Gottfried Wilhelm
Leibniz Universität Hannover
Fakultät für Elektrotechnik und Informatik
Institut für Praktische Informatik
Fachgebiet Software Engineering

Designing an Experiment on Triggering Explanations Based on Mental Model Conflicts

Bachelorarbeit

im Studiengang Informatik

von

Carolin Isabell Kirchhoff

Prüfer: Prof. Dr. rer. nat. Kurt Schneider
Zweitprüfer: Prof. Dr. Michael Rohs
Betreuer: Jakob Richard Christian Droste

Hannover, 27.03.2023

Erklärung der Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Bachelorarbeit selbständig und ohne fremde Hilfe verfasst und keine anderen als die in der Arbeit angegebenen Quellen und Hilfsmittel verwendet habe. Die Arbeit hat in gleicher oder ähnlicher Form noch keinem anderen Prüfungsamt vorgelegen.

Hannover, den 27.03.2023

_____

Carolin Isabell Kirchhoff

Abstract

Explainability is a non-functional software requirement gaining importance as software complexity increases. Software-sided explanations are a principal way to implement explainability. This supposedly increases the user's trust and the system's transparency. Side effects need to be considered, as explainability can have a negative impact on other non-functional requirements, such as usability. This depends largely on the perspective of the user and their subjective need for explanations. Monitoring user behavior, especially in response to conflicts, as well as their own perception of conflicts could serve as a base to formulate explanation triggers, increasing the chance that an explanation is considered helpful by the user.

Knowing users' mental models may help to classify a certain type of conflict that occurs when mental model and system model do not match, obstructing the strategies to reach a goal. There is very little research on users' subjective perception of mental model conflicts. Existing studies mainly focus on the quality of the mental model, while research on explainability is usually centered around implementation techniques, disregarding the user's perspective.

In this work, we designed and piloted an experiment to assess novice users' perceptions of and reactions to a specific type of mental model conflict. To this end, we induced a mental model conflict in six study participants, using erroneous tutoring material for existing literature management software. Participants solved tasks and afterwards reported their subjective need for explanation. Their mental models were deduced from their behavior in the tasks and a questionnaire. Our results indicate that subjective need for explanation does not directly follow from a mental model conflict. Factors like participants' own experience with the software as well as conviction in their strategy and trust in their own abilities are influencing factors.

This work presents initial insights into constructing mental model conflicts and novice users' reactions to them. It can serve as a methodological basis for future research, which could assess actual reactions to explanations and identify aspects of user behavior that could serve as explanation triggers.

Zusammenfassung

Software wird immer komplexer. Deswegen ist Erklärbarkeit eine nicht-funktionale Softwareanforderung von zunehmender Relevanz. Softwareseitige Erklärungen sind ein grundlegender Ansatz zur Umsetzung von Erklärbarkeit. Hierdurch kann vermeintlich das Vertrauen von Nutzer in und die Transparenz der Software selbst gesteigert werden. Aufgrund negativer Einflüsse von Erklärbarkeit auf weitere nicht-funktionale Anforderungen müssen dabei jedoch Nebenwirkungen berücksichtigt werden. Solche negativen Einflüsse sind größtenteils abhängig von der Nutzerperspektive sowie von dessen subjektivem Erklärungsbedarf. Das Überwachen von Nutzerverhalten, insbesondere die Reaktion auf Konflikte, sowie die subjektive Wahrnehmung solcher Konflikte, können als Basis für die Festlegung von Auslösern für Erklärungen dienen. Hierdurch kann die Wahrscheinlichkeit erhöht werden, dass eine Erklärung tatsächlich als hilfreich empfunden wird.

Das Mental Model von Nutzern zu kennen könnte die Einordnung von Konflikten aufgrund mangelnder Übereinstimmung von Mental Model und Systemmodell erleichtern. Derartige Konflikte stellen Hindernisse bei der Ausführung von Strategien zur Zielerreichung dar. Die Forschungsgrundlage zur subjektiven Wahrnehmung von Mental-Model-Konflikten ist derzeit gering. Bisherige Studien richten den Fokus auf die Beschaffenheit des Mental Model selbst. Forschung zu Erklärbarkeit fokussiert hingegen technische Umsetzungsmöglichkeiten, wobei die Nutzerperspektive eher vernachlässigt wird.

Im Rahmen dieser Arbeit wurde ein Experiment zur Erfassung des subjektiven Erklärungsbedarfs unerfahrener Nutzer infolge gezielt erzeugter Mental-Model-Konflikte entworfen und pilotiert. Hierzu wurde bei sechs Teilnehmern mithilfe von fehlerhaften Anleitungsmaterialien für ein existierendes Literaturverwaltungsprogramm ein Mental-Model-Konflikt verursacht. Die Teilnehmer bearbeiteten Aufgaben und gaben anschließend Auskunft über ihren subjektiven Erklärungsbedarf. Ihr Mental Model wurde aus ihrem Verhalten sowie Antworten in einem dafür entworfenen Fragebogen abgeleitet. Die Ergebnisse weisen darauf hin, dass subjektiver Erklärungsbedarf nicht direkt aus einem Mental-Model-Konflikt entsteht. Der Erfahrungsstand der Nutzer, ihre Überzeugung von ihren Strategien sowie das Vertrauen in ihre eigenen Fähigkeiten sind weitere ausschlaggebende Faktoren.

Diese Arbeit bietet einen ersten Einblick in die Konstruktion von Mental-Model-Konflikten und die Reaktion unerfahrener Nutzer auf diese. Die Ergebnisse dienen als methodologische Grundlage für zukünftige Forschung, welche softwareseitige Erklärungen tatsächlich anbieten und Reaktionen darauf untersuchen könnte. Weiterhin muss die Umsetzbarkeit von Nutzerverhalten als Auslöser für Erklärungen untersucht werden.

Contents

1. Introduction

Nowadays, complex software is ubiquitous and there is no end in sight to growing complexity. While this increase in complexity allows software systems to aid users in processing and evaluating vast amounts of data in high-stakes environments, it also poses a risk by turning these software systems into more and more opaque black boxes. This can make it harder to understand and interpret the system's behavior, or form trust in it [11]. Explainability is a possible means to counteract these negative effects and has been considered in recent research as an addition to an established collection of non-functional requirements (NFRs) [9]. Realizing explainability, i.e., designing a system that explains itself, poses challenges in several aspects. Most notably, its explanations should be necessary, understandable and utilizable from the perspective of the user [9, 13].

Therefore, the user's perspective is central in realizing explainability. To meaningfully engage with a software system, users must hold an understanding of that system that enables them to formulate a goal, choose appropriate actions, build an expectation of the software's reaction and interpret it [19]. These cognitive processes have been termed the user's mental model [18]. A mental model can be understood as a loose and dynamic cognitive representation of the system at hand. It is not defined by its form or structure or even content but by what it allows the user to do. It is a mental representation of a system that summarizes which experiences, deductions and beliefs the user draws from when interacting with a system. These models need not be technically accurate, but they need to enable the attainment of the user's goals.

A mismatch between mental model and system model poses a threat to that goal [18]. A system that is able to mitigate negative consequences stemming from such a mismatch by offering explanations is desirable, but these explanations should be tailored to the user, with regards to timing and content. Therefore, the system needs to establish an understanding of what the user understands. Practical approaches to this task have not reached a satisfying level.

1.1. Problem statement

Due to the growing importance of the concept, approaches to definitions are manifold but a unifying definition has not been established. Several aspects of an explainable system, such as the addressee, the subject of the explanation as well as the content can differ, depending on actual demands on the system [10]. Explainability overlaps with other NFRs, most notably usability and transparency [9]. The addressee takes

a central role as both the requester and receiver of explanations. A user-centered approach is appropriate to do these roles justice.

Making a system understandable and therefore operable is in itself not a new challenge. The overlap of explainability with usability is founded in similar user-centered goals and the use of similar measures to achieve them. Years of research in the fields of psychology and human-computer-interaction has established successful usability concepts and design conventions that we follow to this day [17, 19, 20]. The majority of user-sided confusion and incomprehension that arose in the early days of interface design was mitigated by following these conventions. These insights should not be dismissed in current research on explainability. In fact, their interaction with explainability needs to be considered for beneficial interaction of both requirements. Theoretical concepts from usability research may very well aid explainability research.

The following questions address the user's perspective as well as showcase the relationship between explainability and usability: How can explanations be tailored to the user and not be unclear, distracting or disruptive? Should this additional information only be given upon an active request by the user or should the system decide whether an explanation is needed? How can the system itself identify a need for explanation?

1.2. Solution approach

We attempt to design an experiment that can answer some of the questions introduced above. To formulate a helpful explanation one needs to understand what the user already knows or believes to know about the system. An explanation should either complement this knowledge or correct it. Since the user's actions are guided by what has been summarized as their mental model, it is assumed that a mismatch between their mental model and the system they are operating can lead to difficulties [18]. Within this study we intend for novice users to experience conflict during the operation of an existing software system. We intend to construct this conflict by having the users build a non-functional but plausible mental model of the software, resulting in false predictions of the software behavior. We want to determine under which circumstances such a mental model conflict is actively perceived as a difficulty during operation of the system, and especially whether it results in a need for explanation. If such a need does not arise, an explanation might instead be perceived negatively [9]. This experiment will elicit novice users' self-reported need for explanations in response to the difficulties, with the goal of gaining insights that can allow precise timing of explanations as well as suitable content. We will ask participants who consider an explanation helpful what kind of information they would have liked to receive. Do users prefer being told what actions to take to solve the problem or do they prefer insight into the software's inner processes that resulted in the problem? We expect that their answers in relation to the self-reported difficulties and their mental model provide insight into the approriateness of software-sided explanations to resolve mental model conflicts.

1.3. Thesis structure

In Chapter 2, we establish a framework of existing research for this study, which we base our experiment design choices on. In Chapter 3, we define our research questions and outline all parts of the experiment. Chapter 4 presents the results of pilot testing, which are discussed with regards to our research questions and experimental design in Chapter 5, and with regards to limitations and threats to validity in Chapter 6. In Chapter 7 we draw conclusions to our findings with regard to our problem statement.

2. Background and related work

2.1. Explainability

2.1.1. Current impact

Explainability is an NFR that is still under-specified [6], despite interest in the topic tracing back to the 1980s [16]. NFRs are software quality characteristics that do not apply to a single functionality but to the software system as a whole. As a result, they are traditionally hard to grasp. Their evaluation is often subjective and relative, and their operationalizations can cause wanted or unwanted interactions with one another [9]. Explainability can fundamentally be described as the ability of a software system to provide explanations of itself or its own behavior, although this definition is not sufficient to guide the design process [9]. Many more detailed definitions allow a better grasp of the topic as well as implementation approaches [10, 13].

To give an exemplary scenario described in [9], a user may use a navigation system to navigate in a city not entirely familiar to them. They may have followed the suggested route several times before, when the system one day suggests a different route than usual. This may come as a surprise to the user, who may be interested in an explanation of the *whys* and *hows* of the decision made by the system. An explainable system could provide an explanation on the data that influenced the decision or give reasons for why it did not suggest the usual route [9].

Equipping a system with the ability to explain itself or its behavior is a decision that needs to be made in the early stages of development [9], when system requirements are being defined. Especially in the field of artificial intelligence, where high-stakes decisions are supported by systems only superficially understood by their users [22], explanations can help to prevent the loss of the user's trust and the system's usability. This is applicable to complex systems in general, intelligent or not. Moreover, explanations can increase the modifiability of a software system, thus facilitating practices like debugging [12, 13]. Whether problems like a decrease of trust, usability or modifiability are likely occur and whether explainability is an appropriate remedy depends on proper requirements engineering. Furthermore, successful operationalization of explainability requires thorough analysis of all aspects affecting the generation and reception of an explanation.

2.1.2. Definition and related concepts

Definitions of explainability are approached from many starting points. Philosophical and psychological understandings are among them [16]. Since we believe that insights from outside the realm of software engineering and requirements engineering enrich the discussion, we will use a definition as a starting point that resulted from a combination of the philosophical, psychological and requirements engineering perspectives. This definition excludes questions of implementation, but enable a common ground for communication around explainability [10]:

> *A system **S** is explainable with respect to an aspect **X** of **S** relative to an addressee **A** in context **C** if and only if there is an entity **E** (the explainer) who, by giving a corpus of information **I** (the explanation of **X**), enables **A** to understand **X** of **S** in **C**.*

Consider the example of a text processing system S containing an optical character recognition functionality X which converts scans to editable text. An algorithm E within such a system could explain to an end user A which parts of the scan, that the user has just provided (C), it identified as text and was able to convert. This information I may be given visually, by highlighting parts of the original scan. This should enable A to understand why the recognized text is missing certain parts that are present in the scan. S could, in contrast, provide no explanations to a user that disclose any information about the storage location and duration of scans, obscuring this aspect X' from the user.

The concrete shape of the variables in this definition can differ, depending on the specific requirements on the system. This enables non-experts, such as customers, to consider and express all requirements they have in mind when picturing an explainable system. Whether explainability is achieved in a system ultimately depends on whether the addressee can understand what is being explained to them.

Even an accepted definition of explainability cannot entirely separate it from other NFRs, with which it overlaps and interacts [9, 16]. These interactions must be known before jumping onto explainability as a universal cure for increasing software opacity.

Explainability most notably has similarities with usability and transparency. According to ISO/IEC 25010 [1], usability enables users to achieve their goals effectively, efficiently and satisfyingly by providing them with a system that is operable and learnable. Transparency essentially is the disclosure of information and therefore the opposite of opacity, i.e., the obscuring or withholding of information. This begs the question what explainability brings to the table that cannot be achieved by prioritizing usability and transparency. Notably, however, the interactivity of NFRs did not first emerge with the addition of explainability but rather is a characteristic inherent to many sets of NFRs [9, 12].

The relationship of explainability with usability and transparency, respectively, is illustrated by breaking it down into two aspects – the goals of these requirements and the measures are taken to achieve them. On an operational level, explainability requires transparency. Without disclosing information about an aspect of the system this aspect

cannot be subject of an explanation. Similarly, transparency without explainability disregards the understandability of the disclosed information [9]. Depending on a company's values, this can be an attractive way to satisfy transparency regulations without incurring a competitive disadvantage [12].

Explanations can function as a form of tutorial, making a system easier to learn for novice users. They can further the user's understanding and lessen the likelihood of operation errors. They can even enable users to "debug" a system through exploitation of its inner workings [15]. These effects contribute to the system's usability [9]. This positive relationship is symmetric, since a better understanding on behalf of the user is a goal of both requirements. However, explainability can also negatively impact usability [9]. The wording of an explanation could fail to match the user's language and level of expertise, adding obscurity instead of reducing it. Moreover, an explanation might take too long to read and comprehend in an urgent situation. An explanation provided at the wrong time could interrupt the user's workflow, making the interaction unsatisfying. Determining correct explanation triggers therefore is essential. In general, management of trade-offs is recommended [9].

Established usability design principles can help to mitigate negative effects of explanations on usability [9, 17], by simplifying the dialogue and communicating only necessary details. This is a common approach in usability design. Creating and designing an interface between system and user allows highlighting of certain features and hiding others to reduce complexity. Natural mappings and metaphors are two examples still very present in interfaces today [8, 17, 18, 20]. They abstract from implementation details by translating them. For this translation to have benefit, these design principles rely on knowledge engrained not in an individual but in a culture. This knowledge is exploited in order to decrease the amount of new information a user needs to comprehend in order to operate a device. However, such translations have limits, and as users gain knowledge about implementation details, a metaphor may not be fitting anymore and need to be dismantled in order to allow users to further their understanding of the system at hand. These guiding principles and practices have their place in interface design and should not be dismissed in research on explainability. They enable users to operate a system they only partially understand [10, 18], making systems available to non-experts. Nonetheless, this again illustrates that creating understandable explanations is more complex than committing to transparency and applying usability principles.

To summarize, explainability is an NFR closely tied to usability and transparency. A commitment to transparency or an improvement of the user experience can identify explainability as a requirement for a system. Their interactions, however, can be complex and careful analysis of goals and measures is needed to avoid unwanted negative side effects of one on the other. Especially usability can come into conflict with explainability, despite the user-centered approach that both employ. Understanding the needs of the user – and when exactly they arise – therefore is essential in implementing explainability and triggering explanations.

2.2. Role of the user

A user-centered design approach is generally recommended for explainability, due to its proximity to usability [9]. Insights from social sciences, investigating explanations as given from one human to another, can support this effort, since software should mimic communication and knowledge exchange as observed between humans, especially in the case of highly complex systems to which users may ascribe human-like traits [16]. In user-system-interaction the user – or their subjective needs – act both as the elicitor of an explanation as well as its recipient and judge of its understandability.

2.2.1. Need for explanations and explanation triggers

Once explainability is identified as a requirement to combat complexity, due to a company's commitment or legal requirements, it is easy to lose focus on the user's *subjective* need for explanation. Depending on the task or context it is possible that users do not want an explanation at all or not a *software-sided* one. They may have identified satisfying strategies for operation that let them achieve their goal without any proper understanding of the software's inner working [18, 25]. Alternatively, other sources of information than the software itself are often readily available and may be preferable, as indicated by the popularity of online video tutorials. Therefore, identifying the subjective need for explanation, regardless of the user's level of understanding, is crucial [9].

Unexpected software behavior is a common cause of a need for explanation. In the hypothetical scenario of a navigation system given at the beginning of the chapter, the majority of users considered the idea of an explanation helpful [9]. However, opinions differed on whether the explanation should be given on demand or unprompted in exceptional cases. Both approaches are promising and pose different challenges for implementation.

In generalized view on explainability Fey et al. distinguish passive explanation triggers based on monitoring system and user behavior, and active triggers via a query [13] (see Figure 2.1). Their framework is an attempt at a unifying view of self-explaining systems that abstracts from the concrete system at hand. It approaches the question of implementation strategies by identifying the kind of addressee-system interactions necessary to provide tailored explanations. Its generality allows application both for technical systems as well as humans as addressees. Within it, the explanation Ⓑ, addressee Ⓐ and self-explaining system Ⓓ are considered as individual but interrelated actors. To realize their interactions, several facets influencing one another are added. Notably, the addresse's perception is considered in detail: The algorithm Ⓒ creating Ⓑ takes into account Ⓐ's understanding of the system and background knowledge ⑪. This represents the need for understanding of the communicated information on behalf of the user and adds a reference point for an explanation-generating algorithm to relate to. The user's knowledge or sometimes lack thereof is the an existing set of beliefs to which an explanation should add.

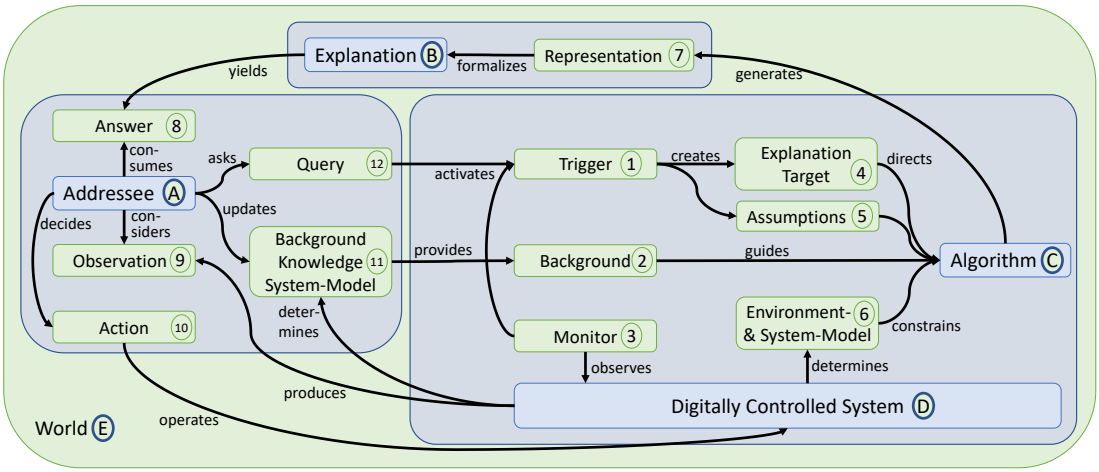

Figure 2.1.: Generalized approach to self-explaining systems [13]

In addition to providing a baseline set of user beliefs, (11) is also involved in triggering explanations. In the case of a human addressee, we suggest that the addition of an explicit connection between (11) and (A)'s inputs to the system is reasonable, since actions and strategies are derived from a set of beliefs about the system [18]. Despite the lack of an explicit connection in the model, Fey et al. consider *action-oriented explanations* in their work, which the addressee uses to derive appropriate further actions, as well as update their system model. Therefore, (11) guides (A)'s actions. Subsequently, (D) reacts to these actions. This behavior is observed both by (A) and by an system-internal monitor (3). From here on, explanation triggers can be realized actively or passively. A query (12) by (A) constitutes an active trigger. Alternatively, exceptional system behavior causes (3) to passively trigger an explanation, bypassing (A). Similarly, exceptional user inputs can be monitored to trigger an explanation.

Active explanation triggers are preceeded by a subjective need for an explanation, else the user would feel no need to request one. In contrast, passive explanation triggers assume or should assume a need for explanation on behalf of the user. That means that it is not exceptional software behavior per se that should be monitored but *unexpected* system behavior, more specifically, system behavior that does not meet the *user's* expectations.

In order to determine what constitutes such unexpected behavior that conflicts with users' expectations, one needs to identify what the usual expected behavior is, possibly based on evaluation of past system behavior and sequences of user actions. Notably, the software's behavior may not be "unusual" at all but still contradict a user's expectations.

Implementing passive explanation triggers linked to unexpected system behavior or exceptional user behavior have the benefit that users who are unaware of a dissonance or an error within their understanding of the system, could still receive an explanation. However, this carries a risk, as it overrules the user's decision and can impair the

interaction [9]. What constitutes exceptional user behavior is a highly individual decision. Examples are canceling and restarting dialogs or long sequences of inaction. In contrast, unexpected system behavior can be investigated more easily by eliciting the user's understanding of the system. Once identified, users can report whether they experienced a need for explanation when confronted with this unexpected system behavior. This is the approach we want to investigate in this work.

Being able to reliably identify cues in the user which indicate a need for explanation, not just confusion, would minimize the risk of bothersome explanations. If such cues can be found, they could be taught to the system to realize the paths outlined in Figure 2.1.

2.2.2. Modeling the user

A self-explaining system with the ability to understand what the user is thinking could satisfy a need for explanation as soon as it arises. With regard to the framework proposed by Fey et al., the addressee has several roles. They act as a system operator, explanation requester and judge of understandability with regard to their background knowledge and system model. The self-explaining system holds some belief about this knowledge. This is what its explanations, content-wise, must relate to.

Miller, in a very extensive survey of research in the social sciences on explanations, provides a similar argument [16]. He states that an explanation-generating system requires a *Theory of Mind*, a psychological concept describing the ability of an entity to infer the mental states of another entity and explain and predict their behavior [4]. Humans acquire this ability naturally at the age of around four years and use it in communication [5]. Human-machine communication could benefit from mimicking this process.

Approaches of monitoring the user's interaction with a system to infer their theory of mind have been applied for digital games but also for more generalized kinds of systems [4]. These approaches promise intelligent system assistance, by identifying strategies that are not goal-directed (in the words of the authors "stabbing around in the dark"). An overview over systems that have employed the idea of user modeling is given in [16]. In these works, the system establishes an understanding of the user it is interacting with in order to provide tailored explanations with regard to the content. Applying such a model to compute the timing of an explanation, i.e., a trigger, is not considered in these works.

Modeling the user is a valid approach to identify problems during system operation. Nonetheless, the gap between difficulties and need for explanation persists. To close it, modeling the user's strategy and understanding of the system, including their perception of difficulties and errors is necessary. Since computers and software are ubiquitous nowadays, users approach software systems with some existing level of understanding. This understanding, correct or incorrect, guides the interaction [13]. It allows the user to form a strategy, which is executed as a series of actions, to which the system reacts. Observations of these reactions continuously influence the user's understanding of the system, completing a feedback loop. A conflict in this interaction potentially leads to a need for explanation.

An externally created representation of the user's background knowledge and system understanding can serve as a placeholder to shed light on the effects of difficulties on the need for explanation. In the following section, we will discuss mental models as a framework that can take this place. A characterization of difficulties during interaction that relates to the user's mental model can subsequently be used to identify the demands for a system-sided modeling of the user.

2.3. Mental models

2.3.1. Established conceptualizations

The mental model, a theoretical concept proposed by Norman in 1983, entails the user's understanding of the system they are working with [18]. The term mental model summarizes all internal conceptions that users develop in order to effectively operate a system. These conceptions and beliefs aid the user in choosing appropriate actions and predicting the software behavior in the attainment of their goals. Mental models are specific to each individual user and therefore lead to different strategies. The term itself is based on the idea that internal conceptions have a degree of similarity to the system they represent, the *target system*, thereby modeling it.

Originally, Norman distinguishes the target system and the model of that target system. In the following, we will refer only to the system model. The distinction between system and system model is not particularly relevant to this study, since we assume the system model to be technically and functionally accurate and comprehensive.

Mental models differ from the system model in their extent and level of detail, practically never presenting a comprehensive picture of the entire system. They may also contain inconsistencies as well as superstitions. These properties often lessen the cognitive load on the user [18]. Despite – or sometimes because of – these differences, they are functional by definition. A user's mental model is whatever cognitive structures and mental representations they draw from to attain a goal. Knowledge that is not strictly required is omitted. One can ascribe a predictive nature to these models, allowing planning of actions and evaluation of system reactions [18].

Of course, the inherent characteristic of functionality cannot mean that users never make mistakes. If a user encounters an unexpected problem, they follow problem-solving strategies concordant with their mental model [14]. If the mental model itself proves to be dysfunctional, it is updated [18]. Therefore, one can argue that with increasing experience with a system, the mental model continually evolves and maintains functionality. This leads to the question of mental model formation. It is assumed that mental models are formed upon experience with a system and similar systems [18]. Therefore the user's background, like their general technical abilites, influence the mental model as well.

2.3.2. Critique and working definition

The original theoretical framework proposed by Norman is not readily applicable to practical approaches because it lacks operational character. Put sharply, the mental model has no scientific, i.e., neuropsychological basis. It is a layer of abstraction between cognition and action. Observations can only be made indirectly. If observable variables, such as behavior and introspection, contradict, the question arises of who holds the sovereignty of interpretation. For example, Norman himself mentions the risk of the user rationalizing their behavior after-the-fact, even though there was no reasoning behind it at the time [18]. The question of whether the rationale was created *because* of the introspection is very hard to answer.

Since the purpose of this study is to better understand explainability as an NFR, and mental model conflicts as a potential trigger for software-based explanations, we are specifically interested in the mental model of users as it predicts and explains software behavior. It would be beneficial to be able to elicit the strength of users convictions and their basis, such as extensive experience or logical conclusions.

Within this work we will therefore adhere to the following defining attributes and use them to define operationalizations:

1. Human users operating any system employ planning and strategy, however basic, short-term, inefficient or illogical the strategy may be. Any cognitive processes and resources underlying planning and strategizing are summarized under the term mental model.
2. Mental models are not stable. They may change over time or as a response to new information.
3. Mental models contain conscious as well as subconscious aspects and are therefore not fully known even to the person holding the model.

The predictive nature of mental models follows from the first attribute. Planning is a mental process of choosing appropriate actions and ordering them to reach a goal [21]. Goal evaluation completes a prediction. Planning is a largely conscious process [21], allowing conclusions about the mental model both from analysis of behavioral data as well as self-reports.

Subconscious aspects of the mental model may especially influence actions that do not follow a prediction-evaluation pattern, like intuition. Such aspects elude themselves from self-reports, making them less reliable and increasing the importance of behavioral data. Our definition lays a foundation, but previous research must be thoroughly analyzed for different conceptualizations of mental models and the elicitation methods used.

2.3.3. Mental model conflicts

Linking the subjective need for explanations with the mental model requires a conceptualization of a conflict as it occurs during the operation of a system. We define a mental model conflict as the mismatch of a user's prediction of the software's behavior with the actual behavior of the software, as it occurs during the execution of some strategy. Within this work, we therefore focus on users' expectations rather than actions taken without conscious reasoning.

2.3.4. Elicitation

Our justification of defining attributes of mental models is given in the light of methodological pitfalls evident in research from previous decades: The elicitation of mental models is inherently challenging [18, 25]. Strategies such as think-alouds, verbal protocols, online protocols, problem-solving performance, information retention over time, observations of system use, users' explanations and predictions of system behavior have been used as indicators of the mental model. An evaluation of common elicitation techniques can be found in [25]. Even if self-reports and performance data are combined, their analysis still entails methodological risks.

Merely observing user and system behavior allows no real inference of the user's reasoning. The same can be true for self-report prompts. Asking users to draw a picture of their understanding of specific software [2, 24] or more generally, a technical system [26], leads them to draw just the software's graphical interface. This can be frustrating for researchers, as it is difficult to decide whether the result is the extent of the user's mental model or whether the prompt was insufficient at eliciting it. Both Anders et al. and Zhang found written or oral data equally rich or even more informative than a drawing or the explanation of a drawing [2, 26]. We therefore assume that a visual representation only appears to free the user from the constraints of textual representation, when they in fact constrain them to visual aspects of their mental model. Those visual aspects are then not easily supplemented with non-visual aspects.

Another aspect to be considered is the vagueness of the prompt or question. A very vague wording, especially when it is not related to specific software, might elicit a wide variety of answers across participants or even knowledge and beliefs irrelevant to operating a system. For example, in [26], users were asked to "draw a diagram or picture of [their] perceptions about the Web", and the relationship between mental model and performance in a search engine task was investigated. This elicited results, of which some were very technical (showing connections between computers, routers, servers and modems), while others showed a browser interface with the Google start page. One cannot be certain that users having a technically accurate understanding of the internet did not *also* have a functional view of it – we believe it is likely they did but found the technical drawing a more appropriate response to the question. The drawing as a response to the prompt does not allow the conclusion that this aspect depicts the cognitive resources users retrieved when completing the search engine task. We

therefore suggest that self-report prompts should be worded as close to a specific use case, task, or goal as possible.

Evaluating behavior in any kind of test or task in terms of success rate and efficiency allows a conclusion about the degree of functionality of the current mental model but not necessarily its nature: According to Sasse [25], even successful task completion can be based on “the most appalling misconceptions”. A similar thing is true for the opposite: Errors made by users can be the result of slips or orientation problems, not a lack of understanding of the system’s functioning, as stated in a critique of several studies in [25]. This is especially true for samples consisting only of novice users. To determine the nature of errors, users should be asked how they perceive their errors and what they expected to happen [25]. Users may, upon making errors in a benchmark test, form opinions on the instruction materials given to them, judging them as unhelpful and disregarding the information obtained from them, causing divergent, potentially inefficient, strategies or further errors. Ignoring the possibility that users do not consume and follow instruction materials uncritically leads to wrong assumptions about the mental model they are actually holding. It is therefore critical not to base judgments of the mental model quality solely on success rates in benchmark tests.

Studies in educational contexts go even further in their assumption that a mental model correctly mirroring the system model is preferrable, regardless of performance measures. In a study by Papastergiou, users are given a large set of questions, with arguably suggestive character, testing not only the nature but the correctness of the mental model not with relation to performance but similarity to the system model [23]. Some questions presumed the existence of certain conceptions. An example is the question “What do you think a server is and does?”. Servers and their role in the internet may not be a part of the individuals mental model. One can interact with the internet without considering servers. Making the user feel like they are under examination is to be avoided especially with self-reports. A preset line of questioning, as with a questionnaire, can easily induce such a feeling and affect answers.

Outside of experimental settings, users are often not required to even use a specific system or software. They may decide to switch systems if they perceive it lacking in usability, like a calculator requiring inputs in reverse polish notation as used by Halasz and Moran [14]. Nonetheless, Halasz and Moran claimed that forming a mental model that is so logically coherent that it allows more effective task-solving, especially in transfer tasks, should be encouraged through the system’s design [14]. However, users presented with usability so low that an extensive mental model is required to solve basic tasks, would probably prefer to use a system with better usability altogether. Therefore, generalizing results from artificial experimental settings, especially obscure and unnecessarily complicated software, is questionable. The users’ motivations outside of experimental constraints should always be scrutinized.

How and under which circumstances users update their mental model is experimentally underrepresented. Applying insights from overly artificial experimental settings using obscure software is dubious. Nonetheless, if the assumptions made when generalizing experimental results are kept track of and minimized where possible, connecting the mental model framework to explainability seems a promising approach. Neverthe-

less, methodological flaws from previous research need to be considered, despite the difference in research questions.

A user's mental model summarizes their knowledge about a system and their approaches to operating it. Insights into the relationship between mental model conflicts and subjective need for explanation can be integrated into a software-held model of the user's knowledge. This might enable explainable software that triggers explanations only when required.

In this work, we present a pilot study aiming to explore mental model conflicts with regard to existing software. We aim to implant these mental models with errors, causing a dissonance between the system model and the user's mental model. The purpose of this is to establish an experimental framework to assess how mental model conflicts may be detectable and how they affect users' need for explanation. This approach aims at closing the knowledge gap between user errors and need for explanation to enable self-explaining systems to provide explanations when needed.

3. Research goal and design

3.1. Research goal

We defined three research questions (RQs) for this study and used the Goal Question Metric (GQM) [3] to identify our metrics. Tables 3.1, 3.2 and 3.3 in the following subsections show the respective abstraction sheets. Details on the instruments used to apply the metrics are discussed in Section 3.2.

3.1.1. RQ1

Purpose	**Issue**	**Object**	**Viewpoint**
Investigate	overlap	system model and mental model	novice users
RQ1: Can we construct a mental model in novice users by presenting a video tutorial of the software?			
Metrics **a** self-reports of experience and understanding of Citavi **b** strategies employed when solving tasks in Citavi or similar common literature management software		**Variation factors** - video tutorial communicating mainly correct system model of Citavi with intentional incorrect aspects (errors)	
Baseline hypotheses **a** novice users report no previous practical experience with Citavi or similarly common literature management software **b** novice users have no meaningful strategies to solve tasks in Citavi*		**Impact on baseline hypotheses** - both correct and incorrect aspects of the video tutorial reflect in **a** and **b**	

*Baseline hypothesis was not measured but inferred from metric **a**

Table 3.1.: GQM abstraction sheet for research question 1

We intend to observe the effects of mismatches between a user's mental model and the system model of the software they are using. To ascertain the existence and nature

of such a conflict, we plan to construct it. In the context of our experiment we view the system model as “fixed”. We are using existing software, namely Citavi 6 (QSR International, February 2018). We need to construct a mental model in the user that differs from the system model in chosen aspects. Since this process serves as the foundational first step of our experiment, we need to make sure that the instructional material presented to the participants has the intended effect. We intend to test novice users without previous experience using the software, assuming that they have not yet formed a mental model specific to the software. We will then present them with instructional information explaining how the software works and how to complete certain tasks. We will purposely place erroneous information in the tutorial, hoping the users will form a mental model on the basis of the information given to them. Users will then be asked to explain how the software works in order to elicit their mental models. Additionally, their task-solving strategies will be analyzed from screen recordings.

3.1.2. RQ2

Purpose	**Issue**	**Object**	**Viewpoint**
Investigate	perception	mental model conflict	novice users
RQ2: Does a mismatch between the user’s mental model and the system model cause a conflict in the user?			
Metrics **a** self-reports of strength and nature of difficulties for conflict tasks **b** self-reports of strength and nature of difficulties for no-conflict tasks **c** task-solving behavior in conflict tasks **d** task-solving behavior in no-conflict tasks		**Variation factors** - match between mental model and system model - mismatch between mental model and system model (due to constructed errors)	
Baseline hypotheses **b** match between mental model and system model shows negatively **d** match between mental model and system model shows negatively		**Impact on baseline hypotheses** - mismatch between mental model and system model shows positively in **a** and **c**	

Table 3.2.: GQM abstraction sheet for research question 2

We want to design an experiment that serves to contribute to our understanding of the role of mental model conflicts in the need for explainability. We expect to be able to successfully construct a user’s mental model that differs from a system model in small but fundamental aspects that are relevant for the completion of certain tasks within the

software. It is yet unclear how users react to being faced with such a mismatch. By answering **RQ2** we hope to gain further insight into the ways users perceive, react to and cope with such conflicts. After having completed a number of tasks in the software at hand, some of them designed to make them aware of the errors in their mental model, users will be asked to report any difficulties they had during task completion.

3.1.3. RQ3

Purpose	**Issue**	**Object**	**Viewpoint**
Determine	presence	subjective need for explanation	novice users
RQ3: Does a conflict between a user's mental model and the perceived system behavior result in a subjective need for explanation?			
Metrics **a** self-reports of need for explanation and respective desired explanation content		**Variation factors** - perception of strength and nature of mental model conflict (as measured in **RQ2**)	
Baseline hypotheses **a** absence of perceived mental model conflict (as measured in **RQ2**) relates to no self-reports of a subjective need for explanation		**Impaxt on baseline hypotheses** - strength and nature of mental model conflict correlates positively with **a**	

Table 3.3.: GQM abstraction sheet for research question 3

When users are faced with system behavior that does not match the expectations formed on the basis of their mental model, they may feel a need for an explanation of said behavior. This is, however, only one way of reacting to such a conflict. Users might also simply "accept" the mismatch, update their mental model and actively figure out a way to complete the task anyway. Alternatively, they might experience frustration and not perceive the idea of an explanation as helpful. By answering **RQ3** we want to clarify whether and under which circumstances these alternative reactions can be excluded. If the proposed mental-model-system-model conflict results in a subjective need for explanation, monitoring the user in order to detect such a conflict might serve as an essential trigger for explanations provided to user, as previously discussed in Chapter 1 and Chapter 2. After reporting on potential difficulties during task completion, participants will be asked for their subjective need for explanations and what a helpful explanation could have looked like.

3.2. Experiment design

3.2.1. Subjects and experiment overview

Participation in the experiment was voluntary. Participants were recruited at the Leibniz University of Hannover. All participants provided informed consent before participation. Participants were told that the purpose of the study was to determine the effectiveness of video tutorials for learning new software, to direct their focus away from their own performance, encourage them to report any difficulties they had during task completion and not feel like their intelligence or technical ability were being tested. After completing the experiment, participants were given information on the true purpose of the study.

The experiment is summarized in Figure 3.1. Participants provided informed consent about participation in the study, then proceeded to fill out a preliminary questionnaire, asking their technical abilites and experience with citation software. Next, they watched a tutorial video on Citavi 6. In a following intermediate questionnaire participants were asked to answer questions about how Citavi functions in order to elicit their mental models, as influenced by the tutorial video. Participants were then presented with four tasks in total, grouped into the two topics discussed in the video (importing literature and searching through imported literature). For each thematic cluster, one task was designed to be completed successfully with the information given in the tutorial, the other task was designed to steer participants into the respective error in the communicated system model, with the intention of causing a conflict between their mental models with the actual system behavior. While participants worked on the tasks, the screen was recorded. Afterwards, participants filled out a questionnaire in which they were asked to report any difficulties they had while working on the tasks and whether they would have liked an explanation from the software. Lastly, they filled out a demographic questionnaire.

3.2.2. Preliminary questionnaire

To be able to detect a potential relationship between general technical expertise and difficulties during task completion or need for explanation, participants self-assessed their general level of expertise at using technology in comparison to the German population, and reported on any previous experience with Citavi or similar citation software (Zotero, Mendeley).

We specifically recruited subjects that had no prior experience with Citavi, but we wanted to be certain they had not previously used similar citation software either. Previous experience could have easily confounded the effect of our tutorial video. Experienced users could have detected the constructed errors in the system model too quickly, or they could have built a rather tried-and-tested mental model already and not let the contents of the video affect it.

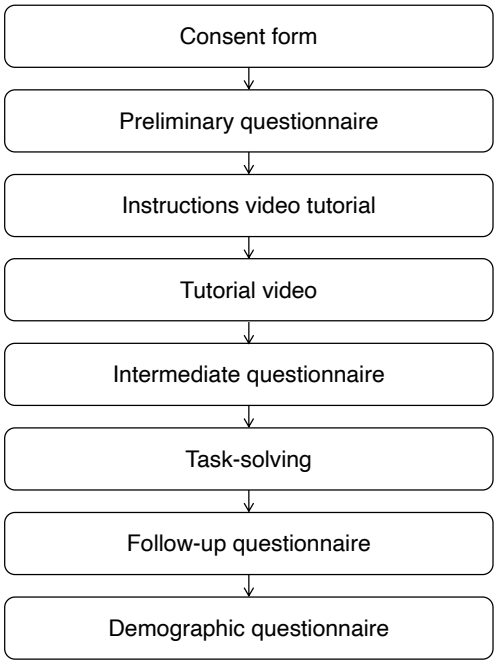

Figure 3.1.: Sequence graph of experiment structure

3.2.3. Tutorial video

Participants were shown a tutorial video of 7 minutes and 46 seconds in length that consisted of simple animated diagrams as well as screen recordings of the Citavi interface, to allow users to feel like they observed someone else using the software. Voiceover explained two general use cases of Citavi. First, how to import literature into the project, and second how to search for titles using search criteria. Script and voiceover were provided in German to avoid any language difficulties. (see Appendix A.1.1 for the original script and Appendix A.1.2 for an English translation).

The script for the voiceover was based exclusively on information from the official Citavi manual [1]. We cannot guarantee that the system model we are basing the tutorial video on is complete and correct, since we do not own Citavi and have no insight into the source code. However, we based our understanding of the system exclusively on official and publicly available material, such as the Citavi 6 Manual. Even in the case that we have derived an incorrect system model, our system model is still correct from a practical standpoint in that it works for the tasks we have set for the participants.

The popularity of software tutorials on video platforms such as Youtube indicates that it is one of the most popular means of teaching oneself how to use new software. Not many new users refer to extensive manuals but rather search for video tutorials by independent parties. This is evidenced by the fact that Citavi does not only provide a textual manual online but also short video tutorials on Youtube.

[1] https://www1.citavi.com/sub/manual6/de/index.html (last accessed: 13.12.2022)

We expected mental model conflicts consisting of disappointed predictions, of which users were very convinced, to be more noticeable to the participants themselves and more resistant to workarounds than conflicts in which users were not entirely sure how Citavi would react. This is why we decided to focus the video on the internal mechanisms of the software but communicate it in a way that is easily understandable to laypeople. To achieve this, we included wordings that focused on Citavi as the actor, reacting to and processing input from the user instead of imperatives that merely tell the user what to do when, but not how the software processes the input. Especially for the two functions we focused on (importing literature and searching imported literature), there exist several redundant options in the interface to achieve one's goal, all of which (presumably) use the same *inner* mechanism which we decided to communicate to the participants instead, abstracting from the interface.

Participants were allowed to pause, rewatch and skip around the tutorial video as well as take notes which they were allowed to use during the tasks. We reminded participants that they would be given tasks to solve in Citavi following the video. We did not restrict them in how many times they watched the tutorial, because the efficiency with which participants formed a mental model sufficient to solve the following tasks was not of interest to us. They were told to watch the tutorial until they felt like they had appropriately understood it. They were however not allowed to ask the instructor if anything in the video was unclear to them to avoid giving different information to individual subjects, thereby potentially influencing how they went about solving the tasks. We hoped that notes taken by the subjects would give additional insight into their mental models.

Importing literature

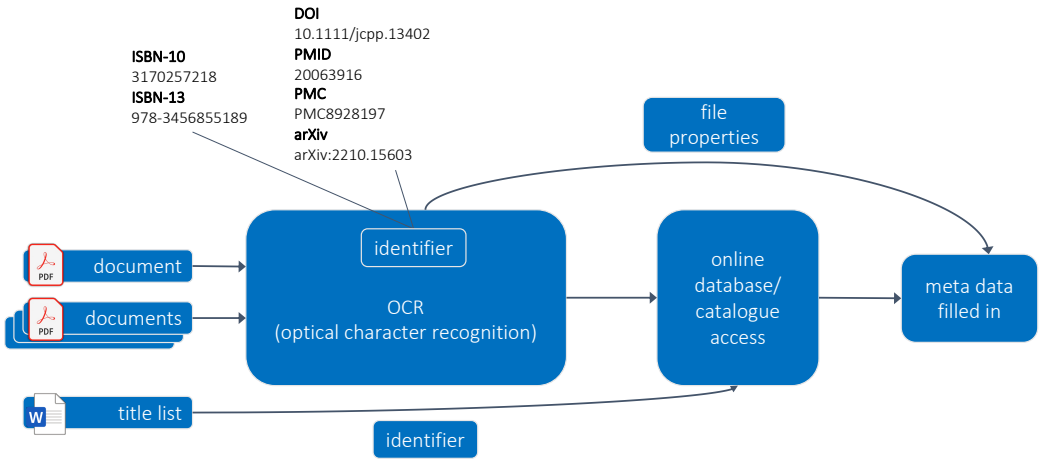

Figure 3.2.: Diagram on the 'import literature' function complex in Citavi 6

The function complex for importing literature in Citavi 6 was the first to be explained in the tutorial video. The diagram shown in Figure 3.2 is correct according to our system model of Citavi 6. We explained to subjects that Citavi accepts PDF documents and processes them via OCR to extract an identifier (such as a DOI). Citavi searches the first five pages for an identifier. If the search is successful, the identifier is used to access online databases or catalogues to obtain correct meta data for the title. If Citavi cannot find an identifier, it uses file attributes to fill out the title field and number of pages according to the file properties. Furthermore we explained that Citavi can also process (unfixed layout) text formats, such as Word files, but does not need OCR to discover identifiers. In truth, Citavi does not need OCR to process so-called “true” text-based PDFs, as it can obtain and search the file text from the file’s code. However for scanned or image PDFs, that do not contain the depicted text in the code, Citavi uses OCR. This process is however not very robust and often fails to decipher even slightly rotated or blurred text. We simplified this distinction in our communicated model for the already mentioned reason of not cognitively overloading our participants and decided to merge distinctions made between text-based PDFs and scanned/image PDFs. Assuming that all PDFs are processed with OCR leads one to a more functional mental model than assuming that only text-based formats can be processed, which is why we designed our system model that way.

We then proceeded to show how to access these functions the Citavi interface. Examples included a PDF file of a scientific article containing a DOI, a PDF file of a scientific article containing no identifier, and a Word file containing several identifiers of different formats. All files were imported via a Drag & Drop functionality. This is not actually possible in Citavi 6. The video was manipulated to show a successful import including correct meta data of all titles corresponding to the identifiers in the Word file.

In truth, Citavi can, in accordance with Figure 3.2, import literature via identifiers from a Word file. However, this requires a different sequence of action in the interface, namely clicking *ISBN, DOI, other ID* and in the dialogue that opens choosing the option *From a file*. Furthermore, for files imported via Drag & Drop Citavi only searches the file text until it discovers the first identifier, which it then uses to obtain meta data.

Searching imported literature

The second part of the video explained how to search and navigate through imported literature. Citavi can search titles by their meta data fields, such as author, title, subtitle, publication date, etc. Search terms can be specified by shorthands for a specific field or group of fields. A table containing all possible shorthands and their respective meta data fields was shown, followed by examples for combining these shorthands with wildcards, namely (*, ?), relational operators (<, >, <=, >=, =) and logical operators (AND, OR, NOT) shown in Figure 3.3. One of the shorthands is *ft* (full text) which searches the file text of attached files. Among the example search expressions we included “*ft*: Ver*ung”, explaining that it is possible to use wildcards in the full text search. This is however not true. Wildcards and other relational operators can be used for all other shorthands but not in the full text search, where the search term is treated

as a literal string. Similar to the first half of the video we proceeded to show how to use the search functionality in the Citavi interface, focussing on the so-called *Quick Search* where the above-mentioned options are available. Examples included a full text search for the term “sampl*”, explaining that this returns titles containing words like *sample*, *sampler*, and *sampling*. Before recording the video we manipulated a number of articles by adding a page with the literal search term “sampl*” to the attached PDF.

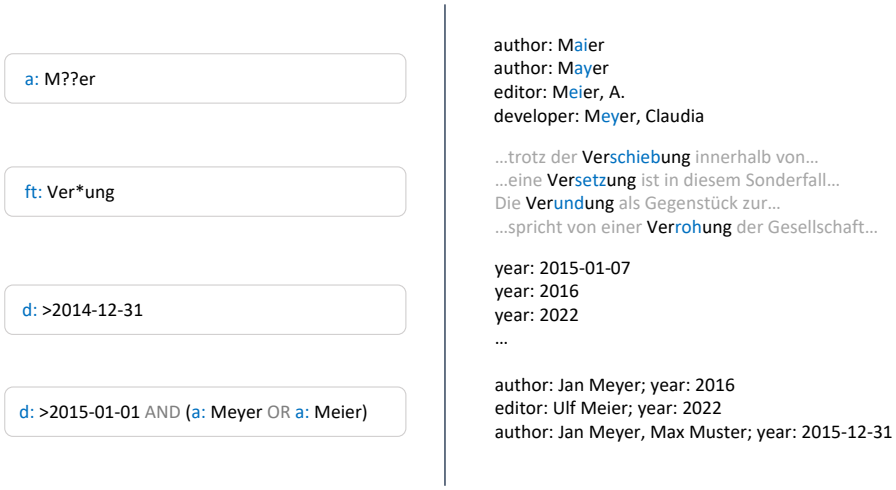

Figure 3.3.: Examples for the ‘search imported literature’ function complex in Citavi 6

3.2.4. Intermediate questionnaire

When participants finished watching the tutorial they were asked to answer three questions about the functionality of Citavi in their own words.

1. How does Citavi work?
2. How does Citavi handle identifiers (ISBN, DOI, ...)?
3. Which attributes of titles (books, journal articles, anthologies, ...) in a Citavi project can be searched?

We see the combination of behavioral data with introspection as the most promising mental model elicitation method, the reasons for which we have layed out in chapter 2.

Answers were to be interpreted as supplementary indicators of the nature of the mental model.

We decided to ask subjects for written answers to the three questions above but gave them the option to draw a supplementary diagram, following insights from studies discussed in chapter 2 [2, 24, 26]. The second and third question are more task-specific

than the first question, in order to prevent subjects from giving only task-irrelevant answers.

3.2.5. Task

Subjects were given four tasks in total, two per thematic cluster, one designed to be solvable with the system model communicated in the tutorial video (no-conflict task), one designed to lead subjects into the "trap" constructed through errors in the communicated system model (conflict task). An overview over the task structure and content is given in Table 3.4.

	Conflict task	**No-conflict task**
Literature import tasks	**Task 1.1** Participants were given a Word document containing a number of identifiers and told to add the corresponding titles to their project and make sure that as many fields as possible were filled in correctly.	**Task 1.2** Participants were given a folder containing a number of PDF files of books and scientific articles and again told to add the titles to their project and make sure that as many fields as possible were filled in correctly.
Search tasks	**Task 2.1** Participants were to find all titles whose full text contained words beginning with "un" and ending with "ing". Example matches *understanding, undermining, undoing* were given.	**Task 2.2** Participants were to find all titles containing the word "nuclear" in the heading, published before 2000 or after 2015.

Table 3.4.: Structure of tasks solved by participants in the experiment

None of the tasks depended on successfully solving another. Participants were free to choose the order in which they solved the tasks.

Participants were encouraged to let the instructor know if they had trouble solving the tasks but were also told that the instructor would not be able to help them.

All tasks corresponded to aspects of the system model given in the tutorial and required participants to make use of their acquired knowledge. For Tasks 1.1 and 1.2 participants at least had to remember that Citavi can use identifiers of different formats to provide meta data for a title. The communicated system model should support the conclusion that Citavi can more easily process a pure text file (like a Word document) than a fixed

layout text file potentially containing scanned text (like a PDF). Since the only import option explained to them was via Drag & Drop onto the title list, they should be confused as to why a Word file is not processed correctly, i.e., why almost no fields are filled in correctly. For Tasks 2.1 and 2.2 participants had to have the understanding that Citavi can search different meta data fields as well as the file text, and that relational and logical operators as well as wildcards can be applied. Since the video showed a wildcard search within file text, we expected participants to be confused as to why they did not find any results, despite several files containing words matching the search term.

Tasks and strategies shown in the tutorial were chosen so that Citavi gives no explanation or hints on its own as to why certain actions are not possible. Notably, this was quite a difficult task, since many dialogue windows in Citavi contain explanations, sometimes even dynamic, responding to the user hovering over a button.

During task completion we recorded the screen. This allowed assessment of participants' strategies and difficulties, as well as their objective success in solving the tasks. We expect their behavior to reflect participants mental models.

3.2.6. Follow-up questionnaire

In a questionnaire following the tasks in Citavi, participants answered the same questions for all four tasks, followed by a question regarding the video tutorial.

This follow-up questionnaire served to answer **RQ2** and **RQ3** by having participants report difficulties (indicating mental model conflicts) during task completions and give their opinion on the helpfulness of an explanation.

For each task participants were asked whether they worked on the task at all and if so, whether they experienced none, mild or severe difficulties, on which they were then asked to elaborate. This was meant to ensure that the difficulties they experienced were (at least largely) due to a conflict between their mental model of Citavi and the behavior Citavi exhibited.

We then asked participants if, regarding any difficulties they experienced, they would have perceived an explanation given by Citavi as helpful or generally unhelpful. In either case participants gave reasons for their answer and in case they would have found an explanation helpful they described what such an explanation should have looked like. This targeted users' subjective need for explanation.

Lastly, we asked participants to recall the video and make suggestions for improving it with respect to its content, if they considered it worth improving. This served two purposes: If participants identified the errors in the video, we would be able to assess how this influenced their behavior. Additionally, it would allow us to evaluate users' attribution targets.

3.2.7. Demographic questionnaire

Lastly, participants filled out a demographic questionnaire including their gender identity, age, native language(s), highest academic and school-leaving qualification and their current employment or educational path. This was done for the purpose of assessing any expected or unexpected effects of any of these factors on the results of the experiment.

3.2.8. Data analysis

Free-text replies in questionnaires as well as notes by the participants were given in handwritten form and transcribed. Behavior during task-solving was screen-recorded and summarized by the experimenter in written form. No coding methods were used for the summary.

4. Results

Six participants were tested for this pilot study, all of which were included in the analysis. 50% identified as male, 50% identified as female. Ages ranged from 20 to 22 years. All except one were German native speakers. Participants had received 12 to 14 years of primary and secondary education and all had completed high school with Abitur. All of them were enrolled in bachelor's degree programs and had completed one to five semesters. The majority of participants were enrolled in a technical study program (1 computer engineering, 4 computer science, 1 English and History double degree). None held a university degree yet.

Only one participant had heard of Citavi before but never used it, all other participants had never heard of Citavi. None had heard of Mendeley or Zotero before. Participants were diverse with regards to their self-reported general technical abilites in comparison with the German population. On a five-point Likert scale (*far below average – below average – average – above average – far above average*) answers ranged from *below average* to *far above average*.

For the sake of clarity, we will identify each of the six participants by numbering them.

4.1. Tutorial video

Five participants spent between 9 and 11 minutes watching the tutorial video. Participant 2 watched the video for a total of 27 minutes. They paused the video for extended times and skipped back to re-watch certain sequences several times.

All participants took written notes during the tutorial video. The amount of notes taken by subject 2 far exceeded the other participants' but followed the tutorial script very closely. Scans of the notes are included in Section B.1.

In general, notes were very task-oriented. Only subjects 2, 4 and 6 took notes on the purpose of Citavi as a whole. The inner workings of Citavi were noted by two participants.

The Drag & Drop functionality was almost never noted, only four subjects took notes on the import functionality at all.

All notes included some or all of the shorthands for searchable attributes, as well as the option of using two types of wildcards (*, ?), relational operators (<, >, all except subject 4), and logical operators (AND, OR, NOT, all except subject 2). Syntax examples were noted by two participants but were very minimal. This caused difficulties during the tasks.

4.2. Self-reported mental models

Despite being given the option to draw a diagram or pictures, answers in the intermediate survey were given in text form only. Across all three questions, participants wrote between 7 and 59 words (median = 14.5 words).

The original German answers can be found in Appendix B.2.1, as well as English translations in Appendix B.2.2.

Question 1: How does Citavi work?

Despite being given lots of space to write out their answers, they extended a single sentence in the case of only one subject. Participants expressed their understanding of how Citavi functions on a use-case level (“a software for helping with essays”, “creating a reference list”, “making highlighting and copying easier”), an action level (“files can be imported via Drag & Drop”), and inner workings (“Citavi searches for the meta data of a medium within an online database”). Answers often were a mix of several levels. The imprecise wording of the question allowed for these different interpretations. Answers relating to the inner workings of Citavi were not very detailed but correct overall.

Question 2: How does Citavi handle identifiers (ISBN, DOI, ...)?

The wording put Citavi as the actor of a more specific action and therefore constrained users more in the interpretation of the question. Only one-sentence answers were given. Answers were more technical than with Question 1, but often imprecise or incorrect, only stating that meta data is “obtained” but not how, or that Citavi searches the internet or its own database for identifiers or meta data. In this sense, answers partly expressed the use case of automatically adding meta data, and partly expressed, with varying degrees of precision, which technical steps on behalf of Citavi are involved in achieving this.

Question 3: Which attributes of titles (books, journal articles, anthologies, ...) in a Citavi project can be searched?

Participants all listed searchable attributes, in two cases in combination with their respective shorthands in the search syntax, or contrasted “regular” attributes with the full text search. Answers were fundamentally correct and contained the most common and useful searchable attributes. The full text search was included in all but two cases, in which only the example shown in the video was paraphrased (“word beginnings/endings”) or in which only a “free text input” is referenced. Two answers are indicative of at least a basic understanding of the search syntax (shorthands). The other answers allow no assumptions beyond the use case implied in the question wording being understood, which is that one might want to find a set of titles characterized by data in specific fields. More advanced syntax options like logical or relational operators and wildcards were not mentioned.

4.3. Task-solving, self-reported difficulties and need for explanation

Except for participants 3 and 6 who did not work on the conflict-free search task (Task 2.2) at all due to the time constraint, all participants worked on all four tasks. Except for participant 1 (18 minutes) everyone took the full 20 minutes to work on the tasks.

Table 4.1 gives an overview over the strategies used by each subject for each task. Subjects following the video's strategies partially are discussed in more detail in the following sections. A summanry of the success rates for all tasks, the number of participants who reported difficulties while solving the task and the number of participants who considered an explanation helpful is given in Figure 4.1 at the end of the chapter.

In the follow-up questionnaire participants interpreted the three questions regarding difficulties experienced during task completion differently. Some described their difficulties very matter-of-factly but gave insights into their personal perception of these difficulties when asked why they thought an explanation would be helpful. We will therefore present a combined analysis of replies to these two questions and participants' strategies.

	Subject 1	**Subject 2**	**Subject 3**	**Subject 4**	**Subject 5**	**Subject 6**
Task 1.1	follows tutorial	follows tutorial	follows tutorial	follows tutorial	follows tutorial	follows tutorial
Task 1.2	follows tutorial	follows tutorial	follows tutorial	follows tutorial	follows tutorial	does not follow tutorial [1]
Task 2.1	follows tutorial [2]	partially follows tutorial [3]	partially follows tutorial 2,4	partially follows tutorial 2,4	partially follows tutorial [4]	does not follow tutorial [3]
Task 2.2	follows tutorial*	does not follow tutorial 2,3	*not worked on*	partially follows tutorial 2,4	partially follows tutorial 2,4	*not worked on*

*Subject made an error in goal formation and did not solve the task but employed the tutorial's strategy.

[1] Unnecessary workaround
[2] Evaluation error
[3] Go-to Search instead of Quick Search
[4] Syntactically or semantically incorrect syntax

Table 4.1.: Strategies employed by each subject during task-solving

4.3.1. Literature import tasks

Task 1.1

Conflict task 1.1 was solved correctly by participants 1, 4 and 6 via a workaround. All other participants failed to complete the task and reported it as such. All participants initially followed the strategy shown in the tutorial exactly (dragging and dropping the Word file into the title list).

Subject 1 reported mild difficulties, all other subjects reported severe difficulties. Some participants even verbally expressed their confusion and frustration during the experiment. Notably, subjects 5 and 6 reported difficulties regarding their attempts at workarounds, not the initial failed strategy.

Subjects attributed their difficulties differently. Subject 1 gave a matter-of-fact description ("Word document could not be imported via Drag & Drop"). Subjects 2 and 3 additionally expressed unmet expectations ("Literature was not taken in as expected", "I thought importing the file was possible via Drag & Drop but that did not work"). Subject 3 further described difficulties experienced while trying to find a workaround. Subject 5 also gave a matter-of-fact description ("Whenever choosing the file an error message appeared"), referring to attempts at a workaround. [1] Subject 2 attributed the difficulties to their own technical inability and not remembering how Citavi recognizes and interprets identifiers. Subject 6 also attributed their difficulties to not remembering "what was done in the video" but then reports slowly remembering "how that went, approximately". The subjective need for explanation for Task 1.1 is given in Table 4.2.

Task 1.2

All but participant 6 successfully solved the conflict-free task 1.2 (importing multiple PDF files via Drag & Drop) and followed the strategy shown in the video tutorial very closely. Participant 6, after having experienced difficulty with task 1.1, which they had previously worked on, diverged from the strategy shown in the video. They employed the same workaround for the PDF files as they had used for the Word file in Task 1.1, resulting in them importing more titles than intended. They made several attempts to import either individual or several PDF files via the dialogue "ISBN, DOI, other ID", leading Citavi to read all (not just the first) identifiers from the file(s). This lead to warnings about double entries and files not containing identifiers. Eventually subject 6 abandonded the task, having imported 36 instead of 26 titles. Subjects 2 and 6 reported mild difficulties during this task. Subject 2 attributed this to their own forgetfulness, having forgotten Citavi could import several PDF files at once. Subject 6 explained difficulties importing three of the 26 documents and reported their failure to complete the task. Subject 2 considered an explanation helpful because of their experienced confusion, whereas subject 6 did not, due to considering the process rather intuitive. Subject 2 described a

[1] Subject 5 found the same functional workaround as subjects 1, 4 and 6. Due to the Word document being opened when choosing the file in the Citavi dialogue, Citavi displayed an error message reading "No valid identifiers were found." This does not occur if the file is closed when chosen in the dialogue.

helpful explanation as a small note or reminder that one can drag and drop several files at once.

Subject 1	yes	It would be good to have another import option pointed out to prevent any problems.	Short description that the option exists and where to find it.
Subject 2	yes	I would have maybe wanted a little shortcut where the topic is explained in bullet points since I personally had no idea where to insert the files	A small window with bullet points
Subject 3	yes	Because I would have liked to know why Citavi imports the meta data from the literature list the way it does (I believe the title was interpreted as a note)	Direct feedback from the program which part of the input (the Word file) was interpreted by Citavi would have very much helped me find the error. A deeper explanation why the respective things were interpreted/read the way they were would have helped as well.
Subject 4	yes	I believe it was even explained in the video and I memorized it incorrectly. So it's my fault.	Explanation from the video
Subject 5	yes	I didn't know what the problem was.	A prompt with an explanation.
Subject 6	no	It is not hard, I think you just need to get acquainted with the program.	

Table 4.2.: Participants' subjective need for explanation (yes/no) as well as their reasoning and desired content of the explanation for Task 1.1

4.3.2. Search tasks

Task 2.1

Solving this task (Full-text search including a wildcard) is not possible in Citavi 6. The strategy shown in the video was to open the Quick Search dialogue and enter the search term “ft: un*ing”. Three participants misinterpreted the result and thought they had solved the task correctly, despite using an incorrect search term (“un*ing”) with regard to the tutorial video. This constitutes an unintended evaluation error. Evaluation strategies for the search tasks were not explicitly included in the tutorial video.

Two participants spent considerable time trying to solve the task via the Go-to search function but eventually switched to the Quick Search function. The Go-to search function was shown and explained in the video but only as a distractor. No search term containing shorthands were shown in the Go-to search bar.

Another participant tried a number of semantically and syntactically incorrect search terms before, apparently by chance, using the search term as would have been correct with regard to the video tutorial, getting an empty result set most of the time.

All but participant 4 reported mild or severe difficulties for this task. Participants who did not experience evaluation errors all described the search syntax causing them problems. All but participant 3 considered an explanation helpful. Participant 1 again did not give an answer. The difficulties participants experienced and described were not exactly the ones we expected from the conflict construction. The evaluation error hugely influenced participants’ perception of the task. The subjective need for explanation for Task 2.1 is given in Table 4.3.

Task 2.2

Of the four participants who worked on the the conflict-free task 2.2, none solved it correctly. However, all four thought they had solved it correctly, indicated by them entering the term “aufg2.2” into the titles in their result sets. Their errors were recognizable, i.e., the result sets contained titles published between 2000 and 2015. This is clearly visible in the Citavi interface. Furthermore, Citavi highlights where it has found the search term. This indicated that it had searched for expressions such as “2000” not only in the field for the publication date but other fields as well, e.g., the page numbers. As with Task 2.1, this constitutes an evaluation error.

Strategies and examples from the video tutorial were partially followed. All but subject 2, who used the Go-to Search, used the Quick Search function, but none entered a semantically correct search term.

Only subjects 1 and 2 reported mild difficulties. Subject 1 realized they had not read the task instructions completely (and had subsequently searched for all titles containing the term “nuclear” in the title with no publication date restrictions using the – for this purpose correct – search term *t: nuclear*). Subject 2 reported trouble confusing the relational operators “<” and “>”, as well as being unable to “undo” the search and subsequently having to re-enter everything every time. Data on whether subject 1 considered an

explanation helpful is missing due to them not filling out the entire questionnaire. Careful assumption suggests that since they attributed the difficulties to not having read the instructions completely, subject 1 experienced no difficulties operating Citavi correctly and would therefore not have considered an explanation helpful. Subject 2 considered an explanation helpful. They experienced frustration not being able to return to a previously entered search term and the corresponding results, and would have liked to be "shown, where you can go back without going back and forth within the titles in the list".

Subject 1 *			
Subject 2	yes	I was very confused, among other things because of the placeholder, and understood nothing in the end.	Maybe a small step-by-step explanation where one needs to click (like tutorials in games)
Subject 3	no	I was looking for a tool to add a note to several documents, of which I am not sure it exists. So I did not need an explanation for that.	
Subject 4	yes	It would save immense amounts of time if it was explained how to add a note to several titles at once.	A cue when editing the notes of a title that it's possible for several titles at once.
Subject 5	yes	I did not get the right combination. (Not sure, what kind of information was given, because I didn't have the time to read it all.)	some exemplary search terms
Subject 6	yes	Not many people are experienced with these kinds of search (using commands). This is why I would find a small tutorial for this function useful.	How to enter the commands correctly. It was shown in the video, but a few explicit examples wouldn't be bad.

* Subject 1 did not completely fill out the questionnaire.

Table 4.3.: Participants' subjective need for explanation (yes/no) as well as their reasoning and desired content of the explanation for Task 2.1

4.3.3. Further observations

All participants gave corrections for the video, but only participant 3 expressed suspicions that it might contain errors, but also gave their faulty memory as an alternate explanation.

Several participants experienced difficulties entering the terms “aufg2.1” and “aufg2.2” into the notes field of each title during the search tasks and spent a lot of time trying to find a way to automatically add the term to all titles at once. Apart from answers in the follow-up questionnaire, this was inferred from the screen recordings.

In the screen recordings it became evident that several participants were confused by the “Selection” view in Citavi, which, for imports and search results, automatically shows only a selection of all titles in the project. Not all participants realized and undid the selection when necessary.

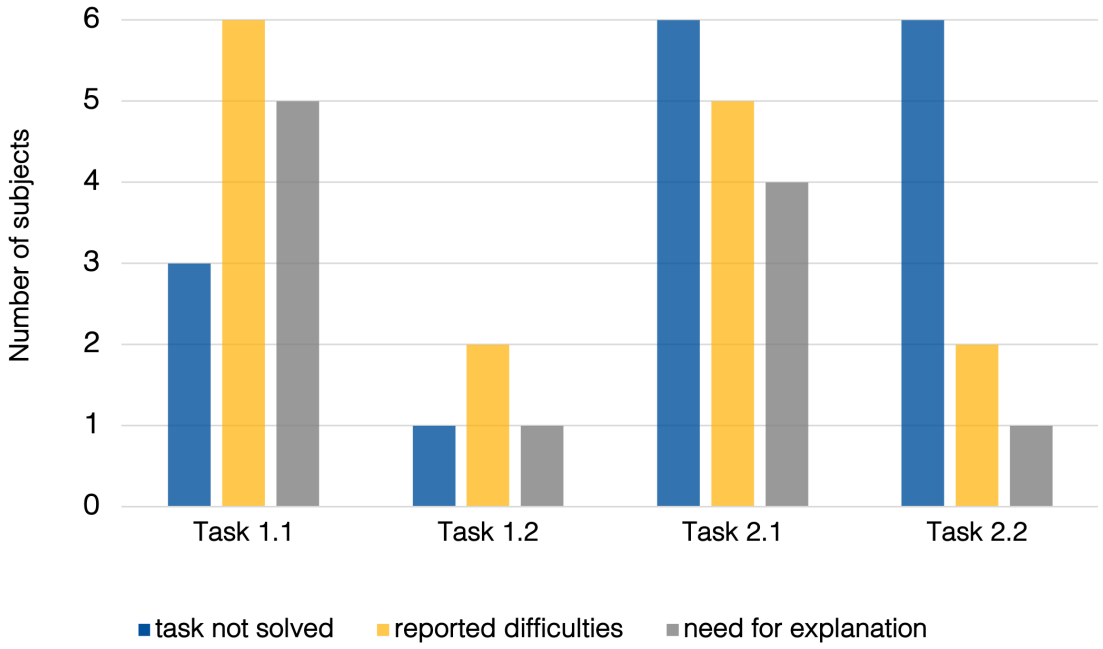

Figure 4.1.: Overview over participants’ behavior and reports during all tasks

5. Discussion

5.1. Research questions

In the following chapter, we will interpret the results with respect to our research questions. Additionally, since this is a pilot study, we will give a short methodological evaluation of our study design.

5.1.1. RQ 1

> **RQ1**: Can we construct a mental model in novice users by presenting a video tutorial of the software?

Behavioral data, i.e., strategies employed while working on the tasks, suggests that for all four tasks participants had internalized the larger part of the information given to them in the tutorial video and had chosen the necessary actions to reach the goal of the task. At no point does their behavior suggest a rejection of the information due to a lack of credibility or preference of a different strategy. We believe that participants following the video's instructions is a strong indicator of successful mental model construction in the sense of our definition in Chapter 2 via the video tutorial. However, not all participants followed the strategies from the video exactly. We discuss the obtained data within the context of possible alternate explanations.

Import tasks

In both import tasks, participants strategies were in line with the tutorial, indicating that they had readily taken up the model. The divergence from the tutorial observed with participant 6 is actually in line with our expectations, even if more extreme, and will be discussed as a reaction to a mental model conflict.

Search tasks

Subjects had difficulties taking up the model as communicated in the tutorial's section on searching literature. In some cases this affected the strength of their predictions of Citavi's behavior, in other cases it resulted in errors evaluating the observed behavior.

In the conflict task, two participants showed an evaluation error we did not expect to occur. Evaluation strategies were not explicitly included in the tutorial video. We assumed that for all tasks judging whether the goal was reached was intuitive. This constitutes a gap in the system model as we communicated it to the users which reflects in their mental model.

Probable reasons for participants using the Go-to Search instead of the Quick Search are memory difficulties combined with general confusion, as supported by their reports of difficulties. Subjects showed no preferred alternative strategy but trouble following the video's strategy, as well as little conviction in their actions. They presumably did not form a strong expectation of how Citavi would behave as a response.

The use of semantically incorrect search terms across participants, despite taking notes on the shorthands, indicates that the search syntax was considered relevant but intuitive. Knowing the shorthands but not the syntax makes no sense, so subjects must have thought they would be able to remember the syntax or that cues from task or the interface would be sufficient. The lack of conviction in their strategies is evident from the self-reported difficulties remembering or choosing the correct syntax. Subject 4, who was the only one not reporting difficulties in the search tasks at all, suffered from evaluation errors, believing they had successfully solved the task.

Intermediate questionnaire

With regard to our research question, data from the intermediate questionnaire asking participants about the functionality of Citavi yielded little additional insight.

Regarding Citavi's general functionality, answers related less to specific strategies than to what users believe they should know or remember about a program. In this way, answers may be more reflective of individual users' general approaches to unknown software than to their strategizing and thus could give an overall "theme" to their task approaches. For exmaple, a user giving a use-case level answer might have more problems solving a problem with the program than a user giving a technical-level answer. However, the demand characteristics of open questions seem to have a great influence on answers. Users often attempt to figure out more clearly what kind of answer is expected of them instead of choosing the appropriate answer themselves. Using answers to open questions to argue for the lack of specific knowledge is one of the pitfalls described in Section 2.3.4. Users may have a multi-layered understanding of a program but if not asked directly they seem to decide to communicate one perspective but not the other(s).

Regarding the import functionality, users did seem to understand that Citavi had to somehow extract the identifier from a file to fill out meta data. Their descriptions of the inner workings were not perfect but indicative of a sufficient level of understanding to use the function in Citavi and form the expectation that an identifier was the key to having the meta data filled in automatically. Regarding the search functionality, answers are best interpreted in combination with notes taken during the video tutorial. Participants were not allowed to use their notes when answering the intermediate questionnaire but their answers seemed to be an attempt to recall what they had written down since all notes included shorthands for searchable attributes. Moreover, notes included wildcards and operators which were never included in the answers to question 3. This shows that most subjects did consider this information important and expected not to be able to remember it - as appears to have proven true - and thus noted it down.

Overall, answers are indicative of the users' knowledge about Citavi and include its inner workings but only on a relatively superficial level. We were unable to see a

direct relationship between replies in the intermediate questionnaire and users' actions, problem-solving strategies and reported difficulties.

In conclusion, subjects appear to have internalized the information given to them, with the exceptions discussed above, and formed a mental model, on the basis of which they chose their actions and predicted the software's reactions. The structure and durability of their mental model cannot conclusively be determined on the basis of the obtained data.

5.1.2. RQ 2

> **RQ2**: Does a mismatch between the user's mental model and the system model cause a conflict in the user?

With regard to users' reactions to difficulties, we will exclude instances in which evaluation errors caused no reports of difficulties and instances in which no concrete strategy and therefore no solid expectations of Citavi's behavior were formed. Difficulties experienced due to a lack of a coherent strategy differ in nature from the mental model conflicts we intended to research.

Participants' reports in the import tasks reflected that they did experience conflict and reacted to it behaviorally, cognitively and emotionally. Retries of the same behavior, that had just proved not to yield the desired result, as seen in the conflict task, is illogical behavior, indicative of confusion and conflict. All participants reported difficulties with at least the literature import conflict task. Some replies allowed insight into their attribution. Attribution targets included Citavi not working, participants' own technical inability as well as them having bad memory. Importantly, despite difficulties clearly being caused by the constructed mental model conflict, some participants reported difficulties and a need for explanation regarding their attempts at workarounds instead of their initial failed strategy. Which part of a conflict a user's perception latches onto is therefore variable.

We want to stress that none of the participants came to the certain conclusion that the tutorial video contained any errors, despite us giving the impression that the video was the entity being tested in the experiment. Suspicions were expressed only once. This indicates that subjects experiencing conflict show only minimal tendency to disregard external sources of information but a tendency to doubt themselves.

In the case of subject 6, the severe conflict experienced in Task 1.1 had a spill-over effect. The participant must have come to the conclusion that the Drag & Drop strategy did not work in Citavi at all and therefore abandoned it, causing them to opt for a much more labour-intensive workaround which additionally did not yield the correct result. Them disregarding the tutorial's strategy even in the no-conflict task initially seems indicative of damaged trust in the tutorial. However, in the end, they believed they had remembered and followed the tutorial's strategy in both import tasks. All in all, their behavior and attribution is suggestive of a severe lack of trust in their own memory. The other participants' behavior and reports, in contrast, showed next to no signs of conflict

for Task 1.2: Their strategy was straight-forward, in line with the tutorial, and almost no difficulties were reported.

Participants occasionally showed signs of frustration and impatience and sometimes asked if it was possible that some setting in Citavi was wrong. This occured especially often during the first half of the experiment, which the majority of subjects spent trying to solve Task 1.1 correctly.

In summary, all participants showed signs of conflict for the conflict task, albeit differing in nature and intensity. After having watched the tutorial until feeling confident to be able to solve tasks in Citavi, they later expressed doubts in their memory, their technical abilities and sometimes in Citavi. In contrast, almost no conflicts seem to have been experienced in the no-conflict task, with the notable exception of one spill-over effect.

5.1.3. RQ 3

> **RQ3**: Does a conflict between a user's mental model and the perceived system behavior result in a subjective need for explanation?

Combining experienced conflict, strategies and attributions shows that experienced conflict is not the only or most important factor causing a subjective need for explanation.

A need for explanation does not always arise when users experience difficulty, as seen most clearly with subject 6. Instead, users may take their own level of expertise into account, considering more practice and experience with the system a more appropriate way to avoid difficulties. In other cases, users do not even realize their errors or any conflict. They might never feel a need for explanation, or only in later steps, like when attempts at a workaround fail.

In contrast, some users will, when asked, feel a need for explanation even after mild, temporary confusion or insecurity, despite successful goal completion, as seen with subject 2. This may be a demand characteristic of the follow-up questionnaire. Alternatively, it could be a disposition of users with low confidence in their abilities.

Users who do experience a subjective need for explanation related to a specific conflict consider different contents helpful. Subjective memory difficulties lead to users wanting to be given the same explanation again that they believed they had misremembered.

The distance between expected behavior and actual behavior may also affect the subjective need for explanation regarding the explanation content. This may depend on the user's attention to detail as well as the specific task at hand. Citavi importing the Word file in Task 1.1 as an empty title can be interpreted as the software being able to import the file type in general. This might have caused a desire to have Citavi's inner processes – which part of the file contents had been "read" – explained to them in one user. However, this reason is an interpretation on our behalf, for which our results alone do not contain sufficient evidence.

It should be noted that users are not always able to consider different types of explanations or what kind of information explanations could contain. Instead, they may only

have their goal of understanding in the most general sense in mind, unable to specify what kind of explanation might help them achieve this goal. Since users do not always know where the actual problem they are experiencing stems from, their suggestions for explanation contents should be followed only with knowledge of the context in which the problem occured.

In summary, our results show that mental model conflicts as we constructed them do play a role in the subjective need for explanation. However, many other factors need to be considered as well, such as the specific task characteristics, the user's individual preferences and their perception of their own abilities. The experience of a mental model conflict and the need for explanation sometimes lie apart, when users first attempt workarounds but then still cannot reach their goal.

5.2. Study design

The structure of the experiment conducted for this work had not been tested before. From a methodological standpoint it was largely successful. However, as suspected in the study design, the detailed and comprehensive elicitation of the mental model proved difficult and the intermediate questionnaire turned out to yield smaller insight into participants cognitions than expected. The behavioral data was very informative. The users' own descriptions of the difficulties they experienced supplemented behavioral data unexpectedly well.

The merit of the mental model concept per se is questionable to us. Behavioral data was more informative than self-reports. This makes the intermediate step of a mental construct guiding this behavior practically redundant. Trying to fit behavioral data and self-reports into the shape of a model of the system can be frustrating, as this structure seems not to be reflected in users' cognitive processes. The mental model concept appears to be more helpful to communicate the fact that users sometimes follow beliefs that are deeply connected to one another as if they were parts of a system. As a self-standing construct to be elicited and evaluated it seems too far from reality. It does however help to characterize certain types of conflicts, as in our case a violation of users' predictions based on their beliefs. We recommend using the mental model for this purpose but to avoid the expectation that (novice) users' cognitions are a tightly and logically interconnected set of beliefs. For a study like ours, identifying certain essential action-prediction pairs is sufficient.

With hindsight we recommend to clearly communicate goal evaluation techniques when inducing a mental model via tutoring materials. We were interested in software behavior that comes unexpected for the user. Since users had problems evaluating the search results shown to them they could not compare them to their expectation. This should be corrected in future research.

Furthermore, many participants had trouble with the search term syntax despite taking extensive notes during the video tutorial. Explaining the syntax and the hierarchy of shorthands and operators beyond a handful of examples could help users to form a

more solid understanding and reduce trial-and-error approaches, making the search functionality in Citavi a valid task set to investigate mental model conflicts. Additionally, some subjects did not intuitively understand that Citavi has a “selection” view with which it replaces the title view when a search is executed. This selection view shows only the result set, which is not always easily recognizable. Some participants seemed to have trouble telling these views apart. How to do so should have been communicated in the tutorial to separate these difficulties from difficulties caused by constructed mental model conflicts.

Regarding the search tasks, the intermediate questionnaire was not very task-related, as indicated by the discrepancy between correct answers given in the questionnaire (lists of shorthands) and the lack of success in the tasks. A question about the precision with which titles can be searched in Citavi might have better identified the users’ lack of understanding of the search term syntax regarding the combination of shorthands, operators and wildcards, and thus given a more complete view of task-relevant knowledge.

In the search tasks, we asked participants to edit the notes of the titles they had found to mark their solution. This was very time-consuming, as users were relatively slow in operating Citavi’s interface. Moreover, it was mentioned in the follow-up questionnaire and some participants considered an explanation of a less labour-intensive strategy helpful. Choosing a different, simple and quick method for participants to indicate their success in the search tasks is an advisable solution to this problem.

6. Limitations and threats to validity

Due to the partly methodological nature of this study, it is important to us to reflect on limitations of the design and the resulting threats to validity.

We only tested a very small number of novice users, which allows no conclusive answers to our research questions. Small effects of of interindividual differences between users might only come to light with a larger sample. It is therefore important to view the results presented in this work as an idea of the results that might be gained with this type of experiment.

Participants not being allowed to use the internet to solve their problems was a necessary constraint in order to hide errors in the tutorial. This might have affected the subjective need for explanation, since we withdrew a commonly used source of information. We cannot determine the extent of this effect. Notably, none of the participants reported that they would have rather used the internet instead of being given an explanation, but they may not have considered this option because they were thinking within the constraints of the experiment. Despite the option being given in our experiment, we recommend a reminder of alternative sources of information for participants to consider when reporting their need for explanation.

Despite our own criticism of mental model elicitation techniques using the success rates in tasks as an indicitator of mental model quality we use a similar measurement. We viewed this from the perspective of the degree to which users follow the tutorial given to them, but Citavi offers almost no alternative strategies allowing success within the tasks set. In short, it is not possible to fail when following the tutorial and it is rather hard or impossible to succeed with a different strategy. It would have been desirable to construct tasks that allow more workarounds to allow users to choose between more strategies, but we prioritized setting meaningful tasks within Citavi to the detriment of alternative strategies.

Despite this shortcoming, Citavi also allowed the circumvention of an artifical setup. It is not an obscure or overly complicated piece of technology unlikely to have a place in users everyday lives [14]. Instead, the software is widely known in academic circles and likely to be useful to our participants in the future.

Our experimental setup still begs the question as to why an otherwise novice user would come to hold a wrong mental model. We argue that external sources of information like an unofficial video tutorial can easily contain errors or imprecisions. Moreover, users may have watched a tutorial for or gained experience with a previous version of the software at hand. A version switch could entail a system model alteration, rendering the mental model faulty, despite it previously being functional.

Studies using instruction materials based on a system model are sometimes viewed critically, since they might suppress or overwrite the individuality of mental models as they are created during self-directed learning [7]. As a response to this, we want to stress that while we did not study self-directed learning, we opted to script the video tutorial in casual language, using no technical terms without explanation. It was based on the Citavi online manual only content-wise.

7. Conclusion

We will give a conclusive summary of this work and end with implications for future research building on our insights.

In this paper, we have argued that explainability is a non-functional software requirement of growing importance as software complexity increases. Complex software is embedded in more and more parts of our lives, sometimes, in the case of artificial intelligence, supporting decision-making in high-stakes environments. While such functionalities are desirable, they are hard to implement and harder to understand. This is true for experts with a background in computer science and even more so for laypeople. Since areas of application are not limited to specialist niche cases, we consider the perspective of end users without a deep understanding of the algorithms they are interacting with. In order to make use of complex software, it should be able explain itself and its behavior. The design of self-explaining systems aims for maintaining the user's trust, or even just their satisfaction when interacting with the software. We approached the specific question of the user's subjective need for explanation in response to difficulties during goal attainment.

Users normally approach a software system with a goal in mind. They choose appropriate actions to reach their goal, and in turn form expectations of the software's behavior. The cognitive structures underlying this planning behavior – the user's understanding of the system – are referred to as their mental model. The theoretical concept states that if the mental model is flawed in a way that impedes goal attainment, e.g., by leading the user to form false expectations of the software behavior, users are likely to experience conflict. Software-sided explanations might resolve such conflicts. We designed an experiment to answer the question of whether and how users perceive mental model conflicts and the resulting difficulties, and whether they experience a need for explanation.

For this pilot study, we successfully induced a flawed mental model of an existing software system in users via a video tutorial. The success of mental model construction via an external source of information was validated via a questionnaire eliciting the users' understanding of the software. Users then interacted for the first time with that software and solved two types of tasks, either designed to elicit a conflict or be solvable without conflict. We captured their task-solving behavior and success and afterwards had them report any difficulties they had and their subjective need for explanation for each task. Through this method we gained insight into users' subjective need for explanation following difficulties, as well as the type and content of explanation they desired.

Methodologically, our results partly confirm the difficulty of eliciting a mental model. In response to open-ended questions users gave short answers which did not relate to

their approaches to the tasks, but tather their general approaches to unknown software. Such knowledge largely lies outside the mental model as defined for this study, as it is task-irrelevant. Additionally, participants' answers appeared to paraphrase the video tutorial script.

Task-solving strategies in combination with self-reports of difficulties were more insightful. Participants followed the strategies as they were shown in the video tutorial, and did not diverge even upon unexpected software behavior. None of the participants expressed fundamental doubt in the tutorial. Self-reported difficulties and subjective need for explanation never included the tutorial as a possible reason for the difficulties. Instead, users occasionally suggested the tutorial's contents as a desirable software-sided explanation. Credible tutoring materials seem to be a possible means to induce a mental model in novice users. Elicitation methods which combine the analysis of behavioral data as well as some form of task- or strategy-related self-report, such as the user's interpretation of errors, appear to be insightful for research questions such as ours, for which no explicit, full mental models need to be externalized.

With regard to subjective need for explanation, we found that the level of conflict users experience beforehand is not the only decisive factor. In some cases, users wrongly interpreted the software's behavior, leading them to believe they had solved the task correctly. In these instances, users did not report difficulties. In the remaining cases, behavioral data indicated conflicts, which were experienced cognitively and emotionally. Users showed signs of frustration and confusion. Illogical behavior, such as undoing and redoing an action sequence that had just yielded an unexpected result, was observed in several cases. Self-reports confirmed these difficulties.

Whether users considered a potential explanation helpful depended on how they attributed their difficulties. We observed users with low self-assessed technical abilites to attribute difficulties to their own inability, such as a lack of understanding. Those users seem to experience a need for explanation even in response to mild insecurities, when no actual conflict is apparent. Another attribution target was users' own faulty memory. Users with self-assessed high technical abilities considered the possiblity that the software was not working, suspecting a wrong setting. In one case, severe difficulties caused the user to abandon a strategy entirely, not even attempting it in the conflict-free task. In self-reports, they blamed their bad memory for the difficulties. Due to the small sample size of this pilot study, the relationship between self-assessed technical ability and attribution of difficulties is only anecdotal. Doubts in the video tutorial were expressed only once, despite all participants being told that the video was the entity being tested in the experiment.

Our results show that a subjective need for explanation depends on several factors. Unexpected software behavior, conflict and resulting difficulties during operation of a system – constituting a mental model conflict – is one of those factors. Notably, users need for explanation may be experienced with a delay, after attempted workarounds fail. Importantly, users' own attribution of their difficulties can completely change their experience. Users with little trust in their may not experience a need for an explanation, preferring to get acquainted with the software first and to learn from their errors. Users

wtih little trust in their general technical ability may, in contrast, experience a need for explanation even in response to mild confusion or insecurity.

All in all, we consider the experiment successful and appropriate to answer our research questions, if conducted with a larger and possibly more diverse sample size. As mentioned in Section 5.2, our materials were imperfect but can be improved with little effort.

7.1. Implications for future research

For this work we conducted a pilot study with a small sample size of novice users to test our methodological approach. Conducting our experiment with a larger sample size would allow statistical analysis and therefore stronger evidence for the conclusions and assumptions from these anecdotal insights. A larger sample size could also extend our findings to allow for the identification of user behavior that is characteristic of certain types of conflicts. Additionally, non-behavioral user characteristics that influence the need for explanation, such as the attribution style or memory performance could serve as basis for the creation of personas. Using personas in the requirements engineering process has been suggested before [9], to allow tailoring to inter-individually different needs for explanation.

Moreover, it would be insightful to actually present users, even those who do not report a subjective need for explanation, with an explanation anyway. We observed that users – in the case of our experiment – wrongly attributed difficulties to their own lack of expertise or ability. Since this was not actually the cause of their difficulties, they might still react positively to an explanation. In contrast, other users expressed a need for explanation at the first sign of difficulty. Providing an explanation might show that those users react more negatively to its disruptive character than they themselves expected. Further research therefore might help to decide when it is more sensible to follow the user's subjective opinion on the helpfulness of an explanation and when to disregard it. Evaluation erros in users are eligible for a similar research question. They likely strengthended users' conviction of a dysfunctional strategy. Presenting an explanation despite the user not experiencing difficulties provides a different angle to the question of when an explanation should be given and is regarded as helpful. Whether such an explanation would be accepted when the user is convinced of their strategy is another open question.

We tested only novice users and induced a mental model that is likely not very stable over time. Expert users have a more stable, and likely more extensive mental model. They could therefore react very differently to mental model conflicts as we constructed them for this study and have a different perspective both on the proposition of a software-sided explanation as well as actual explanations. They might devalue or ignore explanations that do not agree with their beliefs [16].

Two subjects employed a "stab in the dark" approach to the search terms syntax, both expressed a need for explanation. Even though this is not a mental model conflict as

we defined it, and rather a mental model gap conflict, it results in very characteristic behavior that could be easily identified by a non-human observer, i.e., a self-explaining system. This is however task-specific and whether mental model gaps are always expressed so characteristically requires more research into different task domains and input modalities.

Evaluation errors are the hardest type of conflict to detect through the software as they are not even known to the user themselves. To identify such a conflict the system would need to identify the user's overarching goal which may only be evident after observing long action sequences, if at all. However, depending on the type of system and complexity of a typical task, it might be possible. Our study cannot provide insight into the user's perception of explanations following evaluation errors. For specialist software with few use cases and novice users who make evaluation errors investigating this specific scenario could be beneficial.

A. Materials

A.1. Video script

A.1.1. German script

Hallo zusammen, ich möchte euch heute in einem kurzen Tutorial erklären, wie Citavi 6 funktioniert.

Ein kurzer Überblick zu Beginn: Citavi ist ein Literaturverwaltungsprogramm, das ihr verwenden könnt, wenn ihr eine Hausarbeit oder eine Abschlussarbeit schreibt.
Kurzum: Immer dann, wenn ihr in einer Arbeit Literatur für die Recherche verwendet und zitieren müsst, bietet euch ein Literaturverwaltungsprogramm wie Citavi vielfältige Funktionen, um den Überblick zu behalten – auch wenn ihr mehrere Hundert Quellen verwendet. Zu den Funktionen gehört unter anderem, dass ihr Citavi benutzen könnt, um alle eure Titel, die zu einem Projekt gehören, auf einen Blick zu sehen, euch eine Vorschau des Dateitexts anzeigen zu lassen, eure Titel zu markieren und kategorisieren und sie zu durchsuchen. Ihr könnt in den Titeln auch direkt Zitate anlegen, die ihr dann in Word oder in LaTex in eure Arbeit einbindet. Citavi erstellt dann automatisch ein Literaturverzeichnis im von euch festgelegten Zitationsstil. Gerade beim Einpflegen von Literatur nimmt euch Citavi viele repetitive Aufgaben ab, die ihr sonst mühsam von Hand erledigen müsstet, wobei schnell Fehler passieren, zum Beispiel beim Eintragen von Metadaten für einen Titel. Bei einem wissenschaftlichen Artikel sind das die Autoren (das können auch schonmal 5 oder 6 Autoren sein), der Titel, die Zeitschrift, in der der Artikel erschienen ist, das Volume, das Issue, die Seitenzahlen, sowie das Veröffentlichungsdatum. Je nach Zitationsstil müssen alle diese Felder ausgefüllt sein. Im Folgenden erkläre ich euch die wichtigsten Features von Citavi, und wie ihr sie verwendet, um euch möglichst viel Arbeit zu sparen.

Eines der praktischsten Features von Citavi ist seine OCR-Funktionalität (also Optical Character Recognition), d.h. wenn ihr ein Buch oder einen wissenschaftlichen Artikel zu euren Titeln hinzufügen wollt, kann Citavi den Inhalt der jeweiligen Datei einlesen, um Informationen über den Titel zu erlangen. Ihr könnt Citavi dafür einzelne Dateien oder gleich mehrere auf einmal übergeben. Bei der Verwendung dieser OCR-Funktionalität gibt es zwei Outcomes. Im Optimalfall enthält eure Datei einen Identifier. Bei Büchern ist das die ISBN, im 10- oder 13-stelligen Format. Citavi erkennt beides und kann sie auch konvertieren. Bei wissenschaftlichen Artikeln ist das

- die DOI
- die PubMed-ID
- die PubMed Central ID oder

- die arXiv ID

Wenn ihr Citavi eine PDF-Datei übergebt, wird mittels OCR geprüft, ob sich auf den ersten 5 Seiten ein solcher Identifier befindet. Wenn Citavi einen Identifier findet, dann greift es auf zentral verwaltete Online-Kataloge bzw. -Datenbanken zu, in denen die Metadaten eingetragen sind. Die so ausgefüllten Felder sind zu 99,9% korrekt. Wenn Citavi keinen Identifier findet, nutzt es die Dateieigenschaften, um grundlegende Felder auszufüllen. Das sind oft nur der Titel entsprechend des Dateinamens und die Seitenanzahl, entsprechend der Seitenanzahl der Datei.

Der OCR-Scan ist natürlich nicht notwendig, wenn ihr Citavi direkt eine Text-Datei übergebt, in der einer oder mehrere Identifier stehen. Zum Beispiel eine Word-Datei mit Literaturempfehlungen von eurem Prof, wo vielleicht ein paar Bücher mit ISBNs drin stehen, aber auch Zeitschriftenartikel mit DOIs. Das zeige ich euch jetzt auch nochmal in der Citavi-Oberfläche.

Das einfachste ist es, einen Titel, der euch als PDF vorliegt, per Drag & Drop in eure Titelliste zu ziehen. Das funktioniert wie gesagt auch für mehrere Dateien auf einmal! Dieser hier hat eine DOI auf der ersten Seite. Dann passiert genau das, was ich eben beschrieben habe – Citavi erkennt die DOI, nutzt Online-Datenbanken für einen Abgleich und die Felder werden automatisch ausgefüllt. Und jetzt nochmal zum Vergleich mit einer PDF, in der kein Identifier zu finden ist. Wenn ich diesen in meine Titelliste droppe, dann werden nur die Felder ausgefüllt, deren Werte Citavi anhand der Dateieigenschaften bestimmen kann, also die Seitenzahlen und den Titel, der erstmal dem Dateinamen entspricht. Hier könnt ihr aber leicht nachbessern, indem ihr die korrekten Werte aus der Titelvorschau herauskopiert. Genauso könnt ihr auch mit einer Literaturliste in Word verfahren (wie dieser hier, die ich mit Absicht ganz schrecklich formatiert habe). Einfach per Drag & Drop in die Titelliste ziehen und Citavi liest die Identifier aus, holt sich die Datenbankeinträge und fügt die Titel hinzu. Jetzt habt ihr also eure Titel ohne viel Handarbeit in euer Projekt importiert. Als Nächstes möchte ich euch die Such- und Navigationsfunktionen erklären.

Wenn euer Projekt irgendwann so groß ist, dass die Titelübersicht auf der linken Seite unübersichtlich wird, habt ihr in Citavi mehrere Möglichkeiten, durch eure Titel zu navigieren und bestimmte Titel zu suchen. Citavi hat natürlich, wie ihr eben gesehen habt, zu jedem Titel Metadaten, also Titel, Autor, Erscheinungsjahr usw. Diese Felder könnt ihr durchsuchen. Dafür müsst ihr eurer Suchanfrage bestimmte Kürzel voranstellen, damit nur in einem konkreten Feld nach dem Suchbegriff gesucht wird. Das hier sind die wichtigsten davon.

Mit dem Kürzel “a” könnt ihr nach Namen von Personen, also Autoren, Herausgebern, Mitarbeitern usw. suchen. Das Kürzel “d” durchsucht eure Titel nach Erscheinungsjahr, usw.

Am nützlichsten ist aber meiner Meinung nach die Volltextsuche, für die ihr das Kürzel “ft” für *full text* verwendet. Damit durchsucht Citavi den Dateitext der hinterlegten Datei. Dass Citavi diesen mittels OCR lesen kann, habe ich ja vorhin schon erwähnt. Das funktioniert natürlich nur, wenn ihr für einen Titel auch eine Datei hinterlegt habt – das ist aber automatisch der Fall, wenn ihr den Titel per Drag & Drop hinzugefügt habt.

Alternativ könnt ihr auch nachträglich eine PDF-Datei auf eurem Computer mit dem Titel in Citavi verknüpfen.

Ihr könnt für eure Suchbegriffe auch sogenannte Wildcards benutzen, also Platzhalter. “?” funktioniert dabei als Platzhalter für *einen* Buchstaben, “*” funktioniert als Platzhalter für mehrere oder keinen Buchstaben. Damit könnt ihr also unter anderem Namen unabhängig von der Schreibweise finden. Auch relationale Operatoren wie <, >, <=, >=, = können verwendet werden, da Citavi Text im Format Jahreszahl-Monat-Tag als Datum erkennt. Mit dieser Suchanfrage könnt ihr alle Titel anzeigen lassen, die nach 2014 erschienen sind.

Und sogar logische Operatoren wie AND, OR, NOT sind möglich. Hier müsst ihr allerdings auf korrekte Klammerung achten.

So, und jetzt nochmal live. In der Citavi-Oberfläche habt ihr verschiedene Möglichkeiten eine Suche zu starten. Und zwar ist das als erstes die „Gehe-zu“-Funktion, das ist das Suchfeld hier über der Titelliste. Hier durchsucht Citavi alle Felder und ordnet die Suchergebnisse nach Relevanz (zum Beispiel werden Treffer in der Überschrift gegenüber Treffern in anderen Feldern bevorzugt). Wenn ich hier nach “sample” suche, finde ich zuerst Überschriften die “sample” enthalten und dann den Autor mit diesem Namen. Wenn ihr genauer suchen wollt, findet ihr die Schnellsuche und die Erweiterte Suche hier oben unter dem Lupensymbol oder mit der gewohnten Tastenkombination Strg+F. In der Schnellsuche könnt ihr genau so Suchbegriffe eingeben, wie ich euch das eben gezeigt habe. Ich habe dafür mal eine beispielhafte Anfrage vorbereitet.

„sample“ durchsucht alle Felder, findet also sowohl den Autor, als auch die Überschrift, als auch Vorkommen im Abstract usw.

„t: sample“ findet nur die Überschrift, nicht den Autor oder andere Felder, das sind jetzt nur noch zwei Artikel.

„t: sample AND d: > 1995“ findet nur noch einen der beiden Artikel, weil der andere 1991 veröffentlicht wurde.

„ft: sampl*“ mit Wildcard am Ende des Worts findet alle verknüpften PDF-Dateien, die zum Beispiel „sampling“ oder „samples“ oder „sampler“ enthalten.

Damit habt ihr also viele verschiedene Möglichkeiten, eure gesammelte Literatur zu durchsuchen.

Das war‘s erstmal für den Einstieg in Citavi. Ich hoffe ich konnte euch vermitteln, wieviele Funktionen Citavi mitbringt und wie ihr diese bei der Literaturverwaltung verwenden könnt.

A.1.2. English translation

Hello together,

today I want to give you a short tutorial on how Citavi 6 works.

Let's begin with a short overview: Citavi is a literature management tool, which you can use when writing an essay or a thesis. In short: Whenever you have to use or cite literature you researched, a literature management tool like Citavi offers many functions to keep the overview – even if you're using several hundred sources.

One of the functions is that you can use Citavi to see all of your titles belonging to a project at a glance, see a preview of the file text, tag and categorize titles and search them. You can also directly cite from a title, and embed the citation in your thesis in LaTeX or Word. Citavi automatically creates a reference list in your preferred citation style.

Especially during the import of literature Citavi takes over many repetitive tasks that you would have otherwise have to do by hand, at the risk of making mistakes, for example when entering meta data for a title. For a scientific paper, that is the authors (it can easily be 5 or 6 authors), the title, the journal the article was published in, the volume, the issue, the page numbers as well as the publication date. Depending on your citation style all of these fields have to be filled in.

Next I will explain the most important features in Citavi and how you can use them to save yourself as much work as possible.

One of the most handy features in Citavi is its OCR functionality (Optical Character Recognition), i.e., when adding a book or a scientific article to your titles, Citavi can read the contents of the respective file to elicit information about the title. You can hand Citavi a single file or several at once.

When using the OCR functionality there are two outcomes.

Optimally, your file contains an identifier. For books that is the ISBN with 10 or 13 digits. Citavi recognizes both formats and can convert them. For scientific articles it is

- the DOI
- the PubMed ID
- the PubMed Central ID or
- the arXiv ID

When handing Citavi a PDF file, it checks via OCR whether there is such an identifier on the first 5 pages. If Citavi finds an identifier, it accesses centrally controlled online catalogues or databases containing the meta data. That way, fields are correctly filled in 99.9% of cases.

If Citavi cannot find an identifier, it uses the file's properties to fill out basic fields.

Often that is just the title according to the file name and the page count, according to the file's page count.

Obviously, the OCR scan is not necessary when you're giving Citavi a text file containing one or more identifiers. For example, a Word file with literature recommendations from your professor, that may contain a few books with ISBNs but also journal articles with DOIs.

Now, I'll show this to you in the Citavi interface.

The easiest way is to drag and drop a title for which you have a PDF file into your title list. As I mentioned before, this also works for several files at once!

This title has a DOI on the first page. What happens is what I just described – Citavi recognizes the DOI, uses online databases to find a match and the fields are filled in automatically.

And now for a PDF without an identifier. If I drop this into my title list, then only those values that Citavi can determine according to the file properties are filled in, i.e., the page count and the title, according to the file name. But you can easily improve that by copying the correct values from the preview.

You can proceed in just the same way with a literature list in Word (like this one, which I have deliberately formatted horribly). Just drag and drop it into the title list and Citavi reads the identifiers, gets the database entries and imports the titles.

So now you have imported your titles without much manual work.

Next, I want to explain the search and navigation functions.

When at some point your project is so big that the title list on the left gets confusing, you have several options in Citavi to navigate through your titles and search for certain titles. Of course Citavi has, as you've just seen, meta data for each title, i.e., the title, author, publication date and so on. These fields are searchable.

To do this you have to add certain shorthands to the beginning your search term, so that only certain fields are searched. These are the most important ones.

With the shorthands a: you can search for people's names, i.e., authors, editors, collaborators, and so on. The shorthand d searches your titles for the publication date and so on.

The most handy feature is, in my opinion, the full text search, for which you use the shorthand ft for full text. This has Citavi search the text of the attached file. As I have mentioned before, Citavi can read this via OCR. Of course this only works if you have a file attached to a title. This is automatically the case if your imported the title via drag and drop. Alternatively you can later add a PDF file from your computer to the title in Citavi.

You can also use so-called wildcards for your search terms, i.e., placeholders.

"?" works as a placeholder for ONE letter, "*" works as a placeholder for several or no letters.

Among other things, this allow you to find names regardless of a specific spelling.

Relational operators like <, >, <=, >=, = can also be used since Citavi can regonize text of the format year-month-day as a date.

With this search term you can find all titles published after 2014.

Even logical operators like AND, OR, NOT are possible. With these you must pay attention to use the correct bracket syntax.

Now, the same thing live.

In the Citavi interface you have several options to start a search. The first is the “Go-to” function. That is the search field here above the title list. Here Citavi searches all fields and orders search results by relevance (for example, matches in the title are preferred over matches in other fields).

If I search for “sample” here, I’ll find headings containing “sample” first and then the author with that name. If you want to search more precisely, you’ll find the Quick Search and the Advanced Search here under the magnifying glass or with the usual keyboard shortcut Ctrl+F.

In the Quick Search you can enter search terms in just the way I have just shown you. I have prepared an exemplary query.

„sample“ searches all fields, so it finds both the author as well as the heading, as well as matches in the abstract and so on.

“t: sample” returns only matches in the heading, not the author or other fields, that’s only two articles now.

„t: sample AND d: > 1995“ finds only one of those two articles because the other was published in 1991.

„ft: sampl*“ with a wildcard at the end of the word finds all attached PDF files containing for example “sampling” or “samples” or “sampler”. This gives you lots of ways to search your collected literature.

For now this is it for my intro into Citavi. I hope I was able to show you how many functions Citavi offers and how you can use them for managing your references.

A.2. Preliminary survey

Fragebogen

Im Vergleich mit der Gesamtbevölkerung Deutschlands, wie beurteilen Sie Ihre Fähigkeiten im Umgang mit Computern bzw. Software im Allgemeinen?

weit unterdurchschnittlich		etwa durchschnittlich		weit überdurchschnittlich
o	o	o	o	o

Bitte schätzen Sie Ihre Erfahrungen mit den im Folgenden aufgelisteten Programmen ein.

Citavi:

- o Ich habe noch nie von der Software gehört
- o Ich habe die Software noch nie verwendet
- o Ich habe die Software 1-2 Mal verwendet
- o Ich habe die Software bereits unregelmäßig verwendet
- o Ich habe die Software bereits regelmäßig verwendet

Zotero:

- o Ich habe noch nie von der Software gehört
- o Ich habe die Software noch nie verwendet
- o Ich habe die Software 1-2 Mal verwendet
- o Ich habe die Software bereits unregelmäßig verwendet
- o Ich habe die Software bereits regelmäßig verwendet

Mendeley:

- o Ich habe noch nie von der Software gehört
- o Ich habe die Software noch nie verwendet
- o Ich habe die Software 1-2 Mal verwendet
- o Ich habe die Software bereits unregelmäßig verwendet
- o Ich habe die Software bereits regelmäßig verwendet

A.3. Demographic survey

Demographischer Fragebogen

Geschlechtsidentität:	o männlich o weiblich o divers
Alter:	_____ Jahre
Muttersprache(n):	_____
Höchster erreichter Schulabschluss:	o Kein Abschluss o Volks-/Hauptschulabschluss o Mittlere Reife o Abitur/Fachabitur
Schuljahre:	_____ Jahre
Höchster erreichter Berufsabschluss:	o Kein Abschluss o Lehre/Ausbildung o (Fach-)Hochschule (Bachelor) o (Fach-)Hochschule (Master)
Aktuelle Ausbildung bzw. Beschäftigung:	_____

A.4. Task instructions

Instruktionen

Bitte bearbeiten Sie nun die folgenden Aufgaben in Citavi 6. Falls Sie sich während des Tutorialvideos Notizen gemacht haben, können Sie diese verwenden. In welcher Reihenfolge Sie die Aufgaben bearbeiten, steht Ihnen frei. Die Aufgaben sind nicht voneinander abhängig.

Für die Bearbeitung der Aufgaben wurde ein Projekt mit dem Titel „EXPERIMENT Energy Resources – Current Developments and Risks" eingerichtet und bereits Literatur hinzugefügt. Bearbeiten Sie alle Aufgaben in diesem Projekt. Dafür haben Sie 20 Minuten Zeit.

Wenn Sie Fragen zum **Versuchsablauf** oder zur **Aufgabenstellung** haben, geben Sie ebenfalls der Versuchsleitung Bescheid.
Sollten sie auf anderweitige Schwierigkeiten bei der Aufgabenbearbeitung stoßen, melden Sie sich ebenfalls. Bitte beachten Sie jedoch, dass Ihnen Fragen zur Bedienung von Citavi nicht beantwortet werden können.

Geben Sie der Versuchsleitung Bescheid, sobald Sie mit der Bearbeitung beginnen möchten. Ab diesem Zeitpunkt läuft Ihre Bearbeitungszeit.

Aufgabe 1: Literatur einpflegen

1.1
Auf dem Desktop finden Sie eine Datei mit dem Titel „Literaturliste.docx". In dieser ist Literatur aufgelistet, die Sie in Ihr Projekt einpflegen sollen. Jeder Titel ist mit einem Identifier (ISBN, DOI, PMID, PMC oder arXiv) versehen. Fügen Sie die Titel Ihrem Projekt hinzu. Achten Sie darauf, dass möglichst viele Felder korrekt ausgefüllt sind.

1.2
Des Weiteren finden Sie auf dem Desktop einen Ordner mit dem Titel „pdfs". Darin finden Sie PDF-Dateien von wissenschaftlichen Artikeln und Büchern. Pflegen Sie alle Artikel in Ihr Citavi-Projekt ein. Achten Sie darauf, dass möglichst viele Felder korrekt ausgefüllt sind.

Aufgabe 2: Titel suchen

2.1
Finden Sie alle Titel, in deren Dateitext Worte vorkommen, die mit „un" beginnen und mit „ing" enden, also beispielsweise, aber nicht ausschließlich, „*understanding*", „*undermining*", „*undoing*".
Tragen Sie bei allen Treffern das Stichwort „aufg2.1" in das Feld Notizen ein (Reiter „Übersicht" bei Auswahl des Titels).
(**Hinweis**: Sie sollten mindestens 26 Titel finden.)

2.2
Finden Sie alle Titel mit dem Wort „nuclear" in der Überschrift, die vor 2000 oder nach 2015 erschienen sind.
Tragen Sie bei allen Treffern das Stichwort „aufg2.2" in das Feld Notizen ein (Reiter „Übersicht" bei Auswahl des Titels).
(**Hinweis**: Sie sollten mindestens 6 Titel finden.)

B. Results

B.1. Handwritten notes

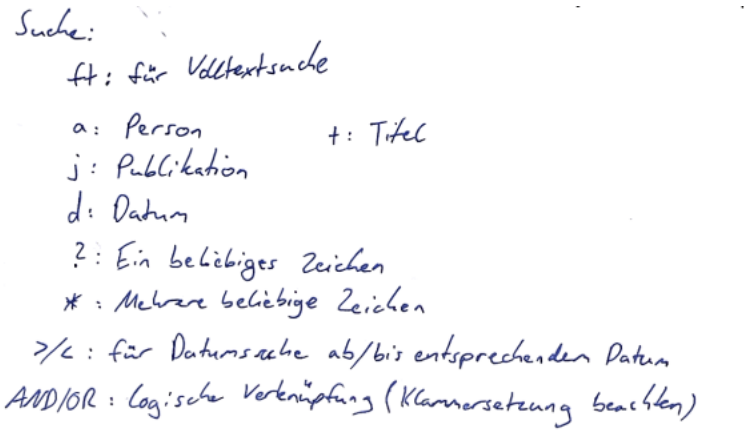

Figure B.1.: Handwritten notes taken by subject 1

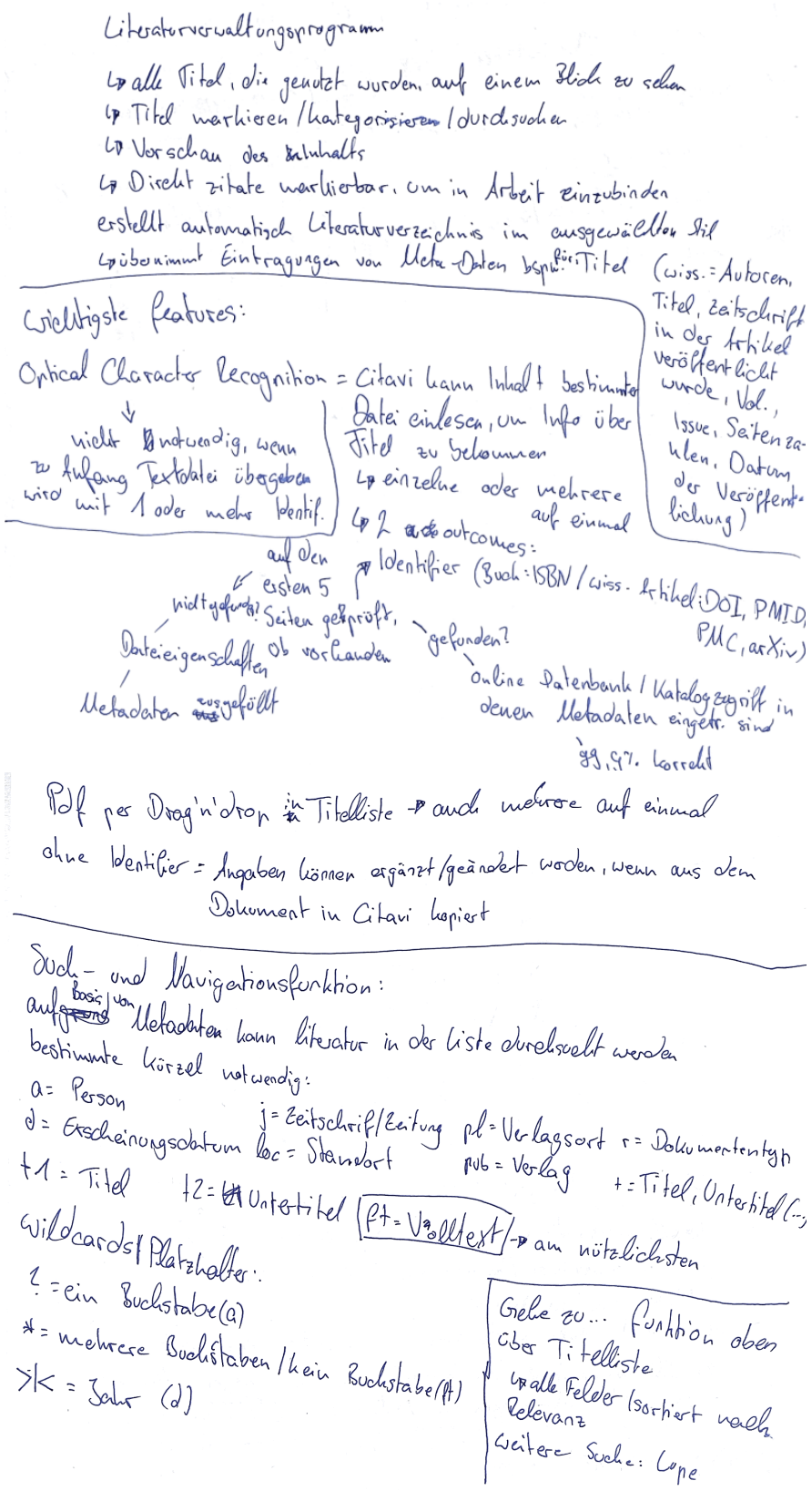

Figure B.2.: Handwritten notes taken by subject 2

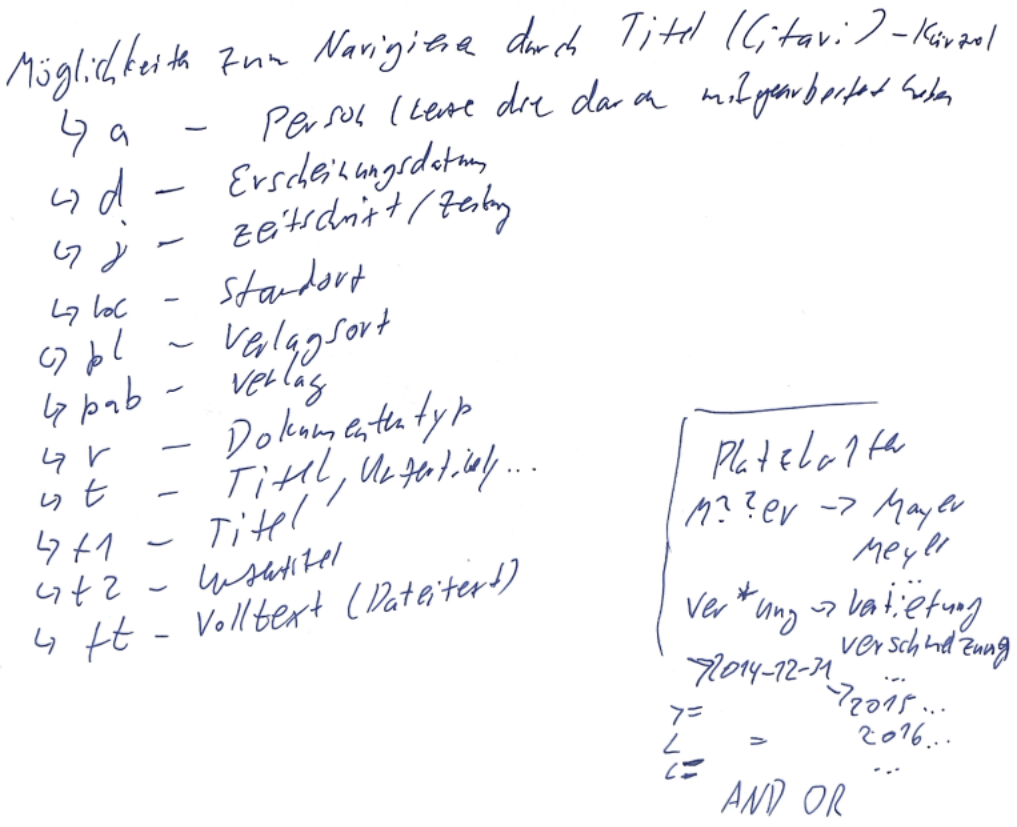

Figure B.3.: Handwritten notes taken by subject 3

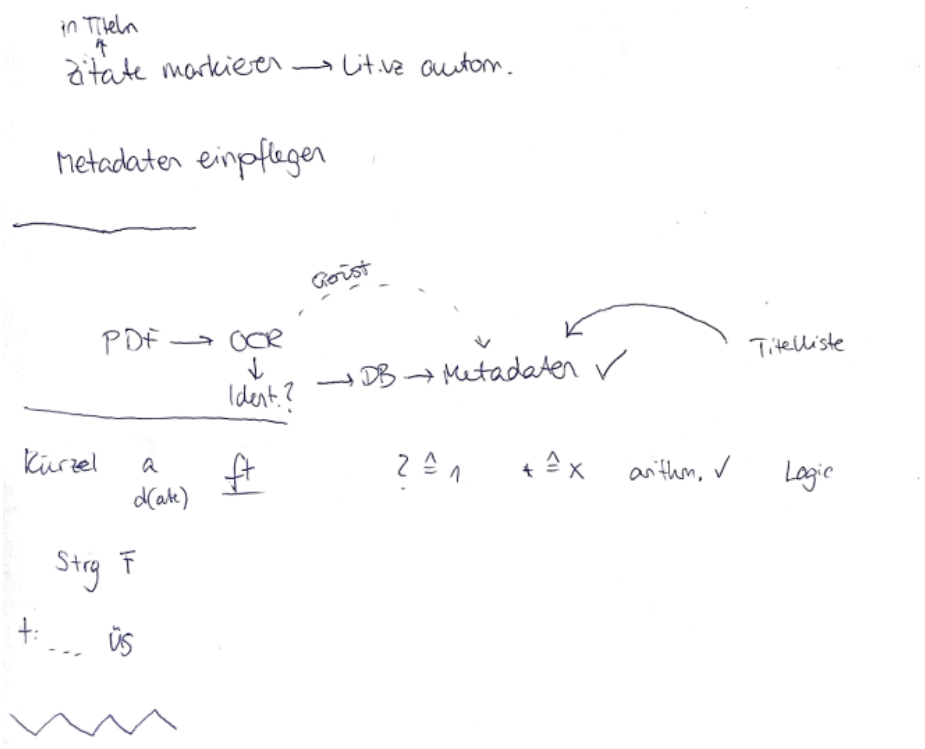

Figure B.4.: Handwritten notes taken by subject 4

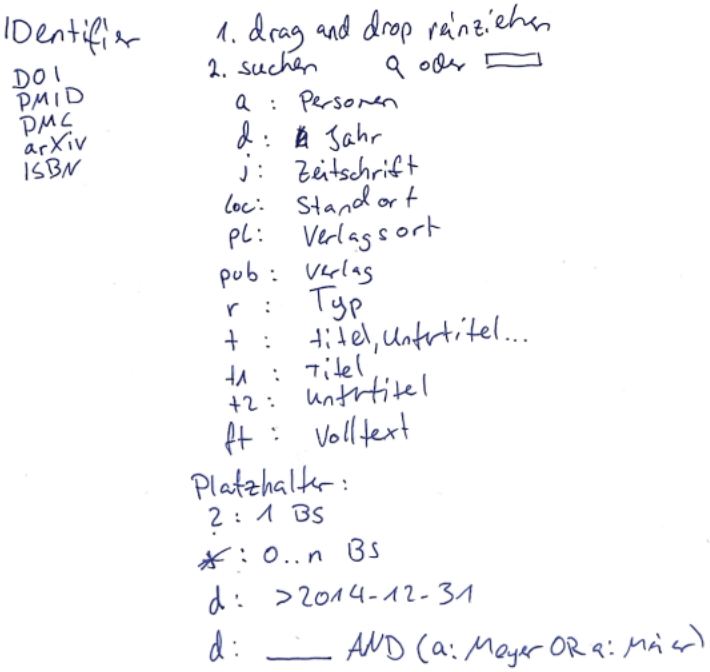

Figure B.5.: Handwritten notes taken by subject 5

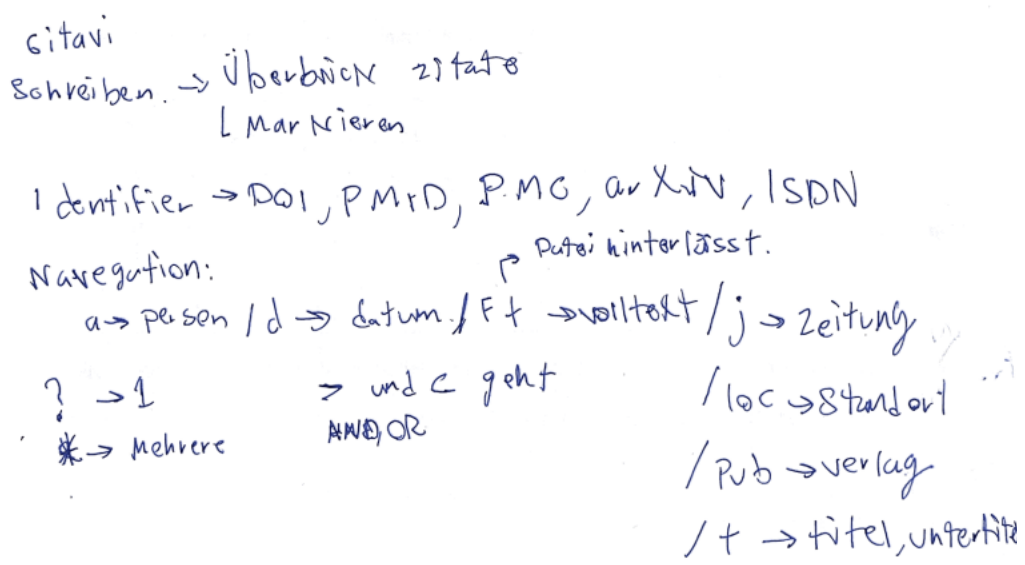

Figure B.6.: Handwritten notes taken by subject 6

B.2. Answers in intermediate questionnaire

B.2.1. German answers

Wie funktioniert Citavi?

Subject 1: Citavi organisiert Literaturquellen und erstellt für fachliche Ausarbeitungen angepasste Referenzen (und Zitate).

Subject 2: Citavi ist eine Software, um bei Arbeiten etc. zu helfen, indem sie bspw. automatisch ein Literaturverzeichnis im gewählten Stil erstellt und auch Zitate markieren und kopieren erleichtert.

Subject 3: Citavi sucht mithelfe von Identifiern in einer Online-Datenbank nach den Metadaten eines Mediums und nutzt diese dann um einen beim Quellenverzeichnis oder anderen Projekten zu unterstützen.

Subject 4: Citavi funktioniert mithilfe von Texterkennung und Verbindungsstellen zu Datenbanken.

Subject 5: Verwaltung von Literaturquellen. Per Drag and Drop können pdf-Dateien oder Word-Dateien eingefügt werden. Citavi fügt automatisch alle Daten ein, die es daraus bekommt oder durchsucht das Internet, falls es einen Identifier gibt. Wenn viele Quellen eingefügt wurden, kann mittels einer Suche und mehreren möglichen Operationen nach einer Quelle gesucht werden. Dabei werden alle gefiltert, auf die die Suche zutrifft.

Subject 6: Sammelt die Papier und wissenschaftliche Dokumente, und such die Schlüsselworten in den Dokumenten.

Wie geht Citavi mit Identifikationsnummern (ISBN, DOI, ...) um?

Subject 1: Die Identifikationsnummern werden aus Dokumenten ausgelesen um an die entsprechenden Metadaten zu kommen..

Subject 2: Citavi liest die Identifier und gibt daraufhin die benötigten Metadaten ein, nachdem die Datei in bspw. einer Datenbank aufgerufen wurde.

Subject 3: Citavi durchsucht die eingefügten Dateien danach und nutzt sie dann als Identifier.

Subject 4: Die Identifikationsnummern werden erkannt vom OCR und dann werden die dazugehörigen Titel automatisch zu Citavi hinzugefügt.

Subject 5: Citavi liest ID-Nummern raus und sucht im Internet nach den wichtigen Daten und füllt die Felder automatisch aus.

Subject 6: Das Programm sucht nach den Identifikationsnummern und vergleicht sie mit den vorhandenen Daten in dem DB von Citavi, sodass die wichtigste Info schon im Programm sind

Nach welchen Attributen lassen sich Titel (Bücher, Zeitschriftenartikel, Sammelbände, ...) eines Citavi-Projekts durchsuchen?

Subject 1: Volltext (ft), Person (a), Jahr (d), Titel (t), Standort (loc), Publikation (pub), Publikationsstandort (pl), Untertitel (t1)

Subject 2: Nach den Attributen a (Autor), d (Titel), ft (Volltext), Jahr, t, t1, t2

Subject 3: Nach Veröffentlichkeitsdatum, Autor/Namen, Begriffen um Text oder Titel, ISBN, Wortanfängen/Endungen,

Subject 4: Man kann die Titel nach Veröffentlichungsdatum, Autor*innen, Seitenzahl, Titel u.v.m. untersuchen, oder man hat eine "Freitext-Eingabe" und durchsucht die Titel nach diesem User Input.

Subject 5: Titel, Namen, Erscheinungsjahr, Untertitel, Volltextsuche, Verlag (Verlagsort)

Subject 6: Ort, Erscheinungsdatum, Verlag, Untertitel, Identifikationsnummer, Text und Jahr

B.2.2. English translation

How does Citavi work?

Subject 1: Citavi organizes literary sources and creates references (and citations) appropriate for professional essays.

Subject 2: Citavi is a software for helping with essays etc. by, e.g., automatically creating a reference list in a chosen style and making highlighting and copying quotations easier.

Subject 3: With the help of identifiers Citavi searches for the meta data of a medium within an online database and uses these to help you with a reference list or other projects.

Subject 4: Citavi works via text recognition and connections to databases.

Subject 5: Management of literary sources. PDF files or Word files can be imported via Drag and Drop. Citavi automatically adds all the data that it can extract or searches the internet, if there is an identifier. If a lot of sources were added, sources can be searched via several possible operations.

Subject 6: Collects papers and scientific documents and searches for the key words in the documents.

How does Citavi handle identifiers (ISBN, DOI, ...)?

Subject 1: Identifiers are read from the document to obtain the corresponding meta data.

Subject 2: Citavi reads the identifiers and subsequently fills in the required meta data after the file was retrieved from, e.g., a database.

Subject 3: Citavi searches the imported files for them and subsequently uses them as identifiers.

Subject 4: Identifiers are recognized via OCR and then the matching titles are automatically added to Citavi.

Subject 5: Citavi reads the identifiers from the file and searches the internet for the important data and fills out the fields automatically.

Subject 6: The program searches for the identifiers and compares them with the data present in Citavi's database, so that the most important information is already in the program.

Which attributes of titles (books, journal articles, anthologies, ...) in a Citavi project can be searched?

Subject 1: full text (ft), person (a), year (d), title (t), location (loc), publication (pub), publication location (pl), subtitle (t1)

Subject 2: a (author), d (title), ft (full text), year, t, t1, t2

Subject 3: Publication date, author/names, terms around text or title, ISBN, word beginnings/endings

Subject 4: One can search titles for publication date, authors, page numbers, title and more, alternatively there is a free text input and one searches the titles for this user input.

Subject 5: Title, names, publication year, subtitle, full text search, publisher (publication location)

Subject 6: Location, publication date, publisher, subtitle, identifier, text and year

Bibliography

[1] ISO 25000:2014 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE, 2020. URL https://iso25000.com/index.php/en/iso-25000-standards/iso-25010.

[2] Michael Anders, Martin Obaidi, Barbara Paech, and Kurt Schneider. A Study on the Mental Models of Users Concerning Existing Software. In *Requirements Engineering: Foundation for Software Quality*, pages 235–250. Springer International Publishing, 2022. doi: 10.1007/978-3-030-98464-9_18.

[3] Neil Anderson, Peter Herriot, and Gerard P. Hodgkinson. The practitioner-researcher divide in Industrial, Work and Organizational (IWO) psychology: Where are we now, and where do we go from here? *Journal of Occupational and Organizational Psychology*, 74(4):391–411, nov 2001. doi: 10.1348/096317901167451.

[4] Oksana Arnold and Klaus P. Jantke. Mining HCI Data for Theory of Mind Induction. In *Data Mining*, chapter Chapter 4, pages 47–68. InTech, 2018. doi: 10.5772/intechopen.74400.

[5] Christian Bellebaum, Patrizia Thoma, and Irene Daum. *Neuropsychologie*, chapter Emotionen: Selbst erfahren und bei anderen erschließen, pages 127–144. VS Verlag für Sozialwissenschaften, 2012. ISBN 978-3-531-16827-2.

[6] Dimitri Bohlender and Maximilian A. Köhl. Towards a Characterization of Explainable Systems. *CoRR*, abs/1902.03096, 2019. URL http://arxiv.org/abs/1902.03096.

[7] Pamela Briggs. What we know and what we need to know: the user model versus the user's model in human-computer interaction. *Behaviour & Information Technology*, 7(4):431–442, 1988. doi: 10.1080/01449298808901887.

[8] John M. Carroll, Robert L. Mack, and Wendy A. Kellogg. Chapter 3 - Interface Metaphors and User Interface Design. In Martin Helander, editor, *Handbook of Human-Computer Interaction*, pages 67–85. North-Holland, Amsterdam, 1988. ISBN 978-0-444-70536-5. doi: https://doi.org/10.1016/B978-0-444-70536-5.50008-7. URL https://www.sciencedirect.com/science/article/pii/B9780444705365500087.

[9] Larissa Chazette and Kurt Schneider. Explainability as a non-functional requirement: challenges and recommendations. *Requirements Engineering*, 25:493–514, June 2020. doi: https://doi.org/10.1007/s00766-020-00333-1.

[10] Larissa Chazette, Wasja Brunotte, and Timo Speith. Exploring Explainability: A Definition, a Model, and a Knowledge Catalogue. In *2021 IEEE 29th International Requirements Engineering Conference (RE)*, pages 197–208, 2021. doi: 10.1109/RE51729.2021.00025.

[11] Arun Das and Paul Rad. Opportunities and Challenges in Explainable Artificial Intelligence (XAI): A Survey. *CoRR*, abs/2006.11371, 2020. URL https://arxiv.org/abs/2006.11371.

[12] Julio Cesar Sampaio do Prado Leite and Claudia Cappelli. Softwaretransparenz. *Wirtschaftsinformatik*, 52(3):119–132, 2010. doi: 10.1007/s11576-010-0219-1.

[13] Goerschwin Fey, Martin Franzle, and Rolf Drechsler. Self-Explanation in Systems of Systems. In *2022 IEEE 30th International Requirements Engineering Conference Workshops (REW)*. IEEE, 2022. doi: 10.1109/rew56159.2022.00023.

[14] Frank G. Halasz and Thomas P. Moran. Mental models and problem solving in using a calculator. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems - CHI '83*. ACM Press, 1983. doi: 10.1145/800045.801613.

[15] Todd Kulesza, Simone Stumpf, Margaret Burnett, Sherry Yang, Irwin Kwan, and Weng-Keen Wong. Too much, too little, or just right? Ways explanations impact end users' mental models. In *2013 IEEE Symposium on Visual Languages and Human Centric Computing*. IEEE, sep 2013. doi: 10.1109/vlhcc.2013.6645235.

[16] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, 2019. ISSN 0004-3702. doi: https://doi.org/10.1016/j.artint.2018.07.007. URL https://www.sciencedirect.com/science/article/pii/S0004370218305988.

[17] Jakob Nielsen. Ten Usability Heuristics, 1994. URL https://www.nngroup.com/articles/ten-usability-heuristics/. obtained: 2023-01-16.

[18] Donald A. Norman. *Mental Models*, chapter Some Observations on Mental Models, pages 7–14. Taylor & Francis Group, 1 edition, 1983. ISBN 9781315802725.

[19] Donald A. Norman. *The Design of Everyday Things*, chapter The Psychology of Everyday Things, pages 141–143. Basic Books, 1988. ISBN 978-0-465-06710-7.

[20] Donald A. Norman. *The Design of Everyday Things*, chapter 3: Knowledge in the Head and in the World, pages 54–80. Basic Books, 1988. ISBN 978-0-465-06710-7. metaphors, natural mappings.

[21] Adrian M. Owen. Cognitive planning in humans: Neuropsychological, neuroanatomical and neuropharmacological perspectives. *Progress in Neurobiology*, 53(4):431–450, 1997. doi: 10.1016/s0301-0082(97)00042-7.

[22] Arjun Panesar. *Machine Learning and AI for Healthcare*, chapter Ethics of intelligence, pages 207–254. Springer-Verlag GmbH, 2019. ISBN 1484237994. URL https://www.ebook.de/de/product/35418157/arjun_panesar_machine_learning_and_ai_for_healthcare.html.

[23] Marina Papastergiou. Students' Mental Models of the Internet and Their Didactical Exploitation in Informatics Education. *Education and Information Technologies*, 10(4):341–360, 2005. doi: 10.1007/s10639-005-3431-7.

[24] Soo Young Rieh, Ji Yeon Yang, Elizabeth Yakel, and Karen Markey. Conceptualizing Institutional Repositories: Using Co-Discovery to Uncover Mental Models. In *Proceeding of the third symposium on Information interaction in context - IIiX '10*. ACM Press, 2010. doi: 10.1145/1840784.1840809.

[25] Martina-Angela Sasse. How to T(R)AP Users' Mental Models. In M. J. Tauber and D. Ackermann, editors, *Mental Models and Human-Computer Interaction 2*, volume 2 of *Human Factors in Information Technology*, pages 59–79. North-Holland, 1991. doi: https://doi.org/10.1016/B978-0-444-88602-6.50007-4. URL https://www.sciencedirect.com/science/article/pii/B9780444886026500074.

[26] Yan Zhang. The influence of mental models on undergraduate students' searching behavior on the Web. *Information Processing & Management*, 44(3):1330–1345, 2008. doi: 10.1016/j.ipm.2007.09.002.