals of *association chunks* are successful, the *version specific category membership chunk* is supplemented. Finally, when the requested product is found and clicked on, all slots of the *version specific category membership chunk*s are filled with content. The chunk is then cleared from the imaginal buffer and placed into declarative memory. Thus, when the product is requested a second time, a complete product path is available and can be retrieved.

b) Explanation: A shallower menu structure requires fewer clicks than a deeper menu structure. But more interesting in this context is, the question to what extent higher level cognitive processes such as memory retrievals are responsible for the difference in product search time between the two versions. We argue that more memory retrievals can be seen as a higher amount of cognitive load. Issues concerning cognitive load, can best be answered via cognitive modeling.

c) Modeling: The building of *version specific category membership chunks* out of *association chunks* continues until all product pathways are established. This building process of *version specific category membership chunks* takes longer for *3L version* than for *2L version*. Since the *3L*

version requires more interaction steps and therefore more encoding of these steps than the *2L version*. Furthermore, for the *3L version* more retrievals of general knowledge (about which shop holds which subcategory, and which subcategory holds which product) are needed. The knowledge of subcategories is unnecessary for the *2L version*. An "expert" model can retrieve *version specific category membership chunks* for all products. Retrievals from declarative memory are much less frequently then for a "learning" model, functioning with *association chunks.* For both versions the number of retrievals is the same as soon as interaction with the application is realized solely via *version specific category membership chunks*. So, for an "experienced" model, the number of clicks alone is responsible for differences in search time.

Note: If simply motor processes (e.g., clicks) differ between the two versions, a new study should investigate if the benefit of the *2L version* is still measurable for products that require menu scrolling, or if the *3L version* may be more favorable for such products.

In general linear hierarchical applications with shallower menu structures require more scrolling than those with a deeper menu structure. It is very plausible that for expert users, a deeper menu-structure with less scrolling processes is more beneficial than a shallower menu structure, since the amount of cognitive load (memory retrievals) is the same for both versions.

a) Does Learning occur?

a. *Empirical:* The data show a clear *experience effect* as participants become more familiar with the application, the mean trial duration decreases.

b. *Modeling:* Learning, defined as the reduction of product search time, can be explained by the modeling approach as follows: As long as a mental model of the product pathway for all products is not complete, the constant retrieval of *association chunks* is necessary. For each processed word a retrieval request for an *association chunk* containing both the product and the current word is made. If such an *association chunk* cannot be retrieved the next word is read and the process is continued until an *association chunk* is found. Then the word is selected and *version specific category membership chunk* is built up. Searching, encoding and retrieving chunks take time. When *version specific category membership chunks* are available in declarative memory the number of retrievals is reduced-resulting in reduced product search time. So, a "novice" model is constantly visual searching, encoding and retrieving chunks. An "expert" model, on the other hand has the relevant knowledge about specific product pathways and therefore, less time is spent on retrieving chunks.

In conclusion, the main reason for learning is that a mental model of the application is built. This mental model consists of the relevant product pathways. As soon as the *version specific category membership chunks* can be retrieved, performance increases. Furthermore, an adequate mental model results in less cognitive load, since less memory retrievals are necessary for navigating.

a) How do version specific expectations influence performance?

Transfer effect: From the *3L* to the *2L version.*

Empirical: Performance improvements between *3L* to the *2L version* are labeled *transfer effect.*

Modeling: After repeatedly interacting with the *3L version* of the application, a mental model for the *3L version* exists. This means that *version specific category membership chunks* containing the correct product pathway for this version for all products are represented in declarative memory. The version of the application is then changed, but the task is kept the same. Without any kind of disturbance, the *3L-version specific category membership chunks* are adequate to fulfill the task with the *2L version* of the application. This is the case, since no additional information needs to be learned when switching from *3L version* to the *2L version* (note that the *3L version* includes all menu-structures of the *2L version*, but has more menu depth). Therefore, performance does not drop when switching from the *3L version* to the *2L version*.

Switching effect: From the *2L* to the *3L* version

Empirical: A *switching effect* occurs when participants familiar with the *2L version* change to the *3L version*. In the empirical data the effect is visible, in the increasing product search time from *2L first expert* to *3L second new*.

Nevertheless, participants who use the *3L version* second benefit from their experience with the *2L version* (product search time for *3L second* is lower, than for *3L first*).

TABLE V. DESIGN OF STUDY 2

order of versions	***2L* (new)**	***2L* (expert)**	***3L* (new)**	***3L* (expert)**
***2L* first, *3L* second**	Block 1	Block 2	Block 3	Block 4

Modeling: Switching from the *2L version* to the *3L version* irritates the users because they end up with a menu they did not expect and are not familiar with. In terms of the modeling approach this implicates that *2L-version specific category membership chunks* do not lead to the required product, when these are deployed with the *3L version*. On the third page of the application redemption of strategy is required and the problem is solved via *association chunks.* New *version specific category membership chunks* are built for the *3L version.* When the *3L version* is presented a second time, these chunks can be retrieved and the product search time decreases.

b) General remarks for Modeling Menu Structures: In the presented approach, mental models of product pathways for linear hierarchical menus are represented as chunks. The slot values of these chunks depict the categories, subcategories and the target in the hierarchical menu. In

order to build a mental model, *association chunks*, containing associations between words are used. These *association chunks* serve as general semantic knowledge. Performance improvements occurring when novice become expert users, are explained through the change of strategy. Novice users rely on *association chunks* and experts on chunks containing the mental model of the specific product pathway. A strategy depending on associations requires more retrievals than a mental model strategy. The more retrievals a strategy needs, the higher the cognitive load. If experienced users are confronted with a different menu hierarchy then the one they are familiar with, depending on the fashion of the new hierarchy, two opposite effects can occur. In general, if hierarchy levels are reduced, existing mental models of product pathways are still useful and productive, a *transfer effect* occurs. On the other hand, if hierarchy levels are added, new mental models of product pathways are required, a *switching effect* is found.

A. Study 2

A second study was conducted, to substantiate findings and model assumptions of the first study and also to investigate the influence of expectancies on product associations. Due to the small number of participants in the second study, statistical tests are not computed. As in the first study participants were asked to search for products with the shopping list application. The products were read to the participants and the participants had to search for the products in the application, select the products and then return to the first page. This procedure was the same, as in the first study except, that participants could only search via the stores pathway. This constraint guaranteed that participants built up a mental model of the product path from the start. The study design is another difference to the first study. In the second study all of the participants first worked with the *2L version* and then switched to the *3L version*.

1) Hypothesis

a) Learning and Version Update:

The same *experience* and *switching effects* as in the first study were assumed. We expected product search time to decrease, if the same version is used the second time and to increase after the version switches to a version with extra menu layers.

b) *The Influence of Expectations:* We predicted that longer product search times occur for category pairs that are more unfamiliar than others. We further predicted that as category affiliation become more familiar, differences in search time between products disappear.

2) Results

a) Empirical Results: As Fig. 4 shows, there is a clear switching effect (e.g., an increase in mean product searches time, after participants switch from the *2L* version to the *3L* version). Similar to the previous study, the data show a clear experience effect, with product search time decreasing as participants become more familiar with the version of the application; therefore, for both versions (*3L* and *2L*) product search in the new condition takes longer than the expert condition. To decipher how product search time differs between different products, learning curves were computed (see Fig. 5). These present the mean product search time for each individual product. With such en detail information, differences between specific products can be uncovered. The exact product search times are also presented in Table VI. In both of the new conditions strong time variations can be observed.

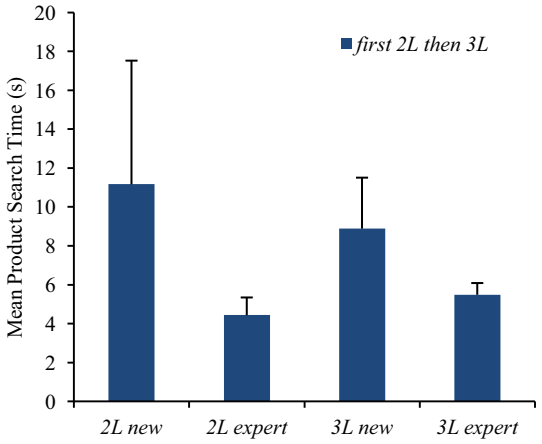

Figure 4. Mean trial time of study 2.

In the expert conditions, on the contrary only small variations between the different products exist.

A possible explanation for the observed variations in the new condition is, that some products are easier to find than others. The non presence of variations in the expert conditions indicates, that users have a correct mental model, e.g., they have learned the item labeling of the application. Hence, a qualitative explanation for product search time variations between different products will be provided.

Products that result in large search time in the new condition are the second *clabbered milk*, the third *canned pineapple* and the eighth product *gilthead*. Products with rather short product search times in the new condition are product number four *body wash* and product number seven *top-fermented dark beer*.

In post-hoc questioning the participants revealed that they expected *clabbered milk* in the *beverage store* and not in the *deli* as it was presented in the app. They also reported that they did not expect *canned pineapple* in the *corner store* and some participants were not aware that *gilthead* is a fish. A plausible explanation for variations between products is the fact, that some category pairs are more familiar for the participants than others. Higher standard deviations for the more uncommon products in the new conditions also provide evidence for this explanation (see Table II).

TABLE VI. MEAN TRIAL TIME PER ITEM FOR STUDY 2.

	new					expert			
	***3L* mean**	***3L std***	***2L* mean**	***2L std***		***3L* mean**	***3L std***	***2L* mean**	***2L std***
Alkoholfreies Bier alcohol free beer	8.918	*8.268*	10.408	*8.510*	*Alkoholfreies Bier* alcohol free beer	5.521	*1.643*	3.984	*0.907*
Dickmilch clabbered milk	10.513	*9.029*	20.596	*8.242*	*Dorade* gilthead	5.143	*1.213*	5.378	*5.758*
AnanasDose canned pineapple	10.007	*13.380*	17.091	*9.222*	*AnanasDose* canned pineapple	5.706	*2.026*	5.249	*4.337*
Duschgel body wash	7.346	*2.103*	4.844	*2.117*	*Duschgel* body wash	5.566	*1.410*	4.108	*0.747*
Amerikaner black and white cookie	11.241	*7.187*	5.585	*4.836*	*Edamer* Edam cheese	5.274	*1.242*	5.514	*2.394*
Edamer Edam cheese	7.053	*2.413*	11.114	*8.260*	*Altbier* top-fermented dark beer	4.920	*1.556*	3.895	*1.678*
Altbier top-fermented dark beer	4.717	*1.238*	3.970	*1.054*	*Acrylfarbe* acrylic paint	5.536	*1.909*	4.259	*1.706*
Dorade gilthead	13.232	*15.250*	19.189	*21.592*	*Dickmilch* clabbered milk	6.896	*2.357*	4.913	*3.161*
Acrylfarbe acrylic paint	7.024	*2.069*	7.759	*5.885*	*Amerikaner* black and white cookie	4.853	*1.034*	2.683	*1.230*

For most of the products, participants take longer with *2L version new* than with *3L version new*. This is probably due to learning transfer over the version, as discussed in the first study.

For product number four *black and white cookie* product search time with the *3L version new* is longer than with the *2L version new*. Post-hoc questioning revealed, that participants did not expect *black and white cookie* in the subcategory *danish (pastry)*.

Modeling: *The modeling approach from the first study can easily be complemented to explain the effects revealed by the learning curve. The results discussed indicate that uncommon products and product category pairs result in longer search times.*

To model such an effect, the general semantic knowledge of the model should be modulated, so that *association chunks* exists, that are correct in daily life (e.g., *clabbered milk* is associated with *beverage)* but misleading for the application. This would make it possible for the model to make errors. These errors would then result in longer product search times, if *association chunks* retrieved from declarative memory result in misleading product path. Another possibility to model longer product search times for uncommon words would be through parametric adjustments to the model. Common *association chunks* are required more often, than uncommon *association chunks*, therefore the activation of the common chunks should be higher, making unsuccessful retrievals of uncommon *association chunks* possible and therefore resulting in longer product search times.

3) Conclusion: In the second study, the *same experience effect* (experts are faster than novices) and the same *transfer effect* as in the first study are observed. These effects are even found when only *the categorical pathway* is used. This restriction ensures that the *product pathway* is represented by version specific *category membership chunks* as described in the first study.

Furthermore, an extending modeling approach offers two explanations that can account for variations in product search time in the new conditions. One explanation is that for uncommon product category pairs misleading *association chunks* are retrieved.

The other is that for unfamiliar products *association chunks* are used very rarely and therefore these chunks have a lower activation and retrieval failures occur.

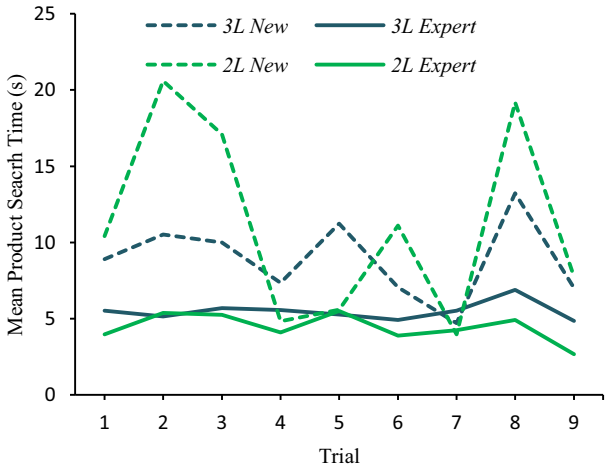

Figure 5. Mean learning curves for study 2.

Thus, designer should use only such labels for categories that are unambiguously linked to the products. Especially for first time users of an application intuitive labeling is

important. Expert users, on the other hand, can cope with uncommon labels, since they have a correct mental model.

V. DISCUSSION

A. Summary study 1

Two versions of a linear hierarchical real-life shopping list application for Android, differing in menu depth, were compared via user test. Product search time for different products gave insight into the following usability factors; efficiency, effectiveness and suitability for learning. Furthermore, novice and expert behavior and switching between both versions were investigated. The user study revealed an experience effect, namely that expert users are faster than novice users- a clear indications for suitability for learning. Overall product search with the *2L version* is faster than with the *3L* version, making the shallower version slightly more efficient. But the difference between the two versions is neither significant for users new to the application, nor for those very experienced with the application. A shallower menu hierarchy seems to be faster to handle, if the users are somewhat familiar with an application, but not jet very experienced. Both versions are effective to compose a shopping list. A *switching effect* was observed; product search time increases when switching from the *2L version* to the *3L version*. The *transfer effect*, on the other hand, is that product search time decreases when the *3L version* is presented as a first version and the *2L version* as a second version. An ACT-R based model was used to explain the results of the user study. It demonstrates how users might build up a mental model of the application. At the beginning, users need to use their general knowledge (*association chunks*) to find associations in between each processed word and the target item. Through successful navigating, they build *version specific category membership chunks* that contain the product path of the target item. These chunks represent the user´s mental model of the application. With this mental model the expert user then navigates quickly to the target item. If the different version is presented, the mental model either is still suitable for navigation (*transfer effect*) or needs to be revised (*switching effect*). Besides having an adequate mental model, the model explains the increase in performance between novices and expert, due to a reduction in cognitive load. Inexperienced users need to retrieve information about associations very frequently from their declarative memory. Each retrieval takes time. The number of retrievals is much less for experienced users, so they are faster.

B. Summary study 2

Study two was conducted to support the findings and modeling assumption of the first study; moreover, to supplement information about the nature of expectations of users new to the application. The task, application and products where kept the same, although the functionality of the application was reduced to the categorical pathway and version switches were exclusive from the *2L version* to the *3L* version. Descriptive results indicated the experience and *switching effect*. Furthermore, the evaluation of learning curves showed that in the new condition some products result in longer search times than others. An extension of the modeling approach of the first study provided two possible explanations for this: Either that the retrieval of misleading *association chunks* for uncommon product category pairs is responsible, or that too low activation of *association chunks* of unfamiliar products leads to retrieval failures.

C. Conclusion and outlook

1) Advice for designers: For a linear hierarchical menu application, that allows users to select items, the number of menu layers is not important. Especially for frequent users searching for products using a 3 or 2 layer menu, is equally efficient and effective. On the other hand, as long as users are not completely familiar with the application, a shallower menu is beneficial. It is important that these findings need to be verified with products that require scrolling. It seems plausible that more layers (resulting in more selection time) reduce the scrolling time. If version updates are necessary, designers should be aware that introducing an extra layer will reduce efficiency until users are familiar with the modification again (the switching effect). On the other hand, an update which reduces the number of layers, does not have a negative influence on efficiency (the transfer effect). Furthermore, intuitive labeling of categories is very important, especially for first time users. Designers should concentrate on finding categories, where the affiliation to the items found in these categories is immediately clear to the target population of the application. Language effects influencing potential users, such as regional terms should be considered. Besides, common categories should be preferred to uncommon ones. Though, if the scope of an application is essentially experts, with every-day exposure, intuitive labeling is less relevant. Experts can handle uncommon labels as well as common ones.

2) On the specific findings of our model: Two different chunks are used in the modeling approach- *association chunks* and version specific category membership chunks. Association chunks contain the association between the target product and categories- either shops or product categories. They represent general semantic knowledge and a novice model searches for products using its *association chunks*. Such a strategy requires much retrieval of *association chunks*. The *version specific category membership chunks* are the *mental model* of *product pathways* containing the target and the first and second category that leads to the product. Expert users can rely on these chunks to navigate through the application. The approach shows that these two different strategies are used by novice vs. expert users. The difference in efficiency observed between the *2L* and *3L* versions for learning users is explained through the different amount of *cognitive load* when the association chunk strategy is used. Using this strategy requires more retrievals for *3L* than for the *2L*

version. For experts, who rely on their mental model, both versions are equally efficient and require the same amount of *cognitive load*, since the number of retrievals of *version specific category membership chunks* is the same for both. When the other version is presented, the *version specific category membership chunks* are either still suitable for navigation (transfer effect) or need to be revised (switching effect). Variations in product search time in the new conditions are explained by the modeling approach through retrieval failures or through retrievals of non matching *association chunks*.

3) General conceptual modeling remark: This paper illustrates how usability influencing factors, such as efficiency, effectiveness and suitability for learning can be assessed with ACT-R based cognitive models. Further concepts such as users' mental model and cognitive load are evaluated, too. The mental model of the pathway of a linear hierarchical application can be modeled through chunks, which slots contain the target and the categories and subcategories. Such a mental model can be constructed via general semantic knowledge, which consists of *association chunks. Association chunks* have two slots, one for the target and one for the associated category. User expectations influence the handling of applications. We showed that unexpectedly associated word pairs can result in retrieval failures, due to low activation of these *association chunks*. This paper opted for a straight forward approach to the concept of cognitive load- the more retrievals from declarative memory was equalized with more cognitive load. Such an approach needs further validation.

4) Outlook: This paper focused on higher level cognitive processes and usability. Motor and visual processes were covered peripherally. Nevertheless, ACT-R provides possibilities to include exact visual processing of lists in its models and this should be done in the future. This is done best together with a model of the higher level mechanisms identified in this work. We provided a psychological plausible modeling approach for modeling the interaction of a smartphone application. Our approach is straight- forward, making transfer to other applications possible. In this work, the model was used to explain results obtained in user test, measuring efficiency, effectiveness and suitability for learning. The ACT-Droid tool allowed the model to directly interact with the application. In order to reach the long-term goal of using merely cognitive models to evaluate usability, other usability concepts have to be modeled. In the case of linear menu structures one should further model and investigate the following aspects: A considerable issue worthwhile to study is how differences between menu-hierarchies are affected by scrolling. When another layer was introduced to the application, we discovered a switching effect. A transfer effect occurred after removing one layer.

Besides the number of layers, other changes in the workflow of applications could be analyzed and tested for similar effects in the future, without having to rely on expansive user studies. Other important factors to include in the model are visual and motor processes. It would be helpful to evaluate these assumptions of the model with eye-tracking data. Further, it is interesting to see, at which point performance improvement of the model saturates. This would provide the opportunity to create a real expert model, for which learning behavior does not improve further. Especially a precise model of different kinds of user errors and different menus and a study focusing on mobile context is necessary for an overall model on the usability of menus. Furthermore, models of different user groups, such as elderly users, should be constructed. Such user groups would be presented through a set of parameters. For our approach to be a real alternative to user studies, it should allow for testing more complex applications. This will require implementing more actions to be simulated by the model, including scrolling, but also potentially costomization of interfaces (eg. favorits).

Cognitive load is a crucial factor for usabiliy, especially for novice users. Mobile applications are a use case for mental load evaluation. On mobile devices, with very limited space, users usually have no possibility to externalize their working memory to the device (e.g., making notes while working). Therefore, users have to rely on their working memory completely, thereby increasing the cognitive load. Complex applications are more demanding in terms of cognitive load, which is hard to measure in user studies. With cognitive modeling however, cognitive load can be assessed [35]. This provides a clear advantage of our aproach over traditional user studies.

REFERENCES

[1] N. Russwinkel and S. Prezenski, "ACT-R meets usability or why cognitive modeling is a useful tool to evaluate the usability of smartphone applications," in Proc. the Sixth International Conference on Advanced Cognitive Technologies and Applications (COGNITIVE 2014) IARIA, May 2014, pp. 62–65, ISSN: 2308-4197.

[2] 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/ 2014.11.10.

[3] J. Nielsen, "Usability engineering". Morgan Kaufmann Pub, 1994.

[4] C. Engel, J. Herdin, and C. Maertin, "Exploiting HCI pattern collections for user interface generation" in Proc. of the 4th International Conference on Pervasive Patterns and Applications (PATTERNS 2014) IARIA, 2012, pp. 36–44, ISSN: 2308-3557.

[5] P. Kortum and S. C. Peres, "The Relationship Between System Effectiveness and Subjective Usability Scores Using The System Usability Scale," Int. J. Hum. Comput. Interact., vol. 30, no. 7, 2014, pp. 575–584, doi:10.1080/10447318.2014.904177.

[6] J. Nielsen, “User satisfaction vs. performance metrics,” 2012. [Online]. Available from: http://www.nngroup.com/articles/satisfaction-vs-performance-metrics/ 2014.11.10

[7] S. McDougall, M. Curry, and O. de Bruijn, “The effects of visual information on users’ mental models: an evaluation of Pathfinder analysis as a measure of icon usability,” Int. J. Cogn. Ergon., vol. 5, no. 2, pp. 153–178, 2001, doi: 10.1207/S15327566IJCE0501_4.

[8] D. Zhang and B. Adipat, “Challenges, methodologies, and issues in the usability testing of mobile applications,” Int. J. Hum. Comput. Interact., vol. 18, no. 3, pp. 293–308, Jul. 2005, doi: 10.1207/s15327590ijhc1803_3.

[9] R. Harrison, D. Flood, and D. Duce, “Usability of mobile applications: literature review and rationale for a new usability model,” J. Interact. Sci., vol. 1, no. 1, p. 1, 2013, doi: 10.1186/2194-0827-1-1.

[10] C. D. Wickens, “Multiple resources and mental workload,” Human Factors: J. of the Human Factors and Ergonomics Society vol. 50, no. 3, pp. 449–455, June 2008, doi: 10.1518/001872008X288394.

[11] S. Cao and Y. Liu, “Mental workload modeling in an integrated cognitive architecture,” Proc. Hum. Factors Ergon. Soc. Annu. Meet., vol. 55, no. 1, pp. 2083–2087, Sep. 2011, doi: 10.1177/1071181311551434.

[12] M. D. Byrne and R. W. Pew, “A history and primer of human performance modeling,” Rev. Hum. Factors Ergon., vol. 5, no. 1, pp. 225–263, Sep. 2009, doi: 10.1518/155723409X448071.

[13] B. E. John and S. Suzuki, “Toward Cognitive Modeling for Predicting Usability" Proc. Human-Computer Interaction, HCI International, July 2009, pp. 267-276, doi: 10.1007/978-3-642-02574-7_30.

[14] S. Möller, K.-P. Engelbrecht, and R. Schleicher, “Predicting the quality and usability of spoken dialogue services,” Speech Commun., vol. 50, no. 8–9, pp. 730–744, Aug. 2008, doi: 10.1016/j.specom.2008.03.001.

[15] J. R. Anderson, J. M. Fincham, Y. Qin, and A. Stocco, “A central circuit of the mind,” Trends Cogn. Sci., vol. 12, no. 4, pp. 136–143, Aug. 2008, doi: 10.1016/j.tics.2008.01.006.

[16] G. Bailly, A. Oulasvirta, D. P. Brumby, and A. Howes, “Model of visual search and selection time in linear menus,” Proc. 32nd Annu. ACM Conf. Hum. factors Comput. Syst. (CHI ’14), pp. 3865–3874, April 2014, doi: 10.1145/2556288.2557093.

[17] E. L. Nilsen, “Perceptual-motor control in human-computer interaction,” University of Michigan, 1996.

[18] D. E. Kieras and D. E. Meyer, “An overview of the EPIC architecture for cognition and performance with application to human-computer interaction,” Human–Computer Interact., vol. 12, no. 4, pp. 391–438, Dec. 1997, doi: 10.1207/s15327051hci1204_4.

[19] S. Lindner, P. Büttner, G. Taenzer, S. Vaupel, and N. Russwinkel, “Towards an efficient evaluation of the usability of android apps by cognitive models,” in Proc. Kognitive Systeme III, 2014.

[20] A. Cockburn, C. Gutwin, and S. Greenberg, “A predictive model of menu performance,” Proc. SIGCHI Conf. Hum. factors Comput. Syst. (CHI ’07), April 2007, pp. 627-636, 2007, doi: 10.1145/1240624.1240723.

[21] B. Mehlenbacher, T. Duffy, and J. Palmer, “Finding information on a menu: linking menu organization to the user’s goals,” Human-Computer Interact., vol. 4, no. 3, pp. 231–251, Sep. 1989, doi: 10.1207/s15327051hci0403_3.

[22] D. Ahlström, A. Cockburn, C. Gutwin, P. Irani, and I. Systems, “Why it’s quick to be square: modelling new and existing hierarchical menu designs,” Proc. of the SIGCHI Conference on Human Factors in Computing Systems (CHI’10), pp. 1371–1380, April 2010, doi: 10.1145/1753326.1753534.

[23] M. Ziefle and S. Bay, “Mental models of a cellular phone menu. comparing older and younger novice users,” in Mobile Human-Computer Interaction - MobileHCI 2004 SE - 3, vol. 3160, S. Brewster and M. Dunlop, Eds. Springer Berlin Heidelberg, pp. 25–37, Sept. 2004, doi: 10.1007/978-3-540-28637-0_3.

[24] M. D. Byrne, J. R. Anderson, S. Douglass, and M. Matessa, “Eye tracking the visual search of click-down menus,” in Proc. of the SIGCHI conference on Human factors in computing systems the CHI is the limit (CHI ’99), pp. 402–409, May 1999, doi: 10.1145/302979.303118.

[25] J. E. McDonald, J. D. Stone, and L. S. Liebelt, “Searching for Items in Menus: The Effects of Organization and Type of Target,” in Proc. of the Human Factors and Ergonomics Society Annual Meeting, vol. 27, no. 9, pp. 834–837, October 1983, doi: 10.1177/154193128302700919.

[26] M. D. Byrne, “ACT-R/PM and menu selection: applying a cognitive architecture to HCI,” Int. J. Hum. Comput. Stud., vol. 55, no. 1, pp. 41–84, Jul. 2001, doi: 10.1006/ijhc.2001.0469.

[27] V. Kaptelinin, “Item Recognition In Menu Selection : The Effect Of Practice,” in CHI ’93 INTERACT '93 and CHI '93 Conference Companion on Human Factors in Computing Systems, pp. 183–184, April 1993, doi: 10.1145/259964.260196.

[28] E. Lee and J. Macgregor, “Minimizing User Search Time in Menu Retrieval Systems,” Human Factors: The J. of the Human Factors and Ergonomics Society, vol. 27, no. 2, pp. 157-162, April 1985, doi: 10.1177/001872088502700203.

[29] A. Geven, R. Sefelin, and M. Tscheligi, “Depth and Breadth away from the Desktop – the Optimal Information Hierarchy for Mobile Use,” in MobileHCI ’06 Proceedings of the 8th conference on Human-computer interaction with mobile devices and services, pp. 157–164, Sept. 2006, doi: 10.1145/1152215.1152248.

[30] K. Samp, “Designing Graphical Menus for Novices and Experts : Connecting Design Characteristics with Design Goals,” in Proc. of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13), April 2013, pp. 3159–3168, 10.1145/2470654.2466432.

[31] A. Cockburn and C. Gutwin, “A Predictive Model of Human Performance With Scrolling and Hierarchical Lists,” Human-Computer Interact., vol. 24, no. 3, pp. 273–314, Jul. 2009, doi: 10.1080/07370020902990402.

[32] A. J. Hornof and D. E. Kieras, “Cognitive modeling reveals menu search in both random and systematic,” Proc. SIGCHI Conf. Hum. factors Comput. Syst. (CHI ’97), April 1997, pp. 107–114, doi: 10.1145/258549.258621.

[33] J. R. Anderson, M. Matessa, and C. Lebiere, “ACT-R: A Theory of Higher Level Cognition and Its Relation to Visual Attention,” Hum.- Comput. Interact., vol. 12, no. 4, pp. 439–462, Sept. 1997, doi: 10.1207/s15327051hci1204_5.

[34] A. M. Collins and M. R. Quillian, “Retrieval time from semantic memory,” J. Verbal Learning Verbal Behav., vol. 8, no. 2, pp. 240–248, 1969, doi: 10.1016/S0022-5371(69)80069-1.

[35] N. Russwinkel, L. Urbas, and M. Thuering, "Predicting temporal errors in complex task environments: A computational and experimental approach," Cognitive Systems Research, vol. 12, no. 3-4, pp. 336-354, Sept. 2011, doi: 10.1016/j.cogsys.2010.09.003.