Interactive Learning of Dynamic Systems:

A Cognitive Modeling Approach to Mental Model Building and Updating.

vorgelegt von

Dipl.-Psych.

Sabine Prezenski

geb. in Stuttgart

von der Fakultät V - Verkehrs- und Maschinensysteme

der Technischen Universität Berlin

zur Erlangung des akademischen Grades

Doktorin der Naturwissenschaften

- Dr. rer. nat. -

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender: Prof. Dr. Markus Feufel
Gutachterin: Prof. Dr. Nele Rußwinkel
Gutachter: Prof. Ph.D. Robert West

Tag der wissenschaftlichen Aussprache: 29. November 2018

Berlin 2019

Abstract

People who want to achieve a goal with a previously unknown technical system simply try it out and learn through feedback from the system how to use it. This is referred to as interactive Learning. To investigate the usability of such systems, elaborate user studies are carried out. Additionally, changes in the system require new studies. Cognitive models simulate the cognitive processes of users and could be used as a substitute or as a supplement to user testing. However, interactive learning of systems, requires that these cognitive models depict elaborate cognitive processes, such as building and modification of mental models. Such aspects can best be modeled with the ACT-R cognitive architecture. However, there is no generally accepted theory on mental model building and updating. There are different views on how a representation is built-up and changes. Cognitive models which depict mental model building in interactive learning of technical systems are unknown of. This dissertation closes this gap; based on the theory of mental models of Li and Maani (2011), ACT-R mechanism are developed and validated.

To investigate interactive learning in technical systems, a smartphone app and a task were developed, with which empirical data was collected. The app is a shopping list app, with a hierarchical-linear structure in which items can be selected. The task of the participants was to repeatedly search for the same items. After two blocks, an update that partially changed the menu structure of the app occurred. Afterwards, the subjects were required search for the same items again. The empirical data showed that participants learned how to use the app quickly and an update interrupted this learning effect. Based on the study, ACT-R modeling mechanisms for building and updating mental model were implemented. To make sure this model, was not too specific to the shopping list app, a new empirical study was conducted with another hierarchical-linear app with a similar look and task. This app is a real-estate app with which subjects should repeatedly enter the same search criteria. After two blocks, the menu structure and position of some entries in the app changed. To test how well the model can describe the empirical data, the model parameters were adjusted to one part of the empirical data of the Real Estate App study. A high match in terms of trends and absolute values was found. The predictive validity of the model was testet by comparing the model data with the other part of the empirical data of the real estate app study and with the data of the shopping-list app study. The model can predict these data very well.

To ensure that the developed ACT-R mechanisms for building and updating mental models are generally useful to depict interactive learning, another interactive task had to be found. Here for, the data from an auditory categorization experiment was used. The auditory stimuli are composed of several features, a certain feature combination was the target stimulus. This had to be identified using button-presses and subsequent feedback. In the middle of the experiment, the assignment of features combination and target tones changed. The ACT-R model based on the previously developed model mechanisms was capable to map the empirical data. In summary, it can be stated that ACT-R can simulate the usage of technical systems thus usability questions can be answered. In this work, a model for interactive learning of hierarchical linear apps was created. The model can map learning and relearning. To predict the usability related aspects of other apps with ACT-R models, however, new ACT-R models need to be created. Hereby, the developed mechanism for mental model building and updating should be adapted to adequately address the structure of the specific app.

These ACT-R mechanisms for mental model building and updating can describe and predict empirical data from various interactive tasks. Furthermore, the mechanisms can be transferred to other different scenarios. For this, however, a task must be newly modeled. For the assignment of the mechanisms, a task analysis must be carried out, and the new model may also need additional assumptions.

Zusammenfassung

Wenn Personen ein Ziel mit einem Ihnen bisher unbekannten technischen System erreichen wollen, dann probieren sie das System einfach aus und lernen durch die Rückmeldung vom Systems den Umgang damit. Man spricht von Interaktivem Lernen. Um die Usability eines solchen Systems zu untersuchen, werden aufwendige Nutzerstudien durchgeführt. Zudem erfordern Änderungen im System neue Studien. Kognitive Modelle simulieren die kognitiven Prozesse von Nutzern und könnten als Ersatz oder auch Ergänzung zu Nutzertests eingesetzt werden. Interaktives Lernen von Systemen, erfordert allerdings, dass diese kognitiven Modelle elaborierte kognitive Prozesse abbilden. Das sind z.B. der Aufbau und Umbau von mentalen Modellen. Solche Aspekte kann man am besten mit der kognitiven Architektur ACT-R modellieren. Allerdings gibt es keine allgemein anerkannte Theorie zu Aufbau und Umbau von mentalen Modellen. Es gibt unterschiedliche Ansichten darüber, wie eine Repräsentation aufgebaut wird und sich verändert. Kognitive Modelle, die den Aufbau eines mentalen Modells beim interaktives Lernen von technischen Systemen abbilden sind nicht bekannt. Diese Dissertation schließt diese Lücke, in dem aufbauend auf der Theorie zum mentalen Modellen von Li und Maani (2011), ACT-R Mechanismen entwickelt und überprüft werden.

Um interaktives Lernen bei technischen Systemen zu untersuchen, wurde zunächst eine Smartphone App and eine Aufgabe entwickelt, mit der empirische Daten erhoben wurden. Bei der App handelt es sich um eine Einkaufslisten App, mit einem hierarchisch-linearen Aufbau, bei der Produkte selektiert werden können. Aufgabe der Probanden war es, wiederholt dieselben Produkte zu suchen. Nach zwei Durchgängen gab es ein Update, dass die Menüstruktur der App teilweise veränderte. Im Anschluss sollten die Probanden erneut dieselben Produkte suchen. Die empirischen Daten zeigen, dass Probanden den Umgang mit der App zügig erlernen und dass ein Update den Lerneffekt unterbricht. Basierend auf der Studie wurden ACT-R Modelmechanismen für den Aufbau und die Änderung vom mentalen Modell implementiert. Um sicherzustellen, dass dieses Modell nicht zu spezifisch für die Einkaufslisten App konstruiert ist, wurde eine neue empirische Studie mit einer anderen hierarchisch-linearen App mit einem ähnlichen Aussehen und einer ähnlichen Aufgabe durchgeführt. Bei der App handelt es sich um eine Immobilien App, in der die Probanden wiederholt dieselben Suchkriterien eingeben sollten. Nach zwei Durchläufen veränderte sich für manche Einträge die Menü-Struktur und die Position in der App. Um zu überprüfen, wie gut das Modell empirische Daten abbilden kann, wurden die Modellparameter an einen Teil der empirischen Daten aus der Immobilien-App Studie angepasst. Es zeigte sich eine sehr hohe Übereinstimmung in Bezug auf Trends und auch auf absolute Werte. Die prädiktive Güte des Modells wurde überprüft, indem die Modelldaten mit dem anderen Teil der empirischen Daten aus der Immobilien-App Studie und mit den Daten aus der Einkaufslisten App Studie verglichen wurden. Das Modell kann diese Daten sehr gut vorhersagen.

Um sicherzustellen, dass dBie entwickelten ACT-R Mechanismen zum Aufbau und Umbau von mentalen Modellen generell nützlich sind, um interaktives Lernen abzubilden, musste eine andere interaktive Aufgabe gefunden werden. Dazu wurden die Daten eines auditorischen Kategorisierungsexperiments verwendet. Die auditorischen Stimuli setzten sich aus mehreren Eigenschaften zusammen; eine bestimmte Eigenschaftskombination war der Zielstimulus. Dieser musste mittels Tastendruck und anschließendem Feedback identifiziert werden. In der Mitte des Experiments änderte sich die Zuordnung von Zieltönen. Das auf den zuvor entwickelten Modellmechanismen basierte ACT-R Modell konnte die empirischen Daten gut abbilden.

Zusammenfassend lässt sich feststellen, dass sich die Benutzung von technischen Systemen mit ACT-R Modellen simulieren lässt und somit Usability Fragen beantwortet werden können. In dieser Arbeit wurde ein Modell zum interaktiven Lernen von hierarchischen linearen Apps erstellt. Das Modell kann Lernen und Umlernen abbilden. Um auch für andere Apps die Benutzbarkeit mit ACT-R Modellen vorauszusagen, sollten allerdings die in dieser Dissertation entwickelten Mechanismen zum Aufbau und Umbau von mentalen Modellen in Bezug auf die Struktur dieser App angepasst werden.

Mit den entwickelten ACT-R Mechanismen zum Aufbau und Umbau von mentalen Modellen können empirischen Daten aus verschiedenen interaktiven Tasks beschrieben und vorhergesagt werden. Ferner lassen sich die Mechanismen auf komplett andere Szenarien übertragen. Hierfür, muss allerdings eine Aufgabe neu modelliert werden. Dabei muss für das Zuordnen der Mechanismen eine Aufgabenanalyse durchgeführt werden und das neue Modelle benötigt eventuell noch zusätzlichen Annahmen.

Overview of Papers

The following four papers, in the Publisher's Version, make up this dissertation.

Paper No. 1

Prezenski, S. and Russwinkel, N. (2014). Combining cognitive ACT-R models with usability testing reveals users mental model while shopping with a smartphone application. Int. J. Adv. Intell. Syst, 7:700-715.
http://www.iariajournals.org/intelligent_systems/intsys_v7_n34_2014_paged.pdf

Paper No. 2

Prezenski, S. and Russwinkel, N. (2016). Towards a general model of repeated app usage. In Proceedings of the 14th International Conference on Cognitive Modeling, pages 201-207.
http://acs.ist.psu.edu/iccm2016/proceedings/ICCM2016proceedings.pdf

Paper No. 3

Prezenski, S., Brechmann, A., Wolff, S., and Russwinkel, N. (2017). A cognitive modeling approach to strategy formation in dynamic decision making. Frontiers in psychology, 8:1-18.
https://doi.org/10.3389/fpsyg.2017.01335

Paper No. 4

Prezenski, S. (2017). Implementing mental model updating in ACT-R. In Proceedings of the 15th International Conference on Cognitive Modeling, pages 121-127.
http://acs.ist.psu.edu/papers/ICCM%202017%20Proceedings.pdf

Contents

Synopsis

Interaction with various technical systems, such as applications, is an integral part of our everyday lives. Early in the morning, we are woken up by the alarm clock app installed on our smartphone. At breakfast, we read newsfeeds on our tablet. We use a navigation device that indicates the fastest route to our workplace. At work, we accomplish our tasks with the help of numerous computer programs. We even track our evening run with our smartwatch, and later evaluate the success of our training on our computer.

Users do not stick to a handful of applications for which they exactly know how to use them. Rather, unfamiliar applications are tried out. An indication hereof is the increase of available smartphone apps in Google Play Store by 5343 new apps on average each day (statista.com, 2017). Although, many apps are tried out, repeated use of an app occurs only if users can easily reach their goal when initial interacting with an app.

But how do users quickly and seamlessly achieve goals with systems prior unknown to them? This work aims to provide a better understanding of how users learn to interact with unfamiliar systems, such as new smartphone apps.

When required to accomplish a task with an unfamiliar system, very few users choose to consult the operations manual. Instead, a "learning by doing" - approach is followed (Nielsen & Budiu, 2012). Users merely try out the new system. For such an approach to be successful, it is important that systems are designed in a similar fashion and follow common interaction philosophy. This is exactly what guidelines are for. For apps, such guidelines facilitate interaction design and make sure that important aspects of apps (such as their menus) are build-up in the same way (Android Design Guidelines; Apple iOS Human Interface Guidelines). An app that deviates from the recommendations of these guidelines often causes problems (Zapata et al, 2014); or as UX-experts would point out, such an app is not intuitive. Naumann et al. define intuitive use as “A technical system is, in the context of a certain task, intuitively usable while the particular user is able to interact effectively, not-consciously, using previous knowledge” (Naumann et al, 2007, pp.129). This means a new system is usable right away or that only very few erroneous interactions occur. With each interaction, users learn more about the system (and the task they are accomplishing with it). Furthermore, repeated interaction is often accompanied by minor system changes, such as changes of the information displayed on a smartphone screen. Systems that keep changing while users interact with them are referred to as dynamic systems. Changes in dynamic systems are either based on user input or internal system processes. An example for user input is a user selecting search criteria in a smartphone app. Hereby, the selection changes the displayed content of the app e.g., a new page opens. An example for internal system changes are software updates.

This work aims to provide a better understanding of how users deal with changing dynamic systems. Light is shed upon dynamic system interaction that users perceive as effortless, as well as on problematic cases.

System interaction (initial or after occurred change) is effortless, if the user requires only a few interactions until success. It needs to be studied how users directly achieve their goals with dynamic systems. Furthermore, it needs to be analyzed why users can quickly, e.g., within a few additional interactions and without errors learn how to use dynamic systems.

On the other hand, problematic cases need to be identified. An explanation needs to be found, that provides insight into questions such as, why some system interfaces are perceived as difficult to use. Also, why after some system updates, usage frequency declines.

Albeit, there are many different motives why users interact with a system – this work focuses on goal-directed tasks. These are tasks in which users

pursue a specific goal with the use of the system (for example, selecting items for a shopping list smartphone application). In alignment hereof, the questions why and how users can quickly learn to achieve a goal in a task with a new dynamic system need to be addressed.

For the user to learn how to achieve a goal with a dynamic system, interactions with it are crucial (Maani and Li, 2011). With each undertaken interaction more is learned about the system. This is because, each state of the system (change or no change) is feedback for its user (Maani & Li, 2011). For example, in a smartphone app, if the action of the user (e.g., selection of criteria) results in the opening of a screen, and if this corresponds to the user's expectation, it is positive feedback. In this work, the term feedback is used in this manner. Thus, feedback does not necessarily need to be something like "correct" or "wrong". More so, in interaction learning, any system behavior following user behavior is feedback.

For a better understanding of how exactly interactive tasks with dynamic systems are learned, a detailed investigation into the underlying cognitive processes needs to be undertaken.

A well-established hypothesis in cognitive sciences is that internal representations (or mental models) of tasks and systems presuppose our behavior. In this manner, learning can be described either as a mental model building or as an adjustment of similar mental models. In most cases, learning how to achieve a goal with an unknown dynamic system requires only a few interactions with that system. Thus, users can very successfully build or adjust a mental model of sch a task with a system and integrate the dynamic changes into it; provided, they receive feedback during this interaction. According to theories on mental models, systems problematic to learn, are those that do not correspond to our usual experiences or per-knowledge with systems. For such systems, no mental models exist which could be used to integrate new aspects into it.

An example here for, would be a system where procedural prior knowledge is not considered correctly because the user is guided through the system from "right to left" instead of usual way which is "left to right". According to the cognitive-artifact-task triad (Gray and Altmann, 2000), a complete analysis of system interaction requires to consider the task, the artifact and the cognitive processes together. In the case of the unusual reading direction, not only is the mental model of the system itself difficult to learn, but there are also problems with the task solved with the artifact because they are closely linked. In the following section on users' mental models of systems, therefore, the tern system refers to the task and the artifact together.

The concept of mental models is commonly applied to identify usability problems in user studies. Here, users´ mental models of system usage are consulted to identify aspects of interfaces which should be adjusted to achieve better usability (Norman, 2013). The underlying idea hereof is that the more similar the mental model of users and the designers' mental models of users' system usage is, the higher is the system's usability (Norman, 1985). In user studies, the mental models of users are commonly investigated using descriptive qualitative methods, such as verbal or graphical methods. These methods are useful to discover flaws of the studied system. Although mental models are described widely in usability studies, there is no unifying theory on mental models (see Payne, 2003 for a structured discussion on mental models in HCI). Rather, various theories describe what mental models are and how they are built and adjusted. (Norman, 1983; Glenberg & Langston, 1992; Zhang, 2009; Maani & Li, 2011; Revell & Stanton, 2014). It is commonly agreed upon that the term mental model refers to internal, abstract representations of reality (Norman, 2013). Moreover, mental models evolve naturally (e.g., through interaction with a system) (Norman, 1983). Most importantly, they need to be functional but not necessarily technical adequate (Norman, 1983). Additionally, mental models are (mental) simulations to predict future states of systems (Klein et al., 2006). Furthermore, mental models are not static but can be altered during system interaction (Norman, 1983). This dynamic nature means that with every new incoming information (e.g., feedback) mental models are either verified, adjusted or discarded.

A more detailed description regarding the cognitive phases underlying the dynamic processes of mental model building and updating is given by Maani and Li (2011).

Additionally, their approach emphasizes the importance of feedback for adjustments of the mental model. According to Maani and Li mental model updating (and building) can be depicted as a continuous circle, consisting of three reoccurring phases namely conceptualization, experimentation and reflection. During conceptualization, an understanding of the situation is acquired, and the outcome of potential actions is mentally simulated. Thus, the current situation is compared to information of the mental model and new information obtained from the environment is integrated into this mental model. During the experimentation phase, the actions derived from the mental model are tested. In the reflection phase, the outcome of the experimentation phase is evaluated by processing feedback. If the outcome is as expected (positive feedback), the mental model is kept. If not (negative feedback), the mental model is updated.

Even though the cognitive phases of mental model updating are labeled and described in Maani's and Li's work, the exact nature of the emerging processes remains unclear. Not because, the theory as such is imprecise, but rather since it is merely specified on a verbal level of description. Something, it has in common with many psychological theories, especially those on mental models. Questions that remain unanswered concern the exact details of the proposed cognitive processes of mental model building and updating. Thus, "*what is meant with an understanding of the situation*", "*how this situation is encoded*" and "*what exactly is updated*".

The questions arise if and how the processes underlying mental model building and updating in dynamic system interaction can be precisely specified. What options are there, besides defined verbal theories, to decipher these processes?

Computational approaches are a promising candidate to achieve more precision. Such approaches use algorithmic descriptions to present details of cognitive processes (Sun, 2008). Already, computational implementations of mental model theories exist (Khemlani, Trafton, & Johnson-Liard, 2013). However, the primarily focus lies on modeling deductive thinking in reasoning task. These tasks give participants plenty of time to find solutions to problems like: "If all models are wrong, and this is a model, then it is wrong".

But, is it possible to use a computational approach for more dynamic scenarios - such as interactive system learning and still be specific about the exact processes?

This is where cognitive architectures come into play. With cognitive architecture approaches complete theories are implemented in a computational model. The crucial aspect hereby is modeling the interplay of different cognitive processes together. This distinguishes cognitive architectures not only from mathematical models of cognitive processes (which express relationships between variables with mathematic formulas (Sun, 2008)), but also from classical experimental approaches widely used in psychology. In experimental approaches specific variables are isolated and the input-output relationship is then analyzed. Newell´s argument against experimental approaches "you can´t play 20 questions and win" (Newell, 1977) is still valid. In this regard, if we want to find out how human cognition arises - a classical psychological approach of isolating aspects cannot provide the answers needed for a comprehensive understanding of cognitive processes underlying human interaction in real-world settings. Instead, the interplay of different cognitive processes, such as perception, stimulus processing and motor reactions, needs to be considered. More so, aiming at the bigger picture seems especially important with the goal of a better understanding of cognitive processes influencing interaction with real systems.

Therefore, this work aims at developing a cognitive architecture approach to mental model building and updating in dynamic systems.

So, what exactly are cognitive architectures?

Various cognitive architectures exist; however, EPIC (Kieras & Meyer, 1997), SOAR (Laird, 2012) and ACT-R (Anderson et al., 2004) are the most prominent ones. They differ mainly in the details of how specific cognitive functions are implemented, for an overview see Byrne (2003). In this work, the cognitive architecture ACT-R (Adaptive Control of

Thought-Rational) was chosen for applied and theoretical reasons. In the following, a short overview of these is given.

Motives on the applied side are: First, there is an active and open-to-the-public ACT-R community which provides feedback and interaction on ACT-R related topics, including an annual conference and workshop and a well-maintained website (http://act-r.psy.cmu.edu/) for models and publication. Second, specific tools exist, which allow ACT-R models to directly interact with real (Android) smartphone applications (Büttner, 2010; Doerr, Prezenski & Russwinkel, 2016). Since this work seeks to address how users interact with real-systems, this is a significant advantage over the other architectures. A final practical advantage of ACT-R is that it has already been used in numerous complex real-world tasks involving system interaction, such as problem-solving of air traffic controllers (Raufaste, 2006) or website navigation (John et al., 2004).

The motivation for ACT-R on the theoretical side is: First that ACT-R has motoric as well as perceptual components, which allows models to interact with its environment system. This is especially important for studying system interaction. Second, ACT-R was originally developed for modeling memory and problem-solving processes closely linked with mental model building. Therefore, extensive evaluation of the implementation of these mechanisms (such as retrieval mechanisms) has taken place. Finally, theoretical concepts closely related to mental model building have been successfully modeled with ACT-R. These are instance-based learning (Gonzalez, 2017), meta-cognition (Reitter, 2010) and sense-making (Lebiere et al., 2013). The core assumption of instance-based learning is that decisions are made depending on previous experiences in similar situations (Gonzalez, 2003). These experiences are called instances and consist of situations and related outcomes which stored in memory. The instances can be seen as the core part of a mental model since they consist of an encoding of the situation and the predicted outcome. However, mental model building and updating also requires reflection and evaluation on the learning process, or in other terms, meta-cognition (Reitter, 2010). Finally, sense-making means finding and interpreting facts that are relevant among all the incoming information. Making sense of a situation requires sorting stimuli into different categories. Here for previous experience and rules about category affiliation are essential.

To conclude, ACT-R is a cognitive architecture qualified for investigating the processes of mental model building during system interaction. **For a deeper understanding of ACT-R,** a brief outline of the processes of this architecture is given subsequently. For more information, each of the four papers of this dissertation introduces the ACT-R mechanism relevant to their specific content in more detail.

ACT-R is a theory about how the interplay of cognitive processes together produce behavior it is based on findings from cognitive psychology and neuroscience. Aspects of the theory are altered if new scientific evidence suggests so. ACT-R is also a computational platform, for modeling these processes. The underlying theory restricts how a model can be implemented. For example, due to limited resources, information units cannot be altered simultaneously by the working memory resource of ACT-R but only one after the other. These restrictions ensure that the assumptions of ACT-R theory are not violated.

ACT-R has a modular structure, resembling the functionality of our brain. These modules process information in parallel; not unlike our ability to simultaneously handle different types of information. Different modules process different types of information through their interfaces, called buffers. For example, there are visual and aural modules and corresponding buffers for sensory processing or motor and speech modules and corresponding buffers for output. Each piece of information is called a chunk. Chunks are often combined into larger pieces of information. All chunks are stored in and retrieved from the declarative memory module via its retrieval buffer. Only one chunk can be in a buffer at any given time, signifying bottlenecks of human information processing. If-then-rules, called productions, are the core part of ACT-R models, they represent the procedural knowledge (in the sense of how to handle specific information) and are stored in the procedural module. When a model is running, only one production can be selected at any

moment (again a bottleneck) and the production is then placed into the procedural buffer. A production is only selected if the states of the specified buffers match. This process is called pattern matching. If more than one production matches, subsymbolic processes, which will be introduced shortly, come into play. The execution of each production takes time, in most cases 50 milliseconds. This corresponds to the duration of a cycle of procedural learning in the brain, thus sending information from the cortex to the basal ganglia, and back (Anderson et al, 2004). The execution of a production can change buffer states, initiate motor output and form new chunks. The forming of new chunks, e.g. combining information into something new, is an important aspect of learning and mental model building and updating. New chunks can only be formed in the imaginal module. The imaginal module and the corresponding buffer depict important processes of working memory, namely manipulating units of information. Also, important for learning a new system are goals and sub-goals. These are represented in ACT-Rs intentional module (goal module) and corresponding buffer. As mentioned earlier, subsymbolic processes are likewise parts of ACT-R. These are based on mathematical formulas that govern how long a specific process takes. Numerous subsymbolic processes exist and most of them are controlled by parameters and can be added to the model or fine-tuned. However, standard values should be used as much as possible. Deviations hereof need to be grounded in a good argumentation. Otherwise, the validity of the model needs to be questioned. The most important subsymbolic parameter for this work is the activation of a chunk. This value influences if a chunk can be retrieved from declarative memory or not. It is a well-known psychological phenomenon that we cannot access every information of our memory. Information that was stored long ago and has not been used since or needs to be retrieved in distracting conditions is more difficult or sometimes impossible to access. However, information we have rehearsed often or a short time ago can be retrieved and this also takes less time. In ACT-R, there is a numerical value for a retrieval threshold, representing how long information it is attempted to retrieve before a failure is reported, e.g. inaccessibility of information. There is a noise parameter, representing the conditions of retrieval. Furthermore, an activation level indicating how highly activated information is; this depends on the time interval and the frequency of past usage. Details on the exact computation and other subsymbolic processes can be found on the ACT-R website (ACT-R, 2017).

Analyzing ACT-R models gives insight into quantitatively and qualitatively aspects of cognition during task performance. The idea hereby is that investigating the models provides understanding on how long such a task will take and more importantly, which cognitive processes lead to the observed behavior. In general, the model performs a task repeatedly, in the same way human participants do in an experiment. But in contrast to experiments with humans, what is going on "in the head of the model" is visible. Thus, an exact specification of occurring cognitive processes is obtained with ACT-R. The direct output of an ACT-R model is time in milliseconds. This way, information on task duration (a quantitative measure) can be obtained. Also, error rates can be acquired by running a model several times and then computing the ratio how often a goal is met. Moreover, a better (qualitative) understanding of specific cognitive processes can be achieved by analyzing the cognitive processes proposed by the model.

But how do we know that the cognitive processes postulated by a model are valid? For this purpose, human data and modeled data of the same task need to be compared. If the model and the participants show the same behavior, then this is an indication that the cognitive processes postulated by the model resemble processes occurring in the human mind. Developing a valid model of a specific aspect of cognition, such as mental model updating requires a two-steps procedure: First, empirical data needs to be matched by a model. Thus, an empirical study needs to be designed, conducted and analyzed. Then, an ACT-R model needs to be developed, and its output analyzed. The output of the model in terms of qualitative trends and quantitative task time should match the empirical data. Second, the model should be able to predict the performance of participants in a slightly different task. Thus, after the behavior of this model in such a task

is analyzed, new empirical data is needed to evaluate the predictions of the model. If the new empirical and the model data match, this is a strong indication for a model validity.

In this manner, the goal of this dissertation is to develop such a valid ACT-R model. Herefor, the question how mental model building and updating during interaction with dynamic systems can be modeled with ACT-R needs to be addressed. Mechanisms, applicable to different dynamic tasks, need to be developed and tested. Hence, the aspects and mechanisms a cognitive model requires for learning and relearning during dynamic system interaction need to be specified.

In addition to in-depth knowledge on key mechanisms of interactive learning and relearning of dynamic interface use, this dissertation aims to address, to what extent these findings can be used to map cognitive processes in the field of "user modeling".

The idea of "user modeling" is to use models to simulate the behavior (and the underlying cognitive processes) of users. The long-term goal hereof is the development of systems that meet the cognitive needs and expectations of their users.

Hitherto, “user modeling" is applied as a tool for evaluating the usability of systems. Usability refers to efficient, effective and satisfactory system use (Standard ISO 9241-11). The models underlying user modeling derive from different disciplines and fulfill different purposes. Mathematical models describe via formulas, specific aspects of user behavior, such as the relationship of the number of items, item position and the search time in linear menus (Bailly, Oulasvirta, Brumby & Howes, 2014). Other approaches use different computational algorithms for aspects such as color, font-size, contrast (Choi et al., 2013, Amalfitano et al., 2012;) These models are useful for designers; they help them determine the efficiency of apps. However, aside from visual processing aspects, these models do not provide insight into cognitive causes of usability problems.

Thus, for designing systems that meet the needs and expectations of their users, it is important that the cognitive processes of these users are simulated.

To a very limited degree, this is attempted by GOMS models. Using GOMS to simulate user behavior requires the modeler to specify Goals, Operators, Methods, and Selection Rules. Even though GOMS models are cognitive, they consist merely of different motor and visual operators. The entire process of thinking is represented by a sole mental operator (assigned a time value). Thus, GOMS models are helpful to find out how long a skilled user requires for a specific task, but they are an oversimplifying approach for more detailed questions.

Such questions can be addressed with models based on cognitive architectures. Already, cognitive architecture tools for user modeling exist. For example, efficiency related aspects can be measured with the ACT-R based tool "CogTool". CogTool provides valid predictions on how long it takes for experienced users on average to find a specific target on a website (John et al., 2004). Furthermore, its successor CogTool Explorer (Teo, John & Blackmon, 2012) can predict search behavior, including errors of users new to a website.

However, models depicting the learning processes (the process of novice turning into expert users) are missing. Furthermore, models resembling interactive learning, learning with very few attempts, as well as a rapid relearning after system changes are needed.

A significant challenge here is of practical matter. In general, developing a model with a cognitive architecture, such as ACT-R is a task that requires experienced modelers to write the model and to prototype the system the model needs to interact with. In work related to this dissertation ACT-Droid was developed, a tool that makes prototyping of smartphone applications written in Android for ACT-R obsolete (Doerr, Prezenski & Russwinkel, 2016). But the modeling effort remains high, and this dissertation, therefore, addresses the question of how it can be reduced. Thus, for a realistic implementation of cognitive models for user modeling, the questions need to be addressed how modeling can be made more

efficient and how this can be achieved without limiting the scope of the model. Hereby, including in depth-cognitive processes, such as mental model updating.

Thus, a model needs to be developed that maps the behavior of users in a goal-directed interaction with an unknown system. First, an empirical study on the dynamics task learning with an unknown interactive system is needed. A straightforward study subject for this is a goal-directed task with an unfamiliar smartphone app. This is because, in general, apps have a limited scope; they are mostly built to solve one specific goal-directed task, such as assembling a shopping list. Furthermore, the most common apps are characterized by a simple and consistent design. This makes tasks with smartphone apps well-suited for a cognitive modeling study. On the one hand, apps provide a controllable study environment, but on the other hand, realistic real-world objects are dealt with.

To verify whether the postulated model processes persist outside of the specific study, the predictive quality of the model needs to be validated with another task. Thus, it is essential that the model not only maps the given empirical data but also can predict new data. This model validation study requires a somewhat different task and system.

Furthermore, a model is particularly useful if its core mechanisms can be transferred to other settings and situations (Marewski & Link, 2014). Hence, the mechanisms for learning and relearning in goal-directed task with dynamic systems should be general. This means they should not only apply to the specific examples studied but are valid in different contexts as well. Here, a different, interactive, dynamic task needs to be found. Thus, by transferring the core assumptions of the model, the usefulness of these for different settings can be examined.

Summary of papers

This dissertation consists of four previously published papers (no. 1: Prezenski & Russwinkel, 2014; no. 2: Prezenski & Russwinkel, 2016; no.: 3 Prezenski, Brechmann, Wolff & Russwinkel, 2017 no. 4: Prezenski, 2017). In the following, a brief overview of each paper is given. The propose hereof is to outline how and to what extent the papers address the research questions and assignments introduced above.

Paper no. 1: Prezenski, S. & Russwinkel, N. (2014). Combining cognitive ACT-R models with usability testing reveals users mental model while shopping with a smartphone application. *Int. J. Adv. Intell. Syst.*, 7(3), 700- 715.

The primary purpose of paper no. 1 is to demonstrate how ACT-R based user-modeling can be applied to test the usability of smartphone applications. Furthermore, how aspects of mental model building and updating are related to usability problems in such an approach.

In paper no. 1, an overview of various assessment usability methods and their benefits and disadvantages is given. The focus hereby lies on methods for testing usability of mobile applications. These are classical usability testing methods, such as thinking aloud and objective user testing, mathematical models of usability aspects and tools based on cognitive modeling. Since a hierarchical list-style smartphone application is studied, special attention is laid on assessment of usability aspects related to menu-design, such as depth of a menu.

Paper no. 1 emphasizes on the benefit of using cognitive architecture approaches for user testing. Firstly, it is argued that these approaches provide insight into the cognitive process such as mental model building and updating of the users and reveal reasons of usability problems. Secondly, the point is made that they are an efficient approach to usability testing since the models can be reused.

An important aim of paper no. 1 is to show that usability testing of real mobile apps is feasible with ACT-R models. Therefore, an app, with a realistic impression and which is usable for an actual non-artificial task was needed. To control the potential influence of confounding variables on usability, such as visual design effects, we opted for a self-designed app instead of an app from an app store. The app is a hierarchical list style app. It supports the task of assembling a shopping list by selecting different items. The items can

either be accessed using an alphabetical ordering or a categorical ordering, e.g., different shops and product categories.

Paper no. 1 aims at providing a better understanding of the cognitive processes of mental model building and updating. Thus, a task that requires users to build up their mental model of the task with the system and update it if changes occur is needed.

Such empirical data is also required for the development of a model. Two empirical studies are designed, conducted and analyzed. For these studies to investigate learning – or mental model building – repeated task processing is needed. For the studies to address mental model updating, aspects of the interface need to change during the study. In both studies, participants are required to search and select the same items repeatedly. This makes it possible to investigate a learning effect by comparing the performances in the first and second run. And in both studies after two blocks (each block consisting of eight items) the interface changes without notice given to the participants. This change affects the number of layers in the app. Either an extra layer was added, or a layer was taken away. The participants were again required to search for the eight items for two more times. The change made it possible to study mental model updating.

The two studies are very similar. The main difference is that in first study, the participants were able to search alphabetically or navigate via the categories- in the other studied navigation was only allowed via categories. The first study´s main purpose is to make sure that mental model building and updating is examinable with the app. Thus, to confirm that statistically significant learning (an improvement from the first to the second run) and re-learning effects (significant behavioral changes after version switches) exist. The second study is restricted because the empirical data is needed to match the model to. Thus, limiting the options of the participants reduces the modeling effort.

The empirical results reveal that there is indeed a learning effect and that changes in the menu-depth also make an effect.

In paper no. 1, the modeling concepts for mental model building and updating are outlined, and it is explained how mental model building and updating can be modeled in such a task. Furthermore, the paper outlines how causes of usability problems can be revealed by the model.

Mental model building is sketched out in paper no. 1 in the following way: In the beginning, the model does not know where a target item is positioned in the menu. Therefore, to navigate to the target (held in the goal buffer) the model uses knowledge of the world chunks. This pre-knowledge chunks represents with each other related or associated aspects. Each item in the menu is visually processed and for each item, the attempt is made to retrieve such an association chunk from declarative memory. On the one hand, if such a chunk cannot be found, the procedure repeats for the next item. On the other hand, if an association between the target item and the current processed menu item is retrieved, menu items are selected, and the core part of a mental model is built-up in form of path-chunks in the imaginal buffer. After this familiarization for target items is complete, the corresponding path chunks can be retrieved and used for direct navigation to these items.

After an update, the previous successful path chunks will not lead to the target. The model will use them until they result in error and then alter them from this point on. More details on the exact implementations are given in paper no. 2.

Paper no. 2 Prezenski, S. & Russwinkel, N. (July 2016b). Towards a general model of repeated app usage. In D. Reitter & F. E. Ritter (Eds.), ***Proceedings of the 14th International Conference on Cognitive Modeling*** **(pp. 201- 207). University Park, PA: Penn State.**

Paper no. 2 presents the implementation of the mechanisms for mental model building and updating in detail.

Mental model building is realized using path chunks build up in the imaginal buffer. But, the model performance is more efficient in the second block, not only because it uses the path chunks to navigate, but also because the position of items is retrieved from memory. Mental model updating is implemented the following way; it uses the successful path chunk after an update until this path chunk leads to an error. The model will try to use the chunk once more (e.g. check if it maybe missed something) and then use its knowledge of the world to build a new path chunk from the point where the old one failed. It will specify the old chunk and the new chunk, thus they become more complex. There is a specific change-detection mechanism in the goal buffer. This specifies if a path chunk has been successful often and if a change in the app has been registered. Thus, if a path chunk has been marked as successful often, but its usage leads to an error it will not directly be discharged. Rather, a second attempt with this path chunk will be made. If this attempt fails again, a new path chunk is build-up using the knowledge of the world chunk.

Furthermore, in paper no 2., modeled data and empirical data of the studies with the Shopping App and the Real-estate App (presented in paper no. 1) are matched and the descriptive and predictive power of the model is evaluated. Thus, the capacity of the model to predict new data with a different app is evaluated. Here for, modeled data and empirical data of a task with this different app are compared. This new app is a real-estate app. It supports the task of selecting criteria for real-estate search. Its appearance is similar to the Shopping App. However, unlike the Shopping App, menu depth in the Real-estate App varies within the same version, and the Real-estate App adapts to preselection of some, but not all criteria. Thus, the position of criteria on a page and the number of layers required to be looked at, before criteria can be found, varies. As with the Shopping App, the task with the Real-estate App required participants to search and select criteria repeatedly. However, some criteria varied, and some stayed the same. Nevertheless, this design allows investigating mental model building and updating.

Also, the paper addresses the practical aspect of how ACT-R can be used to for user modeling and usability prediction without too much effort for the actual modeling process. The model presented is a general model. Consequently, with only small changes, it can predict average user behavior with apps that have a hierarchical list style design. Moreover, the model can handle changes to apps that affect the menu-structure.

Paper no. 3 Prezenski, S., Brechmann, A., Wolff, S. &Russwinkel, N. (2017). A Cognitive Modeling Approach to Strategy Formation in Dynamic Decision Making. *Frontiers in Psychology*. 8 (no 1335). doi: 10.3389/fpsyg.2017.01335

Paper no. 3´s purpose is to test if the core mechanisms for mental model building and updating developed for modeling user behavior with apps (paper no 1. and no. 2) can be extracted and used in a different context. In other words, to test to what extent the prescribe criteria is met (Marewski & Link, 2014).

For this reason, a different task of mental model learning and updating is required. The chosen task is a complex auditory category learning task (Wolf & Brechmann, 2015). Hereby, multi-feature auditory stimuli (complex tones, comprising of four feature combination) are repeatedly presented to the participants. The participants need to learn to discriminate target and non-target stimuli, by using the verbal feedback provided. The alignment of the feature-target relationship changes during the experiment. This task is also a dynamic decision-making task, because it consists of a series of decisions, which depend on the pervious decision and are made in a changing environment under time constraints. Dynamic decision-making is also described as a continuous cycle of mental model updating (Li and Maani, 2011).

Even though the new task requires mental model building and updating, it is nevertheless very different from the tasks with the apps from the previous papers. Hence, a new model was needed to describe the experimental data of this task. However, the core assumptions of mental model building and updating are derived from the model of paper no 2. In the following, a short summary of the new model is given: The model needs to learn which combination of features and button-press is correct. All possible strategies are already

stored as strategy chunks in the model's declarative memory. There are one-feature and two-feature strategy chunks. The model learns which strategy is correct by trying out different strategy chunks. Generally, the model starts with a one-feature strategy and later-on switches to two-feature strategies. Strategies are only changed after negative feedback. They are encoded like the path chunks of paper no 2. The strategy the model is trying out is placed in the imaginal buffer and the goal buffer keeps track of the success of the strategy. For mental model updating the following aspects are the most relevant: If a strategy was successful often, it will not be discharged directly by the model but tested once more (as the path chunks in paper no 2). Furthermore, explicitly registered changes are marked in the goal buffer. After such registered changes, strategy chunks become more specified and thus more complex; differentiating between old and new strategies.

Overall, the comparison of modeled and empirical data indicates that the model captures the data of the participants, including successful initial learning, and reversal learning after the change of feedback contingencies. The modeled data also reflects that not all participants were successful in the two learning phases. However, the overall performance of the model is lower than that of the participants.

Paper no. 3 furthermore, focuses on explaining the theoretical background of the model mechanisms. Thus, the relation of the model mechanisms to dynamic decision making as constant mental model updating (Li and Maani, 2011) is explained. It is outlined to what extent Instance-Based Learning Theory (Gonzalez, 2017) is part of the model and why it is a mixed modeling approach, thus incorporating rule and exemplar-based aspects. Also, meta-cognitive mechanisms required for mental model building and updating are explained.

Paper no. 4 Prezenski, S. (2017). Implementing Mental Model Updating in ACT-R. In M. van Vugt, A. Banks & W. Kennedy (Eds.), *Proceedings of the 15th International Conference on Cognitive Modeling*. (pp. 121-127). Coventry, United Kingdom.

Paper no. 4 provides an answer to the question of how mental model building and updating in interactive tasks can be modeled with ACT-R in general. It is shown how the previously developed core mechanisms for mental model building and updating due to system changes can be implemented using ACT-R. The commonalities of the models of paper no. 1 and 2 and of paper no. 3 are outlined. More specific, it is demonstrated how the mechanism required for modeling tasks of mental model updating can be modeled for these different tasks in the same manner and how they are aligned to and represent a computational implementation of the theory of mental model building and updating by Maani and Li (2001):

The three phases (conceptualization, experimentation and reflection) of the cycle of mental model updating (Li and Maani, 2001) are implemented in the following manner: The central part of a mental model represents the situation, the expected outcome and how successful this mental model was in the past. This is stored as a representation chunk (referred to path chunk in paper no. 1 and no. 2. and to strategy chunk paper no. 3). This part of the mental model is built-up in the imaginal buffer. Thus, the process of building a chunk in the imaginal buffer is a computational implementation of the conceptualization phase. During the experimentation phase, the usefulness of the mental model's prediction is tested. In the reflection phase, the mental model is reflected on and potentially changes to it are initiated. Thus, if the expected outcome is not obtained it is changed. If the predictions of are correct (it is successful), an explicit strengthening mechanism is used to strengthen this representation. As learning evolves mental models become more specific; thus, information is added to the representation chunk. Furthermore, meta-cognitive reflection mechanisms are necessary to record environmental changes and to obtain a learning history. For these kind of meta-cognitive reflection processes the control chunk in the goal buffer is used. Meta-cognition is furthermore required for changing established mental model. So, an in the past useful (or well-established) mental-model does not lead to the expected outcome and the representation chunk needs to be changed (or updated). Hereby, the control chunk is marked with a state of uncertainty. Since, the representation

chunk is established, it is not discharged directly, but tested once more. It is only rebuilt, if it leads to unexpected outcome again. Other meta-cognitive mechanisms, which are also implemented in the control chunk are used to register changes, either to the system, or to the environment.

More details on the exact models, studies and theories are found in the papers and the overall results and implication on mental model building and updating are discussed in the combined discussion of this dissertation.

References

Amalfitano, D., Fasolino, A. R., Tramontana, P., DeCarmine, S., & Memon, A.M. (2012). Using GUI ripping for automated testing of Android applications. *In Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering, ASE* (pp. 258–261), New York, NY, USA. ACM.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review*, 111, 1036-1060.

Anderson, J. R. (2007). *How can the Human Mind Occur in the Physical Universe?* New York, NY: Oxford University Press.

Apple iOS Human Interface Guidelines: http://developer.apple.com/library/ios/#documentation/UserExperience/Conceptual/MobileHIG/Introduction/Introduction.html

Bailly, G., Oulasvirta, A., Brumby, D. and Howes, A. (April 2014). Model of visual search and selection time in linear menus. *Proc. 32nd Annu. Conf. Hum. Factors Comput. Syst. (CHI '14),* (pp. 3865–3874). New York City, NY: ACM. doi: 10.1145/2556288.2557093.

Byrne, M. D. (2003). Cognitive Architectures. In J. Jacko, & A. Sears (Eds.), The human-computer interaction handbook: Fundamentals, evolving technologies and emerging applications (pp. 97-117). Mahwah, NJ: Lawrence Erlbaum Associates.

Buettner, P. (July 2010). "Hello Java!" Linking ACT-R 6 with a Java simulation. In D. D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of the 10th International Conference on Cognitive Modeling* (pp. 289-290). Philadelphia, PA: Drexel University.

Choi, W., Necula, G., & Sen, K. (October 2013). Guided GUI testing of Android apps with minimal restart and approximate learning. In Acm Sigplan Notices, 48(10), (pp. 623-640). NYC: NY: ACM. doi: 10.1145/2509136.2509552.

Doerr, L.-M., Russwinkel, N., & Prezenski, S. (July 2016). ACT-Droid: ACT-R Interacting with Android Applications. In D. Reitter & F. Ritter (Eds.), *Proceedings of the 14th International Conference on Cognitive Modeling* (pp. 225- 227). University Park, PA: Penn State.

Eddy, S. R. (1996). Hidden markov models. *Current opinion in structural biology*, 6(3), (pp. 361-365).

Glenberg, A. M. & Langston, W. E. (1992). Comprehension of Illustrated Text: Pictures Help to Build Mental Models. *Journal of Memory and Language*, 31, 129-151.doi: 10.1016/0749-596X(92)90008-L

Google User Interface Guidelines: http://developer.android.com/guide/practices/ui_guidelines/index.html

Gray, W. D., & Altmann, E. M. (2001). Cognitive modeling and human-computer interaction. In W. Karwowski (Ed.), International encyclopedia of ergonomics and human factors (Vol. 1) (pp. 387-391). New York City, NY: Taylor & Francis, Ltd.

John, B. E., Prevas, K., Salvucci, D. D., & Koedinger, K. (April 2004). Predictive human performance modeling made easy. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 455- 462). New York City, NY: ACM. doi: 10.1145/985692.985750

Kieras, D. & Meyer, D.E. (1997). An overview of the EPIC architecture for cognition and performance with application to human-computer interaction. Human-Computer Interaction., 12, 391-438.

Klein, G.A., Moon, B., Hoffman, R.R. (2006). Making sense of sensemaking 2: A macrocognitive model. *IEEE Intelligent Systems*, 22(5), 88-92. doi: 10.1109/MIS.2006.100

Khemlani, S., Trafton, J. G., & Johnson-Laird, P. N. (July 2013). Deduction as stochastic simulation. In. R. West and T. Stewart (Eds.), Proceedings of the 12th International Conference on Cognitive Modeling (pp. 297- 302). Ottawa, Canada: Carleton University.

Laird, J.E. (2012). The Soar Cognitive Architecture. Cambridge, UK: MIT Press.

Lewandowsky, S., & Farrell, S. (2011). Computational Modeling in Cognition: Principles and Practice. Thousand Oaks, CA, US: SAGE.

Li, A. & Maani, K. (July 2011). Dynamic decision-making, learning, and mental models, in J. Lyneis (Ed.), Proceedings of the 29th International Conference of the System Dynamics Society (pp. 1- 21). Washington, DC: System Dynamics Society.

Marewski J. N. & Link D. (2014). Strategy selection: an introduction to the modeling challenge. *Wiley Interdiscipl. Rev.,* 5, 39- 59. doi: 10.1002/wcs.1265

Naumann, A., Hurtienne, J., Israel, J. H., Mohs, C., Kindsmüller, M. C., Meyer, H. A., & Hußlein, S. (July 2007). Intuitive use of user interfaces: defining a vague concept. In *International Conference on Engineering Psychology and Cognitive Ergonomics* (pp. 128-136). Berlin, Heidelberg: Springer.

Nielsen, J. & Budiu, R. (2012). *Mobile Usability*. San Francisco, CA: New Riders Publishing.

Newell, A. (1973). You can´t play 20-questions with nature and win: Projective comments on the papers of the Symposium. In W.G. Chase (Ed.), *Visual information processing* (pp. 463- 526). New York City, NY: Academic Press.

Norman, D. (2013). *The Design of Everyday Things*. New York City, NY: Basic Books

Norman, D. (1983). Some Observations on Mental Models. In D. Gentner & A. Stevens (Eds.). *Mental Models*, (pp. 7-14). New York City, NY: Psychology Press.

Payne, S. (2003). Users' mental models: The very ideas. In J. Carroll (Ed.), *HCI models, theories, and frameworks: Toward a multidisciplinary science*, (pp. 135-156). Burlington, MA, USA: Morgan Kaufman Publishers.

Prezenski, S. (2017). Implementing Mental Model Updating in ACT-R. In M. van Vugt, A. Banks & W. Kennedy (Eds.), *Proceedings of the 15th International Conference on Cognitive Modeling*. (pp. 121-127). Coventry, United Kingdom.

Prezenski, S.& Russwinkel, N. (2014). Combining cognitive ACT-R models with usability testing reveals users mental model while shopping with a smartphone application. *Int. J. Adv. Intell. Syst.*, 7(3), 700- 715.

Prezenski, S., Brechmann, A., Wolff, S. &Russwinkel, N. (2017). A Cognitive Modeling Approach to Strategy Formation in Dynamic Decision Making. *Frontiers in Psychology*. 8 (no 1335). doi: 10.3389/fpsyg.2017.01335

Prezenski, S. & Russwinkel, N. (July 2016b). Towards a general model of repeated app usage. In D. Reitter & F. E. Ritter (Eds.), *Proceedings of the 14th International Conference on Cognitive Modeling* (pp. 201- 207). University Park, PA: Penn State.

Revell, K. M. A. & Stanton, N. A. (2014). Case studies of mental models in home heat control: Searching for feedback, valve, timer and switch theories. *Applied Ergonomics*, 45(3), 363-378. doi: 10.1016/j.apergo.2013.05.001

Statista.com (Dec 2017). Average number of new Android app releases per day from 3rd quarter 2016 to 3rd quarter 2017. Retrieved from https://www.statista.com/statistics/276703/android-app-releases-worldwide/

Sun, R. (2008). Introduction to computational cognitive modeling. In R. Sun (Ed.), *Cambridge handbook of computational psychology* (pp. 3-19). Cambridge, UK: Cambridge University Press.

Teo, L. H., John, B., & Blackmon, M. (2012, May). CogTool-Explorer: a model of goal-directed user exploration that considers information layout. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 2479- 2488). NYC, NY: ACM. doi: 10.1145/2207676.2208414

Zapata, B. C., Niñirola, A. H., Idri, A., Fernández-Alemán, J. L., & Toval, A. (2014). Mobile PHRs compliance with Android and iOS usability guidelines. *Journal of medical systems*, 38(8), 1-16. doi:10.1007/s10916-014-0081-6

Zhang, Y. (2009). The construction of mental models of information-rich web spaces: The development process and the impact of task complexity (Doctoral dissertation, The University of North Carolina at Chapel Hill).

2 Paper No. 1

Prezenski, S. and Russwinkel, N. (2014). Combining cognitive ACT-R models with usability testing reveals users mental model while shopping with a smartphone application. *Int. J. Adv. Intell. Syst*, 7:700–715.

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.

2014, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

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

2014, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

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

2014, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

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

2014, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

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).

2014, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

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.

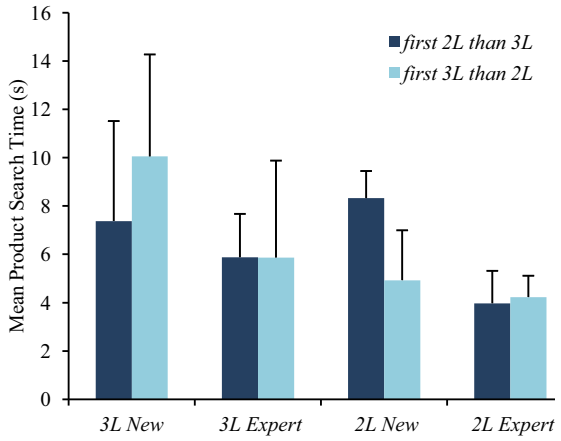

Figure 2. Mean trial time of study 1.

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 $\eta 2=0.363$).

2014, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

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<0.001as 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.

2014, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

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).

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 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.

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

2014, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

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.

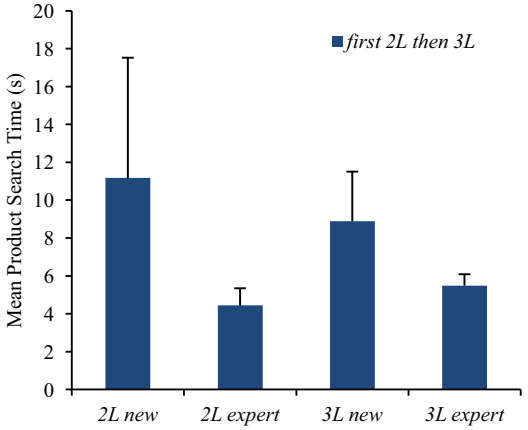

Figure 4. Mean trial time of study 2.

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).

2014, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

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.

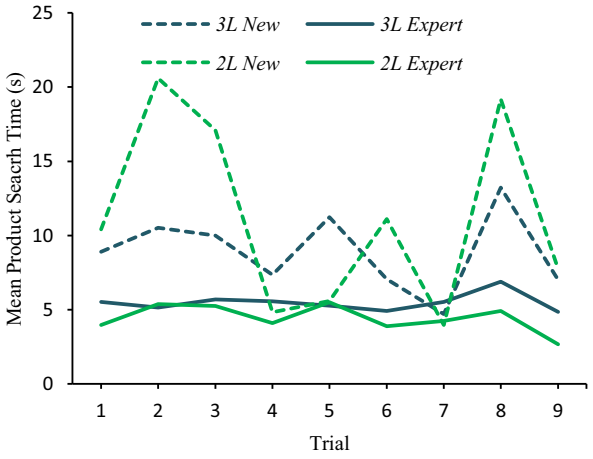

Figure 5. Mean learning curves for study 2.

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

2014, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

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*

2014, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

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.

2014, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[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.

2014, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

3 Paper No. 2

Prezenski, S. and Russwinkel, N. (2016). Towards a general model of repeated app usage. In *Proceedings of the 14th International Conference on Cognitive Modeling (ICCM 2016)*, pages 201–207.

In D. Reitter & F. E. Ritter (Eds.), Proceedings of the 14th International Conference on Cognitive Modeling (ICCM 2016). University Park, PA: Penn State.

Towards a General Model of Repeated App Usage

Sabine Prezenski (sabine.prezenski@tu-berlin.de)
Department of Cognitive Modeling in Dynamic Human-Machine Systems, Technische Universität Berlin
10587 Berlin, Germany

Nele Russwinkel (nele.russwinkel@tu-berlin.de)
Department of Cognitive Modeling in Dynamic Human-Machine Systems, Technische Universität Berlin
10587 Berlin, Germany

Abstract

The main challenge of implementing cognitive models for usability testing lies in reducing the modeling effort, while including all relevant cognitive mechanisms, such as learning and relearning, in the model. In this paper we introduce a general cognitive modeling approach with ACT-R for hierarchical, list-based smartphone apps. These apps support the task of selecting a target, via navigating through subtargets positioned on different layers. Mean target selection time for repeated app interaction, learning and relearning behavior was collected in four studies conducted with either a shopping app or a real-estate app. The predictions of the general modeling approach match the empirical data very well, both in terms of trends and absolute values. We also explain how such a general modeling approach can be followed. The presented general model approach requires little modeling effort to be used for predicting overall efficiency of other apps. It supports more complex interface, as well.

Keywords: ACT-R; usability; apps; cognitive modeling; learning; relearning; updates; general model

Introduction

Numbers of smartphone apps are growing and so is the need for efficient usability testing methods. Cognitive models simulate human behavior and can in theory be utilized either as a supplement to, or instead of real user testing. To achieve this aim, especially in terms of costs and effort, it is crucial to develop valid cognitive models for specific tasks and app characteristics. These models should be written in a general manner, in order to minimize the effort to transfer them to other similar apps. Such general models could then be used to predict specific usability measures like efficiency. This would be particularly helpful in the prototyping phase of apps, where these models could be used instead of user tests. Moreover, the model traces could provide evidence on potential (cognitive) causes of usability problems that are not achievable with user tests.

Theory

Smartphone apps often support a limited amount of tasks and although apps for a great variety of tasks exist, the structures and functionalities of these apps are similar. Consequently, predicting usability of such apps with a general cognitive modeling approach would be useful and worthwhile. In order to provide meaningful predictions, user behavior should be depicted accurately by such general cognitive models. It is crucial that these models are written in a manner that transferability to other similar, but not identical apps, in terms of content and structure, is feasible with minimal effort. Such an approach implicates that not all cognitive mechanisms of users are represented by the models. In respect to transferability, simplifications are necessary for such an approach.

ACT-R has a modularized structure, resembling the architecture of the human brain. Specified modules handle different types of information, called chunks. Each chunk has slots; this is where the smallest pieces of information are stored. The different modules interact via specialized buffers. Visual information is processed by the visual module and its two buffers (visual-object and visual-location). Motor movement is controlled by the manual module and its manual buffer. The declarative module serves as the systems memory and retrieved information from memory is stored in the systems retrieval buffer. The imaginal module and its correspondent buffer are required for learning new information. The steering of the model is governed by the goal module and buffer. The procedural module connects the modules and selects (production-) rules that steer the model behavior. A production is selected and executed, if the states of the buffers are met. The production then alters the states of the modules. Subsymbolic processes are also addressed in ACT-R. If a production requests a chunk and two chunks match the request, then the chunk with the higher activation level is selected. The activation level of a chunk depends on how long ago the chunk was created, on how often it was used and on when it was last accessed. Other parameters are the latency factor which influences the duration until a retrieved chunk is available in the retrieval buffer and the duration until a retrieval failure occurs. The later is also manipulated by the retrieval threshold parameter.

Box 1: A brief introduction to ACT-R

Hierarchical, list style apps are a common type of apps. They are often designed to support the task to find and select a target by navigating through different layers and selecting a subtarget on each layer. This paper presents a general cognitive model of a user interacting with hierarchical list style apps. The model covers repeated interaction, thus investigating learning and relearning effects.

Some cognitive modeling approaches addressing the usability of HMI already exist. The most prominent is CogTool (John, Prevas, Salvucci, & Koedinger, 2004). This is a rapid prototyping tool that enables the creation of cognitive models and predicts execution times for predefined task. But important aspects for the usability of apps such as version updates and learning behavior cannot be modeled with CogTool. A main objective of our work is to develop a model that learns through experience with the interface. A modeling procedure that is strong in

representing learning mechanism is the cognitive architecture ACT-R (for a brief outline of the mechanisms of ACT-R see box 1). Successful ACT-R models of menu and mobile interaction exist. The following aspects of these models are used our model. In an eye-tracking study using desktop computers, Byrne (2001) showed that menu search can be modeled with ACT-R. The fact that this model reads the menu from top to bottom is adopted in our model. In a study on learning of mobile phone usage for elderly novice users (Das & Stuerzlinger 2007), the performance increase in the model was due to the successful recall of locations of keys. Our model also uses the retrieval of locations as a learning mechanism. St.Amant, Horton, & Ritter (2007) developed a model that predicted time on task for expert users searching in hierarchical menus with a feature phone. Their assumptions that experienced users navigate with parent child chunks through hierarchies is implemented in a specialized chunk of our model.

The aim of this paper is to develop and test a general cognitive model with ACT-R that predicts user behavior during repeated interaction. Thus, the model incorporates learning and relearning mechanisms. The modeled task is repeated target selection with different hierarchical list apps.

Methods

Four studies were conducted with either a shopping app (*shopping 3-4, shopping 4-3)* or a real-estate app (*house apartment, apartment house*). Both apps are custom-designed android apps. The empirical studies are presented elsewhere in greater detail (Prezenski, Lindner, Moegele, & Russwinkel, in preparation.; Prezenski & Russwinkel, 2014). Since this paper focuses on the modeling approach, only a brief outline of the apps and the study procedure will be given. See figure 1 for an overview of the apps.

The main functionality of the shopping app is to compose a shopping list. To place products on the list, navigation through various stores and product categories in the menu is required. The real-estate app allows the selection of search criteria for real-estates, such as the number of rooms or the city district. Again, the criteria can be found by navigating through different categories. Both apps are multi-layer hierarchical list apps with variations in menu depth. The main functionality of the apps is target selection via navigating through a number of layers (see figure 1). On each layer a subtarget has to be preselected. For each target, there is only one correct path of subtargets leading to the target. Two versions of the shopping app are used: one with three and one with four layers of menu depth for all targets. As illustrated in figure 1, the path leading to the target e.g. *alcohol free beer* differs between the two versions. The real-estate app has a mixed number of layers (either three or four layers per target). Furthermore, the real-estate app is adaptive. Depending on preselection the paths leading to targets and the position of some subtargets changed. As can be seen in figure 1 the path leading to the target *lawn* differs if either house or apartment has previously been selected.

Figure 1: Screenshots of the apps with the modified paths leading to the targets for the different versions (shopping app) or different previous selections (real-estate app).

Task

Participants repeatedly selected targets using the apps installed on a Google Nexus 5 smartphone, running android 4.1.1. Targets were read to the participants and after selecting the target, participants were required to navigate back to the first layer of the app. In all four studies there were four runs, each run required the participants to select a number of targets. Participants of the studies *shopping 3-4*, and *shopping 4-3* had to select nine targets (products) per run. The same targets were used for all four runs. After the second run the version of the shopping app was updated, either a layer was added (*shopping 3-4*) or removed (*shopping 4-3*). Thus, the paths leading to the targets were the same for the first and second layer but were altered from the third layer on. Participants of the studies with the real-estate apps had to select six or seven targets (criteria) per run. Some of the targets were the same for all runs, e.g. numerical criteria such as *the rent* remained the same for all four runs. Others, like *the city district* varied between all four runs. Participants of the study *apartment house* searched for an apartment in the first two runs and then switched to searching for a house. The order was reverse for participants of the study *house apartment*. Due to the adaptive character of the real-estate app, the pre selection of house or apartment altered the position of the numerical criteria (e.g. the number of rooms) and also changed the path leading to lawn. This path differed for house and apartment from the second layer on.

Model

The data obtained with the studies *shopping 3-4* and *shopping 4-3* was utilized to develop the main model mechanisms and a first ACT-R model. The subsequent studies *house apartment* and *apartment house* were designed for two reasons: First, to test whether the model can predict data obtained with a different app and second, to ensure that the model mechanisms are held in a general matter. Thus, the model incorporates mechanisms for handling variations in depth within an app, changes in paths from varying layers on and variations in locations of targets and subtargets. The task of repeatedly selecting targets in multilayer applications is captured in the model.

Table 1: Examples of the chunk types of the model

meaning chunk	*association chunk*	*path chunk*
NAME "SEARCH" *OBJECT SEARCH*	*OBJECTS HOUSE* *CATEGORY SEARCH*	*FIRST SEARCH* *SECOND WHAT* *THIRD RENT* *FOURTH HOUSE* *TARGET-IM HOUSE* *COUNT FOUR*
chunk with location	*goal chunk*	
SCREEN-POS *VISUAL LOCATION35-0-0* *VALUE "House"* *COLOR BLACK* *HEIGHT 10* *WIDTH 28* *TEXT T*	*STATE PREPARECLICK* *SUBTARGET "SEARCH"* *FINALTARGET "HOUSE"* *TARGET-MEANING HOUSE* *MENUDEPTH FOUR* *IMMOLIST ("MOABIT" ....)* *MENUDEPTHLIST (THREE ...)*	

Summary of Main Mechanisms[1]

Without prior experience with the specific target, visual attention is directed to the top of the page. For each visual processed word, a retrieval request for a *meaning chunk* containing the word as string and as meaning is made (see table 1 for examples of the chunk types used in the model). Navigation through the application is achieved via world knowledge, which consists of associations between two words (*association chunk*). For each read word the attempt to retrieve an *association chunk* with the target is made. If an *association chunk* containing the current word and the target is retrieved, a *path chunk is built*, holding the path leading to the target in the imaginal buffer and the word is selected. A *path chunk* consists of the slots *first, second, third* and *fourth* for the subtargets. The slot *target-im* holds the target word. The *count* slot of the *path chunk* holds information on the current menu depth and is changed if a different subtarget is required.

With experience with the specific target, navigating to this target is realized via the *path chunks* previously built. After a successful retrieval of a *path chunk* a chunk with the location of the relevant subtarget in the path is requested. The retrieved location is visually inspected and the subtarget is selected.

Model steering

Learning mechanisms are incorporated in the model. Furthermore, the model can handle a number of changes to the interface; such as version updates influencing all targets and smaller changes affecting only some targets *Model steering* is implemented with the goal in mind to reuse or extend the model for other applications. Thus, simplifications of some cognitive mechanisms and special chunk slots to account for interface variations are used. Model steering is realized via a *count* slot in the imaginal buffer, which holds the current depth and via different slots in the goal buffer. The *menudepth* slot holds information on changes in depth for the current target (e.g. number of layers leading to the target). The model can handle varying and constant depth values Currently, mechanism exist for a constant depth of three and two layers and for depth changing from three to four and vice versa, with the path leading to the target altered either from the second or from the third layer on. The *menudepth* slot is used to differentiate between strategies for the last versus the other layers in the path leading to the target. The *menudepth* slot is also required after an error in the path leading to the target is noted. From the affected layer on different *path chunks* are built and retrieved. Therefore, the *menudepth* slot holds the assumption that after an error in the path is noted, the erroneous (old) *path chunks* are used only for the layers that have not changed. The *menudepth* chunk furthermore holds knowledge about which layer is the final layer for each target. The *errorpath* slot in the goal buffer holds the knowledge about an occurred change in the path leading to any target; it is not reset between different targets. The *finaltarget* and the *subtarget* slot hold the target and the subtarget as a string. These two slots are used to determine if the target has been found and also required for a superficial visual search utilized on the last page and for researching a subtarget.

Mechanisms en-detail

Initiation In the beginning of each run, the production *start* requests a *meaning chunk*. The building of a *path chunk* is initiated. The production *meaning-in-goal* then copies the retrieved meaning of the target into the slot *target-meaning* of the chunk in the goal buffer and into the *target-im* slot of the chunk in the imaginal buffer. Then, a retrieval request for a *path chunk* leading to the target is initiated with variations[2] of the production *look-for-path.*

Association approach Without prior experience with the specific target, a *path chunk* leading to the target is not retrievable and the production *change-strategy* fires, followed by *find-word* and *reading-word*. Visual attention is directed to the top of the page (to the highest location bellow the current visual attended location) and this location is visually processed. Variations[3] of the production *process-word* then visually encode the current word. For all layers, except the last layer, a request for a *meaning chunk* holding the meaning of the current word is initiated. If such a chunk is found the production *searching association* then initiates the search for an *association chunk* containing the current word and the target. If such an *association chunk* cannot be found, the production *no-association-found* clears the visual buffer and the search continues with the production *find-word*. If an *association chunk* is retrieved, variations[4] of the

[1] The model can be downloaded at https://depositonce.tu-berlin.de/handle/11303/5548.

[2] The variations of *look-for-path* consider two aspects: First, whether or not there was an error in the path and second, the differences in menu depths. This ensures that for a detected change in menu depth the old (misleading) path is not retrieved.

[3] There are variations of *process-word* for the last layer and for the other layers except the last. The variations consider the value of errorpath slot in the goal buffer and the value of the count in the imaginal buffer.

[3] Variations of *association- found* depend on the value of the count slot of the imaginal buffer.

production *association-found* update the *path chunk* in the imaginal buffer. If, for example the first subtarget in path is found, then the value of the *count slot* in the *imaginal buffer* is changed from *one* to *two*. Furthermore, the *first slot* of the *path chunk* is filled with the current *subtarget*, which is also copied into the *subtarget slot* of the goal buffer. A cursor move is initiated and the productions *prepare-click* and *click* initiate the motor movements to press the button. The productions *waiting-click* or *waiting-last-click* (for the final click, in order to initiate the backing procedure) let the model wait until the *manual buffer* is free. After the manual buffer is free a variation of the production *look-for-path* fires again. For the last layer, the elaborate procedure of reading from top to bottom and searching for *association chunks* is replaced by a superficial visual search procedure. If the word in the visual buffer and the word in the slot *finaltarget* of the chunk in the goal buffer are different, a variation of the production *process-word-last-page-wrong* will fire. Via the production *find-word* the next word is searched. If they are the same a variation[5] of *process-word-last-page-correct* will copy the last slot of the path into the *path chunk* in the imaginal buffer, raise the *count slot* and change the value of the *subarget slot* in the chunk in the goal buffer. A cursor move is initiated and the productions *prepare-click* and *click* press the button with the target.

Path Navigation If a *path chunk* leading to the target is retrieved a variation of the production *found-that-path*, depending on the value of the *count slot* in the imaginal buffer, fires. This production copies the value of the relevant slot (e.g. the *subtarget*) from the *path chunk* in the retrieval buffer into the *path chunk* in the imaginal buffer and changes the *count.* The production *find-location* requests for a *meaning chunk* of the relevant subtarget. The production *found-location* indicates that a location was retrieved and the visual attention is moved to the retrieved location. Then the visual buffer and the *subtarget* slot of the chunk in the goal buffer are compared. If they are the same, then the retrieved location is correct and the production *checking-match* fires, followed by *click-location* and *waiting-click*.

Modified Interfaces In the following subsection an overview on mechanism dealing with the modified interfaces is given, for a detailed description see box 1.

The retrieved location is visual inspected and the subtarget is not found at the retrieved location, either because there is a different word, or no word at the retrieved location. This is indicated by the productions *checking-no-match* or *checking-empty*. A visual search for the subtarget is then initiated with the production r*ead-top-to-bottum-again-2*. Visual attention is directed to the top of the page and the production *scan-1* encodes the visual-location. If the word in visual buffer is the subtarget then *scan-correct* fires otherwise *scan-incorrect* moves the visual attention to the next highest word. If the subtarget is found it is selected via *prepare-click* and *click*. If the visual search via *scan-1* and *scan-incorrect* does not lead to the subtarget and the bottom

1. *Depth changes from three to four layers; the path is different from the third layer on.* An update adds a new layer to all targets of the shopping app, e.g. the old path for *alcohol free beer* is *1.stores 2.drinks 3.alcohol free beer* the new path is *1.stores 2.drinks 3.beer 4.alcohol free beer.* In the third run (after an update), *alcohol free beer is searched* on the third layer. But the retrieved location does not contain *alcohol free beer*. The third layer is rescanned from top to bottom in search for *alcohol free beer,* without success. The value of *errorpath* slot in the goal buffer is changed to true. The third layer is then read from top to bottom and for each word an attempt to retrieve *an association chunk* is made. The model visually encodes *beer* and an *association chunk* is retrieved The third slot of the *path chunk* in the imaginal buffer is assigned the value *beer* and beer is selected. On the fourth layer, the attempt to retrieve a *path chunk* that leads to *alcohol free beer* with the slot four having a value is made. Such a chunk cannot be retrieved. The superficial search approach for the last page will be used and directly search for *alcohol free beer is initiated*. For the next product, navigation is realized with the old *path chunk* for layer one and two. For the third and fourth layer the attempt to retrieve a *path chunk* with four layers will fail. The association approach or for the last page the superficial search will be used. In the fourth run the correct *path chunks* for all layers can be retrieved.

2. *Depth changes from three to four layers; the path is different from the second layer on, mixed list*. The path for lawn in the real-estate application changes depending on preselection. If house is preselected, the path is 1.search, 2.garden, 3.lawn and changes to 1.search 2.more 3.garden 4.lawn if apartment is preselected. On the second layer the subtarget garden is not found at the retrieved location. A rescanning of the page is unsuccessful. As in 1) an errorpath in the goal buffer is noted and the approach is changed to reading from top to bottom and searching for association chunks. A new *path chunk* is then built. For the third and fourth layer the attempt will be made to retrieve a *path chunk* with four layers leading to lawn and then utilize the reading and association approach or the scan approach for the last page. For the next target, the value of the menudepth slot in the goal buffer indicates constant depth; therefore it is not influenced by the error in the path. In the fourth run, a *path chunk* for lawn can be found.

3. *Depth changes from four to three layers; the path is different from the third layer on.* An update removes a layer for all targets from the shopping list app. The path for *alcohol free beer* changes from *1.stores, 2 drinks, 3.beer 4.alcohol free beer* to *1.stores, 2.drinks 3.alcohol free beer.* On the third layer the subtarget *beer* is not found at the retrieved location. The third layer is rescanned from top to bottom in search for *beer* and the target *alcohol free beer is found*. The errorpath slot in the goal buffer is changed to true. The third slot of the *path chunk* in the imaginal buffer gets the value *alcohol free beer* assigned and the target is selected,. For the next product navigation is realized with the old *path chunk* for layer one and two. For the third layer the attempt to retrieve a *path chunk* with three layers will fail. The superficial search will be used for the last page and the building of *path chunks will be completed*. In the fourth run the correct path chunks can be retrieved.

4. *Depth changes from four to three layers; the path is different from the second layer on (special case).* First apartment is preselected. Thus, the path for lawn in the real-estate application is *1.search 2.other 3.garden 4.lawn*. Then house is preselected and the path is *1.search, 2.garden, 3.lawn*.On the second layer, the subtarget other is not at the retrieved location. The page is rescanned and *other* is found and selected. On the third page garden is searched for unsuccessfully. The value of the errorpath slot in the goal buffer is then set to true and the model goes back to the second page. A new *path chunk* is built from the second page on via association chunks and reading top to bottom. For the next target, the value of the menudepth slot in the goal buffer indicates constant depth; therefore, it is not influenced by the error in the path. In the fourth run, a *path chunk* for lawn can be found.

Box 2 : A description on how different changes in the interfaces are processed by the model.

of the page is reached, an *errorpath* will be noted in the goal buffer. An *errorpath* in the goal buffer will also be noted, if the target is found while scanning for a subtarget (via *scanning-path-is-wrong*). After the *errorpath* slot in the goal buffer is set to true and the subtarget has not been found, the production *error-in-path-2* resumes the visual attention to the top of the page and via *find-word, reading-word* and *association-found* a *path chunk* is build in the imaginal buffer. The changed value of the *errrorpath* slot will stay in the chunk in the goal buffer for all further model runs. Therefore, if the *errorpath* value in the goal buffer is set to *true*, a wrong *path chunk will not be retrieved*. Either no *path chunk* is retrieved and navigation is implemented via *association chunks*, or *path chunk* are utilized only as long as they are correct. For example, if the *path chunk* is helpful for layer one and two – navigation with this chunk for these layers is implemented, but different *path chunk* are sought for on the third layer.

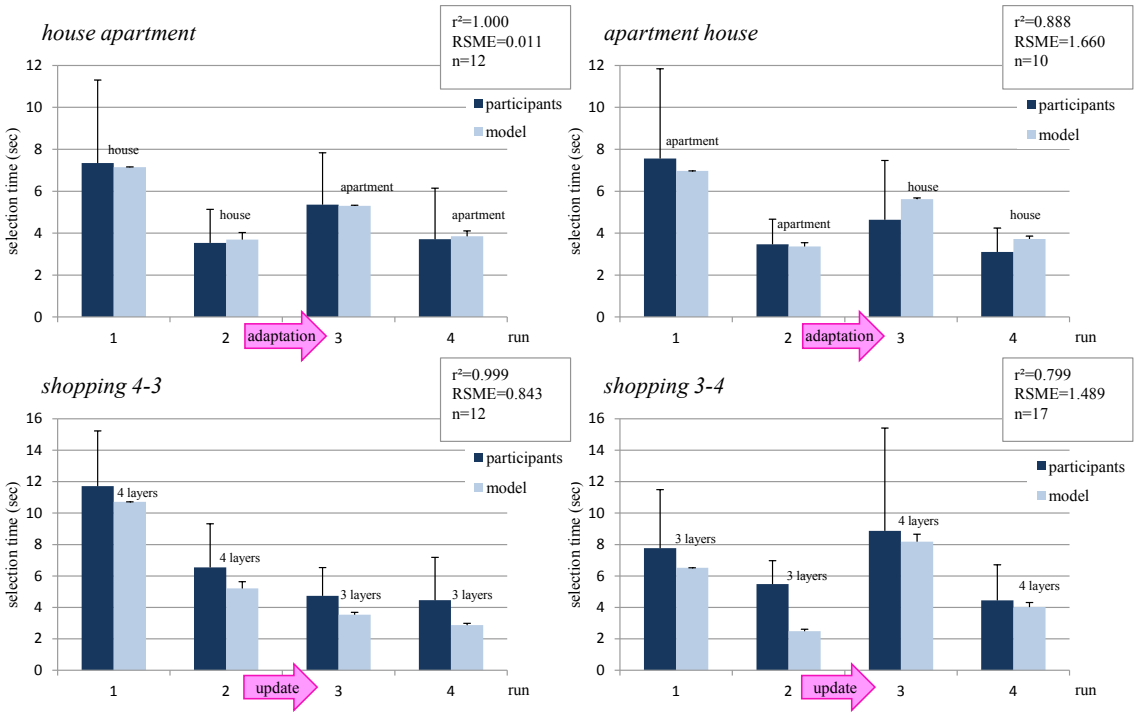

Figure 2: Mean target selection time for the four different studies for the modeled and empirical data.

Results

Data processing

The empirical data comprises of four studies, with a varying number of student participants ($10 < n < 17$). The model was run 10 times per study. Target selection time (for both model and empirical data) is defined as the time between the selection of the target and the selection of the first subtarget. Extreme values were excluded from analysis. Mean target selection time for each target was calculated and averaged over the runs (six to nine targets per run). Mean target selection time for each run was calculated and averaged over the participants. Model and empirical data were compared via goodness of fit indices and qualitative analysis of graphs.

Model parameter

The latency factor (lf) and the retrieval threshold (rt) parameter were fit to match the data of the study house apartment. They are set to the values: lf = 0.1 and rt = -1.5.

Comparison of modeled and empirical data

The model provides a good to very good fit to the data, see figure 2 for the comparison of the mean target selection times of the empirical and modeled data. The main trends of the four studies are mapped in the modeled data, as are most of the absolute values.

1. The *house apartment* study reveals a decrease of mean target selection time followed by an increase and again a decrease. This is exactly represented in the modeled data; $r^2 = 1.0$. The absolute values of the modeled and empirical data are also very close; RSME = 0.011.

2. The *apartment house* study reveals the same pattern (decrease, increase and decrease); $r^2 = 0.888$. The mean target selection time predicted by the model is slightly lower in the first two runs and a bit higher in the last two runs; RSME = 1.660.

3. The empirical and modeled mean target selection time of the *shopping 4-3* study indicate a decrease from run 1 to run 4, with the decrease leveling out towards the last run. The magnitude of the decrease is very similar for both

datasets; r^2= 0.999. In all runs the mean target selection times predicted by the model are lower than those of the participants; RSME = 0.843.

4. For the *shopping 3-4* study, both datasets show a pattern of a decrease, followed by an increase and a final decrease. The modeled increase in mean target selection time between run 2 and 3 is greater than the increase found in the empirical data. The magnitude of the other variations are similar for the empirical and modeled data; r^2 = 0.799. The mean target selection time predicted by the model is lower than that of the participants, especially in the third run; RSME = 1.489.

Summary

In summary, all trends found in the empirical data are predicted by the model, with a high r^2 = [0.799; 1.0] for all four studies. Decreases and increases in target search time are predicted by the model. Therefore, the model provides information on learning and relearning effects for different applications. Moreover, the absolute values are met by the model for the majority of data points, with RSME = [0.011; 1.660], values which are lower than the average STD of the empirical data. Hence, the model can appropriate depict mean user behavior at different time points during a repeated target selection task.

Discussion

The modeling approach provides accurate predictions of the data from four studies with two different apps. Efficiency, learnabilty and the impact of updates or of adaptivty are predicted by the model.

Only a few steps are necessary to alter the model for a different app; world knowledge needs to be provided in form of *association chunks* and reading ability as *meaning chunks*. Furthermore, a compilation of the *targetlist*, containing the targets and of the *menudepthlist*, containing the menu depth of the targets before and after an update, is required.

The model can easily be extended to other list-style hierarchical apps with different menu depths than 3 and 4 layers and to apps with different switches in the number of layers. To do so, variations of the productions *process-word*, *process-word-last-page* and *look-for-path* are required.

Plausibility of Modeling Decision

A general modeling approach facilitates the adaptability of the model. To achieve this, partially simplified assumptions have been made. This applies especially to the *menudepth* slot, which offers a technical solution for the handling of varying menudepth. This slot contains meta-knowledge of the model, in other words knowledge about what kind of update occurred. The *menudepth* slot holds information if the update relates to the entire menu structure or if it relates to individual paths. Furthermore, the *menudepth* slot is useful to identify the last page. It is plausible to assume that users know if the current page is the last page. However, users are likely to obtain this knowledge with the help of visual features. Since these in turn significantly differ between apps, it is useful for a general approach to detect the last page in a simplified manner.

The reading direction of the model is always directed from top to bottom and each item is processed, this is a further simplification. It is possible to model visual processing of menus more precisely (Bailly, Oulasvirta, Brumby and Howes, 2014). Since, the goal of our work was to predict average mean search time for items, our chosen simplification of visual processing was sufficient.

Another simplified assumption is, that the imaginal holds the count of the current page. This is done only for practical reasons to make model steering easy and adjustable to different updates. Furthermore, it does not affect overall item search time.

Limitations and Further Steps

To use the model approach for usability testing, a remaining obstacle is the need of prototyping the interface. In order for the ACT-R model to interact with the application, a copy of the app in Lisp is required. To avoid this effortful step, we are working on a tool called ACT-Droid (Doerr, Russwinkel, & Prezenski, 2016).With ACT-Droid android apps can be connected directly with ACT-R models. This tool is will reduce the practical hurdle of testing usability of apps with cognitive models further. Further steps to improve the model are to replace the mouse movements in the model with touch movements, as provided by ACT-Touch (Greene & Tamborello, 2013). Besides, an in-depth validation of the model with eye tracking data is planned. A new study should also investigate if the model can predict average click times for each menu layer as well. The empirical data of the current studies, do not allow such predictions, due to the study design. Nevertheless, on an individual level the model and empirical data show, that after an update (shopping app) and after an unexpected adaption (real-estate app) search time increases.

We are also intending to look into how far mechanisms of the current model can be used to develop a model for hierarchical apps with icons instead of text.

Moreover, our general modeling approach for hierarchical list-style apps is not only useful for such apps. It is well suited to predict the average user search time for all kinds of list-based interfaces that spread semantically related subtargets across multiple layers.

References

Bailly, G., Oulasvirta, A., Brumby, D. P., & Howes, A. (2014). Model of visual search and selection time in linear menus. *Proc. of the 32nd Annual ACM Conference on Human Factors in Computing Systems - CHI '14, (pp.* 3865–3874**).** Toronto, Canada: ACM.

Byrne, M. D. (2001). ACT-R/PM and menu selection: applying a cognitive architecture to HCI. *Int. J. Human-Computer Studies*, 55, 41-84.

Das, A., & Stuerzlinger, W. (2007). A cognitive simulation model for novice text entry on cell phone keypads. In *Proc. of the 14th European Conference on Cognitive Ergonomics*, (pp.141-147). London, UK: ACM.

Doerr, L.-M., Russwinkel, N., & Prezenski, S. (in submission). ACT-Droid: ACT-R Interacting with Android Applications. In D. Reitter & F. Ritter (Eds.), In *Proc. of the 14th Int. Conference on Cognitive Modeling.* Pennsylvania, USA.

Greene, K. K. & Tamborello, F. P. (2013). Initial ACT-R extensions for user modeling in themobile touchscreen domain In *Proc. of the 12th Int. Conference on Cognitive Modeling.*(pp.348-353)Ottawa, Canada.

John, B. E., Prevas, K., Salvucci, D. D., & Koedinger, K. (2004). Predictive human performance modeling made easy. In *Proc. of CHI'04*, (pp. 455–462). New York, USA: ACM.

Prezenski, S., Lindner, S., Moegele, H., & Russwinkel, N. (in preparation). Archetyping the User.

Prezenski, S., & Russwinkel, N. (2014). Combining Cognitive ACT-R Models with Usability Testing Reveals Users Mental Model while Shopping with a Smartphone Application. *Int. J. on Advances in Intelligent Systems*, 7(3), 700–715.

St.Amant, R., Horton, T.E., & Ritter, F.E. (2007). Model-based evaluation of expert cell phone menu interaction. *Transactions on Computer-Human Interaction*, 14(1), 1-14.

4 Paper No. 3

Prezenski, S., Brechmann, A., Wolff, S., and Russwinkel, N. (2017). A cognitive modeling approach to strategy formation in dynamic decision making. *Frontiers in psychology*, 8:1–18.

frontiers in Psychology

ORIGINAL RESEARCH
published: 04 August 2017
doi: 10.3389/fpsyg.2017.01335

A Cognitive Modeling Approach to Strategy Formation in Dynamic Decision Making

Sabine Prezenski[1]*, *André Brechmann*[2]*, *Susann Wolff*[2] and *Nele Russwinkel*[1]

[1] Cognitive Modeling in Dynamic Human-Machine Systems, Department of Psychology and Ergonomics, Technical University Berlin, Berlin, Germany, [2] Special Lab Non-Invasive Brain Imaging, Leibniz Institute for Neurobiology, Magdeburg, Germany

Decision-making is a high-level cognitive process based on cognitive processes like perception, attention, and memory. Real-life situations require series of decisions to be made, with each decision depending on previous feedback from a potentially changing environment. To gain a better understanding of the underlying processes of dynamic decision-making, we applied the method of cognitive modeling on a complex rule-based category learning task. Here, participants first needed to identify the conjunction of two rules that defined a target category and later adapt to a reversal of feedback contingencies. We developed an ACT-R model for the core aspects of this dynamic decision-making task. An important aim of our model was that it provides a general account of how such tasks are solved and, with minor changes, is applicable to other stimulus materials. The model was implemented as a mixture of an exemplar-based and a rule-based approach which incorporates perceptual-motor and metacognitive aspects as well. The model solves the categorization task by first trying out one-feature strategies and then, as a result of repeated negative feedback, switching to two-feature strategies. Overall, this model solves the task in a similar way as participants do, including generally successful initial learning as well as reversal learning after the change of feedback contingencies. Moreover, the fact that not all participants were successful in the two learning phases is also reflected in the modeling data. However, we found a larger variance and a lower overall performance of the modeling data as compared to the human data which may relate to perceptual preferences or additional knowledge and rules applied by the participants. In a next step, these aspects could be implemented in the model for a better overall fit. In view of the large interindividual differences in decision performance between participants, additional information about the underlying cognitive processes from behavioral, psychobiological and neurophysiological data may help to optimize future applications of this model such that it can be transferred to other domains of comparable dynamic decision tasks.

Keywords: dynamic decision making, category learning, ACT-R, strategy formation, reversal learning, cognitive modeling, auditory cognition

OPEN ACCESS

Edited by:
Wolfgang Schoppek, University of Bayreuth, Germany

Reviewed by:
Robert Lawrence West, Carleton University, Canada
David Reitter, Pennsylvania State University, United States

***Correspondence:**
Sabine Prezenski
sabine.prezenski@tu-berlin.de
André Brechmann
brechmann@lin-magdeburg.de

Specialty section:
This article was submitted to Cognitive Science, a section of the journal Frontiers in Psychology

Received: *15 March 2017*
Accepted: *20 July 2017*
Published: *04 August 2017*

Citation:
Prezenski S, Brechmann A, Wolff S and Russwinkel N (2017) A Cognitive Modeling Approach to Strategy Formation in Dynamic Decision Making. Front. Psychol. 8:1335. doi: 10.3389/fpsyg.2017.01335

INTRODUCTION

Backcountry skiers (and snowboarders) strive for the unique thrill of skiing or snowboarding down powder covered mountains, drawing the first line into freshly fallen snow. Before deciding to go down a particular mountain slope, they check the snowpack, the temperature and wind conditions to avoid setting off an avalanche. Often not a single snow characteristic is crucial but conjunctions of them can change the conditions of safe skiing. The decision to continue on a slope is re-evaluated often, depending on the feedback from the snow (e.g., collapsing snow, snow-brakes vs. nice powder snow) and previous experience.

The described scenario gives a good example of complex cognition. Complex cognition (Knauff and Wolf, 2010) investigates how different mental processes influence action planning, problem solving and decision-making. The term "mental processes in complex cognition" includes not only cognitive but also motivational aspects. Naturalistic decision-making research investigates how decisions are made "in the wild." Real-life decisions made by people with some kind of expertise are investigated in the context of limited time, conflicting goals, dynamically changing conditions, and information sources of varying reliability.

Such complex situations involve further aspects that cannot all be covered in combination when studying complex cognition. Nevertheless, researchers should aim at describing, understanding and predicting human behavior in its complexity.

A model situated within cognitive architectures can simulate multiple parallel processes, thereby capturing multifaceted psychological phenomena and making predictions, sometimes even for complex tasks. Nevertheless, developing such models requires a stepwise procedure to distinguish different influencing factors. For our skiing example, first a model of the core decision-making process (e.g., based on category learning from snow characteristics and feedback) of a backcountry skier needs to be developed and tested. Afterwards this approach can be extended with modeling approaches of other decision influencing processes (e.g., motivation) to predict decision-making in the wild.

To come closer to the overall goal of understanding cognition as a whole, studying dynamic decision-making with cognitive architectures constitutes a step in the right direction. In dynamic decision-making, decisions are not seen as fixed but can be modified by incoming information. So, not only singular aspects of decision-making are considered, such as attentional influence, but also environmental factors that give feedback about an action or lead to major changes requiring an adaptation to new conditions.

In real-life decisions, however, our future choices and our processing of decision outcomes are influenced by feedback from the environment. This is the interactive view on decision-making, called dynamic decision-making (Gonzalez, 2017), of which the scenario presented above is an example. According to Edwards (1962), three aspects define dynamic decision-making. First, a series of actions are taken over time to achieve a certain goal. Second, the actions depend on each other. Thus, decisions are influenced by earlier actions. Third, and most difficult to investigate, changes in the environment occur as a result of these actions but also spontaneously (Edwards, 1962). According to Gonzalez (2017), dynamic decision-making is a process where decisions are motivated by goals and external events. They are dependent on previous decisions and outcomes. Thus, decisions are made based on experience and are dependent on feedback. Most of the time, these kinds of decisions are made under time constraints. Therefore, long mental elaborations are not possible. To sum up, dynamic decision-making research investigates a series of decisions which are dependent on previous decisions and are made under time constraints in a changing environment.

Another view on dynamic decision-making as a continuous cycle of mental model updating is introduced by Li and Maani (2011). They describe this process using the CER Cycle. CER stands for Conceptualization–Experimentation–Reflection. Conceptualization is obtaining an understanding of the situation and mentally simulating the outcome of potential decisions and related actions. Thus, the decision maker compares the given situation with related information in his or her mental model and integrates new information obtained from the environment to develop a set of decisions. During experimentation, the decisions and interventions devised from the decision-maker's mental model are tested in the dynamics of the real world. In the reflection phase, the outcome of the experimentation phase is reflected on, e.g., feedback is processed. If the expected outcome is achieved (e.g., positive feedback), the initial decisions are sustained. If, however, the outcome is unexpected (e.g., negative feedback) or if obtained results differ from the expected outcome, the decision maker updates his or her mental model. To do this, he or she decides for alternative actions such as searching for new sources of information for making better decisions.

These kinds of decision-making procedures have been suggested to share many processes with the procedure of category formation (Seger and Peterson, 2013). Categorization is a mental operation that groups objects based on their similar features. When new categories are formed from a given set of items without explicit instruction, the features distinguishing the different items must first be extracted. Then hypotheses about the relevant features must be formed and tested by making serial decisions.

Category learning experiments in cognitive science often require participants to establish explicit rules that identify the members of a target category. The serial categorization decisions are reinforced by feedback indicating whether a decision was correct or not. The success in such rule-based category learning experiments critically depends on working memory and executive attention (Ashby and Maddox, 2011). The fact that real world decisions critically depend on success and failure in previous trials qualifies category learning as a model for dynamic decision-making.

There are numerous advanced computational models of categorization which explain behavioral performance of subjects in various categorization tasks (e.g., Nosofsky, 1984; Anderson, 1991; Ashby, 1992; Kruschke, 1992; Nosofsky et al., 1994; Erickson and Kruschke, 1998; Love et al., 2004; Sanborn et al., 2010). These competing models differ in their theoretical assumptions (Lewandowsky et al., 2012) and there is currently no

consensus on how different models can be compared and tested against each other (Wills and Pothos, 2012).

Another requirement for dynamic decision-making is the occurrence of changes in the environment. A well-known categorization task using such changes is implemented in the Wisconsin Card Sorting Test (WCST; Berg, 1948). In this test, participants must first select a one-feature rule (color, shape, number of symbols) and are then required to switch to a different one-feature rule. This task tests for the ability to display behavioral flexibility. Another experimental approach to test for behavioral flexibility in humans and animals is reversal learning (e.g., Clark et al., 2004; Jarvers et al., 2016). Here, subjects need to adapt their choice behavior according to reversed reinforcement contingencies.

Thus, category learning experiments with changing rules can serve as suitable paradigms to study dynamic decision-making in the laboratory, albeit with limited complexity as compared to real world scenarios.

The majority of rule-based category learning experiments are simple and only use one relevant stimulus feature specification (e.g., a certain color of the item) as categorization basis. In principle, however, such a restriction is not required and rule-based category learning experiments can become more complex by using conjunction rules. These can still be easily described verbally (e.g., respond A if the stimulus is small on dimension x and small on dimension y). It has been shown that conjunction rules can be learned (e.g., Salatas and Bourne, 1974) but are much less salient and are not routinely applied (Ashby et al., 1998).

In the following, the main points mentioned above are integrated in our backcountry skiing example: Since feedback from the environment plays a central role in building a correct mental model, feedback in the form of great powder snow indicates that the current strategy is correct. By contrast, negative feedback, for example breaking snow, indicates that one should change the strategy, perhaps search for different feature specifications or even for a different combination of features that might promise a better outcome for skiing. Furthermore, sudden changes of environmental conditions can result in a change of which feature combinations are indicative of a positive outcome. In our example, a change could be a different hill side with more exposure to the sun or a rise in temperature, requiring that other feature combinations should be taken as an indication for a safe descent. There are a lot of possibilities which features and feature combinations could indicate safe or unsafe conditions, making such a task complex.

Thus, to study dynamic decision-making in a category learning experiment requires a task with the above-mentioned characteristics (successive decisions with feedback, multiple feature stimuli, and switching of category assignments). To determine how humans learn feature affiliation in a dynamic environment and to investigate how strategies with rising complexity emerge, a modeling approach addressing these aspects first needs to be developed. If this model is useful and plausible it should match average behavioral data. This is an important milestone toward a more precise model which in turn should predict more detailed empirical data (e.g., individual behavioral or neural data). If this step is achieved, then models can be used as decision aiding systems at an individual level.

In this paper, we use the behavioral data of an experiment, described in the following, to develop an initial cognitive model as described above. In the experiment, a large variety of multi-feature auditory stimuli were presented to participants in multiple trials. The participants were then required to learn by trial and error which combinations of feature specifications predict a positive or negative outcome. Since perceptual learning of stimulus features is not the focus of our research, we used salient and easy-to-recognize auditory features. To meet all of the above-mentioned criteria for dynamic decision-making, we further introduced a spontaneous change in the environment such that previous decisions on feature combinations suddenly needed to be re-evaluated to obtain positive feedback.

In particular, we would like to demonstrate how different aspects that influence dynamic decision-making can be addressed through a combination of existing and validated cognitive mechanisms within an architecture. These are: learning to distinguish positive and negative feature combinations depending on feedback; successive testing of simple one-feature rules first and switching to more complex two-feature rules later, and using metacognition to re-evaluate feature combinations following environment changes. Other modeling approaches are also able to replicate such data, what distinguishes our approach is that it has a theory grounded interpretation of plausible cognitive mechanisms.

Why Use Cognitive Modeling?

The method of cognitive modeling forces precision of vague theories. For scientific theories to be precise, these verbal theories should be formally modeled (Dimov et al., 2013). Thus, theories should be constrained by describable processes and scientifically established mechanisms. As Simon and Newell (1971) claim, "the programmability of the theories is a guarantee of their operationality and iron-clad insurance against admitting magical entities in the head" (p. 148).

Cognitive models can make predictions of how multiple aspects or variables interact and produce behavior observed in empirical studies. In real-life situations, multiple influences produce behavior. Cognitive models are helpful to understand which interrelated cognitive processes lead to the observed behavioral outcome. Cognitive models can perform the same task as human participants by simulating multiple ongoing cognitive processes. Thereby, models can provide insight into tasks that are too complex to be analyzed by controlled experiments. Nevertheless, studying such a task with participants is mandatory to compare the outcomes of models and participants. However, understanding the process leading to an outcome is more important than perfectly fitting a model to a given set of experimental results. Our goal in this regard is to understand the processes underlying human decision-making, not least to aid humans in becoming better at decision-making (Wolff and Brechmann, 2015).

Predictions made by cognitive models cannot only be compared to average outcome data (such as reaction times, or percentage of correct decisions) but also to process data. Process

data represent patterns of information search, e.g., neural data. In this regard, cognitive models can be informed by EEG and fMRI data to achieve an empirical validation of such processes (Forstmann et al., 2011; Borst and Anderson, 2015).

The development of neurobiologically plausible models is specifically the focus of reinforcement learning (e.g., Sutton and Barto, 1998). The aim of such computational models is to better understand the mechanisms involved on the neural network level as studied using invasive electrophysiological measures in different brain regions in animals (e.g., sensory and motor cortex, basal ganglia, and prefrontal cortex). Such neural network models have recently been applied to learning tasks requiring flexible behavior (e.g., contingency reversal tasks). The reader is referred to a recent paper by Jarvers et al. (2016) that gives an overview of the literature on reversal learning and describes a recurrent neural network model for an auditory category learning task such as the one applied in the current paper. This probabilistic learning model resulted in a good fit to the empirical learning behavior, but does not interpret the cognitive processes that lead to this behavior. It postulates an unspecified metacognitive mechanism that controls the selection of the appropriate strategy. This is where the strength of our approach comes into play; It is specific about the metacognitive mechanisms that drive behavior in such tasks. An example would be processes which assure that after a number of negative results, a change in strategy will be initiated.

To summarize, cognitive modeling is a falsifiable methodology for the study of cognition. In scientific practice, this implies that precise hypotheses are implemented in executable cognitive models. The output of these models (process as well as product) is then compared to empirical data. Fit-Indices such as r^2 and RSME as well as qualitative trends provide information on the predictive power of the cognitive models.

More specific, the central goals of cognitive modeling are to (a) describe (b) predict, and (c) prescribe human behavior (Marewski and Link, 2014). A model that *describes* behavior can replicate the behavior of human participants. If the model, however, reproduces the exact behavior found in the human data, this is an indication of overfitting. In this case, the model has parameters that also fit the noise found in the empirical data. To address such issues of over specified models, it is important to test the model on a new data set and thereby evaluate how well it can *predict* novel data. *Prescribe* means that the model should be a generalizable model so that it can predict behavior in different situations. Moreover, robust models are preferable this implies that the output of the model is not easily influenced by specific parameter settings.

The term cognitive model includes all kinds of models of cognition—from very specific, isolated cognitive aspects only applicable in specific situations to more comprehensive and generalizable ones. The latter candidates are cognitive architectures that consider cognition as a whole. They aim at explaining not only human behavior but also the underlying structures and mechanisms. Cognitive models written on the basis of cognitive architectures therefore generally do not focus on singular cognitive processes, such as some specific learning process. By contrast, interaction of different cognitive processes and the context of cognitive processes are modeled together. Modeling the relations between different subsystems is especially relevant for applied research questions. The structures and mechanism for this are provided by the cognitive architecture and should be psychologically and neurally plausible (Thomson et al., 2015).

The most commonly used cognitive architectures, such as ACT-R, predict processes at a fine-grain level in the range of 50 ms. These processes can be implemented computationally. However, they are embedded in cognitive theories—this is what distinguishes cognitive models built with cognitive architectures from mathematical models such as neural networks. The latter models formally explain behavior in terms of computational processes. Thus, their explanation of behavior can be seen in terms of computational processes but do not aim at cognitive interpretations (Bowers and Davis, 2012).

The Cognitive Architecture ACT-R

The cognitive architecture ACT-R (Adaptive Control of Thought—Rational) has been used to successfully model different dynamic decision-making tasks and is a very useful architecture for modeling learning (Anderson, 2007; Gonzalez, 2017). In the following, a technical overview of the main structures and mechanisms that govern cognitive models in ACT-R is given. We will focus only on those aspects that are important to understand our modeling approach. For a more detailed insight into ACT-R, we recommend exploring the ACT-R website[1].

ACT-R's main goal is to model cognition as a whole using different modules that interact with each other to simulate cognitive processes. These modules communicate via interfaces called buffers. ACT-R is a hybrid architecture, thus symbolic and subsymbolic mechanisms are implemented in the modules of ACT-R.

Our model uses the motor, the declarative, the imaginal, the goal, the aural[2], and the procedural module. The motor module represents the motor output of ACT-R. The declarative module is the long-term memory of ACT-R in which all information units (chunks) are stored and retrieved. The imaginal module is the working memory of ACT-R in which the current problem state (an intermediate representation important for performing a task) is held and modified. Thus, the imaginal module plays an important role for learning. The goal module holds the control states. These are the subgoals that have to be achieved for the major goal. The aural module is the perceptual module for hearing. The procedural module plays a central role in ACT-R. It is the interface of the other processing units, since it selects production rules (see below) based on the current state of the modules.

Writing a model requires the modeler to specify the symbolic parts of ACT-R. These are (a) the production rules, and (b) the chunks. Chunks are the smallest units of information. All information in ACT-R is stored in chunks. Production rules

[1] http://act-r.psy.cmu.edu/

[2] Please note that the aural module has two buffers. A tone is first encoded in the aural-location buffer and its content can then be accessed using the aural buffer.

(e.g., productions) consist of a condition and an action part. Productions are selected sequentially, and only one production can be selected at a given time. A production can only be selected if the condition part of the production matches the state of the modules. Then, the action part modifies the chunks in the modules. If more than one production matches the state of the modules then a subsymbolic production selection process chooses which of the matching productions is selected.

A further subsymbolic process in ACT-R is the activation of a chunk. It determines if a chunk can be retrieved from memory and how long this retrieval takes. The past usefulness of a chunk (base-level activation), the chunk's relevance in the current context (associative activation) and a noise parameter sum up to the chunk's activation value. Modifying the subsymbolic mechanisms of ACT-R is also part of the modeling procedure. This can be done using specific parameters—however, most parameters have default values derived from previous studies (Wong et al., 2010) which should be used.

How Can Decision-Making and Category Learning Be Modeled in ACT-R?

Many different styles for writing models in ACT-R exist (Taatgen et al., 2006). The following modeling approaches have been used for decision-making: (a) strategy or rule-based, (b) exemplar or instance-based, and (c) approaches that mix strategies and exemplars. These approaches will be compared to motivate our chosen modeling approach.

In *strategy or rule-based models,* different problem solving strategies are implemented with different production rules and successful strategies are rewarded. Rule-based theories in category learning postulate that the categorizer must identify the category of an object by testing it against different rules. So, to find a solution for a problem, strategies in the form of rules are used.

Exemplar or instance-based models rely on previous experience stored in declarative memory to solve decision-making problems. The content and structure of the exemplars depend on individual framing. It is not a complete representation of the event, but represents the feature specifications the problem solver is focused on, together with experienced feedback. Exemplar theories of category learning postulate that category instances are remembered. To decide if an instance belongs to a category, a new instance is compared to an existing instance. Instance-Based Learning (IBL) builds upon instances in the context of dynamic decision processes and involves learning mechanisms such as recognition-based retrieval. The retrieval of instances depends upon the similarity between the current situation and instances stored in memory. In IBL situations, outcome observations are stored in chunks and retrieved from memory to make decisions. The subsymbolic activation of the retrieved instances determines which instances are likely to be retrieved in a given situation. Instance Based Learning requires some amount of previous learning of relevant instances. Then, decision makers are able to retrieve and generalize from these instances (Gonzalez et al., 2003).

Mixed approach models use both rules and instances to solve decision-making problems.

Several authors implemented the described approaches in category learning and decision-making environments. In a strategy-based ACT-R model, Orendain and Wood (2012) implemented different strategies for complex problem solving in a microworld[3] game called "Firechief." Their model mirrored the behavior of participants in the game. Moreover, different training conditions and resulting behavior of the participants could be modeled. The model performed more or less flexibly, just as the participants, according to different training conditions. This demonstrates that success in strategy learning depends on the succession of stimuli in training conditions. Peebles and Banks (2010) used a strategy-based model of the dynamic-stocks and flows task (DSF). In this task, water level must be held constant but the inflow and outflow of the water changes at varying rates. An ACT-R model of strategies for accomplishing this task was implemented in form of production rules. The model replicated the given data accurately, but was less successful in predicting new data. The authors proposed that by simply extending the model so it contains more strategies and hypotheses, it would be able to predict such new data as well. Thus, specifying adequate rules is crucial for rule-based models.

Gonzalez et al. (2009) compared the performance of two ACT-R models, an instance-based model and a strategy-based model, in a RADAR task. In this task, participants and the model had to visually discriminate moving targets (aircrafts) among moving distractors and then eliminate the targets. Both models achieved about the same overall fit to the participants' data, but IBL performed better in a transfer task.

Lebiere et al. (1998) tested two exemplar models that captured learning during a complex problem-solving task, called the sugar factory (Berry and Broadbent, 1988). The sugar factory task investigates how subjects learn to operate complex systems with an underlying unknown dynamic behavior. The task requires subjects to produce a specific amount of sugar products. Thus, in each trial the workforce needs to be adjusted accordingly. The two exemplar models produced adequate learning behavior similar to that of the subjects. In a subsequent study, Fum and Stocco (2003) investigated how well these original models could predict participants' behavior in case of a much lower target amount of the sugar product than in the original experiment. Furthermore, they investigated if the models could reproduce behavior in case of switching from a high product target amount to a low product target amount and vice versa during the experiment. The performance of the participants increased significantly in the first case. The original IBL models were not able to capture this behavior. The authors therefore developed a rule based model that captured the subjects, switching behavior.

Rutledge-Taylor et al. (2012) compared a rule-based and an exemplar-based model for an intelligence categorization task where learned characteristics had to be studied and assigned. Both models performed equally well in predicting the participants' data. No model was superior to the other.

[3]Microworlds are computer simulations of specific problems. They are applied to study real-world problem solving in dynamic and highly complex settings.

In a different categorization study, Anderson and Betz (2001) studied three category-learning tasks with three different ACT-R models, an exemplar-based model, a rule-based model and a mixed model. The mixed model fitted best, reproducing learning and latency effects found in the empirical data.

In summary, there is no clear evidence that one or the other modeling approach is superior. In their paper, Anderson and Betz (2001) state that the mixed approach is probably the closest to how humans categorize, because the assumption that categorization is either exclusively exemplar-based or exclusively rule-based is probably too close-minded. Furthermore, stimulus succession and adequate rule specification are important for dynamic decision-making and category learning tasks.

In addition, models of complex tasks should incorporate metacognitive processes such as reflecting and evaluating the progress of the selected approach (Roll et al., 2004; Reitter, 2010; Anderson and Fincham, 2014). Reitter's (2010) model of the dynamic stocks and flow tasks investigated how subjects manage competing task strategies. The subject-to-subject analysis of the empirical data showed that participants exhibited sudden marked changes in behavior. Learning mechanisms which are purely subsymbolic cannot explain such behavior, because changes in model behavior would take too long. Furthermore, the strategies of the participants seemed to vary with the complexity of the water flow. Thus, a model of this task must address switches in strategy and not only gradual learning. Reitter (2010) assumes that humans' solutions to real-world problems emerge from a combination of general mechanisms (core learning mechanisms) and decision-making strategies common to many cognitive modeling tasks. His model implements several strategies to deal with the basic control task as well as a mechanism to rank and select those strategies according to their appropriateness in a given situation. This represents the metacognitive aspect of his model.

Our Aim

Our aim is to develop an ACT-R modeling approach for dynamic decision-making in a category learning task. A suitable task for such a modeling approach needs to fulfill several requirements. First, it should use complex multi-feature stimuli for the model to build categories from combined features. Second, the task needs to provide feedback, thereby allowing the model to learn. Third, changes in the environment should occur during the task forcing the model to act on them by refining once learned category assemblies.

To model performance in such a task, the modeling approach will need to incorporate mechanisms for strategy learning and strategy switching. It should precisely specify how hypotheses about category learning can be implemented with ACT-R. A mixed modeling approach of rules and exemplars should be used since previous work indicates that such models are most suitable for dynamic decision-making tasks. Furthermore, since switches in category assignments as well as monitoring of the learning progress need to be addressed, metacognitive aspects should be incorporated in the modeling approach.

Our modeling approach should provide information on the actual cognitive processes underlying human dynamic decision-making. Hence, it should be able to predict human behavior and show roughly the same performance effects that can be found in empirical data reflecting decision-making, e.g., response rates. Even more importantly, we aim at developing a general model of dynamic decision-making. For the model to be general (e.g., not fit exclusively to one specific experimental setting or dataset), it needs to be simple. Thus, only few assumptions should be used and unnecessary ones avoided. As a result, the modeling approach should be capable to predict behavior with other stimulus materials and be transferable to other similar tasks.

To summarize the scope of this article, our proposed modeling approach aims to depict the core processes of human decision-making, such as incorporating feedback, strategy updating, and metacognition. Building a model with a cognitive architecture ensures that evaluated cognitive processes are used. The quest is to see whether these cognitive aspects including the processes of the architecture can produce empirical learning behavior:

First, performance improvement through feedback should be included in the model. In the case of feature learning and strategy updating, improvements in one's strategy are only considered in the case of negative feedback (Li and Maani, 2011). If feedback signals a positive decision, people consider their chosen strategy for later use. Thus, people update their mental model during dynamic decision-making only if they receive negative feedback (Li and Maani, 2011). For our feature learning model, this implies that once a successful strategy has been chosen over alternatives, revisions to this strategy will require negative feedback on that strategy rather than positive experience with others, as these are no longer explored.

Second, the model should include transitions from simple to complex strategies. Findings suggest that people initially use simple solutions and then switch to more complex ones (Johansen and Palmeri, 2002). The modeling approach under discussion should be constructed in a similar fashion. In the beginning, it should follow simple one-feature categorization strategies and later switch to more complex two-feature strategies.

Third, the model needs to use metacognitive mechanisms. For example, it needs specifications for which conditions switching from a single-feature strategy to a multi-feature strategy is required. The metacognitive aspects should furthermore reflect previous learning successes. Thus, keeping track of which approaches were helpful and which were not, or of how often a strategy has been successful in the past, should be implemented in the model. Moreover, such mechanisms should ensure that if a strategy was successful in the past and fails for the first time, it is not discarded directly, but tested again. Furthermore, metacognitive mechanisms should not only address the issue of switching from single-feature to multi-feature strategies but also incorporate responses to changes in the environment.

MATERIALS AND METHODS

In the following, an experiment of dynamic decision-making and our model performing the same task are presented. The model

includes mechanisms to integrate feedback, to switch from simple to complex strategies and to address metacognition. The model was built after the experimental data were obtained.

This section is subdivided in the following manner: First, the participant sample, setup and stimuli of the empirical experiment are described. Then, the modeling approach is explained in detail. Afterwards, the model setup and stimuli are presented. Finally, the analytical methods to evaluate the fit between the model and the empirical results are outlined.

Experiment Participants

55 subjects participated in the experiment that took place inside a 3 Tesla MR scanner[4] (27 female, 28 male, age range between 21 and 30 years, all right handed, with normal hearing). All subjects gave written informed consent to the study, which was approved by the ethics committee of the University of Magdeburg, Germany.

Experimental Stimuli

A set of frequency-modulated different tones served as stimuli for the categorization task. The tones differed in duration (short, 400 ms, vs. long, 800 ms), direction of frequency modulation (rising vs. falling), intensity (low intensity, 76–81 dB, vs. high intensity, 86–91 dB), frequency range (five low frequencies, 500–831 Hz, vs. five high frequencies, 1630–2639 Hz), and speed of modulation (slow, 0.25 octaves/s, vs. fast, 0.5 octaves/s), resulting in $2 \times 2 \times 2 \times 10 \times 2$ (160) different tones. The task relevant stimulus properties were the direction of frequency modulation and sound duration, resulting in four tone categories: short/rising, short/falling, long/rising, and long/falling. For each participant, one of these categories constituted the target sounds (25%), while the other three categories served as non-targets (75%).

As feedback stimuli, we used naturally spoken utterances (e.g., ja, "yes"; nein, "no") as well as one time-out utterance (zu spät, "too late") taken from the evaluated prosodic corpus MOTI (Wolff and Brechmann, 2012, 2015).

Experimental Paradigm

The experiment lasted about 33 min in which a large variety of frequency-modulated tones (see Section Experimental Stimuli above) were presented in 240 trials in pseudo-randomized order and with a jittered inter-trial interval of 6, 8, or 10 s. The participants were instructed to indicate via button-press whether they considered the tone in each trial to be a target (right index finger) or a non-target (right middle finger). They were not informed about the target category but had to learn by trial and error. Correct responses were followed by positive feedback, incorrect responses by negative feedback. If participants failed to respond within 2 s following the onset of the tone, the time-out feedback was presented.

After 120 trials, a break of 20 s was introduced. From the next trial on the contingencies were reversed such that the target stimulus required a push of the right instead of the left button. The participants were informed in advance about a resting period after finishing the first half of the experiment but they were not told about the contingency reversal.

[4]The experiments were performed inside an MR scanner to study the specific neural correlates of strategy formation which is the subject of another paper.

Model in Detail

In the following, the model is presented in detail. First, a description of the main declarative representations (chunks) is provided. They reflect strategy representations and metacognitive processes. This is followed by a description of how the model runs through a trial. Finally, the rules that govern strategy learning are summarized.

Chunks and Production Rules Used in the Model

The chunks implemented in the model are shown in **Figure 1**. "Strategy chunks" hold the strategies in form of examples of feature-value pairs and responses. They are stored in and retrieved from long-term memory (declarative module). The current strategy is held in working memory (imaginal module). Strategy chunks contain the following information about the strategy: which feature(s) and what corresponding value(s) are relevant (e.g., the sound is loud or the sound is loud and its frequency range is high), what the proposed response is (categorization, 1 or 0), and the degree of complexity of the strategy (e.g., one or two-feature strategy). Furthermore, an evaluation mechanism is part of this chunk. This includes noting if a strategy was unsuccessful and keeping track of how often a strategy was successful. This tracking mechanism notices if the first attempt to use this strategy is successful. It then counts the number of successful strategy uses; this explicit count is continued until a certain value is reached. We implemented such a threshold count mechanism to reflect the subjective feeling that a strategy was often useful. We implemented different threshold values for the model. We also differentiated between the threshold for one-feature strategies (first count) and for two-feature strategies (second count). The tracking mechanism can be seen as a metacognitive aspect of our model. Other metacognitive aspects are implemented in the "control chunk" which is kept in the goal buffer of the model. These metacognitive aspects include: first, the level of feature-complexity of the strategy, i.e., if the model attempts to solve the task with a one-feature or with a two-feature strategy; second, whether or not a long-time successful strategy caused an error, this signifies the model's uncertainty about the accuracy of the current strategy; third, whether changes in the environment occurred that require to renew the search for an adequate strategy.

Trial Structure

Production rules govern how the model runs through the task. The flow of the model via its production rules is illustrated in **Figure 2**. The following section describes how the model runs through a trial, the specific production rules are noted in parentheses.

A tone is presented to the model and enters the aural-location buffer *(listen)*. After the tone has finished, it is encoded in the aural buffer *(encode)*. Thus, a chunk with all audio information necessary (duration, direction of pitch change, intensity, and

control chunk	
complexity	(one or two)
uncertain	(nil – yes)
environment	(nil-change)

strategy chunk	
complexity	(one or two)
1.feature-value-pair	(e.g. duration short)
2.feature-value-pair	(nil, e.g. volume high)
response	(0 or 1)
unsuccessful	(nil-yes)
first attempt	(nil-yes)
1.count	(nil, 1,2…threshold)
2.count	(nil, 1,2…threshold)

FIGURE 1 | Schematic build-up of the structure of the control and the strategy chunk. Nil indicates that the variable has no value.

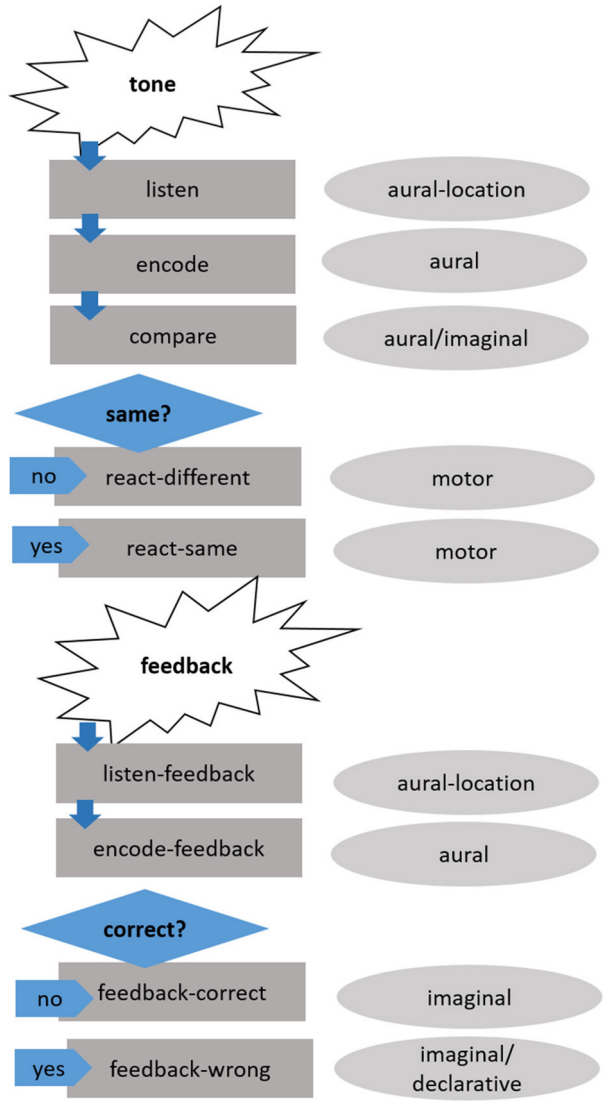

FIGURE 2 | Schematic overview of how the model runs through a trial. The dark-gray boxes on the left represent the production rules, the light-gray ovals on the right the main buffers involved.

frequency range—see Section Modeling Paradigm and Stimuli below) is in the aural buffer and all four characteristics of the tone are accessible to the model. The audio chunk in the aural buffer is then compared to the strategy chunk held in the imaginal buffer *(compare)*. If the specific features (e.g., intensity is high) of the strategy chunks are the same as in the audio chunk, the response is according to the strategy proposed by the model *(react-same)*, if not, the opposite response is chosen *(react-different)*. The presented feedback is listened to and held in the aural-location buffer *(listen-feedback)* and then encoded in the aural buffer *(encode-feedback)*. If the feedback is positive, the current strategy is kept in the imaginal buffer and the count-slot is updated *(feedback-correct)*. If the feedback is negative, the strategy is updated depending on previous experiences *(feedback-wrong)*. Thus, a different strategy chunk is retrieved from declarative memory and copied to the imaginal buffer.

Finding an Adequate Strategy

All possible strategies are already available in the model's long-term memory. The currently pursued strategy is maintained in working memory and evaluated regarding the feedback. For positive feedback, the strategy is retained and it is counted how often it is successful. If feedback is negative, the strategy is usually altered. The following subsection is a summary of how strategy updating is implemented. For more information see **Figure 3**.

The model always begins with a one-feature strategy (which strategy it begins with is random) and then switches to another one-feature strategy. The nature of the switch depends on how often a particular strategy was successful. When the model searches for different one-feature strategies, it retrieves only strategies which were not used recently. In case of immediate failure of a one-feature strategy, a different response is used for the feature-value pair. In other cases, the feature-value pair is changed, but the response is retained. If a one-feature strategy has been successful often and then fails once, the strategy is not directly exchanged, but re-evaluated. However, it is also noted that the strategy has caused an error. Two possibilities explain why switches from a one-feature to a two-feature strategy occur: Such a switch can happen either because no one-feature strategy that was not negatively evaluated can be retrieved or because an often successful one-feature strategy failed repeatedly. Switches

Switching between different one-feature strategies

If the current strategy is a one-feature strategy and it was not successful at the first attempt then retrieve any other one-feature strategy that has not been evaluated as unsuitable and has not been retrieved recently.

If the current strategy is a one-feature strategy and it was successful at the first attempt but has not been successful often then retrieve any other one-feature strategy that has a different feature and has not been evaluated as unsuitable and has not been retrieved recently.

If the current strategy is a one-feature strategy and was successful often then keep the strategy, but note that this strategy caused an error.

Switching from one-feature to two-feature strategies

If the current strategy is a one-feature strategy that was successful often but it caused an error before then retrieve a two-feature strategy which includes the current feature value pair and response.

If the attempt to retrieve a one-feature strategy has failed then retrieve a two-feature strategy.

Switching between different two-feature strategies

If the current strategy is a two-feature strategy and was not successful at the first attempt then retrieve any other two-feature strategy that has not been evaluated as unsuitable and has not been retrieved recently.

If the current strategy is a two-feature strategy and was successful at the first attempt but has not been successful often then retrieve a two-feature strategy which includes the current feature value pair and response but has a different feature and has not been evaluated as unsuitable and has not been retrieved recently.

If the current strategy is a two-feature strategy and has been successful often, then keep the strategy but note that this strategy caused an error.

Detecting a change

If the current strategy is a two-feature strategy and it caused an error before then retrieve a different two-feature strategy and note that something in the environment has changed and evaluate the strategy as unsuitable for this change.

Searching for a strategy after a detected change

If the current strategy is a two-feature strategy and a change in the environment is detected and the first attempt to use this strategy was not successful then retrieve a different two-feature strategy that has not been evaluated as unsuitable for this change.

If the current strategy is a two-feature strategy and a change in the environment is detected and the strategy was successful more than once but not often, then keep one feature and the response and retrieve a strategy that has not been evaluated unsuitable for this change.

FIGURE 3 | Rules governing when and to what degree the strategies are changed after negative feedback is received.

within the two-feature strategy are modeled the following way: If a two-feature strategy was unsuccessful at the first attempt, any other two-feature strategy is used (which one exactly is random). If a two-feature strategy was initially successful and then fails, then a new strategy which retains one of the feature-value pairs and the response will be selected. This strategy only differs in the other feature-value pair. When the environment changes, a previously often successful two-feature strategy (and also a one-feature strategy) will fail. Then a retrieval of another two-feature strategy is attempted. If at the time the environment changes, the model has not found a successful two-feature strategy, it will continue looking for a useful two-feature strategy, and thus not notice the change.

Modeling Paradigm and Stimuli

The following section briefly describes how the experiment was implemented for the model. This includes a short overview of how the stimulus presentation was modified for the model.

The task of the participants was implemented for the model in ACT-R 7.3 with some minor modifications. The same four pseudo-randomizations used for the participants were also used for the model. Thus, 25% of the stimuli were target stimuli. A trial began with a tone, which lasted for 400 ms. To model the two stimulus durations, we used two different features in the new-other-sound command. As soon as the model responded via button press, auditory feedback was presented. Overall, a trial lasted for a randomized period of 6, 8, or 10 s, similar to the original experiment. There was no break for the model after 120 trials, but the targets switched after 120 trials, too.

Instead of employing all 160 different tones, sixteen different tones were presented to the model. Each of the tones is a composition of four characteristics of the four binary features: duration (long vs. short), direction of frequency modulation (rising vs. falling), intensity (low intensity, vs. high intensity), and frequency range (low vs. high). Only binary features were used for the model because the perceptual difference between the two classes of each selected feature was high, except for speed of modulation, which was therefore not implemented in the model. For the participants, more feature variations were used to ensure categorical decisions and to prevent them from memorizing individual tone-feedback pairs. This is not an issue for the model, since no mechanism allowing such memorizing was implemented. As for the participants, auditory feedback was presented to the model.

The modeling approach is a mixed modeling approach, the strategies are encoded as instances, but which instance is retrieved is mainly governed by rules.

To test if the model is a generalizable model, different variations were implemented. The learning curves found in the empirical data should still be found under different plausible parameter settings. However, specific parameter settings should influence the predictive quality of the model. The approach typically chosen by cognitive modelers is to search for specific parameter settings that result in an optimal fit and then report this fit. The objective behind such an approach is to show that the model resembles the ongoing cognitive processes in humans. We have chosen a different approach. Our objective is to show that our modeling approach can map the general behavior such as learning and reversal learning as well as variance found in the data. By varying parameter settings, we want to optimize the fit of the model and examine the robustness of the model mechanisms to parameter variations.

Regarding the choice of varying parameters, we use an extended parameter term which includes not only subsymbolic

ACT-R parameters (which are typically regarded as parameters) but also certain (production) rules (Stewart and West, 2010). In the case of this model, productions that control the tracking mechanism of successful strategies are varied. The tracking mechanism keeps track of how often a strategy is successful. However, the model does not increase the count throughout the entire experiment. After it reaches a threshold, a successful strategy is marked as "successful often." Thereafter, it is not discharged directly in case of negative feedback but instead reevaluated. So, to answer the question what the most suitable values for the threshold of the first and second count are, these values were varied. Another implemented model assumption is that this threshold is different for single-feature vs. two-feature strategies. We assumed that the threshold for two-feature strategies should be double the value for one-feature strategies, as if the model was counting for each feature separately. The first count was varied for three, four and five and the second count for six, eight, and ten.

Besides the parameters that control the tracking mechanism, we also investigated a parameter-controlled memory mechanism. The latter controls for how long the model can remember if it had already used a previous strategy. This is the *declarative-finst-span*[5] parameter of ACT-R. We assumed that participants remember which strategy they previously used for around 10 trials back. We therefore tested two different values (80 and 100 s) for this parameter, determining whether the model can remember if this chunk has been retrieved in the last 80 (or 100) s. The combination of the declarative-finst-span (80, 100), three values for the first count (3, 4, 5) and three values for the second count (6, 8, 10) resulted in 18 modeling versions (see **Table 1**).

TABLE 1 | Resulting modeling versions from combining the different parameter settings for the first and second count and the declarative-finst-span.

		First count 3	First count 4	First count 5
:declarative-finst-span 80	Second count 6	3_06 _080	4_06 _080	5_06 _080
	Second count 8	3_08 _080	4_08 _080	5_08 _080
	Second count 10	3_10 _080	4_10 _080	5_10 _080
:declarative-finst-span 100	Second count 6	3_06 _100	4_06 _100	5_06 _100
	Second count 8	3_08 _100	4_08 _100	5_08 _100
	Second count 10	3_10 _100	4_10 _100	5_10 _100

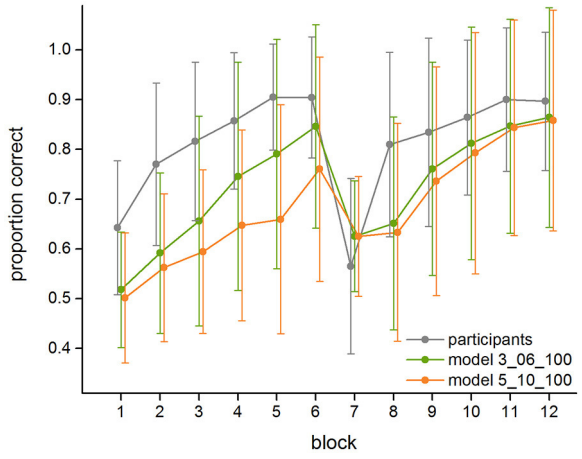

FIGURE 4 | Average performance and standard deviations of the human participants, the best fitting model (3_06_100), and the worst fitting model (5_10_100) in the 12 blocks of the experiment.

Analyses

Each of the models was run 160 times, 40 times for each pseudo-randomized order, using ACT-R 7.3. The data were preprocessed with custom Lisp files and then analyzed with Microsoft Excel.

The model data and the empirical data were divided into 12 blocks, with 20 trials per block. The average proportion of correct responses and the standard deviation per block was computed for the experiment as well as for each of the 18 models.

One aim of this study was to predict average learning curves of the participants. Thus, the proportion of correct responses of the participants was compared to the proportion of correct responses of each of the models. Visual graphs comparing the modeled to the empirical data were analyzed with regard to increases and decreases in correct responses.

As an indication of relative fit, the correlation coefficient (r) and the determination coefficient (r^2) were computed. They represent how well trends in the empirical data are captured by the model.

As an indication of absolute fit, the root-mean-square error (RMSE) was calculated. RMSE represents how accurately the model predicts the empirical data. RMSE is interpreted as the standard deviation of the variance of the empirical data that is not explained by the model.

To compare the participant-based variance found in the empirical data with the variance produced by the 160 individual model runs, a Levene's test (a robust test for testing the equality of variances) was calculated for each block of the experiment.

[5]The declarative-finst-span parameter controls how long a finst (fingers of instantiation) can indicate that a chunk was recently retrieved. The number of items and the time for which an item can be tagged as attended is limited. These attentional markers are based on the work of Zenon Pylyshyn.

RESULTS

In the following sections, the empirical data, the modeled learning curves, and the results regarding the general fit of the different model versions to the data are presented.

Empirical Learning Curves

The descriptive analysis of the empirical data (see **Figure 4** and **Table 2**) shows that on average, in the first block the participants respond correctly in 64.3% (±13.5%) of the trials. The response rate of the participants increases until the sixth block to 90.4% (±12.2%) of correct trials. In the seventh block, the block in which targets and non-targets switch, it drops to 56.5% (±17.7%) of correct trials. It then increases again and reaches 81.0% (±18.5%) of correct trials in the eighth block and 89.7% (±13.9%) of correct trials in the last block. Across all 12 blocks, the standard

TABLE 2 | Average proportion of correct responses and standard deviations (in %) of the participants and the 18 versions of the model in the 12 blocks of the experiment.

	1	2	3	4	5	6	7	8	9	10	11	12
Participants	64.3 ± 13.5	77.0 ± 16.4	81.6 ± 15.9	85.7 ± 13.7	90.5 ± 10.7	90.4 ± 12.2	56.5 ± 17.7	81.0 ± 18.5	83.4 ± 18.9	86.4 ± 15.6	90.0 ± 14.4	89.7 ± 13.9
MODELS												
3_06_080	51.1 ± 13.9	59.0 ± 15.5	67.3 ± 20.9	73.9 ± 22.3	75.9 ± 24.5	82.8 ± 21.1	62.9 ± 11.8	64.4 ± 20.2	71.3 ± 22.5	73.9 ± 25.9	81.0 ± 22.3	83.8 ± 23.6
3_06_100	51.8 ± 11.6	59.2 ± 16.1	65.6 ± 21.0	74.6 ± 22.9	79.1 ± 23.0	84.7 ± 20.4	62.5 ± 11.1	65.1 ± 21.4	76.1 ± 21.4	81.2 ± 23.4	84.7 ± 21.5	86.4 ± 22.1
3_08_080	50.1 ± 11.7	56.0 ± 16.0	62.0 ± 17.9	68.5 ± 20.7	72.3 ± 23.8	81.5 ± 20.9	62.8 ± 11.4	66.7 ± 20.4	75.6 ± 22.4	77.6 ± 25.6	83.7 ± 22.1	87.0 ± 21.3
3_08_100	50.0 ± 11.5	57.9 ± 14.9	66.7 ± 21.1	73.8 ± 22.2	77.2 ± 23.9	86.5 ± 19.8	61.9 ± 12.3	61.6 ± 21.2	73.8 ± 24.0	79.9 ± 22.7	82.8 ± 23.3	87.2 ± 20.7
3_10_080	51.3 ± 11.9	54.5 ± 16.8	66.9 ± 20.6	71.8 ± 22.4	75.9 ± 24.2	82.7 ± 22.5	62.6 ± 11.7	63.6 ± 19.4	70.3 ± 22.7	75.6 ± 23.6	81.1 ± 21.6	83.6 ± 22.9
3_10_100	51.6 ± 12.5	60.6 ± 17.1	66.3 ± 21.0	73.5 ± 22.8	74.1 ± 25.4	83.8 ± 20.6	63.0 ± 11.4	65.3 ± 21.1	74.7 ± 23.1	79.8 ± 23.8	84.2 ± 22.7	87.7 ± 21.1
4_06_080	50.5 ± 12.3	58.5 ± 16.0	65.6 ± 20.3	71.7 ± 20.8	72.2 ± 24.7	81.6 ± 20.9	61.6 ± 12.2	64.9 ± 20.9	73.5 ± 24.1	76.7 ± 24.4	82.6 ± 22.6	83.7 ± 23.9
4_06_100	52.3 ± 11.8	58.3 ± 15.4	65.5 ± 20.9	71.4 ± 21.8	74.5 ± 23.4	81.9 ± 21.1	62.2 ± 11.8	63.9 ± 20.9	74.0 ± 22.4	76.0 ± 25.5	84.2 ± 21.6	85.7 ± 21.9
4_08_080	50.3 ± 12.6	56.2 ± 14.0	61.0 ± 17.7	70.9 ± 21.5	72.9 ± 24.4	78.4 ± 23.4	64.0 ± 11.1	63.1 ± 21.6	75.3 ± 22.8	77.7 ± 25.1	84.3 ± 20.7	85.0 ± 23.3
4_08_100	51.1 ± 12.1	57.8 ± 15.1	63.3 ± 20.0	68.7 ± 20.8	70.1 ± 24.5	78.7 ± 23.4	62.8 ± 12.7	66.2 ± 20.5	74.8 ± 23.1	77.6 ± 25.2	83.3 ± 21.9	85.9 ± 22.0
4_10_080	49.5 ± 11.3	58.9 ± 15.5	63.0 ± 20.1	69.9 ± 21.1	70.2 ± 25.6	80.0 ± 21.8	63.3 ± 11.3	61.3 ± 20.1	70.0 ± 23.7	73.2 ± 24.7	81.7 ± 21.8	81.9 ± 24.8
4_10_100	51.7 ± 12.5	56.8 ± 16.1	64.5 ± 16.9	68.3 ± 22.2	71.9 ± 24.3	81.3 ± 22.4	63.3 ± 11.7	64.2 ± 20.7	72.6 ± 22.7	79.0 ± 23.2	83.5 ± 22.4	86.3 ± 22.0
5_06_080	51.3 ± 12.2	56.5 ± 14.3	61.0 ± 16.2	66.8 ± 20.1	69.1 ± 23.3	78.8 ± 21.9	61.4 ± 11.7	64.1 ± 20.5	72.8 ± 22.9	74.9 ± 25.0	83.2 ± 21.9	85.1 ± 22.8
5_06_100	53.0 ± 11.6	58.5 ± 16.5	61.3 ± 18.8	65.1 ± 21.2	65.9 ± 23.1	76.2 ± 20.7	59.8 ± 11.9	64.1 ± 19.7	72.0 ± 23.0	74.7 ± 24.3	81.3 ± 21.6	85.9 ± 21.2
5_08_080	50.2 ± 11.2	55.3 ± 16.2	58.6 ± 16.0	64.3 ± 19.5	65.3 ± 22.6	72.6 ± 22.2	58.8 ± 12.2	62.6 ± 19.6	72.7 ± 21.3	75.6 ± 23.1	82.6 ± 21.9	86.6 ± 20.9
5_08_100	49.1 ± 12.0	55.0 ± 14.1	60.7 ± 18.2	67.4 ± 20.2	67.0 ± 23.4	74.1 ± 22.6	61.2 ± 12.0	63.8 ± 21.2	73.1 ± 22.4	73.5 ± 24.1	81.5 ± 21.8	83.4 ± 23.3
5_10_080	50.3 ± 12.5	56.4 ± 14.8	60.8 ± 18.9	68.1 ± 19.7	69.3 ± 23.6	76.5 ± 22.7	62.3 ± 13.4	65.5 ± 21.9	74.9 ± 23.0	76.4 ± 25.7	84.5 ± 20.3	85.5 ± 22.0
5_10_100	50.2 ± 13.1	56.2 ± 14.9	59.4 ± 16.4	64.7 ± 19.2	65.9 ± 23.0	76.0 ± 22.5	62.5 ± 12.0	63.3 ± 21.9	73.6 ± 23.0	79.3 ± 24.3	84.3 ± 21.7	85.8 ± 22.2
MODELS OVERALL												
MEAN	50.8	57.3	63.3	69.6	71.6	79.9	62.2	64.1	73.4	76.8	83.0	85.4
MIN	49.1	54.5	58.6	64.3	65.3	72.6	58.8	61.3	70.0	73.2	81.0	81.9
MAX	53.0	60.6	67.3	74.6	79.1	86.5	64.0	66.7	76.1	81.2	84.7	87.7

deviation of the empirical data ranges from 10.7% minimum to 18.9% maximum, with an average standard deviation of 15.1%. The standard deviation of the participants derives from the fact that different participants showed different learning curves, and not all participants reported to have found the correct strategy in a post interview. Correspondingly, eleven participants (20.0%) showed a performance below 85% by the end of the first part of the experiment (Block 6), and 12 participants (21.8%) stayed below 85% correct responses at the end of the second part (Block 12).

Modeled Learning Curves

Figure 4 further shows the means and standard deviations of the proportion of correct responses of the best (3_06_100) and worst fitting (5_10_100) model (see below, Section Model Fit). In addition, **Table 2** lists the model performance means and standard deviations for each of the twelve blocks for all 18 models, and **Figure 5** shows the learning curves of all 18 models.

Both the best and the worst fitting model (as do all others) capture the overall shape of the learning curve found in the data. They both show an increase in the learning rate in the first six blocks. Similarly, all models show a drop in performance in the seventh block, which is followed by another increase in performance. Even in the best fitting 3_06_100 model, however, the proportion of correct responses is underestimated by the model, especially in the first blocks. Also, the participants show a more severe setback after the switch but then recover faster, while the model takes longer until its performance increases again. Nevertheless, for the best fitting model, the modeled data are always within the range of the standard deviation of the empirical data.

As **Table 2** shows, each of the models shows a large degree of variance across its 160 runs. The standard deviation averaged across all 12 blocks ranges from 18.9 to 20.4%, depending on the model's parameter settings. For the best-fitting model, the standard deviation in the individual blocks ranges from 11.6 to 23.4% and is significantly larger than the standard deviation found in the empirical data, except for the first two blocks of the experiment and the first two blocks after the switch (for all blocks except Block 1, 2, 7, and 8: all Fs > 6.79, all ps < 0.010). This high variation of the individual model runs indicates that the same underlying rule-set with the same parameter settings can still result in very different learning curves, depending on which exact strategies are chosen at each point when a new strategy is selected (e.g., initial strategy, alteration of one-feature strategy, alteration of two-feature strategy). Furthermore, similarly to the non-learners among the participants described above (see Section Empirical Learning Curves), not every model run was successful, resulting (for the best fitting model) in a performance below 85% in 35.6% of the runs for Block 6 and in 30.0% of the runs for Block 12.

Model Fit

The average correlation of the model and the empirical data is 0.754. Between 43.9% and 67.1% of the variance in the data is explained by the different models. The average standard deviation of the unexplained variance is 0.136. All r, r^2, and RMSE values for the 18 model versions are presented in **Table 3**.

As **Table 3** and **Figure 5** show, the model shows relative robustness to the influence of varying parameter settings. For the first count, a lower value is somewhat better for the fit—there is a stronger increase in the first part of the experiment (until Block 6) for a lower than for a higher first count value. For the second count, a lower value results in a better fit as well. The influence of the declarative-finst-span parameter on the fit-indices is very small, resulting in a slightly better fit either for a declarative-finst-span of 80 s or of 100 s, depending on the settings of first and second count.

The best fit in terms of correlation was achieved for the model with the declarative-finst-span value set to 100 (i.e., the model was able to remember if it had already used a previous strategy for 100 s), a first count of three (i.e., a one-feature strategy needed to be successful at least three times to be considered as "often successful") and a second count of six (i.e., a two-feature strategy

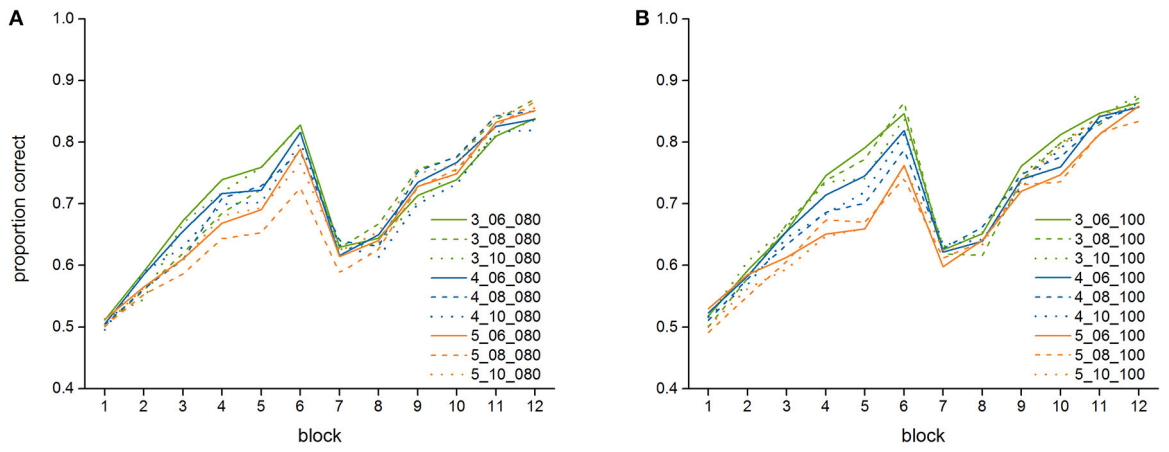

FIGURE 5 | Average performance of the 18 versions of the model in the 12 blocks of the experiment, **(A)** models with a declarative-finst-span of 80 s, **(B)** models with a declarative-finst-span of 100 s.

TABLE 3 | Values of r, r^2, and RMSE of the 18 versions of the model.

	r	r^2	RMSE
3_06_080	0.812	0.659	0.124
3_06_100	0.820	0.672	0.109
3_08_080	0.745	0.555	0.134
3_08_100	0.803	0.645	0.119
3_10_080	0.785	0.616	0.132
3_10_100	0.798	0.636	0.114
4_06_080	0.805	0.649	0.128
4_06_100	0.794	0.631	0.124
4_08_080	0.726	0.527	0.138
4_08_100	0.743	0.552	0.135
4_10_080	0.745	0.555	0.146
4_10_100	0.741	0.549	0.133
5_06_080	0.733	0.537	0.146
5_06_100	0.722	0.521	0.152
5_08_080	0.697	0.485	0.164
5_08_100	0.718	0.516	0.158
5_10_080	0.721	0.520	0.144
5_10_100	0.663	0.439	0.156
OVERALL			
MEAN	0.754	0.570	0.136
MIN	0.663	0.439	0.109
MAX	0.820	0.671	0.164

needed to be successful at least six times to be considered as "often successful"). The worst fit was observed for the model with the declarative-finst-span value set to 100, a first count of five and a second count of ten.

The RMSE varies from a minimum of 0.106 (3_06_100) to a maximum of 0.164 (5_08_100). Thus, the model with a first count of three, a second count of six and a declarative-finst-span set to 100 performs best, both in terms of correlation (r) and absolute prediction (RMSE).

Summary

In general, the models predict the data well. The modeled learning curves resemble the form of the average empirical learning curve, with an increase in the first half of the experiment, a short decrease at the beginning of the second half, followed by another increase in performance. The correlation indices of the best fitting model show a good fit, with 67.2% of the variance of the data being explained by the model with a declarative-finst span of 100 s, a first count threshold of three and second count threshold of six. Note that this is also the model with the closest absolute fit (RSME is 0.109).

However, in absolute percentages of correct responses, all of the models perform below the participants in all blocks (except Block 7). Also, the models show greater overall variance than the empirical data. Furthermore, the models are initially less affected by the switch in strategies but take longer to "recover" from the switch in strategies.

In summary, the model replicates the average learning curves and large parts of the variance. It does so with a limited set of rules and the given exemplars, covering learning and relearning processes which take place in dynamic environments. Moreover, we found differences in model fit depending on the exact specification of the parameters, with the best fit if the model remembers previously employed strategies for 100 s, marks a one-feature strategy as "often successful" after three successful uses and a two-feature after six successful uses. However, all of the 18 different parameter settings we tested resembled the main course of the empirical data, thereby indicating that the mechanisms of the model are robust to parameter variations.

DISCUSSION

The discussion covers three main chapters. First, the fit of the model is discussed and suggestions for possible improvements are given. Second, the broader implications of our approach are elaborated. Finally, future work is outlined.

Discussion of the Modeling Approach

Our modeling account covers relevant behavioral data of a dynamic decision-making task in which category learning is required. To solve the task, two features have to be combined, and the relevant feature combination needs to be learned by trial and error using feedback. The model uses feedback from the environment to find correct categories and to enable a switch in the assignment of response buttons to the target and non-target categories. Metacognition is built into the model via processes that govern under what conditions strategic changes, such as transitions from one-feature to two-feature strategies, occur.

Overall, the fit indices indicate that this model solves the task in a similar way as participants do. This includes successful initial learning as well as the successful learning of the reversal of category assignment. Moreover, the observation was made that not all participants are able to solve the task, and the same is observed in the behavior of the modeling approach. Thus, the model is able to generate output data that, on a phenomenological level, resemble those of subjects performing a dynamic decision-making task that includes complex rule learning and reversal processes. Although the overall learning trends found in the data can be replicated well with the general rules implemented in our model, there are two limitations: The variance of the model is larger than that of the participants, and the overall performance of the model is lower than the performance of the participants.

It is likely that the participants have a different and perhaps more specific set of rules than the model. For example, the participants were told which of the two keys to press for the target sound. However, it is unclear if they used this knowledge to solve the task. To keep the model simple, it was not given this extra information, so there was no meaning assigned to the buttons. This is one possibility to explain the model's lower performance, especially in the first block. Another example for more task specific rules used by the participants compared to the model is that the four different features of the stimuli may not be equally salient to the subjects, which may have led to a higher performance compared to the model. For example,

it is conceivable that the target-feature direction of frequency modulation (up vs. down) was chosen earlier in the experiment than the non-target feature frequency range, while the model treated all features equally to keep the model as simple as possible. Finally, after the change of the button press rule, some participants might have followed a rule which states to press the opposite key if a strategy was correct for many times and then suddenly is not, instead of trying out a different one- or two-feature strategy, whereas the model went the latter way.

Adding such additional rules and premises to the model would possibly reduce the discrepancy between the performance of the model and the behavioral data. However, the aim of this paper was to develop a modeling approach that incorporates general processes important for all kinds of dynamic decision-making. This implies using only assumptions that are absolutely essential (meta-cognition, switching from one-feature to two-feature strategies, learning via feedback) and keeping the model as simple as possible in other regards. As a consequence, adding extra rules would not produce a better general model of dynamic decision-making, but would only lead to a better fit of the model for a specific experiment while making it prone to overfitting. As mentioned earlier, good descriptive models capture the behavioral data as closely as possible and therefore always aim at maximizing the fit to the data they describe. Good predictive models, on the other hand, should be generalizable to also predict behavior in different, but structurally similar situations and not just for one specific situation with one set of subjects. In our view, this constitutes a more desirable quest with more potential to understand the underlying processes of human dynamic decision-making. This is supported by Gigerenzer and Brighton (2009), who argue that models that focus on the core aspects of decision-making, e.g., considering only few aspects, are closer to how humans make decisions. They also argue that such simplified assumptions make decisions more efficient and also more effective (Gigerenzer and Brighton, 2009).

As stated earlier, one way to model dynamic decision-making in ACT-R using only few assumptions is instance based learning (IBL). This approach uses situation-outcome pairs and subsymbolic strengthening mechanisms for learning. However, IBL is insufficient to model tasks which involve switches in the environment (Fum and Stocco, 2003). Such tasks require adding explicit switching rules. Besides these rules, our task needed mechanisms that control when to switch from simple one-feature strategies to more complex strategies. Since meta-cognitive reflections are not part of IBL, we used a mixed modeling approach which incorporates explicit rules and metacognitive reflection. IBL is insofar part of our approach as the strategies are encoded as situation-outcome pairs and subsymbolic strengthening mechanisms of ACT-R are utilized.

To evaluate if our modeling approach of strategy formation and rule switching is in line with how participants perform in such tasks, data reflecting learning success need to be considered. Such data are the learning curves reported in this paper. We believe that an IBL model alone cannot produce the strong increase in performance after the environmental change in the empirical data.

For a further understanding of complex decision-making, other behavioral data, such as reaction times, could also be modeled. However, not all processes that probably have an impact on reaction time are part of our general modeling approach. This is especially the case for modeling detailed aspects of auditory encoding with ACT-R; for example, the precise encoding of the auditory events can be expected to comprise a different gain in reaction time for short compared with longer tones. However, our modeling approach is expandable, allowing the incorporation of other cognitive processes such as more specific auditory encoding or attention. This extensibility is one of the strengths of cognitive architectures and is particularly relevant for naturalistic decision-making, where many additional processes eventually need to be considered.

Scope of the Model

A formal model was built with ACT-R, it specifies the assumptions of dynamic decision-making in category learning. This model was tested on empirical data and showed similar learning behavior. Assumptions about how dynamic decisions in category learning occur, e.g., by learning from feedback and switching from simple to more complex strategies, and metacognitive mechanisms were modeled together. ACT-R aims at modeling cognition as a whole, thus addressing different cognitive processes simultaneously, an important aspect for modeling realistic cognitive tasks. Moreover, the model is flexible. Thus, the model chooses from the available strategies according to previous experience and random influences.

Our modeling approach is simple in the sense that it comprises only few plausible assumptions, does not rely on extra parameters and is nevertheless flexible enough to cope with dynamically changing environments.

To test the predictive power of the model, it needs to be further tested and compared to new empirical data that are obtained using slightly different task settings. Our aim was to develop a first model of dynamic decision-making in category learning. Thus, relevant cognitive processes that occur between stimulus presentation and the actual choice response are included in the model. Furthermore, we wanted to show how a series of decisions emerge in the pursuit of an ultimate goal. Thus, as a first step we needed a decision task that shows characteristics similar to natural dynamic settings. Such aspects include complex multi-feature stimuli, feedback from the environment, and changing conditions. Since explicit hints on category membership are usually not present in non-experimental situations, it is furthermore reasonable to use a task without explicit instructions regarding which features (or stimuli) attention should be focused on. The downside of using unspecific instructions as done in our study is that from the behavioral data, it will remain unclear how exactly individual participants process such a task, since aspects such as which exact rules are followed or which features are considered at the beginning of a task, are uncertain.

As a next step we aim at modeling and predicting the dynamic decision-making course of individual participants. In general, a big advantage of cognitive modeling approaches is that they can predict ongoing cognitive processes at any point in time. To

evaluate the validity of such predictions, different approaches can be followed.

One approach to constructing models in accordance with the cognitive processes of participants is the train-to-constrain paradigm (Dimov et al., 2013). This paradigm requires instructing participants in a detailed step-by-step procedure on how to apply specific strategies in decision tasks. This approach gives the modeler insight into the strategies that participants are using at a given time point. This again can be used to constrain ACT-R models in the implementation of these strategies. In future studies, we plan to adopt this paradigm by (a) instructing the participants and (b) adjusting our model accordingly. To ensure that the train-to-constrain paradigm was successfully implemented, self-reports of the participants should be used.

Another approach is to conduct interviews while the participant is performing the task. To confirm the model's predictions about the prospective behavior of participants, subjects of future empirical studies should thus be asked about their decisions during the course of the experiment. The first few participant decisions can be expected to be strongly influenced by random aspects (e.g., which feature is attended to first), but after some trials, the modeling approach should be able to predict the next steps of the participants. Thus, it should allow precise predictions of the subsequent cognitive processes. To make such predictions, a revised model would need to use the first couple of trials as information about the strategy an individual participant initially follows.

In a further step, the exact cognitive processes proposed by the model should be tested on an individual level on more fine grain data (e.g., fMRI) and then be readjusted accordingly. Currently, different methods to map cognitive models to finer grain data such as fMRI or EEG data have been proposed (Borst and Anderson, 2015; Borst et al., 2015; Prezenski and Russwinkel, 2016a). These methods are currently investigated and have been applied for basic research questions. Nevertheless, mapping cognitive models to neuronal data is a challenge. More research is needed especially for applied tasks. To supplement neuronal data, additional behavioral data, such as button press dynamics (e.g., intensity of button press), can be added as an immediate measurement of how certain an individual participant is about a decision (Kohrs et al., 2014).

Besides using cognitive models to predict individual behavior, we aim to develop more general cognitive mechanisms to model learning, relearning and metacognition that are valid in a broad range of situations. To test the applicability of our modeling approach in a broader context and different situations, variations of the experiment should be tested with different tasks and materials. For example, the model proposed here should be able to predict data from categorization experiments using visual stimuli such as different types of lamps (Zeller and Schmid, 2016) with some modifications to the sensory processing of our model. Furthermore, the model should be capable of predicting data from different types of categorization tasks, for example a task using a different number of categorization features, more switches or different sequences. Such a task would be a predictive challenge for our model; if it succeeds, it can be considered as a predictive model.

The developed general mechanisms can also be used in sensemaking tasks. Such tasks require "an active process to construct a meaningful and functional representation of some aspects of the world" (Lebiere et al., 2013, p. 1). Sensemaking is an act of finding and interpreting relevant facts amongst the sea of incoming information, including hypothesis updating. Performance in our task comes close to how people make sense in the real world because it involves a large number of different stimuli, each carrying different specifications of various features. Thus, "making sense of the stimuli" requires the participants to validate each stimulus in a categorical manner and use the extracted stimulus category in combination with the selected button-press and the feedback that follows as information for future decisions.

To conclude, such a cognitive model which includes general mechanism for learning, relearning and metacognition can prove extremely useful for predicting individual behavior in a broad range of tasks. However, uncertainty remains regarding whether this captures the actual processes of human cognition. This is not only due to the fact that human behavior is subject to manifold random influences, but also to the limitation that a model always corresponds to a reduced representation of reality. The modeler decides which aspects of reality are characterized in the model. Marewski and Mehlhorn (2011) tested different modeling approaches for the same decision-making task. While they found that their models differed in terms of how well they predicted the data, they ultimately could not show that the best fitting model definitely resembles the cognitive processes of humans. To our knowledge, no scientific method is ever able to answer how human cognition definitely works. In general, models can only be compared in terms of their predictive quality (e.g., explained variance, number of free parameters, generalizability). Which model ultimately corresponds to human reality, on the other hand, cannot be ascertained.

Outlook

One reason for modeling in cognitive architectures is to implement cognitive mechanisms in support systems for complex scenarios. Such support systems mainly use machine learning algorithms. Unfortunately, those algorithms depend on many trials to learn from before they succeed in categorization or in learning in general. Cognitive architecture inspired approaches, on the other hand, can also learn from few samples. In addition, approaches that rely on cognitive architectures are informed models that provide information about the processes involved and the reasons that lead to success and failure.

Cognitive models can be applied to a variety of real-world tasks, for example to predict usability in smartphone interaction (Prezenski and Russwinkel, 2014, 2016b), air traffic control (Taatgen, 2001; Smieszek et al., 2015), or driving behavior (Salvucci, 2006). Moreover, cognitive modeling approaches can also be used in microworld scenarios (Halbrügge, 2010; Peebles and Banks, 2010; Reitter, 2010). Not only can microworld scenarios simulate the complexity of the real world, they also have the advantage of being able to control variables. This implies that specific variations can be induced to test the theoretical approach or model in question (as demonstrated in Russwinkel et al., 2011).

Many applied cognitive models are quite specific task models. Our model, in contrast, aims at capturing core mechanisms found in a variety of real world tasks. As a consequence, it has the potential to be applied in many domains. So, our model of dynamic decision-making in a category learning task makes predictions about the cognitive state of humans during such a task. This involves predictions about strategies (e.g., one-feature or two-feature strategies), conceptual understanding (e.g., assumptions about relevant feature combinations) and metacognitive aspects (e.g., information on the success of the decision maker's current assumption), all of which are aspects of cognition in a multitude of tasks and application domains.

Our general modeling approach therefore has the potential to support users in many domains and in the long run could be used to aid decision-making. For this, the decisions of individual users during the course of a task could be compared to the cognitive processes currently active in the model. If for example a user sticks to a one-feature strategy for too long or switches rules in an unsystematic manner, a system could provide the user with a supportive hint. Other than regular assistant systems, such a support system based on our model would simulate the cognitive state of the user. For example, this online support system would be able to predict the influence of reoccurring negative feedback on the user, e.g., leading him to attempt a strategy change. If, however, the negative feedback was caused by an external source such as a technical connection error, opting for the strategy change would result in frustration of the user. The proposed support system would be able to intervene here. Depending on the internal state of the user, the support system would consider what kind of information is most supportive or if giving no information at all is appropriate (e.g., in case of mental overload of the user). As long as no support is needed, systems like this would silently follow the decisions made by a person.

Moreover, if the goal of the user is known, and the decisions made by the user have been followed by the system, it would be possible to predict the user's next decisions and also to evaluate whether those decisions are still reasonable to reach the goal. Many avalanches have been caused by repeated wrong decisions by backcountry skiers stuck in their wrong idea about a situation (Atkins, 2000). A support system that is able to understand when and why a person is making unreasonable decisions in safety critical situations would also be able to present the right information to overcome the misunderstanding. A technical support system for backcountry skiers would need information about current avalanche danger, potential safe routes and other factors. Such information is already provided by smartphone applications that use GPS in combination with weather forecasts and slope-steepness measures. In the future, when this information is made available to a cognitive model-based companion system that predicts the decisions of the users, it could potentially aid backcountry skiers. Cognitive model-based support systems designed in a similar manner could equally well be employed in other safety-critical domains, as well as to assist cyclist, drivers or pilots.

AUTHOR CONTRIBUTIONS

AB and SW designed the auditory category learning experiment. SW performed human experiments and analyzed the data. SP and NR designed the ACT-R modeling. SP implemented ACT-R modeling and analyzed the data. SP, SW, and AB prepared figures. SP, NR, and AB drafted manuscript. SP, NR, AB, and SW edited, revised and approved the manuscript.

FUNDING

This work was done within the Transregional Collaborative Research Centre SFB/TRR 62 "A Companion-Technology for Cognitive Technical Systems" funded by the German Research Foundation (DFG) and supported by funding by the BCP program.

ACKNOWLEDGMENTS

We thank Monika Dobrowolny and Jörg Stadler for support in data acquisition within the Combinatorial Neuroimaging Core Facility (CNI) of the Leibniz Institute for Neurobiology.

REFERENCES

Anderson, J. R. (1991). The adaptive nature of human categorization. *Psychol. Rev.* 98, 409–429. doi: 10.1037/0033-295X.98.3.409

Anderson, J. R. (2007). *How can the Human Mind Occur in the Physical Universe?* New York, NY: Oxford University Press.

Anderson, J. R., and Betz, J. (2001). A hybrid model of categorization. *Psychon. Bull. Rev.* 8, 629–647. doi: 10.3758/BF03196200

Anderson, J. R., and Fincham, J. M. (2014). Extending problem-solving procedures through reflection. *Cogn. Psychol.* 74, 1–34. doi: 10.1016/j.cogpsych.2014.06.002

Ashby, F. G. (1992). "Multidimensional models of categorization," in *Multidimensional Models of Perception and Cognition*, ed F. G. Ashby (Hillsdale, NJ: Erlbaum), 449–483.

Ashby, F. G., Alfonso-Reese, L. A., Turken, A. U., and Waldron, E. M. (1998). A neuropsychological theory of multiple systems in category learning. *Psychol. Rev.* 105, 442–481.

Ashby, F. G., and Maddox, W. T. (2011). Human category learning 2.0. *Ann. N. Y. Acad. Sci.* 1224, 147–161. doi: 10.1111/j.1749-6632.2010.05874.x

Atkins, D. (2000). "Human factors in avalanche accidents," *Proceedings. Int'l Snow Science Workshop* (Big Sky: MT), 46–51.

Berg, E. A. (1948). A simple objective technique for measuring flexibility in thinking. *J. Gen. Psychol.* 39, 15–22. doi: 10.1080/00221309.1948.9918159

Berry, D. C., and Broadbent, D. E. (1988). Interactive tasks and the implicit-explicit distinction. *Br. J. Psychol.* 79, 251–272. doi: 10.1111/j.2044-8295.1988.tb02286.x

Borst, J. P., and Anderson, J. R. (2015). "Using the ACT-R Cognitive Architecture in Combination with fMRI data," in *An Introduction to Model-Based Cognitive Neuroscience*, eds B. U. Forstmann, and E.-J. Wagenmakers (New York, NY: Springer), 339–352.

Borst, J. P., Nijboer, M., Taatgen, N. A., Van Rijn, H., and Anderson, J. R. (2015). Using data-driven model-brain mappings to constrain formal models of cognition. *PLoS ONE* 10:e0119673. doi: 10.1371/journal.pone.0119673

Bowers, J. S., and Davis, C. J. (2012). Bayesian just-so stories in psychology and neuroscience. *Psychol. Bull.* 138, 389–414. doi: 10.1037/a0026450

Clark, L., Cools, R., and Robbins, T. W. (2004). The neuropsychology of ventral prefrontal cortex: decision-making and reversal learning. *Brain Cogn.* 55, 41–53. doi: 10.1016/S0278-2626(03)00284-7

Dimov, C. M., Marewski, J. N., and Schooler, L. J. (2013). "Constraining ACT-R models of decision strategies: an experimental paradigm," in *Cooperative Minds: Social Interaction and Group Dynamics. Proceedings of the 35th Annual Conference of the Cognitive Science Society*, eds M. Knauff, M. Pauen, N. Sebanz, and I. Wachsmuth (Austin, TX: Cognitive Science Society), 2201–2206.

Edwards, W. (1962). Dynamic decision theory and probabilistic information processing. *Hum. Factors* 4, 59–73. doi: 10.1177/001872086200400201

Erickson, M. A., and Kruschke, J. K. (1998). Rules and exemplars in category learning. *J. Exp. Psychol.* 127, 107–140. doi: 10.1037/0096-3445.127.2.107

Forstmann, B. U., Wagenmakers, E.-J., Eichele, T., Brown, S., and Serences, J. T. (2011). Reciprocal relations between cognitive neuroscience and formal cognitive models: opposites attract? *Trends Cogn. Sci.* 15, 272–279. doi: 10.1016/j.tics.2011.04.002

Fum, D., and Stocco, A. (2003). "Instance vs. rule-based learning in controlling a dynamic system," in *Proceedings of the Fifth International Conference on Cognitive Modelling* (Bamberg: Universitaets-Verlag Bamberg), 105–110.

Gigerenzer, G., and Brighton, H. (2009). Homo heuristicus: why biased minds make better inferences. *Top. Cogn. Sci.* 1, 107–143. doi: 10.1111/j.1756-8765.2008.01006.x

Gonzalez, C. (2017). "Decision making: a cognitive science perspective," in *The Oxford Handbook of Cognitive Science (Vol. 1)*. ed S. E. F. Chipman (Oxford: Oxford University Press), 249–263.

Gonzalez, C., Dutt, V., Healy, A. F., Young, M. D., and Bourne, L. E. Jr. (2009). "Comparison of instance and strategy models in ACT-R," in *Proceedings of the 9th International Conference on Cognitive Modeling—ICCM2009*, eds A. Howes, D. Peebles, and R. Cooper (Manchester).

Gonzalez, C., Lerch, J. F., and Lebiere, C. (2003). Instance-based learning in dynamic decision making. *Cogn. Sci.* 27, 591–635. doi: 10.1207/s15516709cog2704_2

Halbrügge, M. (2010). Keep it simple - A case study of model development in the context of the Dynamic Stocks and Flows (DSF) task. *J. Artif. Gen. Intell.* 2, 38–51. doi: 10.2478/v10229-011-0008-2

Jarvers, C., Brosch, T., Brechmann, A., Woldeit, M. L., Schulz, A. L., Ohl, F. W., et al. (2016). Reversal learning in humans and gerbils: dynamic control network facilitates learning. *Front. Neurosci.* 10:535. doi: 10.3389/fnins.2016.00535

Johansen, M. K., and Palmeri, T. J. (2002). Are there representational shifts during category learning? *Cogn. Psychol.* 45, 482–553. doi: 10.1016/S0010-0285(02)00505-4

Knauff, M., and Wolf, A. G. (2010). Complex cognition: the science of human reasoning, problem-solving, and decision-making. *Cogn. Process.* 11, 99–102. doi: 10.1007/s10339-010-0362-z

Kohrs, C., Hrabal, D., Angenstein, N., and Brechmann, A. (2014). Delayed system response times affect immediate physiology and the dynamics of subsequent button press behavior. *Psychophysiology* 51, 1178–1184. doi: 10.1111/psyp.12253

Kruschke, J. K. (1992). ALCOVE: an exemplar-based connectionist model of category learning. *Psychol. Rev.* 99, 22–44. doi: 10.1037/0033-295X.99.1.22

Lebiere, C., Pirolli, P., Thomson, R., Paik, J., Rutledge-Taylor, M., Staszewski, J., et al. (2013). A functional model of sensemaking in a neurocognitive architecture. *Comput. Intell. Neurosci.* 2013, 1–29. doi: 10.1155/2013/921695

Lebiere, C., Wallach, D., and Taatgen, N. A. (1998). "Implicit and explicit learning in ACT-R," in *Proceedings of the Second European Conference on Cognitive Modelling,* eds F. Ritter and R. Young (Nottingham: Nottingham University Press), 183–193.

Lewandowsky, S., Palmeri, T. J., and Waldmann, M. R. (2012). Introduction to the special section on theory and data in categorization: integrating computational, behavioral, and cognitive neuroscience approaches. *J. Exp. Psychol. Learn. Mem. Cogn.* 38, 803–806. doi: 10.1037/a0028943

Li, A., and Maani, K., (2011). "Dynamic decision-making, learning and mental models," in *Proceedings of the 29th International Conference of the System Dynamics Society* (Washington, DC), 1–21.

Love, B. C., Medin, D. L., and Gureckis, T. M. (2004). SUSTAIN: a network model of category learning. *Psychol. Rev.* 111, 309–332. doi: 10.1037/0033-295X.111.2.309

Marewski, J. N., and Link, D. (2014). Strategy selection: an introduction to the modeling challenge. *Wiley Interdiscipl. Rev.* 5, 39–59. doi: 10.1002/wcs.1265

Marewski, J. N., and Mehlhorn, K. (2011). Using the ACT-R architecture to specify 39 quantitative process models of decision making. *Judgm. Decis. Mak.* 6, 439–519. doi: 11/101112/jdm101112.pdf

Nosofsky, R. M. (1984). Choice, similarity, and the context theory of classification. *J. Exp. Psychol.* 10, 104–114. doi: 10.1037/0278-7393.10.1.104

Nosofsky, R. M., Palmeri, T., and McKinley, S. (1994). Rule-plus-exception model of classification learning. *Psychol. Rev.* 101, 53–79. doi: 10.1037/0033-295X.101.1.53

Orendain, A. D. O., and Wood, S. (2012). "An account of cognitive flexibility and inflexibility for a complex dynamic task," in *Proceedings of the 11th International Conference on Cognitive Modeling* (Berlin), 49–54.

Peebles, D., and Banks, A. P. (2010). Modelling dynamic decision making with the ACT-R cognitive architecture. *J. Artif. Gen. Intell.* 2, 52–68. doi: 10.2478/v10229-011-0009-1

Prezenski, S., and Russwinkel, N. (2014). Combining cognitive ACT-R models with usability testing reveals users mental model while shopping with a smartphone application. *Int. J. Adv. Intell. Syst.* 7, 700–715.

Prezenski, S., and Russwinkel, N. (2016a). "A proposed method of matching ACT-R and EEG-Data," in *In Proceedings of the 14th International Conference on Cognitive Modeling*, eds D. Reitter, and F. E. Ritter (University Park, PA: Penn State), 249–251.

Prezenski, S., and Russwinkel, N. (2016b). "Towards a general model of repeated app usage," in *In Proceedings of the 14th International Conference on Cognitive Modeling*, eds D. Reitter and F. E. Ritter (University Park, PA: Penn State), 201–207.

Reitter, D. (2010). Metacognition and multiple strategies in a cognitive model of online control. *J. Artif. Gen. Intell.* 2, 20–37. doi: 10.2478/v10229-011-0007-3

Roll, I., Baker, R. S., Aleven, V., and Koedinger, K. R. (2004). "A metacognitive ACT-R model of students' learning strategies in Intelligent Tutoring Systems," in *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*, eds J. C. Lester, R. M. Vicari, and F. Paraguacu (Maceió), 854–856.

Russwinkel, N., Urbas, L., and Thüring, M. (2011). Predicting temporal errors in complex task environments: a computational and experimental approach. *Cogn. Syst. Res.* 12, 336–354. doi: 10.1016/j.cogsys.2010.09.003

Rutledge-Taylor, M., Lebiere, C., Thomson, R., Staszewski, J., and Anderson, J. R. (2012). "A comparison of rule-based versus exemplar-based categorization using the ACT-R architecture," in *Proceedings of 21st Annual Conference on Behavior Representation in Modeling and Simulation, BRiMS 2012* (Amelia Island, FL), 44–50.

Salatas, H., and Bourne, L. E. Jr. (1974). Learning conceptual rules: III. Processes contributing to rule difficulty. *Mem. Cogn.* 2, 549–553. doi: 10.3758/BF03196919

Salvucci, D. D. (2006). Modeling driver behavior in a cognitive architecture. *Hum. Factors* 48, 362–380. doi: 10.1518/001872006777724417

Sanborn, A. N., Griffiths, T. L., and Navarro, D. J. (2010). Rational approximations to rational models: alternative algorithms for category learning. *Psychol. Rev.* 117, 1144–1167. doi: 10.1037/a0020511

Seger, C. A., and Peterson, E. J. (2013). Categorization = decision making + generalization. *Neurosci. Biobehav. Rev.* 37, 1187–1200. doi: 10.1016/j.neubiorev.2013.03.015

Simon, H. A., and Newell, A. (1971). Human problem solving: the state of the theory in 1970. *Am. Psychol.* 26, 145–159. doi: 10.1037/h0030806

Smieszek, H., Joeres, F., and Russwinkel, N. (2015). "Workload of airport tower controllers: empirical validation of a macro-cognitive model," in *Online Journal Kognitive Systeme*, eds D. Soeffker, and A. Kluge (Duisburg: DuEPublico). doi: 10.17185/duepublico/37699

Stewart, T. C., and West, R. (2010). Testing for equivalence: a methodology for computational cognitive modeling. *J. Artif. Gen. Intell.* 2, 69–87. doi: 10.2478/v10229-011-0010-8

Sutton, R. S., and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. London: MIT Press.

Taatgen, N. A. (2001). "A model of individual differences in learning Air Traffic Control," in *Proceedings of the fourth International Conference on Cognitive Modeling*, eds E. M. Altmann, A. Cleeremans, C. D. Schunn, and W. D. Gray (Mahwah, NJ: Erlbaum), 211–216.

Taatgen, N. A., Lebiere, C., and Anderson, J. R. (2006). "Modeling Paradigms in ACT-R," in *Cognition and Multi-Agent Interaction: From Cognitive Modeling*

to Social Simulation, ed R. Sun (Cambridge: Cambridge University Press), 29–52.

Thomson, R., Lebiere, C., Anderson, J. R., and Staszewski, J. (2015). A general instance-based learning framework for studying intuitive decision-making in a cognitive architecture. *J. Appl. Res. Mem. Cogn.* 4, 180–190. doi: 10.1016/j.jarmac.2014.06.002

Wills, A. J., and Pothos, E. M. (2012). On the adequacy of current empirical evaluations of formal models of categorization. *Psychol. Bull.* 138, 102–125. doi: 10.1037/a0025715

Wolff, S., and Brechmann, A. (2012). "MOTI: a motivational prosody corpus for speech-based tutorial systems," in *Proceedings of the 10th ITG Conference on Speech Communication* (Berlin: IEEE), 1–4.

Wolff, S., and Brechmann, A. (2015). Carrot and Stick 2.0: the benefits of natural and motivational prosody in computer-assisted learning. *Comput. Hum. Behav.* 43, 76–84. doi: 10.1016/j.chb.2014.10.015

Wong, T. J., Cokely, E. T., and Schooler, L. J. (2010). "An online database of ACT-R parameters: Towards a transparent community-based approach to model development," in *Proceedings of the 10th International Conference on Cognitive Modeling*, eds D. D. Salvucci, and G. Gunzelmann (Philadelphia, PA: Drexel University), 282–286.

Zeller, C., and Schmid, U. (2016). "Rule learning from incremental presentation of training examples: Reanalysis of a categorization experiment," in *Proceedings of the 13th Biannual Conference of the German Cognitive Science Society*. (Bremen), 39–42.

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Copyright © 2017 Prezenski, Brechmann, Wolff and Russwinkel. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) or licensor are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.

5 Paper No. 4

Prezenski, S. (2017). Implementing mental model updating in ACT-R. In *15th International Conference on Cognitive Modeling*, pages 121–127.

In M. van Vugt, A. Banks & W. Kennedy, Proceedings of the 15th International Conference on Cognitive Modeling (ICCM 2017). Coventry, United Kingdom.

Implementing Mental Model Updating in ACT-R

Sabine Prezenski (sabine.prezenski@tu-berlin.de)
Cognitive Modeling in Dynamic Human-Machine Systems, Department of Psychology and Ergonomics, Technical University Berlin, Berlin, Germany

Abstract

This paper demonstrates how mental models and updates of mental models due to system changes can be modeled with the cognitive architecture ACT-R using explicit mechanisms. The mental model building and updating is modeled with a representation chunk and a control chunk. The representation chunk holds the strategy, the expected outcome and an evaluation mechanism of the strategy. The control chunk holds information over environmental conditions and the learning history. This modeling approach was developed and tested for smartphone application tasks and then implemented in a dynamic decision-making task investigating strategy development with complex stimuli. The later task used different multi-feature auditory stimuli material. The modeling approach explained data of participants in the smartphone studies very well and met the trends found in the dynamic decision-making task.

Keywords: *ACT-R; mental model updating; general model; learning; dynamic decision-making, applied*

Introduction and Theory

Our behavior is guided by our internal representation of tasks and situations (Norman, 1983). However, such representations or mental models are not static but they change and are adjusted, due to experience gain, environmental changes etc. Understanding how people update or adapt their mental model is relevant in many fields, from updates in technical systems to real-life tasks that require strategy learning and dynamic decision-making. The later investigates serial decisions. Such decisions are dependent on previous decisions and are made under time constraints in a changing environment (Edwards,1962; Gonzalez, 2014). Dynamic decision-making can be seen as a continuous cycle of mental model updating, made up of conceptualization – experimentation – reflection (Li and Maani, 2011). In the conceptualization phase a general concept of the situation is obtained. Hereby, the outcome of potential decisions is mentally simulated. The current situation is compared to information in the decision maker's mental model.

New information obtained from the environment is integrated to develop a set of decisions. In the experimentation phase, these decisions are tested. The outcome (e.g. feedback) of the experimentation phase is evaluated on in the reflection phase. If the expected outcome is achieved (e.g. positive feedback), initial decisions are kept. If, the outcome is unexpected (e.g. negative feedback) the mental model of the decision maker is updated. Thus, alternatives are sought for, such as new sources of information.

In real-life settings adaptations of mental representations of users are required in many different circumstances. Typical situations which require a user to update his or her mental model are situations leading to errors, due to incomplete or wrong representations. For example, if a user repeatedly fails to install the connection settings for the universities Wi-Fi, he or she needs to adjust his or her mental model, about how to install Wi-Fi on phones. Situations in which changes to the system (due to aspects outside of the person) make the current (in the past correct mental model) inadequate also require adjustment to the user´s mental representation. Examples for the later are a) that due to system-upgrades a new version of an application is launched or b) that past-successful strategies used in decision-making tasks are misleading due to environmental changes.

Nevertheless, the core mechanisms of mental model adaptation should be the same for both situations. This paper demonstrates how mental model build-up and adjustment due to environmental can be addressed using the cognitive architecture ACT-R.

Cognitive architectures allow computationally implementing theories about human cognition in a broad spectrum. The cognitive architecture ACT-R has been applied in many applied domains such as smartphone usage (Prezenski, Bruechner and Russwinkel, 2017) or air-traffic control (Raufaste, 2006) but also in more ground-based research (Halbrügge and Russwinkel, 2016).

ACT-R has symbolic and subsymbolic parts which together produce the modeled behavior. The symbolic parts are chunks, production rules, buffers and modules. The modules resemble the architecture of the human brain. are specified, each of them handles different types of information (chunks). The chunks have slots, they store the smallest pieces of information. The different modules interact through their corresponding buffers. For example, visual information is processed by the visual module and its two buffers. Motor movement is controlled by the manual module and buffer. The declarative module is the long-term memory of ACT-R. Information for this module is retrieved via the retrieval buffer. The imaginal module and buffer are important for learning new information and can be seen as ACT-Rs working memory. Model steering is controlled by the goal module and buffer. The procedural module connects the modules and selects (production-) rules. These production rules are the core part of an ACT-R model- they govern the model behavior. Production-rules can be selected and executed, if buffer states are met. The selected production-rule can then change the states of the modules. An example of a subsymbolic process in ACT-R is the activation level. Thus, if a production requests a chunk and more than one

chunk matches this request, this results in the selection of the chunk with the highest activation level. The activation level of a chunk is composed of how often it was used when it was last accessed and how long ago the chunk was created. There are many more subsymbolic processes built into the architecture of ACT-R (e.g. blending, partial matching). Subsymbolic processes are used for modeling implicit learning, e.g. usage of activation mechanism to model information that is well- known can be better retrieved than information that is less well-known.

However, learning (especially in early phases) is also an explicit process (Tenison, Fincham & Anderson, 2016). Thus, the learner is deliberately processing information and deciding what to do next stepwise. This can be modeled by building of new chunks via specific production rules. They can represent the strategies given to a model by the modeler. For an overview and a discussion of implicit and explicit mechanisms in ACT-R in context of intuitive decision-making see Thomson et al (2016).

Explicit mechanisms seem especially important in mental model updating. According to Li and Maani (2011) mental model updating occurs in the reflection phase when negative feedback (unexpected outcome) is observed. Then new sources of information need to be sought for. Such processes require the modeler to use explicit mechanism.

Cognitive models are useful to make precise predictions about theories on human cognition. Models build with cognitive architectures moreover allow precise prediction about behavior influenced by different cognitive processes. They try to capture cognition as a whole. Enough effort, modeling skills and free parameters make it possible to precisely match behavior of participants with models. But for models to be useful. they should be able to predict data in other situations as well. Therefore, modelers should avoid using many specifications to match the data, but attempt to use broader concepts. A successful example for this are models using instance based learning (Gonzalez, 2005). Instance based learning is used to model intuitive decision-making (Thomson et al, 2016). Hereby problem-solving instances are stored in declarative memory and decisions are made by retrieving these instances. The activation mechanism of ACT-R is used to determine which instances are retrieved. However, in early phases of learning and when previously-learned instances become invalid (due to changes in the environment) explicit mechanisms are needed. Such explicit mechanism should be constructed in a general manner and thus be applicable in a variety of tasks.

Aim and Previous Work

The aim of this paper is to show how the same modeling approaches and mechanisms relevant for mental model building and updating can be used in very different applied tasks. Both tasks have in common that they require the participants to explicitly a) learn and b) notice changes and thus to readjust their mental model. Otherwise the tasks are different, thus two ACT-R models are used. Nevertheless, this paper resembles a general modeling approach, since it demonstrates how the core model mechanisms developed in one study (Prezenski and Russwinkel, 2016) are applied to a different study (Prezenski et al. submitted).

The first study investigated a search-and select task with two different smartphone applications. One application allows users to select items to assemble a shopping list and the other to select search-criteria for real-estates. Initial and repeated usage of these applications was investigated. Furthermore, users' adaptation to changes in the applications due to updates influencing the menu-structure (shopping application) and adaptations (real-estate application) was studied.

The second study examined strategy learning in an auditory dynamic decision-making task. In this task, multi-feature sounds were repeatedly presented to the participants. The task was to decide if the presented sound was a target or a non-target. To solve this task a combination of features had to be chosen as targets. The relevance of feature combinations had to be learned from the feedback given in the experiment. In the middle of the task a uniformed switch of targets and non-targets occurred. The task can be seen as an example for dynamic decision-making, because it requires participants to repeatedly make decisions on whether or not a stimulus is a target or a non-target and learn (e.g. improve their decisions) from feedback of the previous decisions. The decisions have to be made under time-constraints. Other feature-combinations become targets at a given point in the experiment due to changes in the environment.

Methods

The methods section of this paper is structured in the following way: First, the core mechanisms for mental model building and mental model updating are described. Second, the results of the first study on smartphone interaction and the implementation of the mechanisms in the first study is summarized. Third, the second study on dynamic decision-making and the transfer and implementation of the mechanisms is explained.

Mechanisms

Mental model building The core part of the mental model (or abstract representation) of a situation, strategy or solution is stored in the *representation chunk* (see figure 1). The slots of this *representation chunk* hold information on the strategy and the expected outcome of applying this strategy. The information on the strategy consists of a representation of the situation and the action.

During mental model building (conceptualisation phase) the *representation chunk* needs to be placed in the imaginal buffer. Only here ACT-R allows chunks to be altered. In the experimentation phase, the expected (or predicted) outcome of this *representation chunk* is tested and then reflected on (reflection phase).

In the reflection phase, mental models can either be revised or strengthened. On the one hand, revision is required, if the outcome is different from what is expected. On the other

hand, if the outcome is as expected mechanisms for strengthening the mental model are needed. Here fore, explicit mechanisms are used; namely a slot that notes if a strategy is correct and other slots that keep track (until a threshold) how often a strategy was correct. Other implicit ACT-R strengthening mechanisms are also used, such as that frequently used chunks, are retrieved more often and have a higher activation and this again makes them more likely to be retrieved.

Furthermore, as learning evolves, mental models often become more specific (Gonzalez and Lebiere, 2005). For example, a user experienced in installing Wi-Fi on phones for university networks might have two or more mental models depending on the different types of phones the user installed Wi-Fi for university network in the past. Thus, learners may know that a solution is only applicable for a specific situation (e.g. for one version of an application) such knowledge should also be stored in the *representation chunk*.

Besides a representation of the situation, expected outcome and observed success (core part of the mental model), the building of such a model also requires some form of control over the environmental conditions and the learning history. Such information is stored in the *control chunk* (see Figure 1). This chunk is kept in the goal buffer.

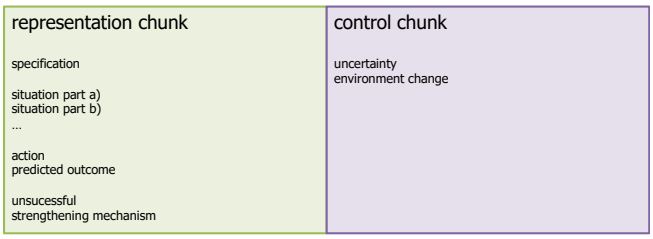

Figure 1: Main chunks and slots required for mental model building and updating

Mental Model Updating In this paper *mental model updating* refers to the modification of an established *representation chunk,* e.g. a strategy that has been successful in the past.

The mechanism, illustrated in Figure 2 works the following way: First, the strategy of the suggested action of the *representation chunk* leads to unexpected outcome. This unexpected outcome is then encoded in a slot of the *control chunk.* This slot represents the uncertainty of the current strategy that something may have changed. The *representation chunk* is nevertheless kept as mental model and tested again. If following the strategy proposed by the *representation chunk* produces unexpected outcome again, this is noted in a slot of the control chunk. This represents that a change has occurred and that a different strategy needs to be built up from now on.

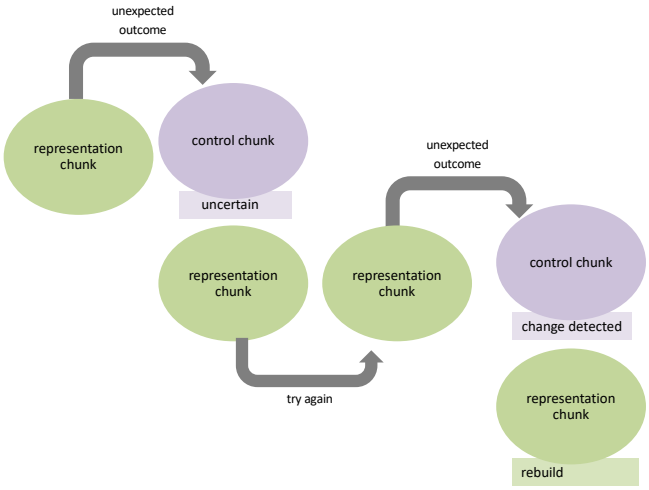

Figure 2: Mental model updating process, governed by specified production rules.

Studies

In the following section the two studies, first the smartphone study and then the decision-making study are presented. Both sections first provide an overview of the tasks and material and then focus on how the core model mechanism from above are implemented respectively.

1) Smartphone Application Study These mechanisms were implemented in a model of users search and select behavior via navigating two smartphone applications. This study has been presented in greater detail elsewhere (Prezenski and Russwinkel, 2016). Thus, only a brief short summary of the applications (material), task, participants, study-design and the implementation of the mental model building and updating mechanism is given.

Material/Applications Two self-designed Android applications (a shopping list application and a real-estate application) each with two different versions were used. The shopping list applications differed in overall menu-depth (three layers vs. four layers). The real-estate application adapted to prior selection, this affected the menu-depth and the positions of some items. These applications were installed on Google Nexus 4.

They are hierarchical-list style applications that support search and select task. Targets and subtargets are spread out over different pages of the applications. See Figure 3 for an impression of the applications.

Task In the shopping-list application participants had to search and select shopping items via navigating through different pages of the application. The participants had to search and select targets (shopping items) via selecting subtargets (e.g. categories, shops) placed on different layers of the application.

Figure 3: Application layout, reprinted from (Prezenski et al, 2017, p. 170)

In the real-estate application participants had to search and select search criteria for real-estates via selecting different subcategories which were again placed on different layers of the application.

Study-Design The design in the four substudies was similar. In the shopping-list study the participants were required to search for the same nine items for four times. In the first two blocks, they used one version of the application (either three or four layers) in the last two blocks the version "updated" and they had to use the other version. They were not informed about the occurrence of a version switch. In the real-estate study the participants were required to search for either a house or an apartment with six or seven other criteria (e.g. specific size, rent) and after two blocks they had to search for the other one twice (e.g. those who searched for a house twice had to search for an apartment and vice versa). Depending on the pre-selection of house or apartment the position and the menu-depth of other search-criteria could differ (e.g. if house was pre-selected the search-criteria 60qm was positioned higher in the list then if apartment was pre-selected).

The dependent variable is the average target selection time per block. Each block consists of the selection of all items (eight items per block for the shopping list studies and six or seven items for the real-estate studies). Thus, four blocks per study existed.

The four sub studies were conducted with student participants. 10 participants took part in the real-estate study where apartment was selected first, and 12 in the one where house was selected first. 17 took part in the shopping list study that used the three-layer version first and 12 in the one that used the four layer version first.

Model implementation The apps were implemented in Lisp and the model was run with ACT-R 7.1. 10 model runs per study were implemented[1]. In the following the modeling principles are summarized. This section focuses on how the mechanism for mental model building and updating are implemented. Other supplemental mechanisms will be briefly introduced in the following section, as well.

Mental model building in smartphone studies The task is to find a target via navigating through different layers of the application. In the beginning of the task, a mental representation of the application is not inherent to the model. Thus, navigation of the application if achieved using *knowledge of the world chunks*. These are made up of associations between different words (e.g. the target-word *alcohol-free beer* is related to the word *bottle shop)*. Thus, each item of the application is read and a request for a *knowledge of the world chunk* linking the current processed item and the target, is made. If such a *knowledge of the world chunk* can be found the item is selected, otherwise the next item is processed. The *knowledge of the world chunk* is used to build up a *representation chunk* in the imaginal buffer. If *a representation chunk* is available, it will be used to navigate to the target. This chunk contains the path leading to the target, e.g. which item needs to be selected in order to reach the target. Thus, the items are the *situation* and the target is the *expected outcome*. There is no strengthening mechanism used in this model. But a *specification* mechanism that clarifies when a *representation chunk* is adequate to be used, e.g. use *representation chunk* for a menu-depth of three. However, this is part of the *control chunk* held in the goal buffer. The control chunk of this model also holds information about *uncertainty* of the current strategy (or path chunk) and on *detected changes* (e.g. updates).

Mental model updates in smartphone studies After the second block a change (either a version update or an adaptation due to prior selection) is made to the application. Thus, the established *representation chunk*s will not lead to the expected outcome anymore. So, targets, or subtargets cannot be found with these *representation chunks*. This *uncertainty* is noted in the *control chunk*. Another attempt to find the target using this *representation chunk* is made. If it again does not lead to the target, then a change in the environment is noted. Thus, a strategy change is initiated and the *knowledge-of the world* chunks are used to build a different *representation chunk*. For the next target, a new *representation chunk* is built directly. If the model is required to search for a target with a new version a second time it can retrieve the correct *representation chunk* using the specification (see Figure 4).

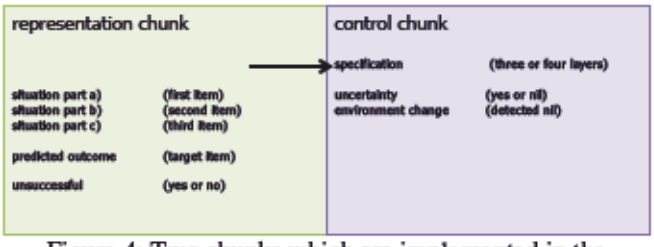

Figure 4: Two chunks which are implemented in the Smartphone application study

2) Dynamic Decision-Making Study Mental model building and updating should be the same process even in very different tasks. Thus, it should be modeled in the same way

[1] The data of the model did not show much variance. Thus, additional model runs were not necessary.

as other tasks that require mental model building and updating. Such another task was investigated in the second study. It required the participants to make sense of multi-feature auditory stimuli. The experiment and the model are presented in more detail in Prezenski et al (submitted). In the following section, a short overview of material, task, participants will be given. Followed by a more description of how the mental model building and updating mechanisms were implemented.

The stimuli were 160 different tones. These were made up of a combination of different category features, namely duration (short vs. long), direction of frequency modulation (rising vs. falling) and intensity (quiet vs. loud) and frequency (high vs. low). Tones which included a combination of specific category feature (e.g. loud and falling) were the target stimuli (25%), while the other where the non-targets (75%). Different category-feature combinations were the target for different participants.

In each trial (there were 240 altogether), a tone was presented to the participant and he or she was required to press one of two buttons to classify if the tone was a target or a non-target. After the button-press auditory feedback was presented ("wrong" or "correct") and then after a pseudo-random time of six, eight or ten seconds the next trial began. After 120 trials, there was a switch of the button allocations, the participants were not informed about this. There were four different randomizations of the experiment; each had different category features as targets.

The dependent variable was the average percentage of correct responses per block. 20 trials were always grouped together as a block. Thus, the experiment consisted of 12 blocks.

55 student participants took part in the experiment.

Model implementation The experiment for the model was implemented in Lisp using the new-other-sound command for the tones and using 16 tones (all possible combinations of the category-feature) pairs as auditory stimuli. The model was written with ACT-R 7.1.

Mental model building The task is to find the correct strategy to classify tones into targets and non-targets. The fact, that a combination of feature-value pairs is the correct solution is unknown to model. Thus, first a single feature-value-pair strategy is used and this is changed to a two feature-value-pair strategy in the course of the experiment.

Two main chunks are part of the model (see Figure 5). The first is a *representation chunk* which holds the current strategy in the imaginal buffer. The second is a *control chunk* in the goal buffer. In the beginning of a trial a tone is heard and a decision has to be made if the tone is a target or not.

The *representation chunk* holds the current strategy the in the imaginal buffer. It contains information about the relevant feature(s) and value(s) (e.g., *the sound is quiet* or the *sound is quiet and its frequency range is high*) and the proposed response (0 or 1). This can be seen as the *situation* and the *predicted action*. Furthermore, the *specification slot* of the *representation chunk* holds information on the degree of complexity of the strategy (e.g. one or two-feature strategy). An evaluation mechanism is part of the representation chunk as well. The evaluation's result determines if a strategy was unsuccessful and keeps record of how many times a strategy was successful. It marks if the first attempt to use this strategy is successful. Furthermore, the number of successful strategy uses are counted until a certain value is reached. This is meant to reflect the subjective feeling that a strategy was useful often. If a strategy was useful often, then is well-established. The same representation chunk is held in the imaginal buffer as long as feedback is positive. If feedback is negative a different *representation chunk* will be retrieved from memory.

The control chunk holds information on the *uncertainty* about a current strategy and on detected *environmental changes*.

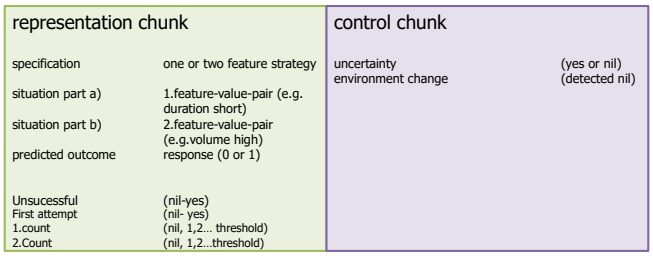

Figure 5: Two chunks which are implemented in the dynamic decision-making study

Mental model updating If an established strategy (in other words representation chunk) causes unexpected negative feedback uncertainty about this current strategy is noted in the control chunk. Nevertheless, this strategy is used a second time. If again unexpected outcome occurs, the strategy will be changed using the mechanism seen in figure 2. In the course of the experiment, this can occur in two different situations. The first situation is, when a one-feature strategy (e.g. volume loudness is 1) is successful often but after repeated unexpected outcome (negative feedback) it is changed into a two-feature strategy. Thereby, the first feature-value pair (volume loudness is 1) is kept as part of the strategy and complemented by another feature-value pair.

The second situation is, after the environment changed when a past establish two-feature strategy repeatedly leads to unexpected outcome. Then different two-feature strategies are sought for.

To sum up, both the smartphone and the decision task implemented mental model building and updating in the same way. Mental model building and updating is modeled using a representation and a control chunk. The followed strategy is held in the representation chunk. This chunk is retrieved from declarative memory and altered using working memory. Information over environmental conditions and the learning history is encoded in the control chunk which is held in the goal buffer. In both models, well-established strategies are not discarded directly in case of unexpected outcome, but

tested once more. If they lead to failure again, they are partly revised and rebuilt.

However, the type of behavioral data that the models´ performance was compared to, differed: average item selection time was used for the smartphone studies and percentage of correct responses for the dynamic decision-making experiment.

Results

The results section briefly presents the results of the empirical data together with the modeled data. The results of the smartphone studies are presented in greater detail in Prezenski & Russwinkel, 2016. The results of the decision-making experiment in Prezenski et al (submitted).

Study 1: Smartphone Interaction

In all smartphone sub studies, the model captured the trends found in the empirical data. The trends show a decrease in item selection time from the first to the second block in all four studies. An increase from the second to the third block found in three studies (both real-estate app studies and the shopping-list app, that added an additional layer (shopping 3-4), see Figure 6). In the other shopping-list app study the model also captured the decrease found between the second and the third block. Finally, in all four studies there is a decrease in the mean item-selection time this was again captured by the model.

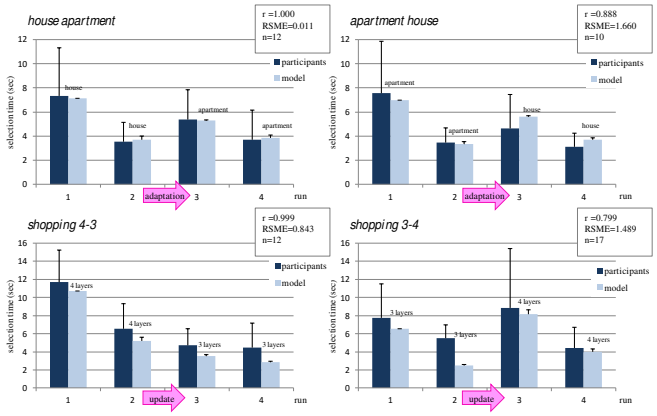

Figure 6: Mean target selection time, reprinted from (Prezenski and Russwinkel, 2016, p. 205)

In the other shopping-list app study the model also captured the decrease found between the second and the third block. Finally, in all four studies there is a decrease in the mean item-selection time this was again captured by the model.

To sum up, the model captured learning and relearning (update detection and new learning). It matched the participant's behavior in mean item selection time very well for all four studies ($r^2 > 0.799$).

Study 2: Dynamic Decision-Making

In this study, the empirical data show an increase in the proportion of the correct response from the first to the sixth block (see Figure 7). This is followed by a drop in correct responses in the seventh block, which is pursued by a performance increase until the twelfth block. The model resembles these trends. The overall r^2 is at 0.672. Nevertheless, the descriptive data indicates that the participants have almost "recovered" from the change in the eighth block, while the model takes longer.

In summary, the model captured the empirical data well; an improvement in performance in the first half of the experiment, the performance drop after the strategy changed and the recovering in performance in the second half of the experiment.

The overall fit of the dynamic decision-making task is not as precise as the fit of the model in the smartphone studies. One explanation hereof is that more measurement points in the decision-making study (12) then in the smartphone study (4) make it less likely to achieve a good fit.

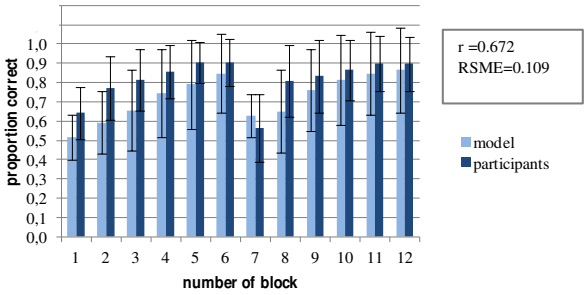

Figure 7: Proportions of correct responses of the model and participants

Another explanation could be that the participants need less long to find an adequate strategy (adequate update of their mental model) after the switch, because they tried the strategy of pressing the other button for the same strategy. Such an explicit strategy was not modeled to keep the model simple and more general.

Discussion

Two very different real-life tasks were modeled with ACT-R using the same explicit mechanisms for mental model building and updating.

The building process of a mental model involves implementing a preliminary version of a mental model and a subsequent testing of this model or strategy. If the strategy performs as expected, it is strengthened, if not it will be updated with a different strategy. Established strategies are not changed immediately in case of unexpected outcome but tested another time before they are changed. Changes to the strategy are gradual; a strategy is not completely discarded; some aspects are kept.

Explicit mechanisms were used, because the changes investigated are registered by the humans. Such distinct noticed changes lead to changes in behavior. Examples for

these kinds of changes in real-life settings are software-updates or changes in environmental conditions during outdoor activity (e.g. sudden rain while climbing).

The scope of the presented model mechanisms is not mental model updating during highly automated processes for very skilled users. However, the presented mechanisms can reproduce initial learning, usage and relearning of strategies. Implicit mechanisms are nevertheless part of the models. For example, the previous activation of chunks, as well as if a chunk has been retrieved recently, influence the course of the model.

Modeling the change in strategy and the relatively fast occurring relearning of the participants using solely implicit mechanism with ACT-R is a challenge.

From a cognitive psychological point of view, explicit mechanisms are superior to data driven machine learning approaches, such as deep neural networks because they provide explanations of the underlying mechanisms of participants. Knowledge about explicit strategies of participants is valuable for the design and testing of interactive systems because such knowledge does not only provide summative performance metrics of an interface but also gives hints towards the causes of usability shortcomings and possible solutions.

The examples that have been demonstrated assume specific mental models and provide mechanisms on how such mental models might be updated in human cognition. There is of course no guarantee that such strategies and mental models closely resemble the real strategies, this is not at last grounded on the fact that the human brain does not employ explicit symbol manipulation mechanisms, such as the explained process of building and updating of mental models does. However, the studies that were presented here show that such models provide a reasonable approximation of participant performance.

Potential next steps are investigating the proposed mental model building and switching strategies empirical, with studies targeting these mechanisms.

Summary

This paper demonstrates how mental models updating due to system changes can be modeled using explicit mechanism. This explicit mental model updating mechanism was first implemented in a model of smartphone application usage (Prezenski & Russwinkel, 2016). The mechanism was then applied to a dynamic decision-making task, where participants were presented with different multi-feature auditory stimuli material (Prezenski, Brechmann, Wolff & Russwinkel, submitted). While the model explained data of participants in the smartphone studies very well, the data in the dynamic decision-making task was not explained to such extend.

References

Anderson, J. R. (2007). *How Can the Human Mind Occur in the Physical Universe?* New York: Oxford University Press.

Edwards, W. (1962). Dynamic Decision Theory and Probabilistic Information Processing. *Human Factors, 4*, 59–73.

Gonzalez, C., Lerch, J. F., & Lebiere, C. (2003). Instance-based learning in dynamic decision making. *Cognitive Science, 27*(4), 591–635.

Gonzalez, C., & Lebiere, C. (2005). Instance-based cognitive models of decision-making.

Gonzalez, C. (2014). Decision Making: A Cognitive Science Perspective. In S. E. F. Chipman (Ed.), *The Oxford Handbook of Cognitive Science* (Vol. 1). Oxford University Press.

Halbrügge, M., & Russwinkel, N. (2016). The sum of two models: How a composite model explains unexpected user behavior in a dual-task scenario. *D. Reitter & F. E. Ritter (eds.), Proceedings of the 14th international conference on cognitive modeling* (pp. 137–143). University Park, PA: Penn State.

Norman, D. A. (1983). Some observations on mental models. *Mental models*, *7*(112), 7-14.

Prezenski, S., Bruechner, D., Russwinkel, N. (2017). Predictive Cognitive Modelling of Applications. In *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 2: HUCAPP, (VISIGRAPP 2017)* (pp. 165-171).

Prezenski, S., Brechmann, A., Wolff, S. and Russwinkel, N. (submitted). A Cognitive Modeling Approach to Strategy Formation in Dynamic Decision Making. *Frontiers in Psychology*.

Prezenski, S. and Russwinkel, N. (2016). Towards a general model of repeated app usage. *D. Reitter & F. E. Ritter (Eds.)*, *Proceedings of the 14th International Conference on Cognitive Modeling* (pp. 201–207). University Park, PA: Penn State.

Raufaste, E. (2006). Air traffic control in ACT-R: A computational model of conflict detection between planes. In *Proceedings of the International Conference on Human-Computer Interaction in Aeronautics (HCI - Aero'06)*, (pp. 258–259).

Thomson, R., Lebiere, C., Anderson, J. R., & Staszewski, J. (2015). A general instance-based learning framework for studying intuitive decision-making in a cognitive architecture. *Journal of Applied Research in Memory and Cognition*, *4*(3), 180-190.

Tenison, C., Fincham, J. M., & Anderson, J. R. (2016). Phases of learning: How skill acquisition impacts cognitive processing. *Cognitive psychology*, *87*, 1-28.

Discussion

The combined discussion of this dissertation is structured in the following manner; first a short summary of each of the four papers is given. Then, the limitations of the three main studies presented in the four papers are analyzed in terms of design and methodologies, empirical data-analysis, model mechanisms and general weaknesses. This is followed by the overall discussion concerning main research questions aligned to this dissertation. Finally, an outlook is given.

Summary of papers

- The main aim of paper no. 1 (Prezenski & Russwinkel, 2014) was to outline how and to what extend the usability of smartphone applications can be addressed with an ACT-R user-modeling approach. Here for, two empirical studies of a repeated search and select task with a smartphone app provided information on usability aspects linked to mental model building and updating. Finally, a model concept was developed.

- This model concept was implemented in paper no 2. (Prezenski & Russwinkel, 2016) with ACT-R and the descriptive and predictive power of the model was tested using data from the main Shopping App study and the Real-estate App study. The model was matched to one condition of the Real-estate App study and could predict the data of the other three conditions (Shopping App study and Real-estate App study). The model captured the trends and absolute values in terms of average data points for eight items per block. The results reveal that the model has high predictive power. It can predict data from different empirical datasets and different apps, with a similar layout but varying menudepth.

- The aim of paper no. 3 (Prezenski et al, 2017) was to investigate the prescriptive capacity of the core mechanisms for mental model building and updating. Thus, to test if the mechanisms developed for modeling user behavior with apps can be applied for a completely different task, the model concept was mapped to a dynamic decision-making task. This task required participants to decipher complex auditory stimuli and assign categories (Wolff & Brechmann, 2015). A new ACT-R model, using the core mechanism developed in paper no. 1 and 2., was developed and the model and empirical data were compared. The results indicate that the model concept can be matched to the dynamic decision-making task. The model matches the trends (learning and relearning) found in the data to a large extent and can capture individual differences.

- Paper no. 4 (Prezenski, 2017) provides a comprehensive analysis and discussion of the model approach from papers no. 1 to no. 3. Hereby, it focuses on explaining the details of how mental model building and updating in different interactive tasks can be modeled with ACT-R in the same manner.

Limitations /Points of Critiques

First Study: Shopping App study (Paper No. 1 and Paper No. 2)

DESIGN/METHODOLOGIES

An important aim of the Shopping App study was to show, that usability tests with cognitive models are realizable. As a study subject for usability, smartphone apps were chosen. To demonstrate that usability tests, based on cognitive models are feasible in an applied setting, a real app or at least an app with a realistic appearance which supports an actual task was needed. To meet this demand, the Shopping App was designed and implemented as an Android app. Commercial shopping apps, such as the “ImmoScout24” app or “Ebay-Kleinanzeigen” app served as a role-model for the Shopping App; regarding menu-structure, item-selection and content it was built in the same manner as these apps. Nevertheless, its design is very simple and lacking many visual features (e.g. background colors, frames) of commercial apps. However, the general structure and selection mechanism used in the study is typical for such apps. To enhance the realistic impression, the labeling of categories, subcategories and items were taken from these commercial apps.

The downside of using an approach which aims at realistic settings is, that some aspects relevant for experimental control are not respected. As a counter measure, when designing the study, it was attempted to balance possible error variables, such as item position, word

length and word frequency in common language. But, balancing these aspects was not priority and thus, it is possible that effects in the data exist which are due to such error variables. Nevertheless, such effects will also be found if actual apps are analyzed and the models should be robust to possible side effects and able to map main effects of learning and relearning.

Regarding the aim of modeling real apps, it is important to use a task, which is close to an actual task in a setting that is not artificial. So, the participants, were told to imagine that they were entering items for shopping into the Shopping App and that these items were dictated to them by a spouse. Something that is common in a household setting. Since learning effects were relevant for the research question, participants had to select the same items repeatedly, which makes the task somewhat unnatural. To control, if this task and the resulting learning effect is not too artificial, the second Shopping App was designed. Here, the participants could use the full capacity of the app. Again, learning effects were found; an indication that the app is learnable.

Now, besides investigating learning, it was of interest to develop modeling mechanism that can handle changes occurring to the application. Updates represent such changes. As a first step, updates that effect the entire menu-depth were looked at (adding a layer or deleting a layer). Again, one can argue that an update which influences all layers in a menu and not only some, is somewhat artificial. However, it is a starting point and more realistic updates should be looked up once general mechanism are developed.

EMPIRICAL DATA

The empirical data presented in this paper is the averaged item selection time and the development of the item selection time over different blocks. Thus, the time it takes for the participant from the starting page until the target item is selected. This is averaged over the items per block. Because, the aim was to look at learning and mental modeling effects on average, the paper does not report more fine grain data, such as the selection time for specific items or information on errors. So, the modeling concepts derived from this data will not be able to provide information that requires greater detail than average item selection times or trends over blocks.

MODEL-MECHANISMS

In short, three chunks are part of the model introduced first in paper no 1. First, a goal chunk which contains all information necessary for model control, second, a declarative chunk that contains knowledge of the world on which items and categories belong together. This chunk is used if the model has no specific knowledge about how to find a target. This specific knowledge is built up in the third chunk which contains the item belonging to the target in a path chunk which is built in the imaginal buffer.

The length of the path chunk is a weak point of this modeling approach. Further studies need to clarify until how many items a singular path chunk is enough and from what number of items on, more than one path chunk should be used.

Another shortcoming is the implementation of menu search for a specific item in the model. Namely, the assumption that each item is visually processed, is too simple. A more precise implementation of item search in menus is found in (Bailly et al, 2014). Future work should change the model in this regard.

Finally, the modeling approach introduced in the first study might be a bit overfitted to the exact specification of the Shopping App and the exact study design. Therefore, it was tested, if it can be applied to a different app.

Second Study: Real-estate App study (Paper No. 2)

The extent to that the second app is different is debatable. The impression of the Real-estate App is very similar to the Shopping App. Furthermore, both apps have a hierarchical list style build-up and consist of at least 3 layers. Nevertheless, the content of the apps is different, as is the fact that menu-depth varies in the Real-estate App and does not in the Shopping App. Finally, the way changes are implemented differs between both apps. In the first app, an “update” effects the menudepth of all items; in the second the app “adapts” to pre-selection. Thus, for some list-items (but not for all); the menudepth changes as does the item position; this means that the menu-layers vary in this app. In my opinion, the two

apps are indeed different apps not because of the different study content, but because of the varying menu-depth and the adaptive character of one app compared to the static one of the other.

DATA (EMPIRICAL/ MODEL)

The data reported consists of averaged item selection times; it is lacking greater detail such as the selection time per item or even more details like the exact time for selecting subcategories on each page. Since information on these aspects was not obtained from the empirical data, the model approach cannot account for them.

Another critical aspect is that the participants used their fingers to interact with the app; while the model used the ACT-R mouse commands. So, specific touch aspects that influence the empirical data cannot be captured by the model.

Also, the scope of the empirical data was too small to obtain information on errors. Here for, a higher number of participants would have been required.

The first trials in the empirical data, are noisier; assumingly, the participants use them for orientation in the task. Again, this cannot be predicted by the model.

Model fitting: In total, there are four different data sets relevant for this model; two from the Shopping App study and two from the Real-estate App study. The version of the model presented in paper no. 2 was fit to one of the four datasets (Real-estate; house-apartment) and it can predict the three others. The model ran with these settings for the other three groups. This may seem a bit in conflict with the described and predicted criteria. However, I first had a model that could handle the Shopping App, but I never analyzed its data in comparison to the empirical data. To see if this model can also run on the Real-estate App, I had to change the goal chunk in the model, so that the new version of this model (presented in paper no. 2) could handle varying menu depth as well as fixed menu depth. This model and the code are reported in paper no. 2. Thus, the model was developed for the shopping app. However, its descriptive power was tested with one group of the Real-Estate App study and the model can predict the other Real-Estate App group as well as the data from the Shopping App Study.

Third Study: Auditory Categorization Study (Paper No. 3 and Paper No. 4)

To test if the model mechanisms from the previous papers are more general, thus if they are applicable for other tasks that required learning and updating, a different task was chosen.

DESIGN AND METHODOLGIES

This is an auditory category learning task, were participants were required to repeatedly decide if a presented tone was a target, or not. In paper no. 3, the task is described as a dynamic decision-making task, because decisions about category membership are made under time-constraints; feedback is essential for learning and finally the environmental conditions change. This last aspect is debatable, because the environment only changes once. Nevertheless, the task is complex as are real-life decision-making tasks, where people need to make decisions in an environment they do not completely understand or control.

DATA (EMPRIRICAL + MODEL)

Due to other research questions that are aligned to the neurobiology of learning, the data was collected in an fMRI scanner. Such a setting is not ideal for learning and can be quite stressful for participants. Perhaps, this is why the performance of the participants, varied to a large extent and why not all participants managed to find the correct category assignment in the experiment.

Furthermore, due to slow BOLD[1] response, the time between the trials had to be at least six seconds. This is problematic for tasks which require participants to reflect on their learning success and keep track of their learning history (e.g. remembering which features are associated with negative feedback in the last trials). Unfortunately, this is likely to be relevant for this experiment, since the participants needed to remember which strategies

[1] Blood Oxygen Level Dependent

were not useful to avoid there usage later. In the model, we used an ACT-R parameter (namely the declarative finst span) and set it to either 80 or 100 seconds, to resemble that the model can remember if a strategy was misleading for approximately 8 to 10 trials. However, it is possible that this value is too high and that the participants in the scanner did not focus on the task during the long pauses.

The stimuli material was developed and tested by Wolff and Brechmann (2015). The stimuli consisted of 160 tones that each composed of five features with binary values. However, only four of these features were reported to be distinguishable by the participants. Consequently, only these four were used as stimuli for the model. However, if some of the participants did use the feature not implemented in our model (speed of modulation), they had a more difficult task then the model.

The task of model and participants differed critically regarding the following aspect. Namely, the model was provided with less information and thus had to accomplish a somewhat more difficult task than the participants. The instruction of the participants included information on which key was assigned to correct tones, this information was not provided to the model[2]. In a subsequent master's thesis, a new model was created that takes this information into account. However, the performance of the model of paper no. 3 and the new model does not differ, which is an indication that the missing assumption in the model is not that problematic.

MODEL MECHANISMS

Dynamic decision-making model

In the following, critical aspects of the decision-making model are briefly discussed. While main aspects of the model; e.g. how learning and mental model updating is implemented are derived from previous work (Prezenski & Russwinkel, 2014; Prezenski & Russwinkel, 2016); other aspects are modeled more arbitrary.

These are:

The assumptions that the model first tries to solve the task using a simple strategy, which regards one auditory feature relevant for solving the task and if these are not useful then it switches to more elaborate two-feature strategy (a strategy that uses a feature combination). This assumption is in line with other research on category learning, which indicates that people start with easy strategies and then switch to more difficult ones (Johansen & Palmeri, 2002). Furthermore, this assumption has high face validity; it seems plausible that something simple is tried out in the beginning and if this does not work it is later changed into something more elaborate. However, it is also reasonable that some participants do not use one-feature strategies and that some use three-feature strategies in this task.

The switching rules are another point of critique. These assumptions about when the model changes parts, or the whole aspect of the current strategies, where chosen carefully and with the intent to cover all possible cases where unsuccessful strategies need to be altered. The advantage of using these different switching rules is, that the model has a high variance and can simulate different individual learning curves. However, it is difficult to investigate if the participants also used all the specified switching rules and which they prefer. Here for, more studies using qualitative methods, such as questionnaires and retrospective interviews would be needed. Such a study using retrospective questionnaires with a modified version of this categorization task was conducted in a master's thesis (Zhang, 2018). However, interpreting the data was challenging and the results were inconsistent. They also showed that about 10 percent of the participants of this study were not able to name or describe their strategy at all or used guessing as a strategy during the entire study. Thus, in contrast to the model not all participants use a strategic approach. So, since the model always follows a strategy in the task, it cannot account for the data of these "non-strategical" participants.

Comparison of models (app and category task)

An important aim of the third study was to show, if the previously developed model mechanisms for mental model building and relearning are useful. Thus, to examine if they

[2] This was due to a misunderstanding between me and my co-authors on the instructions of the participants.

can be transferred to a different task and perhaps even a different area of research. Nonetheless, this task also needs to be an interactive task where learning and updating occurs. The first two papers investigated usability related aspects of app interaction. Nevertheless, it would be great if the model mechanisms allow to investigate more broader aspects, such as decision-making. Hence, the auditory category learning-task with one switch in target assembly was chosen. This task is very different to the app task in terms of stimulus material (non-verbal tones instead of words in text forms). Also, the data of the new study was collected in a scanner and participants were required to repeat the same task for 240 trials; while in the app studies the participants interacted with a handheld smartphone and there a maximum of "only" 32 trials. The categorization study is more artificial in terms, that it does not resemble an actual real-life task (as the first studies do). Instead, it is more controlled and fewer error variables influence the task.

The goals of both tasks differ as well, the first tasks' goals were to find an item in a hierarchical menu; the new goal was to find a strategy that can correctly discriminate between target and non-target stimuli.

To clarify, these vital differences of task and stimuli required the development of a new model. However, this model has high similarity regarding the way the mechanisms for learning an updating are modeled. In both, a model needs to find a solution; either finding a path and finding a target or finding a strategy that provides positive feedback. The structure of the tasks was modeled in the same way; Possible solutions are retrieved from declarative memory (either as path chunks or as strategy chunks). These solutions are built-up and modified in the imaginal buffer. The goal buffer controls the task. Furthermore, changes are handled in the same way. In case of negative feedback, a previously successful solution is kept and not changed directly. However, if it is again unsuccessful, it is altered, but not completely discarded. Both tasks are interactive learning tasks, where feedback is essential for learning. In the first tasks, positive feedback is finding what was expected; in the second task, this is also the case; but in a more straight-forward way; here feedback is encoded as yes or no.

Overall

Cognitive Modeling of Mental Model Building and Updating:

This thesis aimed, at obtaining a better understanding of the processes of mental model building and updating. Although, various theories on mental models exists (Norman, 1983; Glenberg & Langston, 1992; Zhang, 2009; Maani & Li, 2011; Revell & Stanton, 2014) many are not specified precisely or do not allow an actual implementation for real-life applications. Cognitive modeling with ACT-R allows both, specifying theories computationally, and applying them to actual problems. However, the models developed with ACT-R are not valid as such but need to be developed and tested following the descriptive, predictive and prescriptive challenge (Marewski and Link, 2014). Therefore, to ensure that the model mechanisms are valid, they were implemented in one model that was developed for one dataset and was then able to predict other datasets precisely (smartphone app studies). To check if these mechanisms are prescriptive; thus, if they can deal with novel problems, are robust and are generalizable to a wider range of task, they were successfully applied in the categorization study. It was found that the model and the derived model mechanism (which are summarized in the following section) meet the descriptive, predictive and prescriptive challenge.

The model mechanisms for ACT-R developed in this thesis are a specification of the theory of Li and Maani (2011) on mental models in dynamic decision-making. Li and Maani´s approach was chosen because it is open to a broad range of study subjects and thus useful for modeling a broad range of tasks which require mental model building and updating. According to this theory mental model building and updating is a continuous cycle of conceptualization, evaluation and reflection. In summary conceptualization is obtaining an understanding of the situation and mentally simulating the outcome of potential actions. Hereby, the given situation is compared to related information in the mental model and new information is used to develop a set of decisions. Experimentation designates the phase were actions and decisions devised from the mental model are tested. Finally, in the reflection phase, the outcome of these actions is reflected by processing feedback. Positive feedback strengthens the mental model; negative feedback requires mental model

updating. Furthermore, the finding, that mental models also evolve and become more specific, as learning progress (see Gonzalez & Lebiere, 2005) is also implemented specifically in the model mechanisms of this thesis.

In the next section, the model mechanisms developed in this thesis are summed up and aligned to the theory of Maani and Li. See figure 1 and 2 of paper no 4 for a graphical representation of the mechanisms. The main part of the mental model is a representation chunk (paper no. 1 and no. 2 path chunk, paper no. 3 strategy chunk), containing situation, expected outcome and success. During mental model building, it is placed in the imaginal buffer. Mental model building is the conceptualization phase. The mental model is tested (this is in the experimentation phase) and finally (in the evaluation phase) if the outcome is other than expected, the mental model is changed. If it is successful, an explicit strengthening mechanism is used. Also, mental models become more specific as learning evolves. Building of mental models furthermore requires control over environment changes and learning history. This information is stored in the control chunk, kept in the goal buffer. Mental model updating is the modification of an established representation chunk. A strategy which was successful in the path is altered the following way: An established strategy leads to unexpected outcome. So, the correctness of this strategy is questionable. This "uncertainty about the current strategy" is encoded in the control chunk in goal buffer. Nevertheless, the strategy is not discharged directly but tested once more. If this strategy leads to unexpected outcome once more, then a different strategy is rebuilt. Also, a change, e.g. to the system or the environment, is registered in the control chunk.

The mechanisms developed in this thesis, based on Li and Maani's theory, allow simulating mental model building and updating for different tasks. But this does not imply that the theory is correct and other theories on mental models are wrong. However, the theory of Li and Maani is now computationally implemented and thus precisely specified. Moreover, the mechanisms have proved useful for simulating very different tasks of interactive learning. Nevertheless, different ACT-R implementations of Li and Maani's theory are plausible as well.
In the current implementations, the core part of the mental model is the representation chunk in the imaginal buffer, which holds the situation and the predicted outcome. This could well consist of several interleaved chunks instead of one chunk. For example, the model of the smartphone apps the core part is one chunk with the path leading to the target. However, the core part of the mental model could consist of more chunks with parent child relationships and fewer slots per chunk. More chunks seem especially meaningful for more complex systems, were different parts of information need to be combined.

Another aspect worthwhile discussing, it that in this work modeling interactive learning of dynamic systems requires not only implicit but also explicit mechanisms. These explicit mechanisms where used to model meta-cognitive aspects, that reflected on learning progress. Many models which involve more complex issues use explicit mechanisms as well (Reitter, 2010; Roll, Baker, Aleven, and Koedinger, 2004), but usually the implementation of explicit mechanism vary between models. One of the strengths of the model approach that the explicit mechanisms are implemented in the same way for very different tasks. This reduces the arbitrary aspects of such mechanism and can serve as a basis for future guidelines on how to model these aspects.

As a starting point, the imaginal buffer should be used to build up the mental model. Herby, the core aspects need to be encoded as in instance-based learning, containing the situation and the expected outcome. However, the exact representation of the situation depends very much on the exact task.

Keeping track of learning success during early establishment of mental models requires an explicit strengthening mechanism; this needs to keep track of how often the mental model leads to success until a specific threshold. If the mental model does not reach this threshold due to unexpected (negative) feedback, it should be discharged. If the mental model reaches the threshold and then negative feedback occurs, the mental model should be held on to and tested once more. The exact value of the threshold (e.g. until a mental model is established) presumably depends on the task and may also depend on individual

differences. Nevertheless, more studies on this topic are needed, perhaps general values for such a threshold can be set in the future.

To control if changes to an established mental model are needed, another explicit mechanism should be used. In the case of interactive task learning, such changes are required due to changes in the system or to the environment. Here for, the presented modeling approach utilized two slots of the chunk in the goal buffer. If an established mental model leads to unexpected outcome, then uncertainty needs to be noted in the goal. This mental model is then tested again, if this leads to unexpected outcome again, then changes in the environment should be noted in the goal and the mental model needs to be altered. Future work should validate this changing mechanism separately, using qualitative methods. The assumption that a two-time failure of an established mental model leads to its alteration should be examined in an empirical study.

Even though the exact values for the change and the strengthening mechanisms are currently only rules of thumbs and these values are likely to vary depending on the task, the way the model mechanisms are implemented in ACT-R is very useful for answering applied questions, such as usability prediction or even for the development of intelligent agents. These aspects are discussed in the following sections of this thesis.

User Modeling of higher level cognitive processes/ Usability prediction

Usability describes how efficient, effective and satisfactory the use of a system is (ISO 924-10). On the one hand, satisfaction is a subjective feeling about which information can only be obtained via questionnaires or interviews, something that is time and resource consuming. On the other hand, efficiency and effectiveness are aspects of usability which can be measured with objective methods, such as measuring the time on task. An aim of this thesis was to investigate to what extent objective usability of system interaction can be captured using cognitive models which perform the same tasks as users. Already, the CogTool approach (John et al, 2004) has shown that efficiency of system interaction (especially websites) can be captured with ACT-R based cognitive models. CogTool models the behavior of expert users; it predicts how long users very familiar with am app need to find a goal. However, aspects that are more complex, such as interactive learning of new systems and handling of altered systems have not been addressed. This thesis addresses this gap. For linear, hierarchical smartphone apps and for the task of selecting specific items in such an app a cognitive model with ACT-R was developed and tested. This model demonstrates that usability prediction of apps, using cognitive models is possible. The model can address different aspects of interactive task learning and relearning. This is possible, because the ACT-R is capable of simulating higher-order cognitive mechanisms.

Worthwhile mentioning in the context of user modeling with ACT-R, is that we have also developed a tool (ACT-Droid) that allows ACT-R models to directly access Android apps (Doerr et al, 2016). This greatly lowers the burden of using ACT-R models for usability testing, because the Lisp prototyping of the app is time consuming and this is now obsolete.

A further development of ACT-droid (Russwinkel et al, 2018) namely its connection with ACT-Touch (Greene & Tamborello, 2013), now allows to examine specific touch aspects, such as taps and swipes. Thus, if the motor movement of the model (currently mouse clicks) are changed into touch commands a more accurate prediction by the model is possible.

Finally, for this specific model, where knowledge of the world (or pre-knowledge) is used to model initial app interaction, we have developed a crawler, which creates this knowledge of the world automatically (Prezenski, Bruechner & Russwinkel, 2017). The crawler automatically creates pre-knowledge from the structure of the apps by connecting bottom nodes with all higher-level nodes and automatically transferring them into ACT-R chunks. In this regard, the crawler is an important step towards a more general modeling approach, which aims at minimizing the modeling effort for usability prediction with cognitive models.

In combination with the above introduced tools (ACT-Droid, ACT-Touch and Crawler), we have demonstrated that usability modeling of cognitive aspects can be achieved using ACT-R models that have built-in higher-order cognitive mechanisms; which addresses aspects important for interactive learning such as learning depending on previous knowledge and relearning.

This specific model is extendable to other apps with the same hierarchical style for tasks were specific items need to be searched for. Changing this model, so it can run on other hierarchical list-style apps with more layers is almost effortless. A bit more effort is needed to change this model, so it can deal with lists of icons instead of text. Most mechanisms can stay the same (building up a path chunk in the imaginal and using this to navigate as soon as it exists as well as the meta-cognitive mechanisms in the goal buffer). However, a preliminary study should help to quantify, how long it takes to recognize certain icons.

This model was not designed to predict greater levels of details, such as how long it takes to find a subcategory, or precise visual item search. The later can be changed by implementing the model mechanism of Bailly et al (2014) and changing the visual search pattern, regarding the fact that not every item is visually processed one by one and item search without pre-knowledge requires that only every third item should be visual processed. The first aspect, predicting greater levels of details, requires collecting new data and analyzing to what extent the time it takes to find a subcategory is predicted by the model and subsequently altering the model if necessary.

Answering usability related questions, for very different kinds of apps such as navigation apps or for different tasks, such as searching for an unspecific goal requires the development of other ACT-R models. This is more challenging for several practical and theoretical reasons. These reasons hinder a more widespread usage of ACT-R in user modeling and will now be elaborated on.

The first practical reason is that only few people are familiar with ACT-R. Even though the ACT-R community is active, there is an annual conference, a website, tutorials and help is provided promptly via emails. The second reason is possibly the cause of the first. Namely, that the burden of learning ACT-R is high; the programming in Lisp is difficult, and many students would much rather use a more accessible language such as Python or JAVA. Already implementation of ACT-R in these languages exist (http://cog.cs.drexel.edu/act-r/download.php¸https://github.com/jakdot/pyactr); but they are not used as widely as Lisp ACT-R. However, a decision of the ACT-R community for prioritizing the use of these more accessible language could lead to more exchange with researchers from other fields.

The theoretical reasons are, that first the mechanisms used in the model need to be valid, as well as the assumptions about how a task is done. Mechanism can be derived from related literature or taken from other models. Information on how a task is done requires task analysis and often some empirical data. It is important that modelers make sure, that the model they postulate uses only necessary assumptions and derives them from other studies and empirical data. Only, because an ACT-R model can do the task does not imply that this is the way humans do it. There are many ways to model a task. In this work the attempt was made to follow the describe, predict, prescribe challenge to ensure that valid and useful model mechanisms for mental model building and updating are developed.

Cognitive Intelligent Systems

These ACT-R mechanisms for modeling mental model building and updating can be used to develop intelligent systems or agents. This refers to systems that simulate the cognitive processes of their users emerging during the system interaction. Hence, in theory, such systems are always informed about the cognitive states of their users. In the future, they can then help users in critical situations. For example, such intelligent systems could be used to avoid user errors. More specific, if the agent would predict a critical condition due to high demand of the simulated cognitive processes of the user, a warning would be issued. Currently, most intelligent systems are based on machine learning, such as Hidden Marcov Models (Eddy, 1996). However, unlike humans or models build with cognitive architectures, systems based on machine learning require a lot of samples to learn a task

and can only relearn through many more samples (Alpaydin, 2014). Thus, mechanisms and processes used by cognitive architectures could fill this gap. In this dissertation an ACT-R model mechanism that addressing aspects of interactive learning and updating was advanced. This provides an important step towards the development of learning intelligent agents. In the future aspects of this model approach can be applied to make systems more usable and safer.

References

Alpaydin, E. (2014). *Introduction to machine learning*. Cambridge, MA, USA. MIT press.

Bailly, G., Oulasvirta, A., Brumby, D. P., & Howes, A. (2014). Model of visual search and selection time in linear menus. *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems - CHI '14*, (pp. 3865–3874). Toronto, Canada: ACM.

Doerr, L.-M., Russwinkel, N., & Prezenski, S. (July 2016). ACT-Droid: ACT-R Interacting with Android Applications. In D. Reitter & F. Ritter (Eds.), *Proceedings of the 14th International Conference on Cognitive Modeling* (pp. 225- 227). University Park, PA: Penn State.

Eddy, S. R. (1996). Hidden markov models. Current opinion in structural biology, 6(3), (pp. 361-365).

Greene, K. K. & Tamborello, F. P. (July 2013). Initial ACT-R extensions for user modeling in themobile touchscreen domain. In. R. West and T. Stewart (Eds.), *Proceedings of the 12th International Conference on Cognitive Modeling* (pp.348- 353). Ottawa, Canada: Carleton University.

International Standards Organisation (2018). *Ergonomics of human-system interaction — Part 11: Usability: Definitions and concepts* (ISO 9241-11). Berlin, Germany.

John, B. E., Prevas, K., Salvucci, D. D., & Koedinger, K. (April 2004). Predictive human performance modeling made easy. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 455- 462). New York City, NY: ACM. doi: 10.1145/985692.985750

Li, A. & Maani, K. (July 2011). Dynamic decision-making, learning and mental models, in J. Lyneis (Ed.), *Proceedings of the 29th International Conference of the System Dynamics Society* (pp. 1- 21). Washington, DC: System Dynamics Society.

Marewski J. N. & Link D. (2014). Strategy selection: an introduction to the modeling challenge. Wiley Interdiscipl. Rev., 5, 39- 59. doi: 10.1002/wcs.1265

Prezenski, S. (2017). Implementing Mental Model Updating in ACT-R. In M. van Vugt, A. Banks & W. Kennedy (Eds.), *Proceedings of the 15th International Conference on Cognitive Modeling*. (pp. 121-127). Coventry, United Kingdom.

Prezenski, S. & Russwinkel, N. (2014). Combining cognitive ACT-R models with usability testing reveals users mental model while shopping with a smartphone application. *Int. J. Adv. Intell. Syst.*, 7(3), 700- 715.

Prezenski, S., Bruechner, D. & Russwinkel, N. (March 2017). Predictive Cognitive Modelling of Applications. Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications HUCAPP (pp. 165- 171). doi: 10.5220/0006273301650171

Prezenski, S., Brechmann, A., Wolff, S. & Russwinkel, N. (2017). A Cognitive Modeling Approach to Strategy Formation in Dynamic Decision Making. *Frontiers in Psychology*. 8 (no 1335). doi: 10.3389/fpsyg.2017.01335

Prezenski, S. & Russwinkel, N. (July 2016b). Towards a general model of repeated app usage. In D. Reitter & F. E. Ritter (Eds.), *Proceedings of the 14th International Conference on Cognitive Modeling* (pp. 201- 207). University Park, PA: Penn State.

Reitter, D. (2010). Metacognition and multiple strategies in a cognitive model of online control. J. *Artif. Gen. Intell.* 2, 20–37. doi: 10.2478/v10229-011-0007-3

Roll, I., Baker, R. S., Aleven, V., & Koedinger, K. R. (2004). A metacognitive ACT-R model of students' learning strategies in Intelligent Tutoring Systems. In eds J. C. Lester, R. M. Vicari, & F. Paraguacu (Eds.), *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*, (pp. 854–856). Maceió: Brasil.

Russwinkel, N., Prezenski, S., Doerr, L.& Tamborello, F. (July 2018). ACT-Droid meets ACT-Touch: Modelling differences in swiping behavior with real Apps. In I. Juvina, J. Houpt, & C. Myers (Eds.), *Proceedings of the 16th International Conference on Cognitive Modeling* (pp. 121-125). Madison, WI: University of Wisconsin.

Wolff, S., & Brechmann, A. (2015). Carrot and stick 2.0: The benefits of natural and motivational prosody in computer-assisted learning. *Computers in Human Behavior, 43*, 76-84.

Ziying Zhang. Deciphering Dynamic Decision Making Using Cognitive Modeling Including Subjective Assessment. *Master's thesis*, Technical University Berlin, 2018

References (Complete)

Ahlström, D., Cockburn, A., Gutwin, C., & Irani, P. (April 2010). Why it's quick to be square: modelling new and existing hierarchical menu designs, *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 1371-1380). New York, NY: ACM. doi: 10.1145/1753326.1753534

Alpaydin, E. (2014*). Introduction to machine learning*. Cambridge, MA, USA. MIT press.

Amalfitano, D., Fasolino, A. R., Tramontana, P., DeCarmine, S., & Memon, A.M. (2012). Using GUI ripping for automated testing of Android applications. *In Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering, ASE* (pp. 258–261), New York, NY, USA. ACM

Anderson, J. R. (2007). *How can the Human Mind Occur in the Physical Universe?* New York, NY: Oxford University Press.

Anderson, J. R. (1991). The adaptive nature of human categorization. *Psychological Review, 98*(3), 409-429. doi: 10.1037/0033-295X.98.3.409

Anderson, J. R.& Betz J. (2001). A hybrid model of categorization. *Psychon. Bull. Rev.*, 8, 629-647, doi: 10.3758/BF03196200

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review*, 111, 1036-1060

Anderson, J. R, Fincham, J. M., Qin, Y. & Stocco A. (2008). A central circuit of the mind. *Trends Cogn. Sci.*, 12(4), 136-143. doi: 10.1016/j.tics.2008.01.006

Anderson, J. R.& Fincham, J. M. (2014). Extending problem-solving procedures through reflection. *Cogn. Psychol.*, 74, 1-34. doi: 10.1016/j.cogpsych.2014.06.002

Anderson, J. R., Matessa, M. & Lebiere, C. (1997). ACT-R: A Theory of Higher Level Cognition and Its Relation to Visual Attention. *Hum.- Comput. Interact.*, 12 (4), 439- 462. doi: 10.1207/s15327051hci1204_5

Apple iOS Human Interface Guidelines: http://developer.apple.com/library/ios/#documentation/UserExperience/Conceptual/MobileHI G/Introduction/Introduction.html

Ashby, F. G. (1992). Multidimensional models of categorization. In F. G. Ashby (Ed.), *Scientific psychology series. Multidimensional models of perception and cognition* (pp. 449- 483). Hillsdale, NJ: Lawrence Erlbaum Associates.

Ashby, F. G., Alfonso-Reese, L. A., Turken, A. U., Waldron, E. M. (1998). A neuropsychological theory of multiple systems in category learning. *Psychol. Rev*, 105, 442-481.

Ashby, F. G., & Maddox, W. T. (2011). Human category learning 2.0. *Annals of the New York Academy of Sciences, 1224*(1), 147- 161.doi: 10.1111/j.1749-6632.2010.05874

Atkins, D. (October 2000). Human factors in avalanche accidents. In *International snow science workshop, Big Sky, MT* (pp. 46- 51).

Bailly, G., Oulasvirta, A., Brumby, D. P., & Howes, A. (April 2014). Model of visual search and selection time in linear menus. In *Proceedings of the 32nd annual ACM conference on*

Human factors in computing systems (pp. 3865- 3874). New York, NY: ACM. doi: 10.1145/2556288.2557093

Berg, E. A. (1948). A simple objective technique for measuring flexibility in thinking. *J. Gen. Psychol*, 39, 15- 22. doi: 10.1080/00221309.1948.9918159

Berry, D. C. & Broadbent, D. E. (1988). Interactive tasks and the implicit-explicit distinction. *Br. J. Psychol.*, 79, 251- 272. doi:10.1111/j.2044-8295.1988.tb02286.x

Borst, J. P. & Anderson, J. R. (2015). Using the ACT-R Cognitive Architecture in Combination with fMRI data. In B.U. Forstmann, E.-J. Wagenmakers (Eds), *An Introduction to Model-Based Cognitive Neuroscience* (pp.339- 352). New York City, NY: Springer.

Borst, J. P., Nijboer, M., Taatgen, N. A., Van Rijn, H. & Anderson, J. R. (2015). Using data-driven model-brain mappings to constrain formal models of cognition. *PLoS ONE*, 10(3). doi: 10:e0119673. 10.1371/journal.pone.0119673

Bowers, J. S., Davis, C. J. (2012). Bayesian just-so stories in psychology and neuroscience. *Psychol. Bull.* 138, 389- 414. doi: 10.1037/a0026450

Buettner, P. (July 2010). "Hello Java!" Linking ACT-R 6 with a Java simulation. In D. D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of the 10th International Conference on Cognitive Modeling* (pp. 289-290). Philadelphia, PA: Drexel University.

Byrne, M. D. (2001). ACT-R/PM and menu selection: applying a cognitive architecture to HCI. *Int. J. Hum. Comput. Stud.,* 55(1), 41- 84. doi: 10.1006/ijhc.2001.0469

Byrne, M.D., Anderson, J.R., Douglass, S. & Matessa, M. (Mai 1999). Eye tracking the visual search of click-down menus. Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 402- 409). NYC, New York: ACM. doi: 10.1145/302979.303118

Byrne, M. D. & Pew, R.W. (2009). A history and primer of human performance modeling.*Rev. Hum. Factors Ergon.*, 5(1), 225- 263. doi: 10.1518/155723409X448071

Cao, S. & Liu, Y. (September 2011). Mental workload modeling in an integrated cognitive architecture. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 55 (1) (pp. 2083- 2087). doi: 10.1177/1071181311551434

Choi, W., Necula, G., & Sen, K. (October 2013). Guided GUI testing of Android apps with minimal restart and approximate learning. *In Acm Sigplan Notices*, 48(10), (pp. 623-640). NYC: NY: ACM. doi: 10.1145/2509136.2509552

Cockburn, A. & Gutwin, C. (2009). A Predictive Model of Human Performance With Scrolling and Hierarchical Lists. *Human Computer Interact.,* 24(3), 273- 314. doi: 10.1080/07370020902990402.

Cockburn, A., Gutwin, C. & Greenberg, S. (April 2007). A predictive model of menu performance, *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 627- 636). NYC, New York: ACM. doi: 10.1145/1240624.1240723

Collins, A. M. & Quillian, M. R. (1969). Retrieval time from semantic memory. *J. Verbal Learning Verbal Behav.*, 8(2), 240- 248. doi: 10.1016/S0022- 5371(69)80069-1.

Clark, L., Cools, R. & Robbins, T. W. (2004). The neuropsychology of ventral prefrontal cortex: decision-making and reversal learning. *Brain Cogn.*,55, 41- 53. doi: 10.1016/S0278-2626(03)00284-7

Das, A., & Stuerzlinger, W. (2007). A cognitive simulation model for novice text entry on cell phone keypads. *Proceedings of the 14th European Conference on Cognitive Ergonomics* (pp.141- 147). New York City, NY: ACM. doi: 10.1145/1362550.1362579

Doerr, L.-M., Russwinkel, N., & Prezenski, S. (July 2016). ACT-Droid: ACT-R Interacting with Android Applications. In D. Reitter & F. Ritter (Eds.), *Proceedings of the 14th International Conference on Cognitive Modeling* (pp. 225- 227). University Park, PA: Penn State.

Dimov C. M., Marewski J. N. & Schooler L. J. (2013). Constraining ACT-R models of decision strategies: an experimental paradigm, in Cooperative Minds: Social Interaction and Group Dynamics. In M. Knauff, M. Pauen, N. Sebanz, I. Wachsmuth (eds). *Proceedings of the 35th Annual Conference of the Cognitive Science Society* (pp. 2201-2206). Austin, TX: Cognitive Science Society.

Edwards, W. (1962). Dynamic Decision Theory and Probabilistic Information Processing. *Human Factors*, 4, 59-73. doi: 10.1177/001872086200400201

Eddy, S. R. (1996). Hidden markov models. *Current opinion in structural biology*, 6(3), (pp. 361365).

Engel, C., Herdin, J. & Maertin, C. (Mai 2012). Exploiting HCI pattern collections for user interface generation. In A. Zimmermann & P. Lorenz (Eds.), *Proceedings of the 4th International Conference on Pervasive Patterns and Applications* (pp. 36-44). USA: IARIA.

Erickson, M. A. & Kruschke, J. K. (1998). Rules and exemplars in category learning. *J. Exp. Psychol.* 127, 107-140. doi: 10.1037/0096-3445.127.2.107

Forstmann, B. U., Wagenmakers, E.-J., Eichele, T., Brown, S. & Serences, J. T. (2011). Reciprocal relations between cognitive neuroscience and formal cognitive models: opposites attract? *Trends Cogn. Sci.* 15, 272-279. doi: 10.1016/j.tics.2011.04.002

Fum, D. & Stocco, A. (2003). Instance vs. rule-based learning in controlling a dynamic system. In F. Detje, D. Dörner, H. Schaub (Eds.). *Proceedings of the 5th International Conference on Cognitive Modelling* (pp 105-110). Bamberg: Universitaets-Verlag Bamberg.

Geven, A., Sefelin,R. & Tscheligi, M. (September 2006). Depth and Breadth away from the Desktop - the Optimal Information Hierarchy for Mobile Use. *Proceedings of the 8th Conference on Human-computer Interaction with Mobile Devices and Services* (pp. 157-164). New York: ACM. doi: 10.1145/1152215.1152248.

Glenberg, A. M. & Langston, W. E. (1992). Comprehension of Illustrated Text: Pictures Help to Build Mental Models. *Journal of Memory and Language*, 31, 129-151.doi: 10.1016/0749596X(92)90008-L

Gigerenzer, G. & Brighton, H. (2009). Homo heuristicus: why biased minds make better inferences. *Top. Cogn. Sci.*, 1, 107143. doi: 10.1111/j.1756-8765.2008.01006.x

Gonzalez, C. (2014). Decision Making: A Cognitive Science Perspective. In S. E. F. Chipman (Ed.), *The Oxford Handbook of Cognitive Science* (pp. 249- 263). Oxford, UK: Oxford University Press.

Gonzalez, C., Dutt, V., Healy, A. F., Young, M. D. & Bourne, L. E. (2009). Comparison of instance and strategy models in ACT-R. In Howes, A., Peebles, D., Cooper, R. (Eds.), *Proceedings of the 9th International Conference on Cognitive Modeling*. (paper150, 6 pages), Manchester, UK.

Gonzalez, C., Lerch, J. F. & Lebiere, C. (2003). Instance-based learning in dynamic decision making. *Cognitive Science*, 27(4), 591- 635. doi: 10.1207/s15516709cog2704_2

Gonzalez, C. & Lebiere, C. (2005). Instance-based cognitive models of decision-making. In D. Zizzo & A. Courakis (Eds.), *Transfer of knowledge in economic decision making,* (pp. 148- 165). New York, NY: Palgrave McMillan.

Google User Interface Guidelines: http://developer.android.com/guide/practices/ui_guidelines/index.html

Gray, W. D., & Altmann, E. M. (2001). Cognitive modeling and human-computer interaction. In W. Karwowski (Ed.), International encyclopedia of ergonomics and human factors (Vol. 1) (pp. 387-391). New York City, NY: Taylor & Francis, Ltd.

Greene, K. K. & Tamborello, F. P. (July 2013). Initial ACT-R extensions for user modeling in themobile touchscreen domain. In. R. West and T. Stewart (Eds.), *Proceedings of the 12th International Conference on Cognitive Modeling* (pp.348- 353). Ottawa, Canada: Carleton University.

Halbrügge, M., & Russwinkel, N. (July 2016). The sum of two models: How a composite model explains unexpected user behavior in a dual-task scenario. In D. Reitter & F. E. Ritter (Eds.), *Proceedings of the 14th International Conference on Cognitive Modeling* (pp. 137-143). University Park, PA: Penn State.

Halbrügge M. (2010). Keep it simple - A case study of model development in the context of the Dynamic Stocks and Flows (DSF) task. *J. Artif. Gen. Intell.*,2, 38- 51. doi: 10.2478/v10229-011-0008-2

Harrison, R., Flood, D. & Duce, D. (2013). Usability of mobile applications: literature review and rationale for a new usability model. *J. Interact. Sci.,* 1(1). doi: 10.1186/2194-0827-1-1

Hornof, A. J. & Kieras, D. E. (April 1997). Cognitive modeling reveals menu search in both random and systematic. *Proceedings of SIGCHI Conference on Human factors in Computing Systems* (pp. 107- 114). doi: 10.1145/258549.258621

International Standards Organisation (2018). *Ergonomics of human-system interaction — Part 11: Usability: Definitions and concepts* (ISO 9241-11). Berlin, Germany.

Jarvers C., Brosch T., Brechmann A., Woldeit M. L., Schulz A. L. & Ohl F. W., et al. (2016). Reversal learning in humans and gerbils: dynamic control network facilitates learning. *Front. Neurosci.,* 10 (no: 535). doi: 10:535 10.3389/fnins.2016.00535

Johansen, M. K. & Palmeri, T. J. (2002). Are there representational shifts during category learning? *Cogn. Psychol.* 45, 482- 553. doi: 10.1016/S0010-0285(02)00505-4

John, B. E., Prevas, K., Salvucci, D. D., & Koedinger, K. (April 2004). Predictive human performance modeling made easy. *Proceedings of the SIGCHI Conference on Human factors in Computing Systems* (pp. 455- 462). New York City, NY: ACM. doi: 10.1145/985692.985750

John, B. E. & Suzuki, S. (July 2009). Toward Cognitive Modeling for Predicting Usability. *Proceedings of Human-Computer Interaction HCI International* (pp. 267- 276). Berlin, Heidelberg: Springer. doi: 10.1007/978- 3-642-02574-7_30

Kaptelinin, V. (April 1993). Item Recognition In Menu Selection: The Effect Of Practice. *Proceedings of CHI '93 and INTERACT '93 Conference Companion on Human Factors in Computing Systems* (pp. 183- 184). New York City, NY: ACM. doi: 10.1145/259964.260196

Khemlani, S., Trafton, J. G., & Johnson-Laird, P. N. (July 2013). Deduction as stochastic simulation. In. R. West and T. Stewart (Eds.), *Proceedings of the 12th International Conference on Cognitive Modeling* (pp. 297- 302). Ottawa, Canada: Carleton University

Kieras, D. E. & Meyer, D. E. (1997). An overview of the EPIC architecture for cognition and performance with application to human-computer interaction, *Human–Computer Interact*.,12(4), 391- 438. doi: 10.1207/s15327051hci1204_4

Klein, G.A., Moon, B., Hoffman, R.R. (2006). Making sense of sensemaking 2: A macrocognitive model. *IEEE Intelligent Systems*, 22(5), 88-92. doi: 10.1109/MIS.2006.100

Knauff M., Wolf A. G. (2010). Complex cognition: the science of human reasoning, problem-solving, and decision-making. *Cogn. Process*., 11, 99- 102. doi: 10.1007/s10339-010-0362-z

Koetsier, J. (January 2013). Google Play will hit a million apps in June. Retrieved from: https://venturebeat.com/2013/01/04/google-play-will-hit-a-million-apps-in-2013-probably-sooner-than-the-ios-app-store/

Kohrs, C., Hrabal, D., Angenstein, N. & Brechmann, A. (2014). Delayed system response times affect immediate physiology and the dynamics of subsequent button press behavior. *Psychophysiology*, 51, 1178- 1184. doi:10.1111/psyp.12253

Kortum, P. & Peres, S. C. (2014). The Relationship Between System Effectiveness and Subjective Usability Scores Using The System Usability Scale, *Int. J. Hum. Comput. Interact.,* 30(7), 575-584. doi:10.1080/10447318.2014.904177

Kruschke, J. K. (1992). ALCOVE: an exemplar-based connectionist model of category learning. *Psychol. Rev*., 99(1), 22- 44. doi:10.1037/0033-295X.99.1.22

Laird, J.E. (2012). The Soar Cognitive Architecture. Cambridge, UK: MIT Press.

Lee, E. & Macgregor, J. (1985). Minimizing User Search Time in Menu Retrieval Systems. *Human Factors: The J. of the Human Factors and Ergonomics Society*, 27(2), 157-162. doi: 10.1177/001872088502700203

Lebiere, C., Pirolli, P., Thomson, R., Paik, J., Rutledge-Taylor, M., Staszewski, J. & Anderson, J.R. (2013). A functional model of sensemaking in a neurocognitive architecture. *Comput. Intell. Neurosci*.,5, 1-29. doi: 10.1155/2013/921695

Lebiere C., Wallach D. & Taatgen N. A. (1998). Implicit and explicit learning in ACT-R. In F. Ritter and R. Young (Eds.). *Proceedings of the Second European Conference on Cognitive Modelling* (pp. 183- 193). Nottingham: Nottingham University Press.

Lewandowsky, S., & Farrell, S. (2011). Computational Modeling in Cognition: Principles and Practice. Thousand Oaks, CA, US: SAGE.

Lewandowsky S., Palmeri T. J. & Waldmann M. R. (2012). Introduction to the special section on theory and data in categorization: integrating computational, behavioral, and cognitive neuroscience approaches. *J. Exp. Psychol. Learn. Mem. Cogn*., 38 (4), 803- 806. doi: 10.1037/a0028943

Li, A. & Maani, K. (July 2011). Dynamic decision-making, learning and mental models, in J. Lyneis (Ed.), *Proceedings of the 29th International Conference of the System Dynamics Society* (pp. 1- 21). Washington, DC: System Dynamics Society.

Lindner, S., Büttner, P., Taenzer, G., Vaupel, S. & Russwinkel, N. (March 2014). Towards an efficient evaluation of the usability of android apps by cognitive models. In:

Wendemuth, A., Jipp, M., Kluge, A. & Söffker, D. (Eds.), *Proceedings. 3. Interdisziplinärer Workshop Kognitive Systeme: Mensch, Teams, Systeme und. Automaten* (10 pages) Magdeburg, Germany.

Love B. C., Medin D. L. & Gureckis T. M. (2004). SUSTAIN: a network model of category learning. *Psychol. Rev.,* 111(2), 309- 332. doi:10.1037/0033-295X.111.2.309

Marewski J. N. & Link D. (2014). Strategy selection: an introduction to the modeling challenge. *Wiley Interdiscipl. Rev.*, 5, 39- 59. doi: 10.1002/wcs.1265

Marewski J. N. & Mehlhorn K. (2011). Using the ACT-R architecture to specify 39 quantitative process models of decision making. *Judgm. Decis. Mak.,* 6, 439- 519.

McDougall, S., Curry, M. & de Bruijn, O. (2001). The effects of visual information on users' mental models: an evaluation of Pathfinder analysis as a measure of icon usability. *Int. J. Cogn. Ergon.*, 5(2), 153- 178. doi: 10.1207/S15327566IJCE0501_4

McDonald, J. E., Stone, J. D. & Liebelt, L. S. (October 1983). Searching for Items in Menus: The Effects of Organization and Type of Target. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 27(9) (pp. 834- 837). doi: 10.1177/154193128302700919.

Mehlenbacher, B., Duffy, T. M., & Palmer, J. (1989). Finding information on a menu: linking menu organization to the user's goals. *Human-Computer Interaction*, 4(3), 231-251. doi: 10.1207/s15327051hci0403_3

Möller S., Engelbrecht K.-P. & Schleicher R. (2008). Predicting the quality and usability of spoken dialogue services, *Speech Commun.,*50(8-9), 730- 744. doi: 10.1016/j.specom.2008.03.001

Naumann, A., Hurtienne, J., Israel, J. H., Mohs, C., Kindsmüller, M. C., Meyer, H. A., & Hußlein, S. (July 2007). Intuitive use of user interfaces: defining a vague concept. *In International Conference on Engineering Psychology and Cognitive Ergonomics* (pp. 128-136). Berlin, Heidelberg: Springer.

Newell, A. (1973). You can´t play 20-questions with nature and win: Projective comments on the papers of the Symposium. In W.G. Chase (Ed.), Visual information processing (pp. 463- 526). New York City, NY: Academic Press.

Nilsen, E. L. (1996). *Perceptual-motor control in human-computer interaction*. Ann Arbor: MI: University of Michigan.

Nielsen, J. (October 2012). User satisfaction vs. performance metrics. Retrieved from: https://www.nngroup.com/articles/satisfaction-vs-performance-metrics/

Nielsen, J. (1994). *Usability engineering*. Burlington, MA: Morgan Kaufmann Pub.

Nielsen, J. & Budiu, R. (2012). *Mobile Usability*. San Francisco, CA: New Riders Publishing.

Norman, D. (2013). *The Design of Everyday Things*. New York City, NY: Basic Books

Norman, D. (1983). Some observations on mental models. In D. Gentner & A. L. Stevens (Eds.), *Mental models* (pp. 7- 14). NYC, NY: Lawrence Erlbaum Press.

Nosofsky R. M. (1984). Choice, similarity, and the context theory of classification. *J. Exp. Psychol.*, 10, 104- 114. doi: 10.1037/0278-7393.10.1.104

Nosofsky R. M., Palmeri T. & McKinley S. (1994). Rule-plus-exception model of classification learning. *Psychol. Rev.*, 101, 53- 79. doi: 10.1037/0033-295X.101.1.53

Orendain, A. D. O. & Wood, S. (April 2012). An account of cognitive flexibility and inflexibility for a complex dynamic task. In N. Russwinkel, U. Drewitz, H. van Rijn, H. (Eds.), *Proceedings of the 11th International Conference on Cognitive Modeling* (pp. 49-54), Berlin, Germany: TU Berlin.

Payne, S. (2003). Users' mental models: The very ideas. In J. Carroll (Ed.), *HCI models, theories, and frameworks: Toward a multidisciplinary science*, (pp. 135-156). Burlington, MA, USA: Morgan Kaufman Publishers

Peebles, D. & Banks, A. P. (2010). Modelling dynamic decision making with the ACT-R cognitive architecture. *J. Artif. Gen. Intell.*, 2, 52- 68. doi:10.2478/v10229-011-0009-1

Prezenski, S.& Russwinkel, N. (2014). Combining cognitive ACT-R models with usability testing reveals users mental model while shopping with a smartphone application. *Int. J. Adv. Intell. Syst.*, 7(3), 700- 715.

Prezenski S. & Russwinkel N. (July 2016a). A proposed method of matching ACT-R and EEG-Data. In D. Reitter and F.E.Ritter (Eds.), *Proceedings of the 14th International Conference on Cognitive Modeling* (pp. 249- 251).University Park, PA: Penn State.

Prezenski, S., Bruechner, D. & Russwinkel, N. (March 2017). Predictive Cognitive Modelling of Applications. *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications HUCAPP* (pp. 165- 171). doi: 10.5220/0006273301650171

Prezenski, S., Brechmann, A., Wolff, S. &Russwinkel, N. (2017). A Cognitive Modeling Approach to Strategy Formation in Dynamic Decision Making. *Frontiers in Psychology*. 8 (no 1335). doi: 10.3389/fpsyg.2017.01335

Prezenski, S. & Russwinkel, N. (July 2016b). Towards a general model of repeated app usage. In D. Reitter & F. E. Ritter (Eds.), *Proceedings of the 14th International Conference on Cognitive Modeling* (pp. 201- 207). University Park, PA: Penn State.

Prezenski, S., Lindner, S., Moegele, H., & Russwinkel, N. (unpublished manuscript). Archetyping the User.

Prezenski, S. (2017). Implementing Mental Model Updating in ACT-R. In M. van Vugt, A. Banks & W. Kennedy (Eds.), *Proceedings of the 15th International Conference on Cognitive Modeling*. (pp. 121-127). Coventry, United Kingdom.

Raufaste, E. (Sept. 2006). Air traffic control in ACT-R: A computational model of conflict detection between planes. In F. Reuzeau, K. Corker, G. Boy (Eds.), *Proceedings of the International Conference on Human Computer Interaction in Aeronautics* (pp. 258-259). Toulouse, France: Editions Cépaduès.

Reitter D. (2010). Metacognition and multiple strategies in a cognitive model of online control. *J. Artif. Gen. Intell.* 2, 20-37. doi: 10.2478/v10229-011-0007-3

Revell, K. M. A. & Stanton, N. A. (2014). Case studies of mental models in home heat control: Searching for feedback, valve, timer and switch theories. *Applied Ergonomics*, 45(3), 363-378. doi: 10.1016/j.apergo.2013.05.001

Roll I., Baker R. S., Aleven V. & Koedinger K. R. (September 2004). A metacognitive ACT-R model of students' learning strategies in Intelligent Tutoring Systems. In J.C. Lester, R. M. Vicari, F. Paraguacu (Eds.), *Proceedings of the 7th International Conference on Intelligent Tutoring Systems* (pp. 854- 856). Berlin- Heidelberg, Germany: Springer.

Russwinkel, N. & Prezenski, S. (May 2014). ACT-R meets usability or why cognitive modeling is a useful tool to evaluate the usability of smartphone applications. In (H.-W. Sehrings (Eds.), *Proceedings the Sixth International Conference on Advanced Cognitive Technologies and Applications* (pp. 62- 65). Venice, Italy.

Russwinkel, N., Prezenski, S., Doerr, L.& Tamborello, F. (July 2018). ACT-Droid meets ACT-Touch: Modelling differences in swiping behavior with real Apps. In I. Juvina, J. Houpt, & C. Myers (Eds.), *Proceedings of the 16th International Conference on Cognitive Modeling* (pp. 121-125). Madison, WI: University of Wisconsin

Russwinkel, N., Urbas, L. & Thüring, M. (2011). Predicting temporal errors in complex task environments: a computational and experimental approach. *Cogn. Syst. Res.*, 12(3-4), 336-354. doi: 10.1016/j.cogsys.2010.09.003

Rutledge-Taylor, M., Lebiere, C., Thomson, R., Staszewski, J.& Anderson, J. R. (March 2012). A comparison of rule-based versus exemplar-based categorization using the ACT-R architecture. *Proceedings of 21st Annual Conference on Behavior Representation in Modeling and Simulation* (pp. 44- 50). Amelia Island, FL: BRIMS Committee.

Salatas, H. & Bourne, L. E., Jr. (1974). Learning conceptual rules: III. Processes contributing to rule difficulty. *Mem. Cogn.*, 2, 549- 553. doi:10.3758/BF03196919

Salvucci, D. D. (2006). Modeling driver behavior in a cognitive architecture. *Hum. Factors*, 48, 362- 380. doi: 10.1518/001872006777724417

Samp, K. (April 2013). Designing Graphical Menus for Novices and Experts: Connecting Design Characteristics with Design Goals. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 3159- 3168). doi: 10.1145/2470654.2466432

Sanborn, A. N., Griffiths, T. L. & Navarro, D. J. (2010). Rational approximations to rational models: alternative algorithms for category learning. *Psychol. Rev.*, 117, 1144- 1167. doi:10.1037/a0020511

Seger, C. A. & Peterson, E. J. (2013). Categorization = decision making + generalization. *Neurosci. Biobehav. Rev.*, 37, 1187- 1200. doi: 10.1016/j.neubiorev.2013.03.015

Simon, H. A. & Newell A. (1971). Human problem solving: the state of the theory in 1970. *Am. Psychol.,* 26, 145- 159. doi:10.1037/h0030806

Smieszek, H., Joeres, F. & Russwinkel, N. (2015). Workload of airport tower controllers: empirical validation of a macro-cognitive model, *Kognitive Systeme,*1(2), 10 pages. doi: 10.17185/duepublico/37699

St. Amant, R., Horton, T.E., & Ritter, F.E. (2007). Modelbased evaluation of expert cell phone menu interaction. *Transactions on Computer-Human Interaction,* 14(1), 1- 14.

Statista.com (Dec 2017). Average number of new Android app releases per day from 3rd quarter 2016 to 3rd quarter 2017. Retrieved from https://www.statista.com/statistics/276703/androidapp-releases-worldwide/

Stewart, T. C. & West, R. (2010). Testing for equivalence: a methodology for computational cognitive modeling. *J. Artif. Gen. Intell*, 2, 69- 87. doi: 10.2478/v10229-011-0010-8

Sun, R. (2008). Introduction to computational cognitive modeling. In R. Sun (Ed.), *Cambridge handbook of computational psychology* (pp. 3-19). Cambridge, UK: Cambridge University Press.

Sutton, R. S. & Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. London, UK: MIT Press.

Taatgen, N. A. (July 2001). A model of individual differences in learning Air Traffic Control. In E.M. Altmann, A. Cleeremans, C.D. Schunn, W.D. Gray (Eds.), *Proceedings of the fourth International Conference on Cognitive Modeling* (pp. 211- 216). Mahwah, NJ: Erlbaum.

Taatgen, N. A., Lebiere, C. & Anderson, J. R. (2006). Modeling Paradigms in ACT-R, in Cognition and Multi-Agent Interaction. In R. Sun (Ed.), *From Cognitive Modeling to Social Simulation* (pp. 29- 52). Cambridge, UK: Cambridge University Press.

Tenison, C., Fincham, J. M., & Anderson, J. R. (2016). Phases of learning: How skill acquisition impacts cognitive processing. *Cognitive psychology,* 87, 1- 28.

Teo, L. H., John, B., & Blackmon, M. (2012, May). CogTool-Explorer: a model of goal-directed user exploration that considers information layout. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 2479- 2488). NYC, NY: ACM. doi: 10.1145/2207676.2208414

Thomson, R., Lebiere, C., Anderson, J. R., & Staszewski, J. (2015). A general instance-based learning framework for studying intuitive decision-making in a cognitive architecture. *Journal of Applied Research in Memory and Cognition*, 4(3), 180- 190. doi: 10.1016/j.jarmac.2014.06.002

Wickens, C.D. (2008). Multiple resources and mental workload. *Human Factors: J. of the Human Factors and Ergonomics Society*. 50(3), 449- 455. doi: 10.1518/001872008X288394.

Wills A. J. & Pothos E. M. (2012). On the adequacy of current empirical evaluations of formal models of categorization. *Psychol. Bull.* 138, 102- 125. doi:10.1037/a0025715.

Wolff, S. & Brechmann, A. (September 2012). MOTI: a motivational prosody corpus for speech-based tutorial systems. *Proceedings of the 10th ITG Conference on Speech Communication* (pp. 1- 4). Braunschweig, Germany: VDE.

Wolff, S. & Brechmann, A. (2015). Carrot and Stick 2.0: the benefits of natural and motivational prosody in computer-assisted learning. *Comput. Hum. Behav.* 43, 76- 84. doi: 10.1016/j.chb.2014.10.015.

Wong, T. J., Cokely, E. T. & Schooler, L. J. (July 2010). An online database of ACT-R parameters: Towards a transparent community-based approach to model development. In D.D. Salvucci, G. Gunzelmann (Eds.), *Proceedings of the 10th International Conference on Cognitive Modeling* (pp. 282- 286). Philadelphia, PA: Drexel University.

Zapata, B. C., Niñirola, A. H., Idri, A., Fernández-Alemán, J. L., & Toval, A. (2014). Mobile PHRs compliance with Android and iOS usability guidelines. *Journal of medical systems*, 38(8), 1-16. doi:10.1007/s10916-014-0081-6

Zeller C. & Schmid U. (September 2016). Rule learning from incremental presentation of training examples: Reanalysis of a categorization experiment. In T. Barowski, Z.F.Llansola, H. Schultheis, J.van de Ven, *Proceedings of the 13th Biannual Conference of the German Cognitive Science Society* (pp. 39- 42). Bremen, Germany: Universität Bremen.

Zhang, D. & Adipat, B. (2005). Challenges, methodologies, and issues in the usability testing of mobile applications. *Int. J. Hum. Comput. Interact.,* 18(3), 293- 308. doi: 10.1207/s15327590ijhc1803_3

Zhang, Y. (2009). The construction of mental models of information-rich web spaces: The development process and the impact of task complexity (Doctoral dissertation, The University of North Carolina at Chapel Hill).

Ziefle, M. & Bay, S. (September 2004). Mental models of a cellular phone menu. comparing older and younger novice users. In S. Brewster & M. Dunlop (Eds.), *Proceedings of Mobile Human-Computer Interaction*, 3(3160) (pp. 25- 37). Berlin Heidelberg, Germany: Springer. doi: 10.1007/978-3-540- 28637-0_3

Ziying Zhang. Deciphering Dynamic Decision Making Using Cognitive Modeling Including Subjective Assessment. *Master's thesis*, Technical University Berlin, 2018

THANK YOU NOTE

This thesis would not have been possible without the academic and/or personal help and support and guidance of my supervisors, co-authors, colleagues, friends, and family. Thank you!

First and foremost, I want to thank Nele Rußwinkel, my supervisor and boss, for her everlasting support and constant trust over the last five years, especially while I was on the other side of the planet. Thank you, for your excellent academic guidance, for constantly enabling this work and for always having an open ear and an open door. Also, thank you for your acute advice regarding this thesis and equally important live-practical topics. I greatly enjoyed working on the papers of this thesis as co-authors together. And I thank you for reviewing my thesis.

Second, I would like to thank Robert West for giving his time to review this thesis.

Furthermore, I want to thank my co-authors Susann Wolff and Andre Brechmann for a wonderful team-spirit during the joint paper writing, for excellent contributions and for the discussions which helped me gain new insights.

I would also like to thank the reviewers of the four papers for improving this work greatly through their guidance.

Also, thank you, to Khin Tan Win and the team for hosting me at UOW; Thank you, for providing different perspectives that greatly motivated me to write up this work.

Thanks you, to Markus Feufel for chairing my defence.

Thank you, to Lisa Dörr who supported this work greatly by not only programming the apps, helping with data-collection and pre-analysis but also by her significant contribution to the development of ACT-droid.

Also, thanks to Dominik Brüchner for developing the crawler and to many hours of technical support, discussions and coffee brakes during the most stressful time of this thesis.

Thank you, to André Brandewiede who helped with data collection.

Thank you, to my colleagues Alexander Lotz, Fabian Joeres, Martin Krabbe, Kenneth zur Kammer, Anja Marckwardt, Ziying Zhang, Hardy Smiezek, Marc Hallbrügge, Oliver Klapproth, Stefan Lindner, Alexandra Weidemann and Kai Preuß for all the fun time at work, the philosophical and critical questions and many short and not so short coffee brakes together.

Special thanks to Katrin Lüdtke for having an open-ear and helping with administrative issues and providing advice.

Thank you, to my friends and companions Sandra, Irene, Kerstin, Moni, Joe, Anna, Luara, Maria, Manja, Thekla, Lisa and Claudi for always having an open ear and giving me advice on how to deal with re-occurring crisis-perception during the past 5 years.

I would like to thank, the female researches with whom I discussed issues regarding equality, motherhood, and research careers. These discussions are empowering and helped me to identify burdens and focus on success.

Thank you to my entire family and my partners family for always being there for us, providing babysitting help, general advice, and emotional support.

Many thanks, to my grandparents for taking my side, for as long as I can remember.

Special thanks to my parents. Thank you, for constantly supported me in multiple ways. Which helped me greatly to focus on my thesis! Thank you, for being such wonderful grandparents and for providing practical help but also for putting matters into context.

I would also like to thank my sister for reminding me about the good life at the beach and taking little Maja one there as often as possible.

And finally, thank you, Patrick Ehrenbrink for being my academic mentor, my friend and companion and of course a great daddy to Maja. Thank you, for your help with all programming questions, proof-reading of papers, and not to forget the formatting brake-down comfort. But most of all for your patience, understanding and unconditional support. After nearly 10 years in academia and outside you and I have managed the good and not-so-good times in actual life and uni life together. I am greatly looking forward to what is next.