UI-, User-, & Usability-Oriented Engineering of Participative Knowledge-Based Systems

Martina Freiberg

Würzburg University Press

Martina Freiberg

UI-, User-, & Usability-Oriented Engineering of Participative Knowledge-Based Systems

Martina Freiberg

UI-, User-, & Usability-Oriented Engineering of Participative Knowledge-Based Systems

Würzburg University Press

Dissertation, Julius-Maximilians-Universität Würzburg
Fakultät für Mathematik und Informatik, 2015
Gutachter: Prof. Dr. Frank Puppe, Prof. Dr. Marc Erich Latoschik

Impressum

Julius-Maximilians-Universität Würzburg
Würzburg University Press
Universitätsbibliothek Würzburg
Am Hubland
D-97074 Würzburg
`www.wup.uni-wuerzburg.de`

ISBN 978-3-95826-012-2 (print)
ISBN 978-3-95826-013-9 (online)
URN urn:nbn:de:bvb:20-opus-106072

To my loved ones.
Your sun will always lighten my paths.

Preface

For participative knowledge-based systems, the solution for a problem is derived by close cooperation of user the and the system. In many domains, such as medicine or law, this is the most promising strategy because a complete formalization of the required knowledge can be very expensive. In addition, the user would have to answer a large amount of questions regarding the problem at hand. To avoid such extensive user-system dialogs, a transparent presentation of the solution state and more actions than simply answering the questions strictly as asked by the knowledge-based system are required. Since users have different prior knowledge about the domain and computer handling, the design of user interfaces (UIs) for such participative knowledge systems is a big challenge—and the topic of this book.

Martina Freiberg identifies three core dialog patterns: The Questionnaire, the Interview, and the novel Clarification pattern—as well as one additional pattern hybrid. Particularly this latter Hierarchical Clarification pattern variant enables the user to focus on the critical aspects of a problem in detail whilst at the same time other aspects can be treated on a rather high abstraction level by answering only a few questions. Another core contribution is the design and implementation of the generic knowledge systems engineering tool ProKEt. ProKEt allows for configuring different UIs and framing applications for knowledge-based systems. The tool has been practically applied and evaluated in different domains. Examples are multi centric medical data gathering about special surgeries to identify best surgery practices; or legal consultation systems, where users can interactively clarify questions like *are the circumstances of a job dismissal legally correct* and concentrate on those aspects they feel to be the most critical.

Generally, the success of many knowledge system projects depends strongly on the quality of the UI. This book provides a broad overview on this issue, which in the context of knowledge-based systems is still often neglected. Therefore, tailored KBS UI patterns and development tools which help to solve this problems, as well as the results of diverse user studies, are described.

Frank Puppe

Acknowledgements

First and fore-mostly I want to thank my main supervisor Frank Puppe for providing me the opportunity to pursue and evolve this topic. You always offered an open door to discuss theoretical and practical issues of my research, and you have been a great support also in any other regard. Many thanks also to my co-supervisor Marc Erich Latoschik. Our paths did not cross before a rather late stadium of this work. Still, your advice concerning especially the usability-related aspects, the last user studies, and finally the publication, have been most valuable. Many thanks also to all my kind colleagues at the department—especially, but not exclusively, those that started as part of my ”doctoral researcher generation”. You too always were open to discussions of all kinds and lent helping hands whenever needed—even and especially during those times I did not work at the institute regularly due to the arrival of my two little angels. I also warm-heartedly thank our department's good fairy Petra Braun, who always helped with the smaller and bigger organizational and bureaucratic tasks in all those years.

Next, I want to thank Joachim Baumeister: Our first paper-collaboration on visualization, following my diploma, can surely be blamed to have sparked my wish for staying in academia a bit longer. Several further research cooperations and many hours of valuable discussions also helped this work progress. Thanks also to the entire *denkbares* staff for always being a great support regarding everything related to d3web and KnowWE. Finally, I also thank Dr. Schimmer, Karin Wolz and the entire staff of RenoStar, Dr. Dietz, F. Myusoms, and especially Iris Kyle-Reinhase for their kind and professional cooperation which helped our practical projects succeed.

Last, but definitely not least, I deeply and most cordially thank my family. You all supported my plans from the beginning on and during all those years. You provided the most valuable, priceless personal network for taking care of my little ones so I could pursue my research path without doubts and fear. Especially you, Constantin, always kept up my belief and motivation to finish this work and gave me any required strength. My little angels, you are my true reason for life—your easygoing smile, laughter, and happiness mean everything to me.

Contents

Abbreviations

AI	Artificial Intelligence
APM	Agile Process Model
CBR	Case Based Reasoning
CSV	Comma Separated Value (format)
CW	Cognitive Walkthrough
DSS	Decision Support System
EAM	Encompassing Agile Process Model
ES	Expert System
HE	Heuristic Evaluation
HCI	Human-Computer Interaction
Hi-fi	High fidelity (in prototyping)
KA	Knowledge Acquisition
KB	Knowledge Base
KBS	Knowledge-based System
KBSE	Knowledge-based Systems Engineering
KBS UI Specs	KBS UI Specification
Lo-fi	Low fidelity (in prototyping)
MC	Multiple Choice
OC	One Choice
ProKEt	Prototyping and Knowledge Systems Engineering Tool
PUE	ProKEt Usability Extension
RE	Requirements Engineering
RS	Recommender System

SE	Software Engineering
TA	Thinking aloud technique
UCD	User-centered Design
UE	Usability Engineering
UI	User Interface
UISpecs	ProKEt UI Specification File

Part I

Engineering Participative Knowledge-Based Systems

Chapter 1

Introduction

Make everything as simple as possible, but not simpler.

Albert Einstein (1879–1955)

This notion about *reasonable simplicity* applies particularly well to software intended to support complex tasks. As one of the most popular and established branches of *Artificial Intelligence* (AI), knowledge-based systems (KBS) fall well into that category. Examples of the manifold application contexts of KBS are fault diagnosis for technical devices, e.g., [Cebi et al., 2009, Nghia and Puppe, 2009]; ecological classification or diagnosis applications, e.g., [Neumann et al., 2002, Wan and Lei, 2009]; or advanced medical applications, e.g., [Chen et al., 2012, Huettig et al., 2004, Mersmann and Dojat, 2004].

Basic Paradigm Shift Towards User Participation In the KBS domain, an increasing shift can be observed: From the former aim, to construct the 'perfect intelligent software', towards user-centered software, that provides mutual supplementation and enrichment between the user and the system—pursued, e.g., by [Kajiyama and Satoh, 2014]. Such latter artifacts generally require a higher degree of user participation, in the sense that users contribute their personal knowledge to the problem solving process. For leveraging reuse, such software optimally also is easily adaptable for diverse user groups and needs.

More specifically, for *participative KBS* we define the non-functional requirements *transparency, configurability, quality, and evolvability*. *Transparency* refers to the mediation of the KBS' status—e.g., feedback regarding answered questions and the consequences thereof. *Configurability* both refers to a general user adaptability (e.g., multilingualism features) and to the degree of exploration and user control (e.g., shortcuts for expert users and more precise yet lengthier interrogations for novices). Even if not mimicking the perfect expert anymore, KBS always depend on the *quality* regarding their contents and offered functionality. Especially an ill-defined knowledge base (KB) can quickly render a KBS unusable or, at least, cause severe user frustration. Finally, participative KBS need to be *evolvable*, so that they can grow more mature with their users over time. Therefore, a basic separation of front-end and KB modules, and the possibility to merge those parts into functional KBS artifacts anytime, is highly beneficial. This allows for adapting the system in a targeted, straightforward manner according to the respective user needs.

Many of those requirements, as well as the overall appropriate degree of simplicity of a KBS, strongly depend on the targeted users. Thus, user-centered development becomes and important aspect. On the one hand, this concerns the KB itself. Directly incorporating domain experts in the KB development process offers the chance for a more efficient task sharing; thus, the KBS developer is enabled to concentrate more on UI/usability related development tasks. On the other hand, this can help to reduce semantical and conceptual errors of the KB that might arise in case a knowledge engineer—who typically is no domain expert—formalizes the knowledge all alone. Often, the required KBs also are quite comprehensive, due to the fact that (participative) KBS often target highly expertise domains. There, a carefully considered and tailored UI and interaction solution becomes another essential foundation for easing the KBS usage as much as possible.

Mutual Impact of KBS User Interface (UI) and KB Regarding the KBS front-end design, an important aspect to consider is the *strong mutual impact of KBS UI and KB*. For example, when KB items, such as questions, exhibit highly varying lengths, those might not be well presentable by certain UI types or widgets. This in turn can considerably influence the overall perception of the entire KBS. The other way around, certain widgets or UI configurations already prescribe quite clearly, how questions need to be formulated to be usable with that widget. In current KBS development, however, the UI often is not considered (thoroughly enough) until the KB is ready for deployment. Then, annoying surprises can occur when trying to fit all KB components into the envisioned, assumedly appropriate UI in an appealing manner.

The KBS UI Flexibility Problem with Standard GUI Builders In the context of KBS UI development, another problem arises: KBS are often highly flexible regarding the presented question sequence. For example, follow up- or abstraction questions often only become included once appropriate input for other questions had been provided. Thus, a KBS UI often cannot be designed optimally by tools that produce rather static UI solutions—like most of today's available standard GUI builders or prototyping tools. Mimicking those mechanisms may partly be possible with interactive prototyping approaches—e.g., using slideshows or transparencies—particularly for presenting transitions in single question displays. Yet, especially in cases where a lot of questions are rendered simultaneously, and only several selected parts need to be altered per transition, the appropriate representation by standard approaches may become a rather cumbersome task.

Despite the inarguable importance of the KBS UI, however, current KBS engineering (KBSE) approaches mostly still focus on knowledge acquisition (KA). This in turn often still leads to ad-hoc, non-optimal, and little reusable KBS UI solutions. Undoubtedly, one major factor for the lasting acceptance and usage—i.e., success—of any KBS actually *is* the quality of its KB, c.f., [Baumeister and Freiberg, 2010, p. 2]. Depending on the target domain and application context, KB development alone still often *is* a costly and challenging task [Baumeister, 2004, p. 27]—even despite the availability of manifold mature tools, such as Protégé [Noy et al., 2000, Protégé Stanford, n.d.], myCBR [Roth-Berghofer et al., 2012], or KnowWE [Baumeister et al., 2011]. A *lack of general best practices/patterns* for successful KBS (UI) solutions—or of a public availability of such insights, c.f., [Duan et al., 2005, p. 800/810]—further adds to the issue.

The persistent focus on the KB is a bit surprising, considering that UI design and particularly usability have become highly topical issues in general software engineering, web design, and human computer interaction (HCI). Usability thereby basically describes the quality of a (software) product regarding mainly its efficiency, effectiveness, and satisfaction of its users, c.f. [UPA, n.d.]. In HCI, *Usability Engineering* (UE) is a known usability-fostering approach that provides "structured methods for achieving usability in user interface design during product development" [Mayhew, 1999, p. 2]. Equally renowned is *User-centered Design* (UCD). There, processes basically "focus on users through the planning, design, and development of a product" [UCD def. by UPA, n.d.]. In the KBS research and practitioners' community, however, UE and UCD methods still remain rather unconsidered. Research efforts in that direction are found only sparsely in the literature, e.g., reported for case-based reasoning systems in [He, 2013]. This was further confirmed by our literature review covering the past six years of KBS development—reported in detail in Appendix B. One potential reason may be the still common caveats regarding the application of usability techniques—mostly by those, not yet actively practicing usability. As example, despite the availability of *affordable techniques*—as suggested, e.g., by [Nielsen, 1993b]—usability still often is considered to be quite expensive.

Due to the described paradigm shift of KBS, the mutual effects between KB and UI, the influence of the UI on the overall KBS usability, as well as the general challenge to develop an appropriate UI/interaction in the context of KBS, the need for novel conceptions regarding KBSE approaches is obvious. In this thesis, we present methods and tools regarding the *encompassing* engineering of KBS, i.e., incorporating so far rather neglected UI- and usability-related activities. In the next section, we subsume our approach both regarding the theoretical and the practical contributions.

1.1 Approach

The universal goal of encompassing KBSE support basically led to a rather interdisciplinary orientation of the proposed research; thereby, mainly the areas of software engineering (SE), knowledge acquisition/engineering (KA), user interface (UI) design, interaction design, and usability are touched.

In Theory: Encompassing Engineering of (Participative) KBS Participative KBS solutions are particularly appropriate in highly expertise domains. There, proficient users mostly do *not* require KBS that support the process of finding solutions—such users often are well able to suspect appropriate candidates themselves. Rather, some form of second opinion and detailed inspection regarding a chosen solution is demanded. This in turn requires KBS modules, targeted at a single solution, that allow for an interactive and in-depth exploration of all knowledge related to that solution. For this context, we introduce *Clarification KBS* as an innovative, highly participative KBS paradigm that mashes up consultation and justification components.

To elevate KBSE to a more encompassing level, we suggest the following key components:

- *Extensible prototyping*—a tailored form of evolutionary UI prototyping
- *KBS UI Patterns*
- Suitable *Usability Instruments*

Extensible prototyping founds on treating KBS core UI/interaction, framing functionality, and KB as separate components, and on defining two extension steps for their fusion: Merging the core UI and the KB into functional core KBS artifacts (first extension). And merging the core KBS artifact with the desired framing functionality for retrieving fully productive KBS solutions. Therewith, extensible prototyping explicitly fosters tightly intertwined UI and KB development.

Tailored KBS UI patterns, as second component for encompassing KBSE, offer the major advantage to foster reusing proven solutions in similar contexts efficiently. This not only helps to avoid common mistakes, but can accelerate KBSE in general.

Suitable *usability instruments*, finally, help to assess the KBS solution in a structured manner. Thereby, both implicit usability-fostering activities and explicit usability techniques provide valuable opportunities in the context of KBSE.

Those components are not intended to constitute a stand-alone development methodology. Similar to particular spices for pepping up basic recipes, they rather aim at flexibly upgrading known, KB-focussed KBS development models. This was a deliberate decision due to the manifold existing and proven KBSE approaches in the literature that just require a little adaption for becoming more encompassing models. In this thesis, we demonstrate the integration of the proposed components with an agile KBS development approach—the Agile Process Model [Baumeister, 2004]—and we report resulting positive experiences. The decision for the agile model was mainly owed to the fact, that agile methodologies are able to cope with two problems arising particularly also during KBSE: Unknown technical feasibility of the project, and inability to give a full specification of the final system in advance, c.f., [Baumeister, 2004, pp. 13–14].

In Practice: The ProKEt Toolkit for Encompassing KBSE For practical support of encompassing, user-centered KBSE, we introduce the tailored KBSE toolkit *ProKEt*. ProKEt offers a basic selection of KBS UI patterns as well as corresponding configuration options out of the box. Thereby, ProKEt defines UI widgets and styles in a highly modular manner, allowing for flexibly and interchangeably combining and nesting them. Based thereupon, the tool allows for adapting and extending the predefined KBS UI library on various levels of (programming) expertise. Those features strongly support the motivated user participation regarding the UI development process—e.g., by performing live implementation sessions together with the expert, where the UI is adapted in real time.

For practical usability support, ProKEt offers quantitative (tailored click log files) and qualitative (questionnaire, personal feedback) data collection features which can be activated easily and effortlessly for all ProKEt artifacts anytime. The resulting data can be evaluated by ProKEt's integrated analysis component. This supports the calculation of several basic usability metrics, such as *Average Task Time* or *Error Rate* for the quantitative data. Alternatively, the data can be exported into a comma separated value (CSV) format. CSV can be imported and thus easily further investigated by many comprehensive statistical analysis tools. Potential advantages of a KBSE tool with integrated usability features are: Low costs regarding the development of (KBS/UI) alternatives; and the consequential seamless support regarding highly iterative usability testing (and analysis of the results) of the implemented alternatives (c.f. [Hakim and Spitzer, 2000]).

The practical realization of the suggested paradigm of intertwined KB and UI development, based on extensible prototyping, is one major strength of ProKEt. Primarily developed as a KBS UI development tool, ProKEt does not support the creation of KBs itself. Yet it offers extension points for seamlessly integrating two selected KA tools specifically regarding [d3web, n.d.] KBs: (1) KnowOF, a standard office based KA environment that uses spreadsheet- or word processor based knowledge specifications; and (2) KnowWE [Baumeister et al., 2011], a semantic wiki for collaborative KA. The tight coupling with ProKEt then allows for directly reviewing the KB within a functional KBS artifact—which can be invoked by a mere button click within KnowOF or the wiki. The resulting, visual way of investigating the KB in a realistic context can drastically increase the interest of domain experts to actively participate in KA tasks.

1.2 Results

So far, several projects concerning knowledge-based documentation- and consultation systems proved, that both the encompassing development approach as well as the ProKEt toolset are able to meet their goals: To leverage user-centered KBS development with an increased focus on UI design and usability evaluation. The projects themselves, as well as the corresponding development activities and experiences, have already (partly) been described in former publications: [Freiberg et al., 2012] describes the development of two medical knowledge-based documentation systems, *Mediastinitis Registry* and *EuraHS*. In [Freiberg and Puppe, 2012a], the *JuriSearch* project is reported; this project targets at realizing a legal consultation portal that integrates several knowledge-based clarification consultation modules. Experiences during these projects further showed, that ProKEt with its seamlessly integrated KA extensions indeed enables expert users to support the KA process highly autonomously. Together with the practical application of the proposed encompassing agile development model, this led to a stronger user identification with the system, to the better compliance with their actual needs, and to less general (basically avoidable) usability flaws in the final artifacts.

Regarding specifically the novel *Clarification KBS* type, the JuriSearch project provided the chance for implementing and thoroughly evaluating a tailored instantiation of the *Hierarchical Clarifier* UI pattern. The resulting *ITree* style basically allows for a highly explorative usage and visual, tree-based interaction; further, it enables users to contribute their personal expertise to the problem solving process in autonomously choosing the abstraction level on which to answer questions. ITree further also supports the idea of learnability, i.e., to convey knowledge regarding the target domain to the user by simply using the system. Various evaluations showed promising results for ITree: Whereas its overall (UI and interaction related) perception still offers room for improvement, the achieved success rate was convincing (90–100 %) in the last evaluations. Further, also qualitative feedback confirmed the assumed characteristics and advantages of this KBS type.

ProKEt as KBSE tool further provided valuable support for realizing reference implementations of several of the KBS UI patterns, proposed in this thesis. Five selected variants have been assessed with regards to their usability and utility. Three pattern realizations—Box Questionnaire, Daily Questionnaire, and Strict Interview—excelled as appropriate also and especially for laymen- or one-time users in little- to medium complex domains. Hierarchical Interview and Hierarchical Clarifier, in contrast, seem best applicable in the context of experienced to expert users, especially regarding more comprehensive domains.

The proposed work thus contributes to KBSE research in providing an encompassing theoretical and practical approach to develop and evaluate (participative) KBS solutions—thereby particularly focussing on UI and usability aspects, and additionally also fostering user-centered development at various points.

1.3 Structure of the Thesis

This work encompasses three major parts. The *first part* describes the background and theoretical foundation of the proposed approach. In Chapter 2, we provide a delimitation of the general KBS concept, the introduction of the novel *Clarification KBS* type, and a discussion of related work for the research directions touched in this thesis. Subsequently, we propose the encompassing KBSE approach from the theoretical perspective in Chapter 3.

In the *second part* of this thesis, we discuss the practical aspects concerning encompassing KBSE. Therefore, in Chapter 4 we first introduce general presentation and configuration options for the key important KBS module—the core input module. Based thereupon, we propose a collection of four KBS core input UI patterns, that can be fine-tuned into 10 KBS UI pattern variants. In Chapter 5, we discuss basic options for representing the KBS context. Thereby we understand all core KBS UI modules except the core input—i.e., results presentation, the integration of auxiliary information, and the KBS justification. Afterwards, we introduce the tailored prototyping and knowledge systems engineering tool *ProKEt* in Chapter 6. Therefore, we subsume the (technical) capabilities of the tool and we look at how ProKEt supports the proposed, encompassing development approach. The practical part of this work is concluded by a comprehensive report of diverse evaluation activities and case studies in Chapter 7. Those encompass: Benchmarks regarding the tool ProKEt itself, and a summary of the application ProKEt and the encompassing development approach in actual KBS projects. The experiences regarding the realization and usability assessment of selected KBS UI patterns. And the iterative, evaluation-driven evolvement of ITree, a tailored clarification KBS type for the legal domain.

The *third part* of this thesis first subsumes the core aspects of the proposed work in Chapter 8. A discussion of the main contributions of the proposed work to current KBS research is provided in Chapter 9. An outlook to promising future work finally concludes this work in Chapter 10.

The Appendices of this work provide additional information on selected framing aspects. Appendix A introduces selected KBs used in this thesis. The results of the particularly targeted, extensive literature research, intended to gain insights regarding the integration of prototyping-/UI-/usability activities in current KBS development, are summarized in Appendix B. In Appendix C, we provide exemplary task descriptions, problem statements, and concluding questionnaires that were used in the conducted evaluation studies. Appendix D subsumes renowned usability standards for quick reference. Finally, Appendix E lists all additional materials and sources as provided in a freely accessible online repository.

Chapter 2

Knowledge-based Systems Engineering Today

Knowledge-based systems (KBS) have been the topic of vivid research over the past years. Thus, there exist manifold definitions, as well as differing development methodologies and tools. This chapter first provides an introduction of the core concepts related to KBS in Section 2.1. In Section 2.2 we propose a novel KBS paradigm: *Clarification KBS*, as a mashup of consultation and justification within a single artifact. We conclude this chapter with a discussion of the current state of the art of KBS engineering (KBSE) in Section 2.3.

2.1 KBS—Basic Concepts

The manifold KBS-related concepts and categorizations known today unfortunately often are used in an ambiguous manner. In the following, we define some basic notions related to KBS. Therewith, we delimitate our understanding of terms to those used in the literature, where necessary. We first cover basic aspects such as *KBS core- and UI components* (Section 2.1.1–2.1.2), common *KBS use cases* (Section 2.1.3), and a *user type classification* (Section 2.1.4). Then, we motivate the benefits of *participative KBS* by defining the participation aspect for KBS (Section 2.1.5) in more detail.

Regarding the software artifact itself, the terms most often used ambiguously with KBS are *expert system (ES)*, *decision support system (DSS)*, and *recommender system (RS)*.

KBS basically are systems that use artificial intelligence (AI) to solve problems, e.g., [Akerkar and Sajja, 2010, Preface, p. XVII]. According to, e.g., [Kurbel, 1989, Puppe, 1993, Jackson, 1998, Akerkar and Sajja, 2010], ES basically can be seen as a particular subtype of KBS. This is due to their often stated objective to *mimic the knowledge and the ability for drawing conclusions and making decisions of experts regarding restricted problem areas.* We assent to this view as in turn also more generic types of KBS exist. For example, *documentation* KBS, see [Baumeister, 2004, p. 17]—those may use expert knowledge for implementing an 'intelligent' data input process, yet do not necessarily require decision-making ability.

DSS and *RS*, in contrast, only exhibit a loose relationship to KBS: They may include KBS modules/technologies, but basically follow fundamentally different objectives. DSS are *complex, computer-based information systems that support business or organizational decision-making activities*, c.f. [Akerkar and Sajja, 2010, p. 12]; they consist of several separate modules and interconnected software tools, each supporting specific tasks such as combining and retrieving raw data. RS mainly denote *personalized information filtering systems that infer user preferences by* *analyzing available user data, information about other users, as well as information about the environment*, c.f. [Sebastia et al., 2009]; those are used mostly in an online business context.

2.1.1 KBS Core Collaboration Structure

Figure 2.1 depicts the major components of a KBS and their basic collaboration structure. This consents (mostly) with examples of both seminal basis literature and current works, e.g., [Kurbel, 1989, Gottlob et al., 1990, Puppe, 1993, Akerkar and Sajja, 2010].

The knowledge base (KB) defines the respective domain- or expert knowledge. A KB basically consists of several knowledge classes. In this work we adhere to the view discussed in [Baumeister, 2004], that proposes four typical knowledge classes: *Ontological knowledge*, that mainly defines hierarchies of questions and solutions. *Structural/inference knowledge* for deriving and valuing interview objects (such as abstraction questions) and solutions, thus corresponding to the concept of *derivation knowledge* [Puppe, 1993]. *Strategic knowledge* for guiding the questioning sequence, thus corresponding to *control knowledge* [Puppe, 1993]. And *support knowledge*, that informally describes solutions, questions, and answers in more detail, thus partly corresponding to the auxiliary information concept in this work.

The KB generally is formalized via some kind of knowledge acquisition (KA) module: Either directly by the domain expert; or indirectly, i.e., with the additional support of a knowledge engineer. Therefore, specialist self-contained tools can be applied; examples are Protégé [Gennari et al., 2003] or KnowWE [Baumeister et al., 2011]. Alternatively, KA facilities may also be directly integrated with the KBS itself; or with the respective development tool that is used for implementing the KBS front-end. The latter variant is realized by the KBSE tool *ProKEt*, introduced in this thesis (see Section 6). Optionally, a learning module can enable the KBS to update and adapt its KB according to previously performed sessions and respective results. Such functionality can either be added as explicit separate component, c.f. [Akerkar and Sajja, 2010], or it can be integrated with one of the other modules, c.f., [Kurbel, 1989, Gottlob et al., 1990]. The realization thereof, however, denotes a comprehensive research branch on its own, thus a detailed reflection is out of scope of this work.

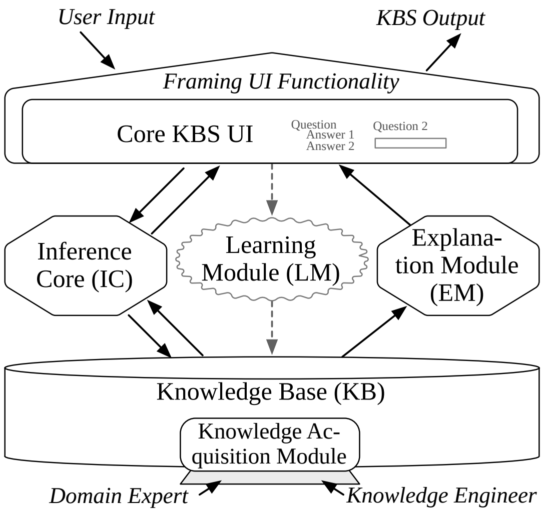


Figure 2.1: KBS Core Components and their Collaboration Structure.

The UI basically is responsible for collecting the input. Input is either explicitly entered by (human) users. Or indirectly fed into the system, e.g., in the form of sensor data in the context of embedded KBS—i.e., KBS, that are completely integrated into the devices they operate with. In general, KBS core UI and framing UI functionality can be distinguished.

The KBS core UI thereby encompasses all UI modules relevant for supporting the data input and results presentation tasks; that is, question and answer widgets, solution panel, etc. The framing UI functionality entails all remaining, potentially valuable features, such as multilingualism, session management, or user management. Based on the user input and the static KB contents, the inference core derives its results: New knowledge facts, or conclusions, which again are presented via the UI. The explanation module generates the relevant justifications regarding the actions and conclusions of the KBS—which again can be presented additionally in the UI. Justifications support reproducing and reviewing the KBS's operation and conclusions, c.f., [Puppe, 1993, Altenkrüger and Büttner, 1992]. Thus apart from leveraging the KBS validation task, justifications help to build trust in the KBS, enable users to learn something about the domain, and thus enhance the overall user experience.

Undoubtedly, the precise KB formalization and validation is a critical relevant task—which justifiably gains much attention in current research. Yet, due to its different responsibilities, we argue that the KBS UI is an equally important module. All the more due to the strong mutual interdependencies and mutual influences of KB and UI.

2.1.2 Basic Conceptualization of KBS UI Elements

The characteristic interaction with KBS can be subsumed as *answering questions*; that is, the KBS presents questions in a (more or less) defined sequence and users answer them with the goal to retrieve some result from the system. The KBS UI basically exists of up to four basic modules, each of which requires a distinct representation: Core input, results, justification, and auxiliary information.

- **The core input module:** Responsible for the main user-system interaction. It presents questions and answers and handles the corresponding actions such as redirecting the user request for setting values.
- **The results module:** Presents the results of the KBS session. Depending on the use case, see Section 2.1.3, we differentiate input data summaries and more comprehensive solution panel displays.
- **The justification presentation module:** Creates an optimally well comprehensible presentation regarding *how* the KBS deduced *which* results.
- **Auxiliary information:** Any additional information for further clarification and explanation of KBS interview items or the KBS state itself. Similar to context sensitive help, auxiliary information provides pointed help slices regarding specific interactions, widgets, or system states. Thereby, add-on information for questions can also account as kind of context sensitive help—i.e., help, to answer the question. Yet, auxiliary information goes beyond that basic concept by providing also additional relevant media, or means for enhancing the overall KBS experience, such as progress information.

Results-, justification presentation-, and auxiliary information modules can be subsumed as *KBS Context*. Depending on the particular application context, such context may additionally be provided but is not necessarily required in addition to the *KBS Core Input Module*. As contribution for the practical implementation of KBS, we suggest relevant presentation options for the KBS Core Input Module in Chapter 4 and for the KBS Context in Chapter 5. The theoretical fundament for this classification of presentation options is a conceptualization of the characteristic base elements of (most) KBS UIs. This has already been subject to prior research, c.f., [Freiberg et al., 2012]. In the following, we refine and extend those concepts in more detail.

KBS Core Input—Questions, Answers, Interview Objects We define the foundational interaction metaphor of a KBS as *answering questions*. Thus, for requesting the user input we define the most basic elements of a KBS UI as *questions* and corresponding *answer options*. In the further course, we address both also as *interview objects*.

Depending on the particular input data, certain fixed question types can be defined: *Choice questions* for selecting from one or several predefined answer choices. *Numerical questions* for entering numerical data. *Text questions* for textual data. And *time/date questions* for dates and points in time. Choice questions further divide into *one choice (OC) questions*, which allow to select only a single answer option; and *multiple choice (MC) questions*, that support the selection of multiple answers. A special case of choice questions are *binary choice* questions; i.e., questions with exactly two oppositional choices, such as *yes/no* or *on/off*.

KBS Core Input (Flexible)—Abstraction- & Follow Up Questions Apart from standard questions, where the user actively provides input, KBS further also encompass the concept of *abstractions*. Those sometimes are also referred to as *symptom interpretations*. Abstractions are questions, that are automatically derived by the KBS, depending on specified input for previously answered questions. A prominent example is the calculation of the body mass index (BMI): There, an abstraction question *BMI* is automatically calculated by the KBS in case the user answers the questions *size* and *weight*.

The other characteristic, flexible concept are *follow up items*. Those are questions or entire questionnaires, that are not activated before defined other questions are answered in a certain way. An example is a questionnaire dealing with female health issues in a medical KBS; such a questionnaire can be specified as a follow up item of a question querying the gender of the user, and is presented only if the user answers this question with *female*.

Structuring the Core Input Module Especially regarding larger KBs, it can be highly valuable to group interview objects topically and/or with respect to the most appropriate workflow. We refer to such groups as *questionnaires*, where each such questionnaire can contain one or many questions.

KBS Results—Finding, Summary, Solution, Case, Justification Once the user has answered a question, we define the resulting question/answer pair as *finding*. Findings commonly occur within the formalized knowledge. For example, heuristic rules or set covering knowledge may define one or several findings as condition for performing a certain action such as rating a solution.

Regarding the results presentation, a KBS should at minimum provide some form of *input data summary*. This allows users to review the entered data as to whether any adaptions of the input is needed (in case this is supported by the system). A simple example is a listing of all entered findings. In many cases, however, the KBS objective is the derivation and rating of defined *solutions*. In a generalized manner, we define a *case* as the solutions (if any) of a certain KBS session with a particular KB along with the corresponding selected findings.

Oftentimes users are not only interested in the results themselves, but also in an explanation of their validity. There, *justifications* come into play. Justifications straighten out *how* the KBS derived its conclusions. They are generated by the explanation module, e.g., in the form of finding- or rule listings—see also Section 5.3. Thus, they support not only a mere understanding of the results of the KBS, but moreover foster the trust in the system and its (correct) operation.

Framing KBS Features—Session-/User-/Language Management Apart from those core KBS UI concepts, further non-mandatory framing functionality exists that is often valuable for actually implementing KBS for productive use.

The first example is *session management*. This allows users to save, resume, and reload KBS sessions—i.e., corresponding data and results. Being able to flexibly use those features—just as required by the respective user—adheres to the principles of *matching system and real world (2)* and *user control (3)* as described by [Nielsen, 1994]. Another opportunity in this regard is, to apply further reasoning techniques, such as *Case Based Reasoning*, that require a collection of stored cases.

Whenever KBS are used not only by dedicated, single experts anymore, but potentially also by larger, diverse user groups, general *user management* mechanisms are profitable. On the one hand for ensuring a certain extent of data security, i.e., users are only able to view and modify their own, entered data. On the other hand, this also allows for offering tailored KBS views for distinct users—e.g., regarding the general UI type, the comprehensiveness of justifications, etc.

Another highly effective feature is a *multilingualism framework* that allows users, to switch between supported languages on the fly. This naturally fosters the correct usage, as users are more likely to understand questions and answer options correctly in their mother tongue. This supports the principles of *matching system and real world (2)*, *user control (3)*, and *flexibility/efficiency (7)*, see [Nielsen, 1994], thus also fostering an overall more positive user experience.

2.1.3 Common KBS Use Cases

Regarding KBS usage, we differentiate between the following basic use cases: *Consultation*, *documentation*, *debugging*, *embedded usage* and *information supply*. The use cases documentation, embedded usage, and information supply adhere to the similarly named *global system metaphors* proposed in [Baumeister, 2004, p. 30 ff.]. Regarding the consultation use case, we extend that former definition by distinguishing a standard consultation and a clarification use case. The latter is the foundation for a novel KBS type introduced in Section 2.2. We further add the use case debugging to the collection; debugging basically requires tailored variants of consultation or documentation KBS for supporting interactive and visual KB assessment.

- ***Consultation:*** Basically, we assent to the definition of consultation KBS as provided in [Baumeister, 2004, p. 30 ff.]: A consultation KBS more or less guides through a questioning sequence. Based on the user-entered findings the KBS presents its results. We differentiate two subclasses that differ regarding the target solution characteristics and the basic deduction procedure: Standard consultation and clarification consultation.
  — *Standard Consultation:* The KBS starts with the assumption that all solutions can be derived likewise during the KBS session—no predefined target solution(s) exist. As result, none, one, or several solution(s) are rated by the system. This conforms to

the known reasoning paradigm *forward chaining* as defined for rule-based systems, e.g. [Russel and Norvig, 2010, Ch. 6]. Examples are medical consultation or fault diagnosis; or all other contexts, where various different solutions may be derived or rated simultaneously.

— *Clarification Consultation:* Always only one single solution is the investigation target. That is, the KBS presents only those questions that in some way contribute to the selected clarification target. This corresponds to the reasoning paradigm *backward chaining*, e.g., e.g. [Russel and Norvig, 2010, Ch. 6] for rule-based systems. Clarification consultation is valuable in contexts where an already suspected solution needs to be investigated in-depth regarding all potentially contributing items. Thus, this use case suits well the testing part of any *hypothesize and test* setting.

- ***Documentation:*** Assenting to the definition in [Baumeister, 2004, p. 30 ff.], the major objective of documentation KBS is the support of high quality data acquisition. Contrasting to that former definition, we suggest that such KBS not necessarily require a comprehensively refined dialog control in the sense of strictly guiding the questioning sequence. Rather, particularly also free and explorative data entry are valuable. Often, users desire to enter their data in the order that fits their current context—e.g., availability of data, working environment, schedule, etc.—best. An example is the medical context, where e.g., operation data are to be collected in a structured manner. There, the respective data may not be available all at once; or it is not possible to enter it immediately as firstly the operation itself needs to be finished. Documentation and consultation KBS mainly differ in the type and extent of results presentation. Thus, they can often easily be adapted mutually, i.e., for the respective other use case, by simply adapting the respective results module(s).
- ***Debugging:*** Another interesting use case for KBS—so far not reported in the literature—is to use them explicitly for *debugging* the KB. That is, to examine the correctness and completeness of the KB contents, as well as to validate the required derivation, indication, or abstraction knowledge—by using specifically configured KBS UI solutions. Both consultation- and documentation KBS can be easily extended to enable such a debugging mode: By simply displaying all interview objects simultaneously, independent from their particular indication state; yet, basically inactive objects, such as follow-up questions, should be highlighted in a clearly distinctive manner and should switch to an activated display style if, and only if, the corresponding question trigger was selected. Based on such a representation, both the completeness of the knowledge as well as the correctness of indication structures can be easily investigated.

Of course, there already exist tools and methodologies for assessing a KB both manually and visually, c.f., [Baumeister and Freiberg, 2010]. However, investigating the 'raw' formalized knowledge, as well as applying external tools that, e.g., support also KB visualization, can be tedious and may require additional efforts or (licensing) money. We do not demand to ban such tools from a knowledge engineer's tool belt—they well provide great benefits. Yet, using a KBS implementation with only a slightly modified UI representation and a sophisticated explanation module is a direct, inexpensive, and efficient first step for KB assessment. In Section 4.3, for each described KBS UI pattern we also consider their applicability for the debugging use case.

2.1.4 KBS User Type Characterization

Typically, different user types are interested in (or overwhelmed by) different information, presentation, and interaction forms. Thus, when striving for increased user participation, delimitating user types and taking into account their diverse needs when developing KBS is an essential issue. Due to today's omnipresence of computer technology, we assume that the greater part of users can handle basic GUIs pretty well. Specifically the younger generation additionally mostly is well familiar with using mobile UIs, such as for smartphones or tablets. In general, today's key expectation towards an UI is an intuitive usage paradigm, requiring as little instructions and familiarization as possible. Categorizing users according to their UI proficiency or expectations thus is rather difficult. Yet there are two other aspects for a reasonable user classification users in the context of KBS: *Usage frequency*, and *domain expertise*.

a. Usage Frequency

Describes the average frequency with which users use the KBS. We differentiate *one-time* users and *frequent users*.

- *One-time users* are most likely to use the KBS only once and then never again. This is the case, e.g., with specifically tailored web recommendation systems for a narrowed context. Such users typically require and expect an overall highly intuitive, usable UI and interaction design, as well as prominently available start-up information or clear usage affordances when starting the KBS session. On the other hand, one-time users are most likely not to accept any required training or familiarization before being able to profitably use the system.
- *Frequent users* use the KBS regularly and thus can familiarize with the KBS' design and interaction specifics successively. Thus, they are more likely to accept also more complex, comprehensive UI/interaction forms that require some familiarization or training. Therefore, however, also pointed instructions, and potentially also initial instructional trainings/tutorials, are essential. Examples for frequent users are staff in the service support domain that use a KBS for additional support on a regular (e.g., daily) basis.

b. Domain Expertise

According to their domain expertise, we distinguish laymen, experienced, and expert users.

- *Laymen* possess only little knowledge regarding the target domain. There, the major requirement is to integrate comprehensive auxiliary information regarding the interview objects. Examples are the clarification of special terms used in the question and answer texts. Such information optimally should be provided in a directly integrated or easily retrievable manner. In turn, laymen are more likely to accept longer KBS sessions as tradeoff for more extensive support. Also, laymen characteristically have only little understanding and/or interest regarding comprehensive (or formal) justifications of the results. Rather, they are content with a compact yet understandable subsumption. Such users basically are qualified for participating in the evaluation of a KBS, e.g., in the course of user studies. Yet, they are not able to actively participate in KA-related tasks, such as KB validation.

- *Experts* possess encompassing knowledge regarding the target domain. Examples are highly trained assistant medical doctors. They require efficient, more compact KBS solutions as they are typically not willing to accept longer KBS sessions. Also, experts do not primarily need general solution derivation systems as they most likely are able to suspect appropriate diagnoses themselves. Rather, they profit from verifying such suspicions by the KBS (second opinion system). Thus, Clarification KBS modules are especially suitable in this context. Those may particularly also contain comprehensive and specialized add-on information which most likely would overwhelm laymen users. Also, the KBS may provide more encompassing justifications, as experts also are likely to scrutinize the KBS results way more critically than laymen. Apart from consultation and providing a second opinion, another relevant use case in this context is the semi-automated documentation of (often critical) data. Domain experts can valuably contribute to formalizing and validating the KB, as well as to assessing the KBS (UI) during user studies.
- *Experienced Users* settle in-between laymen and experts. They possess some knowledge regarding the target domain, yet not as encompassing as experts. Thus, they may well profit from quite comprehensive add-on information, yet a direct integration, beneficial for laymen, is no key requirement here. Also, experienced users may value the benefits of Clarification KBS in case they desire a second opinion regarding a suspected solution—either derived by a standard consultation module, or suspected by themselves. Yet, variants that use more intuitive UI/interaction forms, such as the hybrid patterns proposed in Section 4.3.5, 67 ff., may be more appropriate than specialized, expert variants.

2.1.5 Towards Participative KBS

As motivated already in the beginning of this thesis, we particularly stress the importance—and benefits—of striving for more *participative* KBS solutions. In the following, we discuss some key aspects of *participation*, as well as the potential implications on the development process.

Defining Participation in the KBS Context In general, a certain shift from 'perfect expert software' towards more user-oriented software can be observed, e.g., pursued in [Kajiyama and Satoh, 2014]. Thereby, increased user participation fore-mostly denominates a mutual supplementation and enrichment between the user and the software according to the user's knowledge and capabilities. This typically implies to offer diverse ways of operating the system: Elaborate, yet more intuitive interaction, versus tailored (expert) shortcuts. One example concerns the refinement level of questions. Users with the corresponding (domain) knowledge may be able to shorten the interrogation by answering only more abstract, summarizing questions. Laymen, in contrast, may require more pointed, precise questions. There, KBS solutions that integrate several such questioning levels—and enable users to switch in-between at will—allow both user types to profit from the system. Another example is the provision of both a consultation facility for narrowing down potential solutions and a simple solution listing for direct selection. In that case, expert users, that already suspect certain solutions, can directly choose the desired solution for further in-depth investigation, e.g., by using a clarification module, see also Section 2.2. Non-experts, on the other hand, may first use the consultation module for receiving support in selecting appropriate solution candidates before then also switching to the clarification module.

More specifically, we define participative KBS to emphasize the following, non-functional requirements: *Transparency*, *configurability*, *quality*, and *evolvability*.

- *Transparency* refers to the ability of communicating all aspects to the user that are required for intuitively using the KBS and for understanding its results (and their derivation). This encompasses: (1) The clear indication of the system objective, of the required core interaction, and of functional abilities of the KBS. (2) Understandable system feedback as an immediate response to the provided user input, e.g. regarding the appropriate presentation/highlighting of questions and answers. And (3) also the explanatory capabilities, regarding both the explanation of specialist terms, the provision of further, required media, and the justification of the results.
- *Configurability* basically addresses two aspects in the context of participative KBS: On the one hand, a general user adaptability regarding diverse usage and user contexts. This encompasses, e.g., offering multilingualism features, or letting users fine-tune the granularity of solution justifications according to their needs and proficiency. Also, providing facilities for switching on-the-fly between a comprehensive debugging- and a standard usage view falls into that category. On the other hand, a characteristically high degree of exploration/user control. This refers to the extend and type of different ways of operation—induced, e.g., by expert shortcuts, as described above. Thus, configurability basically complies to the principle of *flexibility and efficiency of use* [Nielsen, 1993b]. Or the claim of Lidwell to adapt the system capabilities to the user's level of expertise [Lidwell et al., 2010, p. 64]. Furthermore, also the usability requirement *User Control* (e.g., [Nielsen, 1993b, p. 115ff.]), which demands that the user should be enabled to master the system and not be surprised by awkward (re)actions, is addressed implicitly in case a high degree of configurability is offered.
- An increased user participation naturally further demands for a high *quality* regarding the respectively provided contents and the functionality. This concerns the core input, i.e., the quality of the presented question/answer contents and of the core interaction. But it equally concerns the results and justification, i.e., the quality of the solution derivation mechanisms and of the explanation and justification of the derived solutions. There, also the mutual influence of KB and UI on each other comes into play: An ill-defined KB—e.g., that lacks the required explanations for special terms, or where the questioning sequence is illogical—can have devastating effects and render a KBS nearly unusable. Or at least cause a good amount of user frustration. Thus, even a sophisticated, thoroughly evaluated and refined UI may be rendered useless. The same is true the other way around: In case the KBS exhibits a counterintuitive, cumbersome presentation or core interaction, the best formalized KB may be useless as the KBS itself cannot be operated well.
- Participative KBS finally also need to be *evolvable* in the sense that they can be easily adapted according to maturing user expertise and resulting needs over time. This concerns the KB and the UI likewise. There, a basic separation of front-end and KB development, and the possibility to merge both components into functional KBS anytime easily, is highly beneficial. This is the motivation for introducing a corresponding separate-and-intertwine development paradigm (Section 3) and tools (see Section 6) in this work.

Participation Induces User-centered Processes The concrete realization and appropriateness of those non-functional requirements strongly depends on the target users. Thus *user-centered* development becomes an important aspect to consider for participative KBS. On the one hand, this concerns the KB itself: Directly incorporating domain experts into KA enables the KBS developer to concentrate more on UI/usability related development tasks overall. And it further offers the chance for increasing the overall KB quality: Experts, once familiar with the KA process and tool, presumably can avoid diverse conceptual KB design errors that base on the mostly lacking domain knowledge of knowledge engineers. On the other hand, this also concerns front-end development and evaluation tasks—e.g., letting users participate more actively in requirements engineering or certain UI adaption processes.

A tight user incorporation further can lead to users feeling more connected, and thus positive, towards 'their' final KBS. In case users have been heard, have actively participated, and ideally frequently been able to evaluate (intermediate) solutions and provide feedback, chances are high that user requirements actually *are* met better. Especially highly evaluation-driven development helps to eliminate potential, yet previously not anticipated problems with the software. A positive attitude and a feeling of solidarity with the KBS, as well as the actual fulfillment of subjective (contextual, personal) requirements, can drastically increase the final acceptance of the system—as already also suggested by [Nurminen et al., 2003].

For increasing users' motivating regarding a more active participation, it is essential to provide appropriate software solutions. Regarding knowledge formalization, we propose a standard office based KA environment (Section 6.2.3) that enables users to formalize knowledge in a well-known software context. This tool is seamlessly coupled to the tailored KBS engineering tool ProKEt (Section 6). The latter also allows for eased, effortless experimentation (in defined scopes), thus fostering the integration of users also in the UI related design process.

Browser-based Solutions for Participative KBS Regarding the realization of (participative) KBS, we are convinced of the benefits of web-/browser-based applications. As confirmed by the encompassing literature review on recent KBS development practices, see Appendix B, the acceptance and distribution of browser-based KBS solutions generally increases. This assumedly is due to the growing maturity of corresponding technologies today. In the often specialist application contexts of KBS, the diverse (expert) target users typically exhibit equally diverse usage contexts, e.g., regarding the basic operation system. There, it is beneficial when KBS solutions can be implemented, maintained, and distributed platform-independently in a straightforward manner. This mostly is the case for remotely installed KBS that are accessed via the browser. Further, KBS artifacts commonly base fore-mostly on user-entered data, independent from whether that data is required for consultation or documentation tasks. Therefore, already several established UI/interaction forms for browser-based software exist, e.g., standard web forms. Those basically tend to be well-accepted by diverse user groups due to their general familiarity. This was confirmed by many domain expert users in several actual KBS projects in the past years.

2.2 Clarification KBS—A Novel Paradigm

The consultation and justification components of KBS are commonly still considered rather independently. This concerns both their responsibilities and their practical implementation and presentation. We introduce a differing point of view: We propose *Clarification KBS* as a novel mashup type of the KBS consultation and justification module. This constitutes an interesting alternative to standard KBS in various contexts. In the following, we first shortly discuss the basic idea of alternative KBS progression types. Afterwards, we specify the *Clarification KBS type* in more detail and we introduce its two general usage paradigms.

KBS Progression—Basic Considerations Regarding rule-based systems, the two known reasoning paradigms *forward chaining* and *backward chaining* exist, c.f., [Puppe, 1993, p. 36 ff.], or [Russel and Norvig, 2010, pp. 276 ff.]. Based thereupon, we differentiate two alternative KBS progression forms: *Forward consultation*, and *backward clarification*.

Forward consultation starts with an empty solution set. That is, potentially all solutions contained in the KB can be derived equally well during the following session. Starting from defined init questions, a forward consultation KBS then potentially questions in all directions. The particular questioning sequence depends on the implemented indication mechanisms the questions.

In contrast, *backward clarification* is initialized with a certain target solution that is to be clarified in detail. Such a KBS then presents only all those questions that potentially contribute to the final state of the chosen target solution.

Clarification KBS Type Clarification KBS blur the borders between consultation and justification modules by entirely uniting consultation and justification interaction and presentation in an all-in-one UI. They base on the backward clarification paradigm, as described above. Thus, always one single solution, the clarification target, is considered at a time. And all contributing questions, i.e., questions that potentially influence the target solution's state either positively or negatively, are processed.

Clarification KBS basically correspond to the testing part in the *hypothesize and test* paradigm. Hypothesizing—i.e., narrowing down the solution space to probable solutions—is performed using forward consulting KBS types. Or by any other means for selecting a solution candidate, e.g. a simple interactive solution listing for experts. Then, for each selected solution a separate corresponding clarification KBS is invoked for its in-depth, interactive inspection or for providing the user with a second opinion.

Due to its mashup characteristic, the clarification KBS type can be initialized in two different ways: *Justification-oriented*, and *Consultation-oriented*. In the *Justification-oriented* variant, the solution already is derived with some value—e.g., as outcome of a preceding forward consultation process. The clarification KBS module itself then is initialized in a filled-in manner; that is, it presents the target solution and all contributing interview items, and highlights those findings, that are responsible for the currently investigated solution state. Such a presentation can further be enhanced as to visually represent the contribution of the knowledge items to the target solution. For example, by indicating the abstract rating value, or the precise score next to the findings. This enables an additional preview regarding the consequences on the target solution even without actively exploring the presentation any further. The main objective of this initialization variant is to provide a highly interactive justification view with exploration facilities; or to retrieve a summarized, second opinion, regarding the validity of the selected solution. In contrast, the *consultation-oriented* variant is initialized in an empty manner, i.e., the target solution possess no rating yet. In the hypothesize and test scenario, this can be valuable in case an expert selects the suspected solution from a plain list and then wants to clarify the confirmation (or exclusion) of the solution candidate entirely by the help of the KBS. Thus, the major objective here is to provide consultation that is limited to the targeted solution. Therefore, the clarification KBS again presents the solution and all contributing questions—yet initially in an entirely non-highlighted manner. However, enhancing answer options by an indication of their effect on the solution for offering already a non-interactive rough preview is equally feasible in this case.

Regarding the particular UI implementations, diverse possibilities exist for clarification KBS, similar as for standard consultation KBS. We present some basic, interesting variants in Section 4.3.4, pp. 63 ff., and in Section 4.3.5, pp. 67 ff.

2.3 KBS Engineering: Status Quo

Regarding an integrated tool or methodology for encompassing, user-centered KBSE as postulated in this work, to the best of our knowledge there exists no previous work to date. We separately discuss relevant publications that cover the aspects touched by our approach: *Basic KBS engineering methodologies and tools*, *prototyping-based KBS development*, *KBS and their UI/interaction design*, *KBS (UI) patterns*, and *usability activities in KBS engineering*.

We conducted an extensive literature review covering the years 2009–2014 for gaining a general overview regarding the latest practices in KBS engineering. This particularly concerned the *application of prototyping strategies*, *KBS UI design*, and *usability activities*. Therefore, foundational journals from the domain of KBS and ES, but also general AI journals, were researched for relevant publications. Of course, manifold publications regarding KBS implementations from earlier years exist—amongst many others [Milne and Nicol, 2000, Neumann et al., 2002, Huettig et al., 2004, Rahimi et al., 2007, Song et al., 2008]. Yet, design paradigms and development tools change rapidly nowadays, due to increasingly faster technology advances. Thus, we deliberately focussed on the recent six years for gaining topical insights. More detailed information regarding the review strategy, the summarized key insights, as well as an extensive listing of the identified publications are presented in Appendix B. The key findings are also discussed in the following sections, where we refer to that literature review as *KBS Development Review*.

2.3.1 Basic KBSE Methodologies and Tools

Regarding general KBS engineering, there exist several tailored software tools and methodologies. Seminal work already suggested development shells for KBS, e.g., [Puppe, 1988, pp. 139ff.], [Kurbel, 1989, pp. 109ff.], or [Altenkrüger and Büttner, 1992, pp. 140ff]. Examples for more recent tools are the KBS development environment *Protégé* [Gennari et al., 2003]; the expert system shell *JavaDON* [Tomic et al., 2006]; the semantic wiki *KnowWE* [Baumeister et al., 2011];

a development tool for web-based expert systems intended for non-AI-experts [Ruiz-Mezcua et al., 2011]; and specifically in the sub-domain of case-based reasoning (CBR), the tools *myCBR* and *COLIBRI Studio* [Roth-Berghofer et al., 2012].

However, in contrast to *ProKEt* (see Section 6), those tools mainly concentrate on fostering the KA task—i.e., design, creation, evolution and (partly) evaluation of the KB. The development of the framing KBS application including UI design and usability activities, in contrast, mostly is not explicitly supported. An exception denote the CBR-related tools *myCBR* and *COLIBRI Studio*, which in principle follow a similar direction as the proposed research: Developing the KB/case base separately from the KBS/case based system, and finally merging the separate artifacts. Thereby, myCBR is responsible for acquiring a case base—conforming to the ProKEt KA extension points, see Section 6.2.3; and COLIBRI Studio is used for developing the case-based application itself—thus relating to the tool ProKEt introduced in this work, see Section 6. Those CBR tools, however, differ from ProKEt as they do not provide a seamless coupling of both tools for fostering an entirely intertwined and thus highly efficient KB/UI development process.

For providing appropriate development methodologies, often process models from general software engineering have been reused and adapted for KBS. A historical overview, covering stage-based, industrial-strength, and formal KBSE models from 1982–2002, is provided by [Plant and Gamble, 2003]. Both [Kurbel, 1989, p. 9ff.] and [Akerkar and Sajja, 2010, p. 58ff.] provide suggestions regarding a KBS development methodology, corresponding process models, or required activities. As a prominent example in the KBS domain, the MIKE methodology, proposed by [Angele et al., 1998], integrates semiformal and formal specification, as well as prototyping activities. Yet, in contrast to the approach in this paper, no straight progression from prototypes to the final productive system is intended. Also, MIKE is quite formalized and comprehensive, and thus may be rather inappropriate for smaller projects, e.g., with limited resources in terms of time and money. The same accounts for COMMONKADS [Schreiber et al., 2000], that also considers prototyping as well as early implementation and evaluation activities as important development tasks, but similarly is a very elaborate, modeling-focussed methodology. In 2003, [Nurminen et al., 2003] already suggested agile techniques as one important factor for successful KBS development. Exemplary, agile approaches are the one of [Knublauch, 2002] that adapts Extreme Programming techniques, or the *Agile Process Model* [Baumeister, 2004]. Those agile models aim at providing more flexible and affordable means for (iterative) KBS development as opposed to aforementioned, extensively formalized methodologies.

In contrast to the development approach proposed in this thesis, those diverse KBSE methodologies again focus on KA. Consequently, they may in cases exploit prototyping-based activities regarding KB development but this still does not much regard the design and development of the UI. Also, those approaches mostly lack the integration of any usability evaluation activities for assessing the KBS as a whole—even though they well might suggest evaluation instruments regarding the KB itself.

As another trend, it can further be observed that KBS are increasingly developed for the web since years, c.f. [Grove, 2000]—i.e., as browser-based applications. This was also confirmed by the *KBS Development Review*. Thereby, KBS may be both integrated as (smaller) modules with websites, or denote complex web applications on their own. This development shift towards the web is probably due to numerous benefits of such applications. For web-based ES, both [Grove, 2000] and [Duan et al., 2005] already discussed such opportunities—e.g., portability, availability across boundaries, online KA, straightforward user testing, or centrally controlled updating and maintenance—and challenges—such as internet speed bottleneck, or responsibility issues regarding management and maintenance. Still, according to the continual trend of moving KBS to the web, advantages seem to outweigh the downsides and challenges. Recent web-based KBS implementations are reported, e.g., by [Chen et al., 2012, Kolhe et al., 2011, Zeng et al., 2012]. Various more examples can be found in the detailed listing of the *KBS Development Review* in Appendix B and in the literature.

Already in her seminal work, [McGraw, 1992, p. 16] motivates, but does not practically provide, a methodology for KBS development, where UI and KBS development are tightly interweaved, and that further accommodates user involvement. Regarding specifically web-based ES, [Dokas, 2005] more recently proposed an approach that merges separately developed ES modules with web site or web application components; there, we agree to the author's note, that the development of those two basic KBS components can influence each other, c.f. [Dokas, 2005, pp. 6–7]. Those approaches basically resemble the intertwined development paradigm for KB and UI proposed in this thesis. In contrast, however, we further specifically postulate the ability to merge KB and UI components together *effortlessly anytime* for a straightforward investigation of their mutual effects.

With respect to tool support for developing web-based KBS, [Grove, 2000] provided an overview of available software. Examples are the Java Expert System Shell (JESS), KB Agent, ExSys, or the XPertRule KBS shell. JESS as well as KB Agent seem to address fore-mostly the creation of rule-based KBs for business users. Regarding ExSys and XPertRule, not much information about their current development-, distribution-, and support state was available. This fosters the impression, that in the 90s up to the millennium, quite a boom in KBS-related development (and corresponding software) existed. Yet, that many of such solutions focussed on the KBS back-end and/or furthermore seem not to be supported/updated any more today. This was similarly perceived by [Duan et al., 2005, p. 809], who stated that some web-based ES tools are commercially available, but no formal evaluation and comparison exists at the time of his writing. ProKEt with its KA extensions aims at closing this gap in offering support for the intertwined development of (browser-based) KBS UIs and the KB.

2.3.2 KBS Engineering and Prototyping

Regarding the design and development of interactive systems, (UI) prototyping has become an established method in various disciplines. Examples are general SE or HCI, c.f., [Bäumer et al., 1996, Beaudouin-Lafon and Mackay, 2003, Arnowitz et al., 2006, Rogers, 2011]. Main recognized advantages of prototyping are the potentially resulting efficiency and affordability. Those in turn allow for exploring the design space and generating ideas, and consequently foster early evaluation and comparison of alternatives. The flexibility of prototyping-based development further can ease dealing with changing base requirements or customer requests. Finally, prototypes provide a visual basis for discussion, thus supporting the communication between diverse people, such as designers, developers, end users, or managers. This can drastically reduce misunderstandings and thus foster a more accurate requirements specification.

In general SE, numerous common prototyping tools and methodologies have been proposed; examples are [Bäumer et al., 1996, Beaudouin-Lafon and Mackay, 2003, Floyd, 1984, Lichter et al., 1993, McCurdy et al., 2006]. Apart from that, particularly tailored approaches have proven valuable in various specific contexts and domains: For example, tailored prototyping tools for the development of cross-device general web UIs [Lin and Landay, 2008], of multimodal systems [McGee-Lennon et al., 2009], or for mobile device applications [Leichtenstern and André, 2010]. Specifically sketching originally was considered more of a low-fidelity, offline method. There, however, [Kieffer et al., 2010] proposed a tool for sketching-based multi-fidelity prototyping; their tool allows for transforming roughly drafted UIs automatically into computer-based representations. Yet, in contrast to evolutionary prototyping in general, and the approach in this paper particularly, no fully productive systems are produced by those approaches.

Nevertheless, those works confirm the general assumption, that prototyping tailored to a specific application context can be a most valuable instrument for fostering the development and success of the respective project and system. Also regarding KBSE, prototyping is successfully applied for creating proof-of-concept implementations of the KBS vision, as reported, e.g., in [Saadé et al., 2004, Sutton and Patkar, 2009, Ting et al., 2010, Afacan and Demirkan, 2011]. However, regarding the use of prototyping for experimenting efficiently with design alternatives and for eventually evolving from initial prototypes to productive systems, still little is published in the context of KBS.

In the early 1990s, [Waterman, 1993] described different stages of prototypes during KBS development. Also, based on his survey of ES developers, [O'Leary Daniel E. , 1991] suggested various advantages of prototyping KBS: Support for requirements engineering (RE), potentially more robust systems regarding the quality of the KB, and requiring less efforts than traditional SE approaches. However, similar to most existing general KBSE methodologies and tools, those seminal works also focus on the KB and not on the entire KBS architecture or specifically its UI. A more recent example from the KBS domain is the development of a mission critical expert system [Bloom and Chung, 2001] by the means of rapid prototyping; this intended to support a multiple steps approach for RE and UI design. There, however, prototypes served as requirements documentation exclusively, thus not being evolved into a productive system at any point in time. In another project, evolutionary prototyping was applied for developing the intelligent fish disease/health management system FIDSS [Xiaoshuan et al., 2009]. Yet again, those authors focussed on the evolution of the KB rather than on designing the UI: Once general UI requirements were decided on during the second development iteration, the UI mostly remained unchanged. Thus the impression arises, that prototyping basically is well-accepted in general SE as well as regarding the KB development. Yet, concerning KBS UIs, prototyping still seems to be somewhat neglected, regardless of the various potential benefits that we are strongly advocating in this work. This assumption was mostly supported by the *KBS Development Review* (Appendix B). There, either no prototyping activities were reported at all; or they resulted in some form of pilot systems. Yet, regarding iterative prototyping-driven development or evolutionary prototyping, no specific publications were found.

Regarding the practical construction of UI prototypes, a vast array of HTML- and/or CSS-based GUI builders exist. Examples are browser-based applications and frameworks, such as *jetstrap* [Jetstrap (Drifty Co.), n.d.]; particular prototyping or wireframing desktop applications, e.g, *Balsamiq Mockups* [Balsamiq Studios, n.d.]; or even more general desktop graphic design software that provide ready-to-use HTML mockup widgets or allow their definition, such as OmniGraffle for Mac OS X [OmniGraffle The Omni Group, n.d.]. However, there is one key difference between KBS and standard questioning software (both paper-based and on the web): The concept of flexibly included follow up questions. Knowledge-based questioning sessions characteristically are not a statically prescribed questioning sequence. Rather, potentially many interview items—such as follow-up or abstraction questions—are included dynamically only if they are useful in the given context. Consequently, when requiring a realistic impression of the target system, KBS UIs cannot be statically prototyped (optimally and entirely) with standard GUI builders such as above. This specifically regards highly complex KBs with comprehensive interview item interdependencies. There, tailored tools become necessary that allow for realistically mimicking also the characteristic, flexible KBS concepts—such as our proposed prototyping and KBSE tool ProKEt.

2.3.3 KBS Engineering, UI design, and Patterns

According to C. Alexander, patterns generally describe a recurring problem and suggest a reusable solution, c.f., [Alexander et al., 1978, p. X]. Since then, patterns have been transferred successfully from their original domain of architecture to various other disciplines, including computer science. There, apart from the probably most renowned software design patterns of Gamma et al. [Gamma et al., 1994], to date patterns have gained strong presence also in UI-, web-, and interaction design. Examples of the manifold publications include [Borchers, 2001, Graham, 2003, Tidwell, 2005], and many more collections are (partly freely) available on the web, e.g., [Laasko, n.d., Erickson, n.d., van Welie, n.d.].

Regarding the UI design of ES, [Hendler, 1988] already published some first research efforts. Hendler's work, however, mainly yielded more general suggestions regarding appropriate KBS interfaces and their development; as an example *the system must be able to explain its behavior*. Those are in some way comparable to general usability heuristics, e.g. as postulated by [Nielsen, 1994], thus no concrete UI design suggestions or even patterns are provided. Also back in 1992, McGraw pointed out the tendency that "[...]the user interface remains the least resarched and developed", [McGraw, 1992, p. 3]. Since then, some singular research efforts have been reported. For example in [Ruckert and Klein, 1996], who report optimizing a KBS UI, yet thereby focussing on the presentation of the explanatory component of the system. Or in [Puppe et al., 2000], who describe some basic UI variants that partly consent with the KBS UI patterns and basic UI styles in this work. Yet in general, the situation did not notably change since then.

The *KBS Development Review* supports that same impression: That today in most cases KBS UIs are still developed in a rather ad-hoc manner; i.e., not following any specific, systematic method, or being explicitly considered and integrated with the applied KBS/KB development process. Further, also no patterns or best practices seem to be (re)used or provided. This might be due to a general lack of scientific research and corresponding little publications regarding web-based ES and their development—which was already noted, e.g. by [Dokas, 2005, p. 2], [Duan et al., 2005, p. 800/810]. In general, the potential value of pattern application—e.g., enabling the reuse of proven solutions and avoiding common mistakes—is acknowledged in various domains. Regarding information retrieval systems and their interaction design, for example, respective efforts are reported in [Schmettow, 2006]. Also, UI related investigations are reported specifically for web-based case-based reasoning systems in [He et al., 2009], yet there, with the focus on presenting UI options for the context of case retrieval. The specification of suitable patterns for more general consultation or documentation KBS UIs, however, seems to have not been thoroughly investigated (or published) so far.

The *KBS Development Review* further suggests, that basically three major categories of UI styles for KBS are applied today: (1) *Form-based*, mapping to the Questionnaire pattern, c.f., Section 4.3.2; (2) *Interview-based*, mapping to the Interview pattern, see Section 4.3.3; and (3) *Visual/Interactive* UI styles.

Form-based UIs thereby denote more or less complex data entry forms, displaying several questions at once to the user before the next page is presented. This corresponds to the *multiple question dialog [Mehrfragendialog]* variant, described in [Puppe et al., 2000]. The *KBS Development Review* basically suggests, that currently most KBS UIs are implemented in a form-based manner. Thereby, sometimes the presented data entry forms are deliberately designed as to resemble existing, originally used paper-based forms. This is intended to enhance the ease of use of the system and for lowering the experts' mental barrier to switch from paper-based to computer-based data entry—and thus, to foster the active usage of the KBS. Such examples mainly were found in medical contexts, where doctors are used to certain paper-based medical records due to their daily practice, see, e.g., [Ting et al., 2010].

Interview-based UIs, contrastingly, seem to be implemented less often. Thereby we mean UIs that present always only one single question to the user before switching to the next question. Optionally, this is augmented by auxiliary information such as additional explanations. Thus, such systems mimic a one-on-one interview, that is conducted between the system and the user. This corresponds to the basic variant *one question dialog [Einfragedialog]* proposed in [Puppe et al., 2000]. A recent implementation is reported, e.g., in [Sarma et al., 2010].

Visual/Interactive UI styles or metaphors seem to be implemented only in the case of very specialized applications. The interactively enhanced *folding dialog [Klappdialog]* introduced in [Puppe et al., 2000] basically falls into this category. Recent examples, found during the *KBS Development Review*, include:

A *UI mashup using SketchUp as basic CAD* package that is enhanced by a plugin that enables the representation and processing of knowledge via form-based message- and dialog boxes; this implementation further includes a semantic wiki for leveraging collaborative design tasks with the tool, see [Afacan and Demirkan, 2011]. A *CAD-based mold base configuration/design ES*, proposed by [Huang et al., 2009]; there, users are enabled to configure mold bases via entering relevant framing data using several web forms, to preview them by an integrated CAD-like presentation mechanism, and to order the resulting mold base directly via the application. And a financial prediction model system proposed by [Cho, 2010]; this system basically is implemented very similar to the existing SPSS module *Clementine* (meanwhile named *IBM SPSS MODELER*). That is, it offers an *own, graphical model builder* that allows for assembling the desired models entirely UI based: By dragging and dropping certain widgets, thus not requiring any programming or formal specification.

2.3.4 KBS Engineering and Usability/Evaluation

When it comes to usability, user experience, and evaluation-related activities in the context of KBS, respective efforts mostly only concern the development and assessment of the KB. One example, that reports some efforts of enhancing the overall user experience of case-based reasoning systems is [He, 2013]. However, the explicit integration of usability-related considerations and evaluation activities especially with regards to the KBS UI in the KBS development cycle has not much been researched—or published—yet.

The *KBS Development Review* also confirms that assumption. Only few publications report the application of usability-related activities at all. Thereof, in two cases more implicit activities are reported: The development according to a thoroughly researched requirements specification—which in turn was based on interviewing the users before development start [Li et al., 2009]; and the implementation of the UI according to a metaphor chosen from the actual work-life of the users [Zeng et al., 2012]. Some other works—e.g. [Hasan and Isaac, 2011, Devraj and Jain, 2011]—report user tests or questionnaire-based interviews that took place after a (pilot) system had been put into use. Overall, this little consideration is a bit surprising, as today, subjective evaluation measures, such as user experience, perceived enjoyment, and usefulness, are gaining importance. This accounts both for general software- and website development, as well as specifically also for DSS, see [Ben-Zvi, 2012].

Quite recently, [Holzinger et al., 2011] highlighted the value of user-centered development in terms of tightly combining (various) prototyping approaches and usability evaluation activities. As examples, evaluation combinations such as thinking-aloud (TA) paper mock-up evaluation and video analysis, or field studies rated with the SUS scale [Brooke, 1996] are described. Named research does not specifically address the KBS domain. Yet the target system—a form of medical questionary—is basically similar to the Box Questionnaire KBS UI style as described in Section 4.3.2.a.. Also, [Holzinger and Slany, 2006] reported positive experiences with their *Extreme Usability (XU)* approach that tightly integrates Extreme Programming and Usability Engineering. This basically resembles the approach proposed in this work, yet again it is not specifically targeted to the KBS domain. Further, [Leichtenstern and André, 2010, p. 94] termed *user-centered prototyping tool* as an all-in-one tool solution that enables developers to efficiently, effectively, and satisfactorily design, evaluate, and analyze developed artifacts—that according to their understanding should be based on evolutionary prototypes. In that sense, the tool ProKEt proposed in Chapter 6 accounts as user-centered KBS prototyping tool.

2.4 Synopsis

In this chapter, we have discussed the state of the art regarding KBSE: We have provided a delimitation of established KBS-related concepts and classifications. Based on that, we have motivated the benefits and provided a definition of *participative KBS*. And we proposed the innovative theoretical conception of *Clarification KBS* as justification-consultation mashup KBS type. Finally, we gave an overview concerning both seminal works and current research efforts regarding the research topics touched by the proposed approach: Basic KBS methodologies and tools, prototyping-based, UI-centered, pattern-based, and usability-aware development.

The development paradigm and practical tools, introduced in the rest of this thesis, basically found on the definitions and concepts provided in this chapter. Thereby, we specifically target participative, browser-based KBS solutions. In the further course of this work, for reasons of briefness those systems are simply referred to as *knowledge-based system* (KBS). In the next chapter, we introduce key activities, concerning mainly UI-/interaction design and usability, that are intended to enhance existing KBSE approaches. We demonstrate their flexible applicability by extending an agile development model for KBS.

Chapter 3

Encompassing, User-centered KBSE

As discussed previously, knowledge-based systems engineering (KBSE) mostly still lacks dedicated UI-, interaction design-, and usability activities. In this chapter, we propose an encompassing approach for leveraging this issue and for fostering a more user-centered development process. Therefore, in Section 3.1 we first summarize key activities that need to be considered more prominently when striving for encompassing KBSE. In Section 3.2 we then demonstrate their integration with an existing agile KBSE approach.

3.1 Key Activities for Encompassing KBSE

Regarding encompassing KBSE, we postulate to add the following *key components* to back-end focussed KBSE approaches:

- *Extensible Prototyping*: A tailored form of evolutionary prototyping; EP fosters an UI- and user-centered development process and the seamless integration of KA activities. In some cases *lo-fi prototyping* can denote a valuable add-on.
- *KBS UI Patterns*: General specifications of proven KBS UI solutions that particularly foster reuse, a more precise RE, and serve as KBS showcase and decision aid.
- *Appropriate Usability Instruments*: Both implicit and explicit means for integrating usability-related activities seamlessly and effortlessly with KBSE processes.

In the following, we discuss those components in more detail. For living up to their full potential, we recommend their iterative application. As already diverse well-tried, mostly KA-centered KBSE development approaches exist, we do not describe yet another own, self-contained methodology. Rather, we demonstrate the integration of the proposed activities with a selected, agile approach in Section 3.2.

3.1.1 Extensible Prototyping

As also published previously, see [Freiberg et al., 2012, Freiberg and Puppe, 2012b], we define extensible prototyping as a *tailored form of evolutionary online prototyping*. Online prototyping basically encompasses prototyping techniques that use software for creating the respective, typically rather mature artifacts. They contrast offline prototyping techniques, which do not make use of any software tools and usually result in throw-away prototypes, c.f. [Beaudouin-Lafon and Mackay, 2003, p. 1014].

Throwaway Prototyping & Evolutionary Prototyping *Throwaway Prototyping*, which is also termed *Rapid Prototyping* sometimes, creates prototypes for a specific purpose and discards them afterwards, c.f. [Beaudouin-Lafon and Mackay, 2003, p. 1014]. Thus, such prototypes are mostly deliberately created in a quick and dirty manner. Characteristic techniques are sketching or paper mock-ups. The resulting artifacts allow for evaluating more design alternatives, to explore ideas more freely, and iterating designs more often. Often, also only specific aspects of the target system are considered, which allows to gain more precise insights regarding the separate aspects. Main shortcoming of this approach is the often lost large amount of development time and efforts as the implementation of the final productive system is always started from scratch. Moreover, customers/users might be biased by the prototype solution and insist on integrating exactly that prototype with the final system—even if that was neither intended nor feasible.

Evolutionary Prototyping, in contrast, denotes a special form of *Iterative Prototyping* according to, e.g., [Beaudouin-Lafon and Mackay, 2003, p. 1009]. There, the developed prototype artifacts are continuously evolved into the final system. An advantage of this approach is the possibility to deliver (intermediate) working releases of the target system more quickly and thus more frequently. Consequently, the artifact can be assessed by actual users in the target context earlier. This again can lead to an increased user engagement and identification with the system. Also, an artifact, that is evaluated under realistic conditions by actual users, is more likely to meet the requirements and satisfy the users. Finally, the availability of prototype artifacts in general fosters the clear communication between developers and customers/users by providing a universal, visual language. A major downside of the evolutionary approach is the impracticability or even danger to evolve such prototypes—rarely completed, robust systems—into final productive systems, c.f. [Beaudouin-Lafon and Mackay, 2003, p. 1026]. Users may also have difficulties to criticize them in a reasonable manner as such artifacts are rather tangible, mature, and sometimes seemingly finished representations of (parts of) the final system. Further problems may arise as evolutionary prototypes and corresponding projects often base on rather vague specifications: The uncertainty regarding a schedule and/or the final artifact might foster the impression of wasted time and money. Also, it might be difficult to finally stop refining the artifact 'yet a bit more'. Such potentially ever-changing requirements can involve the danger that the project turns into an endless (and in the worst case a failure) story (c.f. [Xiaoshuan et al., 2009]).

Extensible Prototyping Based on those considerations, we suggest *extensible prototyping*, a tailored form of online, evolutionary prototyping, as one of the key activities for encompassing KBSE. The main characteristic of an extensible prototype is its ability to be transformed into a productive KBS anytime without efforts. Therewith it contrasts mere evolutionary prototyping, where the process until reaching the final productive system state potentially can be lengthy. As motivated in Sections 2.1.1 and 2.1.2, KBS basically consist of a front-end (UI) and a back-end (KB), as well as of further framing functionality.

Based on treating those three key modules as separate units, we propose the *extensible prototyping strategy*, basing on three steps (EP 1–EP 3), as follows:

- *EP 1: KBS Front-end Prototype (mature and interactive)*: Prototyping itself targets mainly the KBS front-end. Thus in this first step, the desired module(s) of the KBS UI, e.g., core input module, are created. KB contents and KBS core interactions are mimicked by a reduced knowledge specification and a framework supporting the required interactions. The latter may partly be substituted by exemplary actions, as, e.g., actually setting a KB value is not feasible without the inclusion of a functional KB.

- *EP 2: KBS Core Prototype*: A functional KBS core prototype, i.e., without common framing functionality, is created: By merging the front-end prototype with a functional KB. Mimicked functionality from EP 1 is replaced by the actual mechanisms for handling the productive KB.
- *EP 3: Fully Functional KBS Solution*: The core prototype is turned into a fully functional productive KBS by additionally coupling it with common KBS framing functionality.

For supporting this approach practically, a tailored development tool becomes required. This naturally needs to support KBS UI prototyping in the first place. Also, mechanisms for easily adapting the artifacts, and offering a collection of (more and less) comprehensive UI widgets thereby is required. Also, such a tool needs to enable the seamless integration of the created prototypes and KB artifacts. Finally, also at least one default realization of each framing feature is required to ensure that the created KBS prototypes can finally be extended and deployed directly (or only with minimal additional efforts) for actual use. In Section 6, we introduce the tailored KBSE tool *ProKEt*, that fulfills those requirements. ProKEt further uses one single, specification file based mechanism for configuring both the KB to be used as well as the basic framing functionality; thus, ProKEt allows for performing the two extension steps simultaneously, if desired.

Encompassing KBSE with Extensible Prototyping Extensible prototyping firstly helps shifting the focus towards UI- and interaction design by stressing the prototyping / front-end development activities. Also, certain characteristic KBS features, e.g., value abstraction or indication mechanisms, are best understood by future users (i.e., typically non-KBS experts) if actually used. Thus, the anytime availability of interactive KBS artifacts fosters active user participation particularly already at the early development stages—e.g., in the form of joint RE sessions, live-adaption of the prototype with the customers, or early evaluation activities. Further, often prevailing doubts—whether the efforts of developing a KBS are worth regarding its future benefits—can be more easily dismantled by letting customers review a realistic demo system.

Integrated Knowledge Acquisition We regard a tight interconnection of KA and UI development activities a key aspect for encompassing KBSE, due to the already motivated mutual impact of KBS UI and KB on each other. Extensible prototyping basically postulates to merge KB and UI prototype at any desired point in time. This enables the simultaneous development and optimization of both parts, and the anytime investigation of their mutual influences. Also, this strongly supports active user participation in KA. There, domain experts are enabled to formalize knowledge mostly autonomously—even more strengthened by providing tailored KA tools—and to review the outcome directly in a functional KBS artifact. User participation regarding the KA task in turn bears the advantage, that this offers the KBS developer more time for refining or evaluating the KBS front-end or necessary framing functionality.

Possible Add-on: Low-fidelity Prototyping Regarding more comprehensive or also more experimental projects, the available patterns and widgets provided by an extensible prototyping tool may not always fit optimally. There, the additional usage of low-fidelity (lo-fi) prototyping techniques can provide valuable benefits to the proposed approach. Those can foster more creativity-oriented thinking by allowing to sketch possible solutions more quickly than based on a prototyping tool. This may require to use a certain programming- or specification language and its corresponding restrictions. This also applies in cases, where an overall KBS metaphor fits the project requirements generally well, but some certain widgets require adaption or extension for the particular context. An example are standard drop-down widgets that may require to handle multiple choice input for a certain project. Thus, even though lo-fi prototypes alone do not suffice in the context of highly specialized, interactive KBS, they still can be a valuable add-on for specific situations.

Lo-fi prototyping [Rettig, 1994, p. 22] is generally related to the notion of offline prototyping [Beaudouin-Lafon and Mackay, 2003, pp. 1014 ff.]. It encompasses techniques for creating paper and pencil prototypes, sketches, or transparencies that visualize (parts of) the UI in an often deliberately simplistic, rough manner. During own projects, we so far applied paper- and computer-based sketching. One example is the evolution of the ITree (see also Section 4.3.4.a.) KBS UI style. Figure 7.7, p. 120, shows a seminal computer-based mockup of the general vision (I). This mockup was recreated and refined (visually) with the tool ProKEt (II). Based on various development iterations with the project partners, the final ITree style evolved (III).

3.1.2 Core Input KBS UI Patterns

As second key ingredient for encompassing KBSE we suggest tailored KBS UI patterns. Advantages of pattern-based development, that are commonly agreed on, encompass the circumvention of common implementation flaws by using proven solutions, and the opportunity to accelerate the overall development process. Thus, apart from establishing a basic quality of the KBS solution, the application of patterns can further free development time. This in turn can be used for targeted experimentation and (usability) evaluation activities. Thus the KBS quality can be further increased by tailoring it more specifically to users' needs. Patterns thereby should be considered immediately at project start when gathering the main system requirements for envisioning a potentially suitable UI solution. Similar to prototyping, also patterns offer a means for enhancing the communication with customers and users by providing a more tangible, visual basis for discussion. Thus in turn, again also a more active user participation already at early project stages is fostered.

Basically, we recommend that such patterns not exclusively target the KBS UI itself, but also consider typical knowledge characteristics. For example, whether a distinct representation such as a heuristic decision tree (c.f., [Puppe, 2000]) is required. If such patterns further found on a set of key classification criteria for KBS, their clarity can notably increase. We introduce a set of usability-oriented KBS criteria below. Those were used as the foundation for classifying the KBS UI pattern collection later in Section 4.3.

Usability-oriented KBS Classification Criteria Our proposition for encompassing KBSE also entails the consideration of usability issues. We propose tailored, usability-oriented KBS classification criteria on the one hand as a means for delimitating KBS solutions clearly against each other in the form of KBS UI patterns. On the other hand, those criteria help to generally foster the awareness for usability-related properties of KBS solutions, and they support their formal assessment.

Diverse usability guidelines and standards are available today. Known examples are the heuristics of [Nielsen, 1994, Ch. 2], [Shneiderman and Plaisant, 2004], or the norm [ISO 9241–110, 2006]. However, those are all defined at a more general level as they are intended as unified rules applicable for diverse interactive software system types. Thus, those guidelines do not mirror relevant key characteristics of KBS precisely enough. See, e.g., our criterion of *compactness* below: Basically, this resembles to the general claim for aesthetic UI design as postulated in the 8th heuristic of [Nielsen, 1994]. Yet for KBS, we state the meaning more precisely as *the number of interview items that are presented in the KBS UI.*

In previous work, we proposed the *usability stack* [Freiberg et al., 2009] as a collection of usability criteria for KBS. Therefore, various established usability guidelines have been reviewed, partly unified, and prioritized with regards to their suitability for KBS. For defining the stack model, however, we still had kept them formulated at their more general level. Since then, our view on usability criteria for KBS was sharpened due to practical project experiences. Thus, with the stack model in mind, and further considering the requirements for participatory KBS, see Section 2.1.5, we worked out more refined, usability-oriented classification criteria for KBS:

1 ***Compactness:*** *How many interview items are typically presented simultaneously?*

For example, the Daily Questionnaire style (c.f., Section 4.3.2.b.) displays a large number of questions at a time. Yet the clearly structured presentation allows users to process those items quite easily. Thus Daily accounts as a highly compact style.

2 ***Comprehensibility:*** *Does the KBS support users in understanding specialist, complex, or ambiguous KB contents, and in learning something about the domain?*

For KBS, particularly also the understandability of the relevant KB contents is a key factor for success. Yet, especially in more proficient domains this often cannot be achieved easily or automatically, e.g., due to many specialist terms. Then, comprehensibility can be increased by adding comprehensive additional information and explanations to the interview item presentation. Apart from plain explanations, also some form of easily retrievable special terms lexicon (when hovering respective special texts) can be beneficial. Further alternatives encompass the presentation of questions in reasonable context ('neighbor questions'), or the visual representation of inherent knowledge coherences. Also, the provision of different or adaptable justification views can valorize the comprehensibility of a KBS. Consequently, an increased comprehensibility can induce a certain skill-building ability.

3 ***Descriptiveness:*** *Does the KBS provides clues how the respective findings influence the final result of the KBS session?*

It is further highly valuable to provide clues, how each finding (question & answer) influences the final result of the KBS session for fostering both comprehensibility and learnability. However, such clues are only reasonably applicable for KBS that target only a single solution, e.g., for the clarification use case—see Section 4.3.4 for some exemplary solutions for the Clarifier pattern variants.

4 ***Efficiency:*** *How long does a characteristic KBS session take, and how many interview items need to be processed?*

For KBS, efficiency denotes the average length of a session and/or the average estimated number of interview objects that need to be processed until a final result is reached. Efficiency generally can be increased by reasonable expert shortcuts, see item 5.

5 ***Explorability (Participation):*** *May users deviate from the suggested questioning sequence? Are expert shortcuts provided?*

Often, KBS do not only define interview items and solutions, but also certain more or less strictly defined questioning sequences. Explorability addresses the level of freedom a user has to deviate from such prescribed sequences. For example, whether or not it is possible to switch to entirely different questions or questionnaires before having answered the currently active items. Also, this entails the aspect, whether a KBS offers expert shortcuts for more proficient users for finishing a session more efficiently.

6 ***Intuition (usage):*** *Does the KBS offer familiar presentation or interaction forms, or is it otherwise self-descriptive?*

Due to the high relevance of correct data input, a KBS optimally is designed in an intuitively usable way—i.e., either highly familiar to users and/or highly self-descriptive. This might not always be possible; e.g., when the target user population is too diverse as to fulfill all their needs equally. In this case, a KBS should at least provide auxiliaries such as pointed instructions, inductive training (e.g., tutorials), as well as interactive affordances that hint at the required interaction(s).

7 ***Screen Adaptability:*** *Describes whether the KBS UI easily adapts to varying screen sizes and/or resolutions.*

For example, the overall perception of Hierarchical Clarifier UIs (Section 4.3.4.a.) can drastically decrease with decreasing screen sizes. Strict Interview (Section 4.3.3.a.), in contrast, in general is less vulnerable regarding screen/resolution variations.

8 ***Transparency:*** *Does a KBS mediate its current state anytime in a clearly recognizable and comprehensible manner?*

Thereby, the KBS state encompasses: The state of questions, i.e., yet to process, currently inactive, etc.; the state of answered items, i.e., the respectively provided or derived answer value; the state of results, e.g., whether there are any solutions derived/rated and with which current value; justifications regarding how the current state was reached, e.g., by providing the relevant formal background-knowledge; and finally, the current progress in the interrogation sequence, which can be visualized by appropriate progress information widgets (see also see Section 5.2.2).

9 ***Well/clear Arrangement:*** *Does the KBS fulfill some general aesthetical requirements?*

As aesthetics are basically a highly subjective matter, general recommendations are difficult to provide. Yet, actual KBS projects in the last years showed, that two basic criteria seem to influence the perception positively: A small average number of (virtual) lines, as well as a basic degree of symmetry.

3.1.3 Appropriate Usability Instruments

We propose the following set of activities and instruments for further fostering usability in the KBSE context: *Active user participation in development*, *iterative development*, *usability testing*, *querying*, and *log data evaluation*.

Usability-fostering Activities (Implicit Usability Instruments) Active user participation and iterative development *foster usability in an implicit manner*. Regarding the KBS front-end, an early user incorporation increases the chances that the basic system specification fits the user intentions and requirements optimally. An example is active user participation when defining the foundational system image and core requirements. This helps to avoid misunderstandings, frustration, increased development time and efforts later on, and to foster overall user satisfaction and usability. Increased user participation with respect to KA further minimizes the risk of creating a biased KB—e.g., due to a lack of expert knowledge of the knowledge engineer, or due to mistakable communication. There, (active) user incorporation can help to maximize the quality of the KB and thus, to enhance the overall KBS usability. We have already motivated active user participation a key aspect of encompassing KBSE and described, in what regards this is fostered by extensible prototyping.

Iterative development, on the other hand, per se offers the chance to spot and thus timely eliminate more implementation flaws than when applying a sequential development model with only one dedicated testing phase near to the end of the project. This is also suggested by renowned usability practitioners. As, e.g. [Nielsen, 1993a] puts it, "user interfaces should be designed iteratively in almost all cases because it is virtually impossible to design a user interface that has no usability problems from the start. Even the best usability experts cannot design perfect user interfaces in a single attempt, so a usability engineering lifecycle should be built around the concept of iteration". There, extensible prototyping with its claim for the easy anytime production of KBS artifacts strongly fosters frequent iterations. Further, in the following Section 3.2, we describe the integration of the key activities for encompassing KBSE with an agile, and thus also highly iterative, KBS development model. Each of those iterations particularly also should integrate explicit usability activities as described in the following.

Explicit Usability Instruments As *explicit usability instruments*, we particularly found usability testing in conjunction with gathering qualitative user feedback and log data analysis valuable. Saying usability testing, we basically encompass both informal user studies as well as more formal, targeted usability experiments. One established instrument for gaining qualitative feedback during usability testing is the *thinking-aloud (TA)* approach. There test participants are asked to use the system while continuously verbalizing their thoughts as they process the UI, c.f. [Nielsen, 1993b]. Yet for our context, we regard TA only suitable in a restricted manner: Users typically find it harder to openly express their problems or concerns regarding a seemingly finished, mature system. This renders TA rather inappropriate for evaluating the highly mature extensible prototypes. However, the application of TA can become an interesting option when integrating lo-fi prototyping initially for envisioning entirely new solutions for the UI.

In our context, target users often are rather delicate subjects; examples are medical doctors or lawyers with typically little time and thus restricted availability. There, *remote evaluations* are generally favorable; i.e., making the test artifact available on a dedicated demo server which can be accessed by potential participants via the internet. This enables users to test the system in their most convenient (or in the best case: In the targeted) context whenever they have time and will to do so. This alone can increase the general interest in participating such an evaluation for named users.

For collecting qualitative feedback, first some form of *anytime feedback* mechanism can be highly valuable; this basically denotes some mechanism that allows the user to provide feedback to a denoted person in charge at any time during system usage. Further, also the integration of *electronic surveys* is a beneficial option. Those instruments denote a generally valuable and affordable means for gathering qualitative feedback from users, yet they can become a priceless add-on regarding remote evaluations, where feedback cannot be gained easily otherwise.

Finally, also in the context of KBS it can be insightful to *analyze collected, tailored click log data*—e.g., regarding the answered questions, the sequence thereof, the attained solution, etc. Such data can help reveal and better understand both problems with the KBS UI and the KB. For example, if a certain question is almost always skipped, it might be a hint that this question is not appropriately integrated in the questioning sequence. Alternatively, it could also indicate a formulation problem with the question/answers, or a lack of additional information.

Apart from their separate benefits, we especially experienced the joint investigation of log data, utility questionnaires, or success rate analysis with qualitative (free) feedback as most valuable. This is due to the fact, that often the former instruments alone can produce ambiguous results—see the potential reasons for a skipped question, described above. Additionally provided free feedback of users often can clarify the true reason(s) for certain results, thus rendering the overall insights more expressive.

For providing not only a mere collection of KBSE enhancements, we exemplarily describe the integration of the proposed ingredients with the Agile Process Model [Baumeister, 2004] in the following.

3.2 Encompassing, Agile Process Model (EAM)

In past projects, we experienced the Agile Process Model (APM) [Baumeister, 2004] as a favorable approach regarding pragmatic KBS development. However, the APM lacks an explicit integration of UI- and usability related activities—similar to many other renowned KBSE approaches. Therefore, we mashed up the APM with our proposed key activities for UI-, and usability-oriented KBSE. The result is an encompassing, more formal, user-centered KBSE approach: The *Encompassing Agile Process Model (EAM)*.

A preliminary conception for extending the APM was already discussed in previous work, see [Freiberg et al., 2010, Freiberg et al., 2012]. The original APM specifies four phases: System Metaphor, Planning Game, Implementation, and Integration. Shortly summarized, System Metaphor constitutes the initial phase and is passed through once at project start. The remaining phases are traversed in a cyclic manner, starting with the Planning Game and ending with Integration, respectively. In Figure 3.1, the dotted boxes themselves visualize the original APM phases. The items within those boxes are the added ingredients as discussed subsequently.

In the *System Metaphor* phase, KBS developers, stakeholders, and experts specify the basic vision and aims of the KBS. [Baumeister, 2004] distinguishes a global and a local system metaphor: The global system metaphor describes the fundamental application class of the KBS,

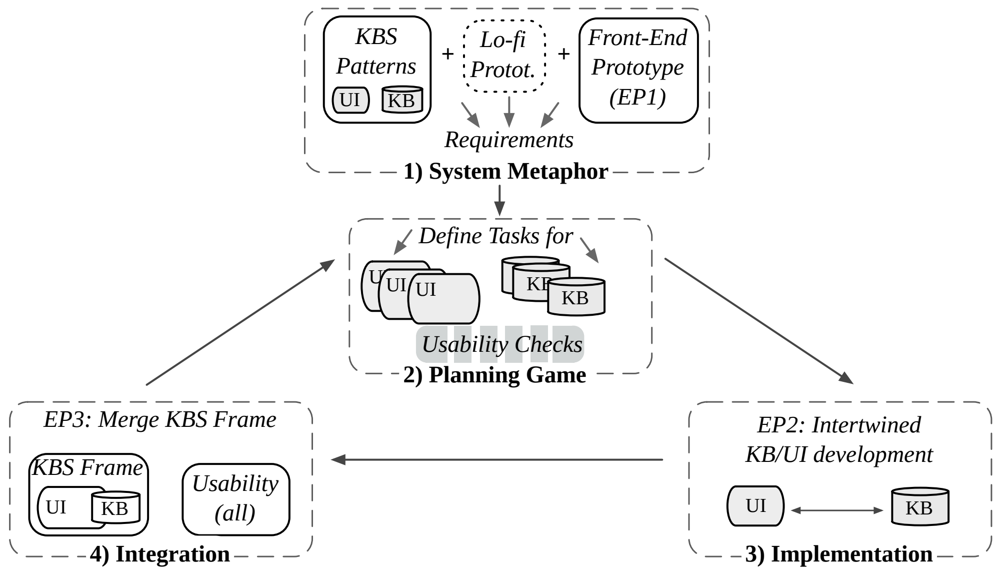


Figure 3.1: Extension of the Agile Process Model [Baumeister, 2004]. For each of the original phases (1–4, depicted as dotted grey containers) selected activities of encompassing KBSE are added.

such as *consultation system*, *documentation system*, *information center*, and *embedded system*. As KBS of the same basic class nevertheless can be equipped with diverse basic UI styles, it is crucial to define not only the application class but also a suitable UI metaphor in that phase. This can be done by viewing, comparing, and/or adapting existing patterns and front-end prototypes (extensible prototyping, step 1). Or by using lo-fi prototyping techniques for envisioning novel solutions. Interactive demo systems, if available with the respectively used toolkit, as well as sound patterns thereby foster discussions on a visual basis. This in turn minimizes misunderstandings, and particularly supports the communication about the KBS and its components. Thus, the definition and usage of the local system metaphor as defined by [Baumeister, 2004] is supported, and the early, active user participation is fostered.

Having defined the System Metaphor, the APM reaches its cyclic phase which begins with the *Planning Game*. There, scope and deadline for the respective next release are defined by exploring possible tasks, assembling them into working packages, and finally prioritizing and scheduling them with regards to implementation risks and importance/desirability. Deciding about those tasks is strongly supported by the anytime availability of interactive KBS prototypes. In the first pass, typically front-end or core KBS prototypes are used—once available, later on also core KBS prototypes or fully functional KBS artifacts are possible. Consequentially, the KBS UI/interactions can be explored and tested in a realistic manner. Based thereupon, both working packages regarding the UI (e.g., adapt the styling of comprehensive questionnaires) and the KB (e.g., include a whole new questionnaire branch in the KB and thus questioning sequence of the system) are defined. In certain cases, the scheduling and prioritizing can be further eased by conducting some usability checks—which at that stage may potentially be less formal; e.g. to what extent potential users will be able to understand, use, and like a new part of the UI. Here, instruments such as smaller-scaled user tests, interviews, or short surveys can be applied.

According to the respective outcome of the Planning Game, both the UI and the KB are adapted according to the scheduled tasks in the *Implementation* phase. The intertwined KB/UI development strategy of extensible prototyping thereby specifically enables the simultaneous modification of both parts, finally resulting in a KBS core prototype (extensible prototyping, step 2).

Once both the front-end prototype and the KB have been adjusted, the resulting KBS core prototype is merged with additional required framing functionality in the final *Integration* phase. This results in a fully functional, productive KBS (extensible prototyping, step 3). Both the KBS core prototype or the fully functional KBS artifact then can serve as the basis for the next cyclic pass of the APM—starting over again with the Planning Game. Regarding usability, the Integration phase offers the chance to reasonably conduct also more comprehensive evaluations based on the productive KBS. Examples are controlled experiments, or usability tests with large, representative user groups.

3.3 Synopsis

In this chapter, we proposed an approach for UI- and usability-oriented, user-centered KBSE. Therefore, we first defined key components for lifting common KBSE approaches to a more encompassing level: Extensible prototyping, KBS UI patterns, and suitable usability instruments. Those components foster active user participation and thus a more user-centered development regarding various aspects. Extensible prototyping firstly supports the active user participation in KA. The same accounts for an incorporation in RE and UI development tasks and can be additionally supported by tailored KBS UI patterns. Selected usability instruments, as well as the basic support for highly iterative development, finally enable and ease user participation regarding diverse evaluation/assessment activities. For demonstrating the flexible applicability of the proposed components, we demonstrated their integration with an existing agile KBSE. This resulted in a more formal, encompassing, user-centered development methodology for KBS: The *Encompassing Agile Process Model (EAM)*.

A general downside of using patterns as well as software prototyping tools is, that those might hinder a entirely free, creative exploration of all options—due to naturally only supporting a portion of the potential design space. This is why we suggest to also include lo-fi prototyping techniques, where appropriate. Also, in the context of KBS basic system requirements often are quite similar; this rather advocates than refrains from the application of patterns and proven solutions. We further argue, that the overall advantages imposed by extensible prototyping and patterns in the KBSE context outweigh the named shortcomings—especially, when using a software tool that offers a reasonable amount of flexibility regarding an adaption or extension of the provided UI widget collection.

The next chapters deal with practical aspects of the proposed approach. In Chapters 4 and 5, we discuss basic presentation options and patterns for KBS UIs. Further, we present the tailored KBSE tool *ProKEt* in Chapter 6 and actual experiences with the approach in Section 7.

Part II

Applied Encompassing KBSE

Chapter 4

KBS Core Input Presentation

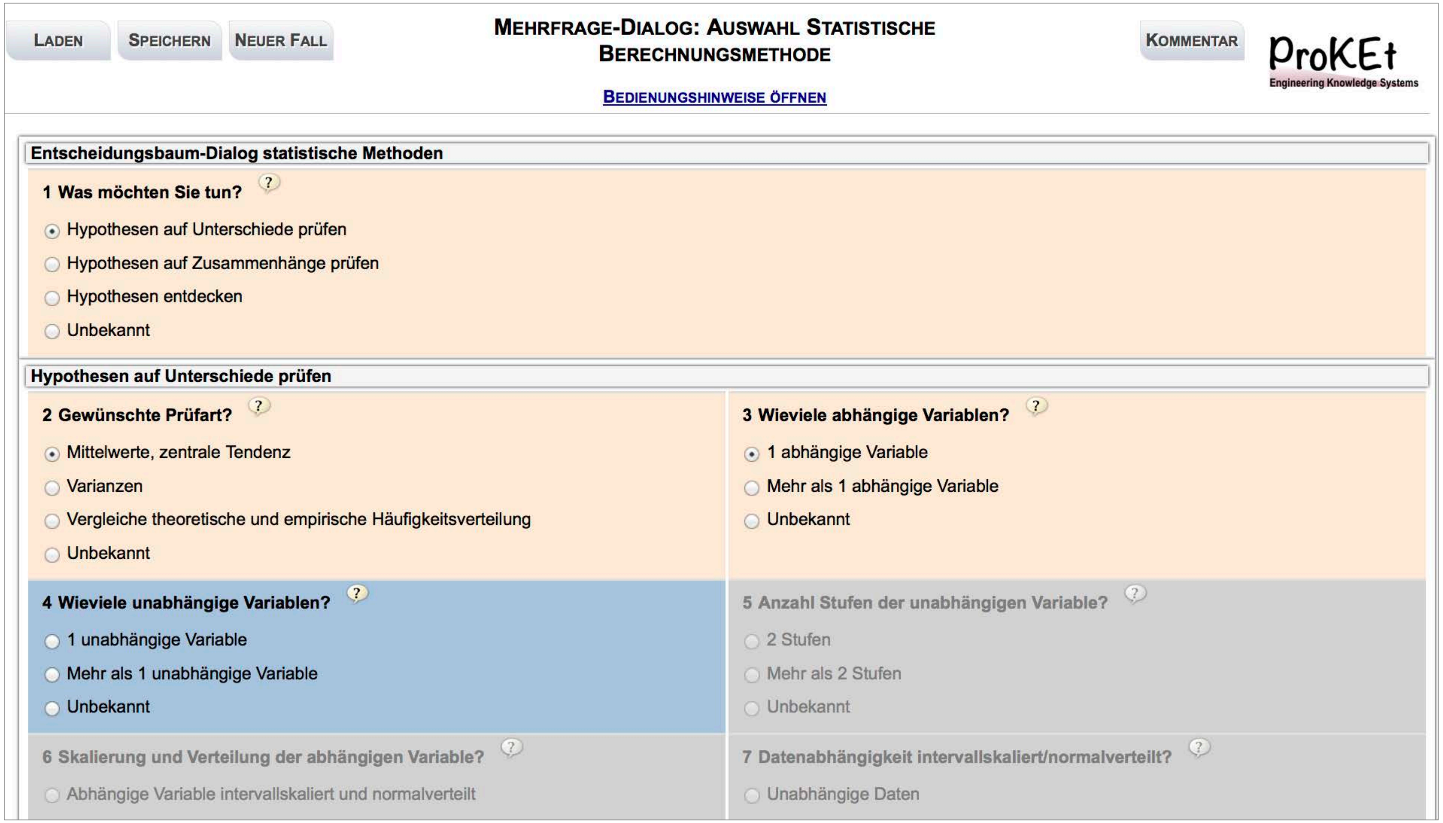
As suggested in Section 2.1.2, KBS exist of up to four UI modules, each of which requires a distinct representation: *Core input*, *results*, *justification*, and *auxiliary information*. Thereby, the core input module denotes the one essential module required for any KBS implementation—the remaining three modules can be summarized as *optional KBS context* (see also Chapter 5), the integration and configuration of which depends on the particular use case, user- and application context of the KBS.

In this chapter, we discuss presentation options for the core input module. Where appropriate, we indicate general usability- or UI design principles that served as the basis for certain design decisions. We begin by introducing *general presentation dimensions and configuration options* concerning the entire core input module in Section 4.1—e.g., whether and how to present non-indicated items. In Section 4.2, we give an overview of *basic UI widgets* for representing particular interview objects—e.g., radio buttons for choice questions. Thereby, we specifically focus on browser-based solutions due to the advantages they offer for the implementation and usage of participative KBS, see Section 2.1.5. Based on those presentation options, we suggest several KBS Core Input UI patterns that unite a distinct basic interaction style with an appropriate UI presentation in Section 4.3.

4.1 General Core Input Presentation Dimensions

The general presentation configuration options introduced in the following basically can be configured in a twofold manner: *Globally*—i.e., for the entire KBS UI core; or *locally*—i.e., for single questionnaires, questions, or answer options. This separate configurability offers the advantage of easily defining global settings that account for the major part of the KBS UI whilst single items can be tailored. Also, the presented configuration options are independent from the particular interview object type. An example is the configuration, whether to hide or show non-indicated interview objects—this basically is equal for all question types alike, such as numerical-, text- or choice questions. Table 4.1 summarizes the proposed *core input presentation dimensions*, their corresponding *configuration options*, and the respective *application level*. Figure 4.1 exemplifies several configuration options at the example of a Box Questionnaire implementation in ProKEt, and will be used for reference in the following sections.

a. Grouped View (global) This dimension applies only to presentation styles, where several to many questions are presented simultaneously. For the grouped view, related questions are assigned to topical display groups such as questionnaires—as opposed to displaying them



MEHRFRAGE-DIALOG: AUSWAHL STATISTISCHE BERECHNUNGSMETHODE

BEDIENUNGSHINWEISE ÖFFNEN

Entscheidungsbaum-Dialog statistische Methoden

1 Was möchten Sie tun? ?

- [✓ Hypothesen auf Unterschiede prüfen]
- Hypothesen auf Zusammenhänge prüfen
- Hypothesen entdecken
- Unbekannt

Hypothesen auf Unterschiede prüfen

2 Gewünschte Prüfart? ?

- [✓ Mittelwerte, zentrale Tendenz]
- Varianzen
- Vergleiche theoretische und empirische Häufigkeitsverteilung
- Unbekannt

3 Wieviele abhängige Variablen? ?

- [✓ 1 abhängige Variable]
- Mehr als 1 abhängige Variable
- Unbekannt

4 Wieviele unabhängige Variablen? ?

- 1 unabhängige Variable
- Mehr als 1 unabhängige Variable
- Unbekannt

5 Anzahl Stufen der unabhängigen Variable? ?

- 2 Stufen
- Mehr als 2 Stufen
- Unbekannt

6 Skalierung und Verteilung der abhängigen Variable? ?

- Abhängige Variable intervallskaliert und normalverteilt

7 Datenabhängigkeit intervallskaliert/normalverteilt? ?

- Unabhängige Daten

Figure 4.1: Questionnaire reference implementation with a german KB on statistical recommendation. It applies several general configuration described in detail on pp. 39 ff.: Grouping in topical questionnaires, brown-blue coloring scheme, show non-indicated items (deactivated), 1-2-1-column configuration, default option *Unbekannt [Unknown]*, interview object numbering (questions), fixed question width (due to column layout).

Presentation Dimension	Configuration Options	Applicability
a) Grouped View	enabled / disabled	global
b) Status Coloring Scheme	enabled / disabled	global
c) Non-indicated Questions	hide / show	global / local
d) Abstraction Questions	hide / show	global / local
e) Number of Columns	multiple (nr) / single	global / local
f) Default Answer Option	enabled / disabled	global / local
g) Numbering	enabled / disabled	local
h) Fixed Width	enabled (width) / disabled	local

Table 4.1: General presentation dimensions for the KBS UI core input module. For each dimension, its respective configuration options as well as the application level (global or local) are provided.

within one single form. Grouped display adheres to the principle of *Chunking*, which aims at "combining many units of information into a limited number of units or chunks, so that the information is easier to process and remember" c.f., [Lidwell et al., 2010, p. 40]. In this regard, groupings enhance the orientation of the user regarding the contents of the KB, due to the topical order and structuring of relevant and related items. This can prove especially helpful in cases of large KBs with hundreds of questions, or where several mutually excluding processing paths exist. On the other hand, in case there is an easily manageable number of total questions, an additional subdivision into groups might rather irritate users.

The reference implementation, shown in Figure 4.1, p. 40, applies a grouped view by assigning questions to basic topical questionnaires. In the figure, the initial group *Entscheidungsbaumdialog statistische Methoden [Decision Tree Interview for statistical methods]*, and a part of the following group *Hypothesen auf Unterschiede prüfen [Test hypotheses for differences]* are presented.

b. Status Mediation Coloring Schemes (global) Regarding UI styles that present many to all interview items at once, often a mediation of the answering status is beneficial. That is, indicating which items have already been answered and which ones are still open. This can be realized easily and clearly by applying distinct coloring schemes. There, we suggest to use two to three different colors at most regarding the answering state mediation. This is based on to the general recommendation, to limit colors within UIs to five at maximum—e.g., [Lidwell et al., 2010, p. 48].

Basically, we propose the integration of *non-indicated/abstraction items* in the commonly known 'deactivated', i.e. greyed, style. This best conveys the intention that such KB items actually exist, yet that they are are currently not activated.

Regarding *answered questions,* we suggest to use rather unobtrusive color hues, that are able to group those answered items visually, yet that do not attract too much attention. Examples are grey to white hues, as used in the EuraHS final implementation—see Figure 7.6, on p. 117. This conforms to the recommendation in [Lidwell et al., 2010, p. 48] to use grey shades for unobtrusively grouping background/currently unused items. Also, the ProKEt reference implementation of the Daily style uses a simple, grey-based coloring—answered questions either are not colored at all (but printed in bold face) or are highlighted with a light grey. This, however, was rated controversially in the user studies in Spring 2014 (c.f., Section 7.3): Some users rated that unobtrusive coloring as very clear, clean, and supporting efficiency (by not distracting users), others stated the design as too unfriendly and too simplistic. Alternatively, lighter shades of brown can be suitable, in case a clearer distinction from deactivated items is desired, as brown basically accounts as trustworthy and neutral color. This was applied, e.g., in the Questionnaire reference implementation in ProKEt, c.f. Figure 4.1 on p. 40. In the KBS UI assessment, performed in Spring 2014 (see Section 7.3), that coloring was rated as clearly distinctive regarding the status mediation, a bit (yet not overly) colorful, and was perceived overall more friendly and beautiful than, e.g., the plainer Daily color scheme. Finally, we also already experimented with a lighter green hue for answered items. Here, the intention was to visually indicate answered questions as green—based on the assumption, that green (in our culture) mostly conveys the meaning *OK/go on*. This was greatly approved of by the domain experts/users in the Mediastinitis project, where this coloring scheme was and still is applied, c.f. Figure 7.5, on p. 113.

Regarding *active yet unanswered* questions, generally brighter, more saturated colors may be applied, c.f., [Lidwell et al., 2010, p. 48]. There, we basically made good experiences with a medium blue tone for still open questions. The selection of blue thereby again based on the assumed (psychological) effects of colors, where blue attributes as calming, trustworthy, professional, dependable, constant, secure, and potentially favored by both women and men. Thus, blue seemed an appropriate choice regarding diverse users in a professional, potentially stressful (medical) usage context as induced by the Mediastinitis (Figure 7.5, on p. 113) and EuraHS (Figure 7.6, on p. 117) projects.

Regarding active questions, another alternative can be interesting: To highlight the suggested next question differently from the remaining active questions as to indicate more clearly the proposed optimal interrogation sequence of the KBS. Therefore, we experimented with a combination of green (currently suggested next question) and yellow (remaining open questions). This adheres both to the recommendation to use an analogous scheme with adjacent colors on the color wheel, c.f., [Lidwell et al., 2010, p. 48]; and to the recommendation to use warm colors for indicating foreground (active, still relevant) questions. Using green for the next suggested question again bases on the assumed cultural meaning of green (*Go on/Go*). This scheme was exemplarily used for the first prototype in the Mediastinitis project, c.f., Figure 7.4, a, on p. 110. Yet, it was not pursued any further in actual projects, as the distinct highlighting of a suggested question so far was required neither in the Mediastinitis, nor in the EuraHS or the JuriSearch project—in all cases, the basic data input sequence is deliberately intended to be freely explorative.

c. Hide/Show Non-Indicated Interview Objects (global / local) Regarding non-indicated interview objects, i.e., that are not (yet) active on the questioning agenda because the corresponding preconditions have not been fulfilled, there exist two basic presentation options: To entirely *hide* them, until they are actually required—which is the default implementation for conversational UIs. Or to *present them distinctly*, for easing their differentiation from regular questions at one glance. Figure 4.1 shows an example where inactive objects are displayed grey (inactive, as also motivated in the previous section) and thus are easily spotted.

Depending on the application context, both options can provide particular advantages. Hiding non-indicated interview objects allows for focussing only on the currently important objects as the display is not cramped with many items that are potentially never required. This form of *constrained presentation*—c.f., [Lidwell et al., 2010, p. 60]—can help to preserve better orientation for the users, especially regarding rather large KBS. Additionally, in data collection systems that handle specific, critical input, such a presentation best avoids a corrupted input data set. On the other hand, when integrating a lot of follow up questions flexibly, the UI representation may constantly change as subsequent questions need to be moved on accordingly. This is especially problematic in multi-column displays, where questions might migrate over several columns. Such an inconsistent presentation form strongly contradicts Nielsen's sixth heuristic—*Recognition rather than recall* [Nielsen, 1994]—as then users are hindered in recognizing the relevant next questions and are forced to 'search' them in other parts of the UI.

The other way around, showing all items at a time yet in distinct manners can be helpful in cases, where the user is required to have an encompassing overview of all items contained in the KB. One example is the debugging use case, c.f., Section 2.1.3. There, a clearly distinct representation of the special status of currently non-indicated questions adheres to the principle of *Self-Descriptiveness*, postulated, e.g., by the second heuristic of [Nielsen, 1994] or in the [ISO 9241–110, 2006]. This option naturally mostly is applicable for form style implementations, where many to all interview objects are presented simultaneously.

d. Hide/Show Abstraction Questions (global / local) Similar to non-indicated interview objects, also abstraction questions may be shown anytime—again distinctly styled as to indicate their special state—or may be hidden until their value becomes derived during the session.

When hiding such questions, however, users may easily get confused regarding suddenly appearing questions with an already set value. This may easily outweigh the potential advantages of a *constrained presentation* as proposed by [Lidwell et al., 2010, p. 60]. Also, in case a lot of abstraction questions need to be integrated in that flexible manner, similar problems with an unsteady display may occur as described in the previous section.

Thus, indicating such questions—e.g., in a greyed manner as motivated before, or explicitly labeled as *abstract/derived*—helps to leverage the understanding of such items and therefore oftentimes provides the more beneficial option. This on the one hand increases a KBS' *Self-Descriptiveness*, c.f., the second norm of the [ISO 9241–110, 2006]; and further it supports the above mentioned heuristic of Nielsen in fostering the recognition of objects that now remain in the same UI place.

As a final variant, abstraction questions can also be hidden entirely from the core input UI and displayed only in separate parts. Examples are, to present them within a solution panel, or in an input data summary (the latter variant was applied in the EuraHS project, see Section 7.1.3).

e. Single/Multiple Columns Presentation (global / local) This option specifies, whether the respective UI elements use a single column presentation style for displaying contained interview objects, or whether they are spread over multiple columns. This can be applied both globally and locally. Globally, it specifies into how many columns each framing UI module is divided, i.e., UI frame, questionnaires, and questions alike. Locally, the columns configuration applies only to the respectively defined element type. For example, on the question level, it specifies, how many questions are placed next to each other within a questionnaire. On the answer level, it specifies, how many columns are used to display answer options next to each other. Figure 4.1, p. 40 shows a 1-column (questionnaire level) / 2-column (question level) / 1-column (answer level) configuration. That is, up to two questions are rendered next to each other, yet answer options are listed only within one column top to bottom.

f. Default Answer Options (global / local) This property specifies default answer options that are added automatically to questions by the KBS UI—i.e., without having to explicitly alter the KB. The most prominent example probably is a default answer option *unknown* (see also Figure 4.1, p. 40). This is typically used and helpful in contexts, where users are likely not to know how to answer each and every question. The unknown option avoids that users get stuck with certain questions and have an alternative for carrying on and completing the KBS session. This in turn fosters the aspect of *effectivity*, e.g. stated in the first norm of the [ISO 9241–110, 2006], as KBS sessions can be finished even in the case of incomplete knowledge. Yet the suitability has to be considered carefully, as also the risk exists, that users choose this comfortable option far too quickly. This potentially can obscure the entered data or complicating a reasonable results inference process. Another option, that has proven valuable so far is the option *indifferent*. Indifferent thereby is closely related to unknown, yet has a different semantic. Unknown basically can imply that the user either is *indifferent* regarding the options, or does not *understand* the option(s). Indifferent, in contrast, implies more clearly that the user has understood the options but deliberately does not want to chose any specific one.

Globally applied, a default answer option specifies, whether to show or suppress the respective option for all questions. This reduces the task to add this option manually for each item of the KB to adding one single property in the UI configuration. For productively used KBSs, we naturally advise to adapt the respective KB accordingly in the end, as any form of knowledge always should be contained in the KB. However, this property fosters the eased experimentation with different default options during the early development phases by reducing the otherwise required potentially huge but noneffective KA efforts for always altering the KB.

g. Interview Object Numbering (local) Especially regarding large KBs, the numbering of questionnaires or questions can result in a more ordered appearance and clearer structure of the UI. Numbering basically constitutes a form of *Affordance*—c.f., [Lidwell et al., 2010, p. 22]—regarding the suggested or required sequence of processing questions. This in turn also increases the *Self-Descriptiveness* of a KBS, see [ISO 9241–110, 2006] or [Nielsen, 1994]. Numbering is most feasible for questionnaires and questions. The framing UI typically is never numbered, and for answer options numbering likewise rarely is suitable. Figure 4.1, p. 40, shows an example where questions exhibit a continuous numbering yet questionnaires are unnumbered.

h. Fixed Interview Object Width (local) Depending on the actual wording of interview objects, it can be advisable to apply a certain, predefined width for their presentation in some cases. This can lead to more distinctly structured, clearer UI representations—especially in cases, where the wording of interview objects varies extensively (very short to extremely long texts). Regarding particularly those extremes, though, it also has to be considered that a fixed width may automatically lead to wasted UI space—concerning all interview items with a shorter length than the specified value. Thus, it has to be evaluated carefully, whether the benefits of enhanced structuring outweigh the disadvantage of a lengthier, not optimally filled UI. This option is only feasible locally. Defining one common width for questionnaires, questions, and answers alike most probably can not lead to any reasonable (presentation) results.

The example implementation of the Daily pattern, applies a fixed width parameter regarding the questions, see Figure 4.4.a, p. 54. This is why answer option listings always begin in the same place left-justified. Yet, it does not apply a fixed width for answer options themselves. Thus the flat answer fields all exhibit varying widths but on the other hand fill the UI space in a highly compact manner.

4.2 Interview Object Presentation Options

Apart from those general presentation options introduced in the previous section, particularly questions can be represented by various UI widgets that depend on the respective target question- and input data type. Table 4.2 provides an overview of *KBS question types*, see also Section 2.1.2, and lists corresponding *input data characteristics*, and suitable *widgets*.

As Table 4.2 indicates, there exist various options for each KBS question type. Further, some of the widgets are applicable for more than one question type. The decision regarding the most appropriate representation thereby depends on the particular input data type and additionally: On whether it is mappable to alternative data representations. On basic constraints that may be imposed by the framing KBS UI pattern—e.g., Check List exhibits its strengths best, if all choices of choice questions are mappable to a fixed answer set. On constraints imposed by the KB contents—e.g., questions with many answer options may not be well presentable as radio button selects. Or on particular user preferences—e.g., certain users may prefer plain familiar widgets over either more space efficient or interaction efficient variants. Specifically such user-induced preferences should be considered carefully. This conforms to the *Aesthetic-Usability Effect* principle—c.f, [Lidwell et al., 2010, p. 20], that assumes that the (perceived) usability of the KBS automatically increases in case user preferences are complied with.

(Binary) OC Questions Probably the commonly best known representation for OC questions are radio buttons, see Figure 4.2, a. Some well established alternatives are listed in Table 4.2 and further are depicted in Figure 4.2, b-e: Flat and button toggle, drop-down/combobox, spinner, and slider.

Further, Figure 4.2, j, sketches also image maps for realizing OC (or also MC) questions based on interactively clickable images. There, both the plain presentation of the image map is feasible—i.e., answering of questions corresponds to clicking areas in the image map. Alternatively, those can be extended by suitable widgets (depending on the question type: radio buttons or checkboxes) that likewise allow to select respective choices. Answering a question then corresponds to either selecting such an option or selecting an image map area likewise. In that case, once a selection has been set in one part, the respective counterpart is equally highlighted.

OC questions exhibit a further special case: Binary OC questions, i.e., questions, that provide only two distinct answer alternatives. Common examples are yes/no, or on/off. Those, of

KBS Question Type	Input Data	Widget(s)
OC Question	categorical → > 2 options	Radio Buttons (Fig. 4.2, a) Flat/Button Toggle (Fig.4.2,b) Drop-down (Fig. 4.2, c) Combo-box (Fig. 4.2, c) Spinner (Fig. 4.2, d) Slider (Fig. 4.2, e) Image Map (Fig. 4.2, j)
(Binary OC Question)	→ exactly 2 options	Switch (Fig. 4.2, f) all above (Fig. see above)
MC Question	categorical → ≥ 2 options	Checkboxes (Fig. 4.2, g) Flat Toggle (Fig. 4.2, b) Image Map (Figure 4.2, j) Tailored Drop-down (Figure 4.2, d)
Numerical/Date/Time Question	continuous, free	Input/Text Field (Fig. 4.2, h) Calendar Widgets (k)
	maps choice set	see One Choice Widgets
Text Question	free	Input/Text Field (Fig. 4.2, h) Input/Text Area (Fig. 4.2, i)
	maps choice set	see One Choice Widgets

Table 4.2: Overview: KBS question types, characteristics of possible input data, and corresponding presentation alternatives (widgets).

course, can be represented also by the general one choice widgets, as shown in Figure 4.2, a-d. Additionally, the distinct representation as a switch (c.f., Figure 4.2, f) is especially suitable as this denotes as an affordance (c.f., [Lidwell et al., 2010, p. 22]) that implicitly but intuitively hints at its intended meaning and usage by using the commonly known metaphor of switches. This further fulfills the *Self-Descriptiveness* criterion of the DIN EN ISO 9241-110 and Nielsen.

MC Questions Apart from their standard checkbox representation, shown in Figure 4.2, g, MC questions can also be visualized by some variants that were already proposed for OC questions. For example, flat/button toggle—c.f., Figure 4.2, b—and image map—c.f., Figure 4.2, j—both those widgets can be implemented easily as to accept more than one input option at a time. Similarly, also the standard drop-down widget can be configured for multiple option selection. Then, however, the base interaction of a simple selection click changes to selecting while pressing a dedicated key (oftentimes: STRG). Alternatively and with some more development effort, also the tailored implementation of a click-only MC drop-down widget, requiring no additional key pressing, is possible—e.g., using scripting languages such as JavaScript. Yet then, it has to be considered that such a non-standard implementation might irritate users, especially if the tailored- and the standard implementation resemble each other too strongly in their UI presentation. Particularly, this contradicts the principle to adhere to established standards, e.g., the fourth heuristic of [Nielsen, 1994] or the [ISO 9241–110, 2006]. For the same reason, even more established one choice widgets—e.g., radio buttons, spinner, sliders, or switches—should not be altered as to support multiple choice input, even if that may technically be feasible with some efforts.

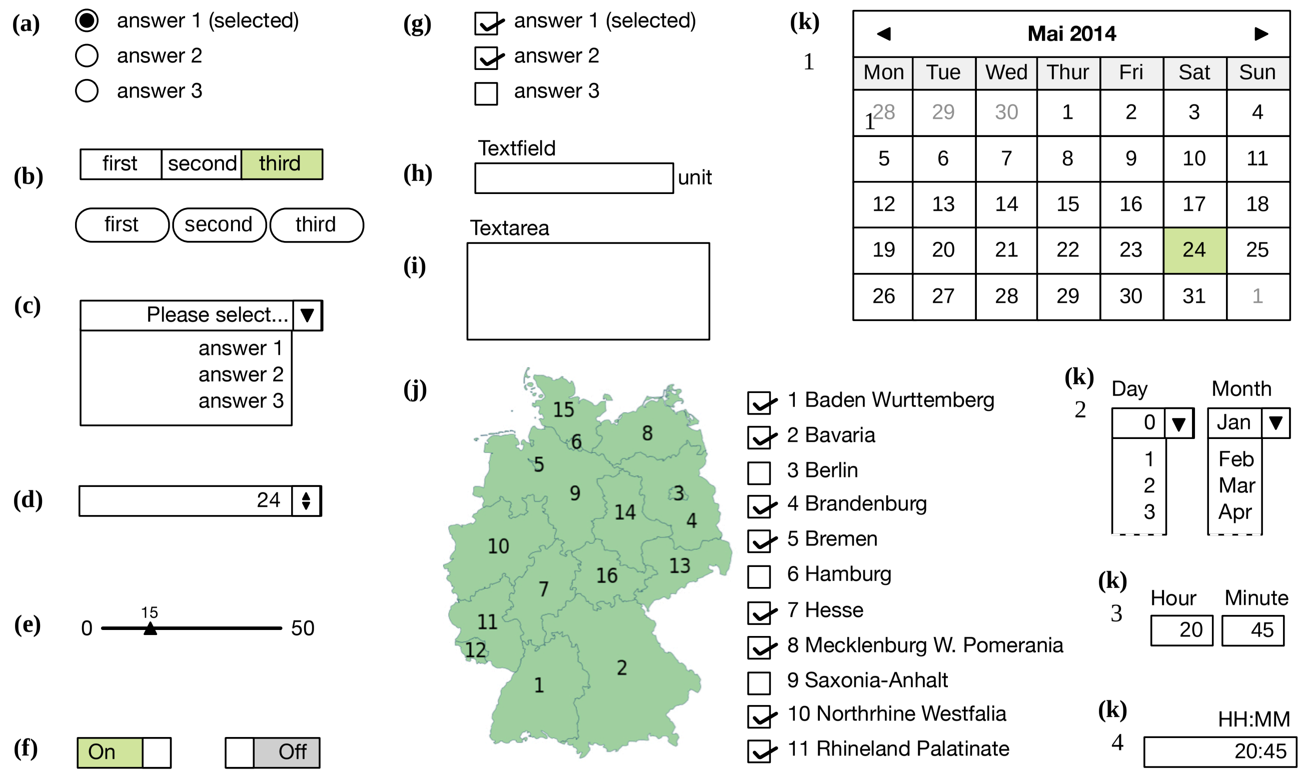


Figure 4.2: Basic UI widgets for representing the most common question types as categorized in Section 2.1.2 and Table 4.2: Radio Button (a); Flat and Button Toggle (b); Drop-down/Combo-box (c); Spinner (d); Slider (e); Switch (f); Check Box (g); Text Field (h); Text Area (i); Image Question (j); Calendar Widget (k1); Calendar Input Variants Realized by base widgets (k2-k4).

Numerical & Date/Time Questions The most basic representation for numerical-, date- or time questions is the standard input (field) widget—either singularly, or in a combined manner, see also Figure 4.2, h, k3, & k4. Further, also general choice question widgets, such as listed in Figure 4.2, a–f, can be applied in case the target input data is mappable onto a categorical data set. As example, for date questions with a constrained and fixed set of target input dates also a combined input using two drop-down fields is feasible, c.f., Figure 4.2, k2. Regarding particularly the input of date and time data, also comprehensive calendar widgets as sketched in Figure 4.2, k1 can be applied.

It has to be noted, that unconstrained input fields generally are the most error prone. Thus, of the depicted date input variations, Figure 4.2, k4 is the least restrictive and thus also the least secure regarding input correctness. In contrast, variations k1–k3 are more secure alternatives. However, also the generally less safe variants can be enhanced: For example, by placing clear, explanatory labels. Or by restricting the potential input technically on the software side—i.e., checking the input on the fly and displaying interactive warnings or hindering any further processing in case of assumedly faulty input. This allows for respecting the principle of avoiding erroneous input wherever possible, stated, e.g., by Nielsen's fifth heuristic, the fifth principle of the [ISO 9241–110, 2006], or by [Lidwell et al., 2010, p. 82].

Text Questions Here, similar base conditions apply as for numerical, date, and time questions: In case the potential input can be mapped to a reasonable categorical choice set, the choice question widgets radio button, flat/button toggle, drop-down, combo-box, or spinner—see Figure 4.2,a–d—become further appropriate presentation forms. Again, those exhibit the additional benefit of helping to avoid typing/input errors. In cases, where text input is completely unpredictable, e.g., if users are asked to give additional personal comments/remarks, either standard text input fields or text areas can be used—c.f., Figure 4.2, h & i.

4.3 KBS Core Input UI Patterns

In this section, we introduce a collection of KBS UI patterns based on experiences in diverse research projects. Those patterns focus on distinct UI and interaction solutions for the core input module. The collection sub-divides into four main patterns, each with several variants: *Questionnaire*, *Interview*, *Clarifier*, and *Clarifier Hybrid*. Each main pattern exhibits a characteristic interaction type as foundation. Further, it specifies the general problem, solved by all its variants, and defines certain common key characteristics. The variants vary regarding the degree of realizing the usability-related KBS classification criteria (see Section 3.1.2) or regarding concrete UI realizations. Basically, the patterns build on the general presentation dimensions and particular interview objects presentation forms as proposed in Sections 4.1 and 4.2.

In the following, we first suggest a pattern template for specifying KBS UI patterns in a uniform, clear manner in Section 4.3.1. Also, we provide an encompassing delimitation of the proposed patterns for quick reference. In Sections 4.3.2 to 4.3.5, we then introduce the patterns in detail.

4.3.1 KBS UI Pattern Template & Quick Reference

a. KBS UI Pattern Template

For the specification of patterns, usually a pattern template is applied. Such a template consists of particular *Sections*, each characterizing a certain aspect of the pattern, and a defined sequence of those sections. Table 4.3 summarizes the template applied for the subsequently proposed KBS UI patterns. Thereby, always the main pattern with its common base characteristics is described first. The *variants* section of each pattern then is further sub-divided according to the table, defining some more fine-grained properties of each pattern variant.

	Section	Description
Main Pattern	Problem Statement	Subsumes the problem that is solved by this pattern—KBS classification criteria (see Section 3.1.2, pp. 30 ff.) are used for a clear delimitation.
	Solution	Describes the general, characteristic (UI/interaction) solution all pattern variants apply for solving the problem.
	Variants	Variations of the main pattern, differing regarding the properties below.
Pattern Variants	Users	Characterizes the typical target users according to *domain expertise* and *usage frequency*; see also the *User Type* classification in Section 2.1.4.
	Knowledge	Summarizes relevant properties of suitable knowledge for that pattern.
	Solution Specifics	Additional characteristics regarding the UI and interaction realization.
	Consequences	Delimitates the pattern against other patterns—based on the usability-related KBS classification criteria (see Section 3.1.2, pp. 30 ff.) for clarity.
	Example	Compact subsumption and screenshot for a reference implementation.

Table 4.3: Basic template for specifying KBS UI patterns. The main pattern specifies several characteristics, see upper part of the table, that apply for all its variants. Pattern variants differ regarding some more fine-grained dimensions, as listed in the lower part of the table.

b. KBS UI Pattern Delimitation—Quick Reference

Table 4.4 provides an overview of the proposed KBS UI patterns and their delimitation. Thereby, the tailored, usability-related KBS classification criteria, introduced in Section 3.1.2, pp. 30 ff., are rated regarding the extent of their realization in the respective pattern, first. We distinguish the values *high (h)*, *medium (m)*, and *low (l)*. In case, two such categories apply, both values are provided—e.g., *mh* for *medium to high*. Secondly, domain expertise (*Expert.*) and usage frequency (*Freq.*), c.f., Section 2.1.4, are taken into account as representative user characteristics. Values in parentheses generally indicate a restricted applicability. For example, *Box Questionnaire* generally is best suitable for laymen and experienced users. Experts, however, might prefer more efficient, compact, or interactive styles with certain expert shortcuts for consultation or documentation tasks. Hence they might consider a *Box Questionnaire* UI mostly regarding the debugging use case, which is why we classify this style suitable for experts in a restricted manner.

Basically, the listed delimitation values found on representative practical experiences with several of the patterns in past and actual KBS projects, as well as on our subjective estimation

		Questionnaire			Interview			Clarifier		Hybrid	
		4.3.2.a. Box Questionnaire	4.3.2.b. Daily Questionnaire	4.3.2.c. CheckList Questionnaire	4.3.3.a. Strict Interview	4.3.3.b. Grouped Interview	4.3.3.c. Hierarchical Interview	4.3.4.a. Hierarchical Clarifier	4.3.4.b. Form Add-On Clarifier	4.3.5.a. Clarifier Interview	4.3.5.b. Clarifier Questionnaire
Usability Criteria	**compact**	m	h	h	l	m	h	h	h	l	m
	comprehensive	m	m	m	h	h	m	mh	m	h	m
	descriptive	l	l	l	l	l	m	h	h	l	l
	efficient	m	h	h	l	m	m	m	lm	l	m
	explorable	h	h	h	l	lm	mh	h	h	l	h
	intuitive	h	m	h	h	m	lm	l	lm	mh	mh
	adapts size	m	m	h	m	h	m	m	m	h	m
	transparent	h	h	h	m	m	m	h	h	m	h
	structured	h	m	h	l	m	lm	m	lm	l	h
Expert.	**Laymen**	X	X	X	X	X	(X)	/	(X)	X	X
	Experienced	X	X	X	X	X	X	X	X	X	X
	Experts	(X)	X	X	/	(X)	X	X	X	(X)	X
Freq.	**One-time**	X	X	X	X	X	(X)	/	(X)	X	X
	Frequent	X	X	X	(X)	X	X	X	X	(X)	X

Table 4.4: Core KBS UI patterns (with section reference): Encompassing delimitation according to *tailored usability-related KBS classification criteria* with extent of their realization (high/h–medium/m–low/l); two values such as *lm* mean a *low to medium* degree. Further, the *user characteristics* domain expertise (Expert.) and *usage frequency* (Freq.) are rated. Marks in parentheses indicate a restricted applicability.

regarding their generalization. However, the KBS UI patterns evaluation studies (reported in detail in Section 7.3, pp. 134 ff.) already confirmed the general tendency for the assessed patterns Daily and Box Questionnaire, Strict and Hierarchical Interview, and Hierarchical Clarifier.

The table is intended as a short-hand guide for finding a suitable pattern for a given context or problem description based on the KBS usability criteria. For example, if a pattern is required that should be *participative*, *descriptive*, rather *compact*, and *efficient*, the *Hierarchical Clarifier* is the most appropriate candidate as it supports all three aspects with a rather high degree—as shown in Table 4.4, fourth last column.

4.3.2 Questionnaire Base Pattern

Problem Statement: The KBS should compactly display a greater part or the entire KB, provide a high level of transparency, and intuitive usage. Also, a certain explorability is desired, in the sense that the user may deviate from the suggested interrogation sequence. Auxiliaries, such as popup explanations, may increase the per default not inherent comprehensibility of the interview objects presentation.

Solution: The Questionnaire pattern implements an interaction- and UI design that resembles standard paper- or web-based questionnaire forms. Depending on the particular realization, many to all indicated interview objects are displayed simultaneously. Typically, the interview items are ordered in some form of grid-based layout. Questionnaire implementations may suggest (visually), but do not necessarily prescribe, any optimal interrogation sequence and thus foster explorative usage. Per default, forward knowledge is used. Questionnaire is quite universally applicable regarding documentation, consultation, and debugging tasks, given that the KBS presentation is configured accordingly: Documentation requires no solution panel whereas this is essential for consultation tasks; in debugging mode all interview objects are displayed regardless their indication state, thereby highlighting non-indicated items, e.g., in a greyed manner.

Variants: Box-, Daily-, and CheckList Questionnaire.

a. Box Questionnaire

Users: Laymen, experienced, (experts) | one-time, frequent.

Knowledge: Box Questionnaire is less critical to varying or extensive interview object texts than Daily pattern. In non-debugging mode, the more follow-up questions are contained, the less constant the presentation gets. This is due to the fact that—especially in multi-column views—preceding interview objects are moved further each time a new item is flexibly integrated. Also, this pattern is rather inappropriate for single-question indication- or strict hierarchical knowledge, such as heuristic decision trees [Puppe, 2000]. Then, it rather approximates conversational styles, and thus looses its original key characteristic and strength of presenting many items simultaneously. When used for debugging, however, Box Questionnaire is equally well applicable for all knowledge types.

Solution Specifics: Box Questionnaire closely adheres to the design of classical paper-based questionnaires. In case of multicolumn configurations, the basic reading direction is left-to-right and then top-to-bottom. Box Questionnaire majorly uses familiar question presentation forms—e.g., checkboxes and radio buttons for choice questions. This strengthens the resemblance to a paper-based questionnaire. The optimal interrogation flow may be (visually) suggested by applying distinct coloring schemes for answer status mediation, see also Section 4.1.4.1, p.41 ff. Box Questionnaire works with heterogenous question types and basically allows for the inclusion of all types, see Section 4.2, pp. 45 ff.

Consequences: Box Questionnaire is less compact but more clearly and regularly structured than Daily Questionnaire. This is caused by the presentation of each question within a self-contained framing box. Compactness may be increased by rendering a collection of similar questions with a *Check List* sub-module, see Figure 4.6, p. 57. Further, the more space consuming presentation reduces the overall efficiency and transparency—especially for comprehensive KBs, where the overview can get lost and a lot of scrolling may be required.

Example: Figure 4.3, p.53, shows a Box Questionnaire implementation example for a plants recommendation system (in german, see Appendix A.2)—used in the *consultation variant* with integrated anytime solution panel (a). Distinct coloring underlines the current state of questions (answered=light brown/open=blue). Auxiliary information for interview objects is integrated by clickable info icons, here shown for the solutions (b). The respective auxiliary information is shown in separately opened info pages (c) and is invoked by clicking info-icons for the respective interview object. Solution justifications are displayed in a popup window (d) when clicking on the desired solution.

b. Daily Questionnaire

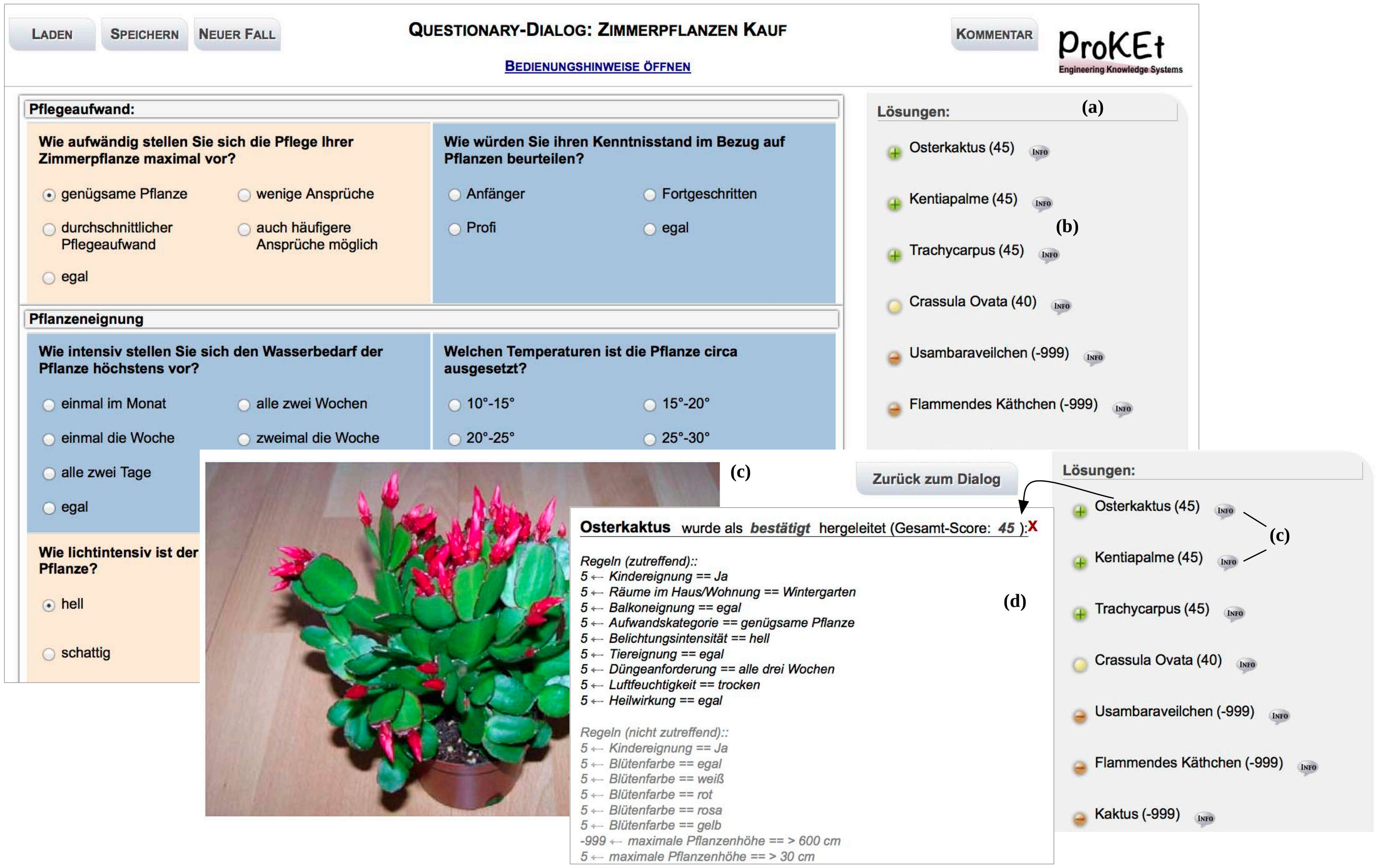
Users: Laymen, experienced, experts | one-time, frequent.

Knowledge: Best presentation results for rather short answer texts with not too much varying lengths. In non-debugging mode, the presentation becomes less constant the more follow up items are required to be integrated flexibly as preceding items consequently are moved. Also, in non-debugging mode this pattern is not suitable for single-question indication- or strict hierarchical knowledge, such as heuristic decision trees [Puppe, 2000]. Then, it looses its original strength of presenting many to all interview items simultaneously and thus rather approximates conversational styles. Yet when used for debugging, Daily Questionnaire is well applicable for all knowledge types.

Solution Specifics: The Daily variant presents questionnaires and their included questions as a column-wide, visual entity. This induces a basic resemblance to daily newspapers, especially in multi-column configurations. Adding to that impression is the flat, juxtaposed question presentation style. There, answer options are selectable by simply clicking the plain, listed text instead of using standard checkboxes or radio buttons. The already high base compactness of Daily is further enhanced by the possibility to collapse questionnaires that are already processed or not (yet) required. Thus, only the questionnaire name keeps displayed, but contained questions are hidden. This fosters the concentration on selected questions/questionnaires and a clearer, less cluttered UI impression. Regarding choice questions with a lot of answer options, the usage of drop-down widgets further increases compactness. Daily style can process heterogenous question types, and can in principle include all question types as introduced in Section 4.2, pp. 45 ff. As tradeoff, it may loose its key strength—compactness—with an increasing number of specialist widgets, such as image questions.

Consequences: Daily Questionnaire is one of the most compact styles, but it is less clearly structured than Box Questionnaire. As more overview is provided, less scrolling is required. The additional simplicity of the text-click interaction can induce a high efficiency of usage. This makes Daily Questionnaire also especially apt for debugging, where an overview of the greater part of the KB and an efficient processing of diverse interrogation paths is a key priority.

Example: Figure 4.4, p. 54, depicts a Daily implementation example with a statistical methods recommender KB (in german, see Appendix A.1). It shows a *debugging (consultation) and two-column presentation variant* with integrated anytime solution panel (a). Answered questions



QUESTIONARY-DIALOG: ZIMMERPFLANZEN KAUF

BEDIENUNGSHINWEISE ÖFFNEN

Pflegeaufwand:

Wie aufwändig stellen Sie sich die Pflege Ihrer Zimmerpflanze maximal vor?

[✓ genügsame Pflanze] ○ wenige Ansprüche ○ durchschnittlicher Pflegeaufwand ○ auch häufigere Ansprüche möglich ○ egal

Wie würden Sie ihren Kenntnisstand im Bezug auf Pflanzen beurteilen?

○ Anfänger ○ Fortgeschritten ○ Profi ○ egal

Pflanzeneignung

Wie intensiv stellen Sie sich den Wasserbedarf der Pflanze höchstens vor?

○ einmal im Monat ○ alle zwei Wochen ○ einmal die Woche ○ zweimal die Woche ○ alle zwei Tage ○ egal

Welchen Temperaturen ist die Pflanze circa ausgesetzt?

○ 10°-15° ○ 15°-20° ○ 20°-25° ○ 25°-30°

Wie lichtintensiv ist der Pflanze?

[✓ hell] ○ schattig

Lösungen:

- Osterkaktus (45)
- Kentiapalme (45)
- Trachycarpus (45)
- Crassula Ovata (40)
- Usambaraveilchen (-999)
- Flammendes Käthchen (-999)

Osterkaktus wurde als ***bestätigt*** hergeleitet (Gesamt-Score: ***45***):

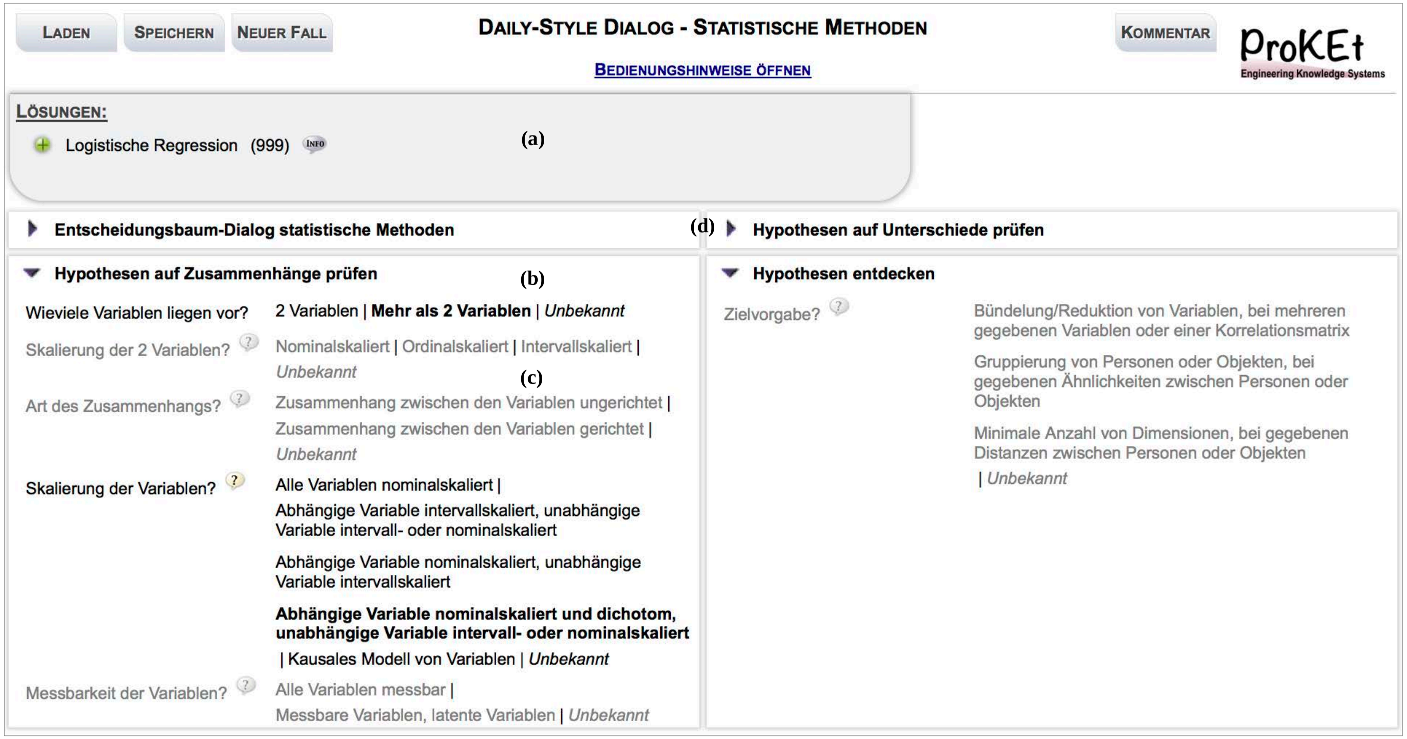
Regeln (zutreffend)::
5 ← Kindereignung == Ja
5 ← Räume im Haus/Wohnung == Wintergarten
5 ← Balkoneignung == egal
5 ← Aufwandskategorie == genügsame Pflanze
5 ← Belichtungsintensität == hell
5 ← Tiereignung == egal
5 ← Düngeanforderung == alle drei Wochen
5 ← Luftfeuchtigkeit == trocken
5 ← Heilwirkung == egal

Regeln (nicht zutreffend)::
5 ← Kindereignung == Ja
5 ← Blütenfarbe == egal
5 ← Blütenfarbe == weiß
5 ← Blütenfarbe == rot
5 ← Blütenfarbe == rosa
5 ← Blütenfarbe == gelb
-999 ← maximale Pflanzenhöhe == > 600 cm
5 ← maximale Pflanzenhöhe == > 30 cm

Lösungen:

- Osterkaktus (45)
- Kentiapalme (45)
- Trachycarpus (45)
- Crassula Ovata (40)
- Usambaraveilchen (-999)
- Flammendes Käthchen (-999)
- Kaktus (-999)

Figure 4.3: Box Questionnaire reference implementation in ProKEt for a german plants recommendation system. See p. 52 for a detailed description.



DAILY-STYLE DIALOG - STATISTISCHE METHODEN

LÖSUNGEN:
Logistische Regression (999)

Entscheidungsbaum-Dialog statistische Methoden

Hypothesen auf Unterschiede prüfen

Hypothesen auf Zusammenhänge prüfen

Wieviele Variablen liegen vor? 2 Variablen | **Mehr als 2 Variablen** | *Unbekannt*

Skalierung der 2 Variablen? Nominalskaliert | Ordinalskaliert | Intervallskaliert | *Unbekannt*

Art des Zusammenhangs? Zusammenhang zwischen den Variablen ungerichtet | Zusammenhang zwischen den Variablen gerichtet | *Unbekannt*

Skalierung der Variablen? Alle Variablen nominalskaliert | Abhängige Variable intervallskaliert, unabhängige Variable intervall- oder nominalskaliert Abhängige Variable nominalskaliert, unabhängige Variable intervallskaliert **Abhängige Variable nominalskaliert und dichotom, unabhängige Variable intervall- oder nominalskaliert** | Kausales Modell von Variablen | *Unbekannt*

Messbarkeit der Variablen? Alle Variablen messbar | Messbare Variablen, latente Variablen | *Unbekannt*

Hypothesen entdecken

Zielvorgabe? Bündelung/Reduktion von Variablen, bei mehreren gegebenen Variablen oder einer Korrelationsmatrix Gruppierung von Personen oder Objekten, bei gegebenen Ähnlichkeiten zwischen Personen oder Objekten Minimale Anzahl von Dimensionen, bei gegebenen Distanzen zwischen Personen oder Objekten | *Unbekannt*

Figure 4.4: Daily Questionnaire reference implementation, used with a statistical methods recommender KB in german. For details, see p. 52.

are highlighted in bold face (b) and currently inactive objects are presented greyed (c). Further, foldable questionnaires (d)—per default initialized opened—are used for a more compact presentation.

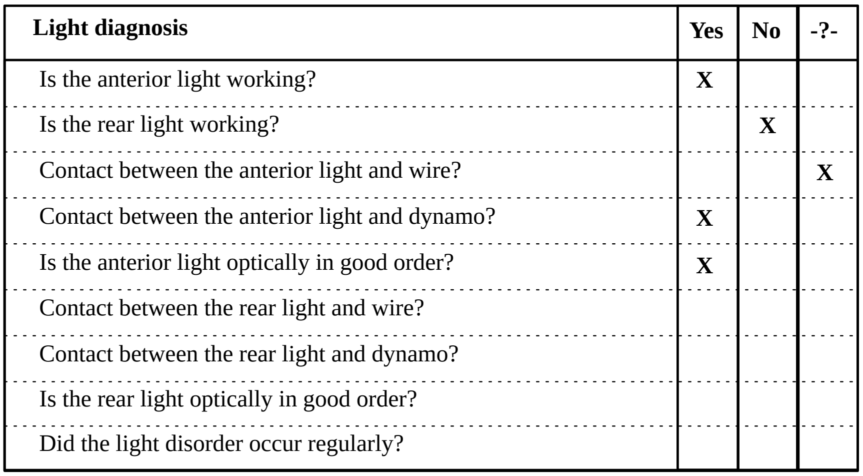
c. CheckList Questionnaire

Users: Laymen, experienced, experts | one-time, frequent

Knowledge: All choice questions need to be mappable on a fixed answer set (homogenous questions). Further, KBs for CheckList preferably contain little to no follow up questions—the efficiency of CheckList decreases with increasing numbers of flexibly integrated interview objects and a resulting an unsteady UI display.

Solution Specifics: The CheckList Questionnaire imitates classical paper-based check lists. This is achieved by listing the questions in a compact, mostly fixed manner, each question followed by check field columns for each answer option, c.f., Figure 4.5, a. Answering a question then simply corresponds to ticking off the appropriate check field—and click repeatedly for deselecting again. CheckList works best with homogenous questions, thus the integration of further question types is not recommended.

Consequences: The highly compact and regularly structured appearance of CheckList fosters highly efficient data input: First, the simple base interaction always takes place in the same part of the UI. Second, users do not need to cater with how or what type of answer to provide—only, that they select the correct field. Due to its familiar presentation and interaction form, CheckList suits both laymen and experts well who require a highly efficient form of data input.



Light diagnosis	**Yes**	**No**	**-?-**
Is the anterior light working?	X		
Is the rear light working?		X	
Contact between the anterior light and wire?			X
Contact between the anterior light and dynamo?	X		
Is the anterior light optically in good order?	X		
Contact between the rear light and wire?			
Contact between the rear light and dynamo?			
Is the rear light optically in good order?			
Did the light disorder occur regularly?			

Figure 4.5: Exemplary sketch of CheckList Questionnaire for a bicycle fault detection KB. Fixed answers are *Yes/No/Unknown(-?-)* and correspond to check field columns on the righthand UI side.

It is, however, rather inappropriate for debugging tasks due to its knowledge-related restrictions. CheckList can serve as sub-module for other patterns if a question subset is mappable to fixed answers.

Example: Figure 4.5, p. 55, shows a basic sketch of CheckList using exemplary questions from a bicycle fault detection KB. Questions map to fixed answers *Yes/No/Unknown(-?-)*. Answers are displayed as check field columns on the righthand UI side and answering simply corresponds to ticking off the respective fields—similar as in paper-based check lists.

Figure 4.6, p. 57, demonstrates a CheckList sub-module for a Box Questionnaire implementation. This demonstrates the highly compact appearance, which can strongly foster an eased, efficient data input regarding questions with similar answer sets.

4.3.3 Interview Base Pattern

Problem Statement: The KBS UI should be highly intuitive and comprehensible, thus specifically easing its usage for first-time users and domain laymen. For that context, compactness, descriptiveness, efficiency, and explorability can be neglected. Further, the core input UI does not need to be transparent in itself.

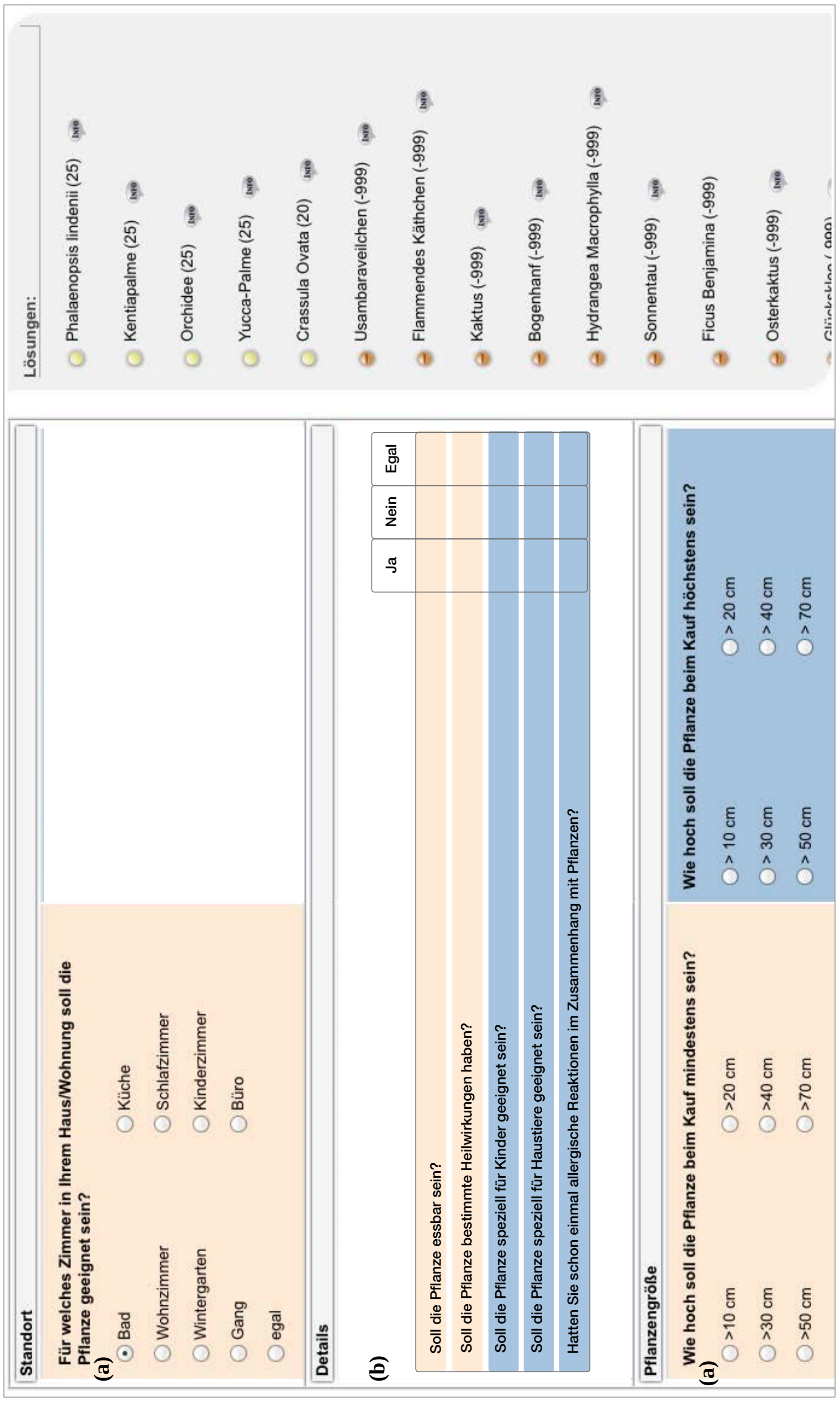
Solution: The Interview pattern imitates human conversations by presenting always a single question—or several related questions—at a time. Typically, the corresponding explanation(s) are integrated directly within the question presentation display or are easily retrievable (e.g., by hovering the question). Thus, auxiliary information are readily available anytime. The one by one presentation style basically follows a rather fixed data input sequence, not much supporting any explorability. Thus, Interview is applicable both for consultation or documentation tasks where a defined questioning sequence is a premise. Debugging, in contrast, is only partly feasible: The correctness of questioning sequences may be well investigated, yet in contrast, KB completeness or the overall structuring and comparison of different interrogation paths in parallel is not feasible. The basic lack of transparency can be alleviated partly by integrating auxiliaries such as an interview object history—listing the already processed interview items—and a progress information display. Section 5.2.2 shortly discusses different progress display types for KBS.

Variants: Strict-, Grouped-, and Hierarchical Interview, where Hierarchical Interview exhibits a very distinct own tree-based style.

a. Strict Interview

Users: Laymen, experienced | one-time, (frequent)

Knowledge: Especially regarding large KBs, carefully designed interrogation paths are necessary for only querying the relevant objects. Optimally, interview items are enriched by extensive auxiliary information, as those are automatically integrated for display.



Standort

Für welches Zimmer in Ihrem Haus/Wohnung soll die Pflanze geeignet sein?

(a)

[✓ Bad]	Küche
Wohnzimmer	Schlafzimmer
Wintergarten	Kinderzimmer
Gang	Büro
egal	

Details

(b)

	Ja	Nein	Egal
Soll die Pflanze essbar sein?			
Soll die Pflanze bestimmte Heilwirkungen haben?			
Soll die Pflanze speziell für Kinder geeignet sein?			
Soll die Pflanze speziell für Haustiere geeignet sein?			
Hatten Sie schon einmal allergische Reaktionen im Zusammenhang mit Pflanzen?			

Pflanzengröße

(a)

Wie hoch soll die Pflanze beim Kauf mindestens sein?

>10 cm	>20 cm
>30 cm	>40 cm
>50 cm	>70 cm

Wie hoch soll die Pflanze beim Kauf höchstens sein?

> 10 cm	> 20 cm
> 30 cm	> 40 cm
> 50 cm	> 70 cm

Lösungen:

- Phalaenopsis lindenii (25) INFO
- Kentiapalme (25) INFO
- Orchidee (25) INFO
- Yucca-Palme (25) INFO
- Crassula Ovata (20) INFO
- Usambaraveilchen (-999) INFO
- Flammendes Käthchen (-999) INFO
- Kaktus (-999) INFO
- Bogenhanf (-999) INFO
- Hydrangea Macrophylla (-999) INFO
- Sonnentau (-999) INFO
- Ficus Benjamina (-999)
- Osterkaktus (-999) INFO

Figure 4.6: Sketch of CheckList- integrated as a sub-module (b) of Box Questionnaire (a), demonstrating its highly compact representation.

Solution Specifics: Strict Interview displays always only a single question at a time. The corresponding auxiliary information is automatically presented near the question/answer text as to reduce retrieval efforts, and to foster its direct comprehensibility. Also, Strict Interview follows a prescribed interrogation sequence. Thereby, it closely imitates one-by-one human conversations where interviewer and interviewee take exact turns in posing/answering questions. This is further underlined by the automated progression interaction: Once the user selects an answer, the next question is displayed automatically. In case more user control is desired or required, Interview can equally well include navigation buttons for enabling users to move from an answered question to the next question manually. As for all Interview variants, an interactive interview objects history and a progress display can enhance the transparency of this style.

Consequences: With its comprehensive single question presentation and the strict interrogation guidance, Strict Interview denotes a highly intuitive and comprehensible style especially for one-time users and/or domain laymen. In turn, this style is not compact, and consequently not very efficient to use, especially regarding lengthy interrogation sequences. Thus, it is particularly not suitable for large documentation systems where often more freedom regarding the interrogation sequence is required. The general lack of an encompassing KB overview renders Strict Interview inappropriate regarding debugging KB contents (e.g., completeness of the questions)—yet, it may be applied for debugging the questioning sequences.

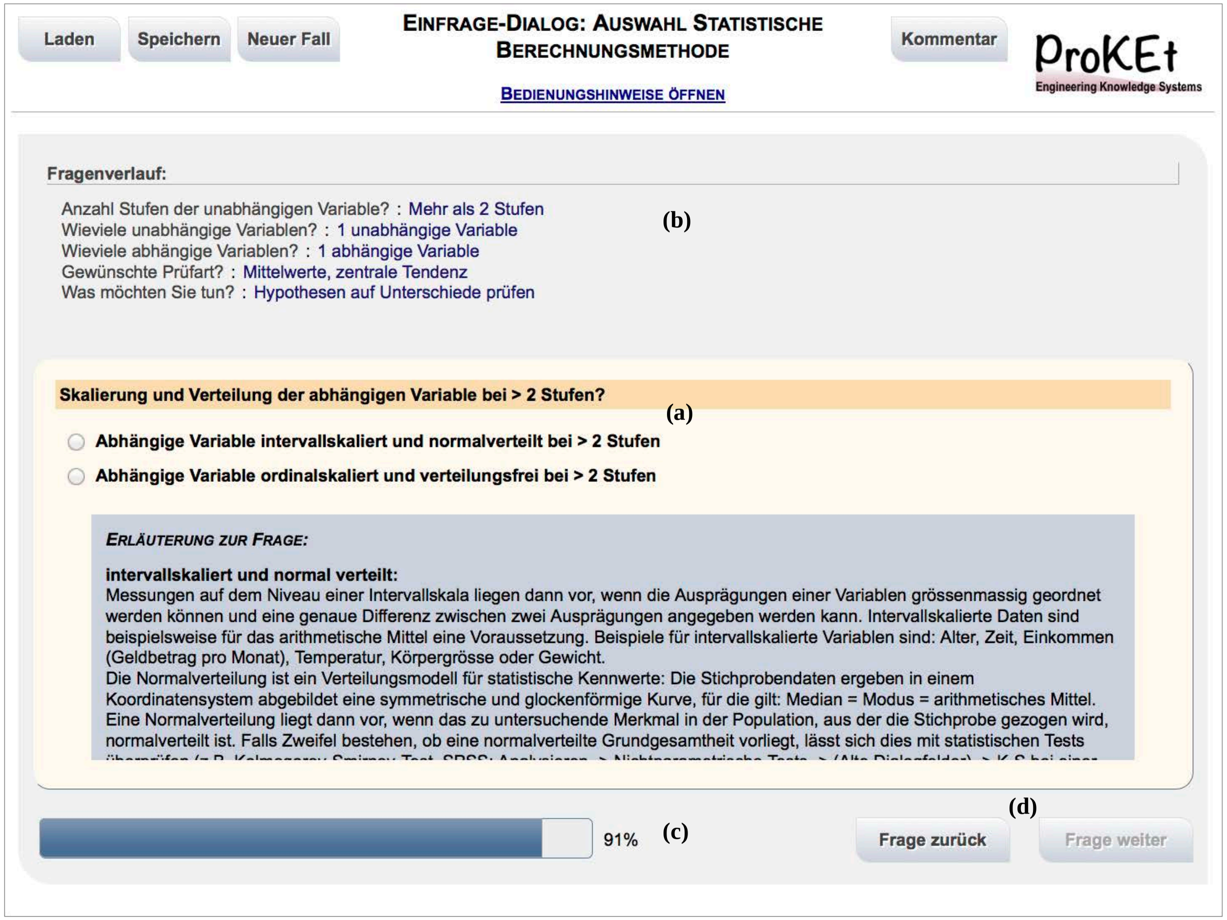
Example: Figure 4.7, p.59 shows the ProKEt reference implementation of Strict Interview. It uses a decision tree KB for statistical methods recommendation (in german, see Appendix A.1). The example shows the core data input module (a) that consists of the main question display with integrated auxiliary information display. Further, it offers a navigable interview objects history (b) and visual progress information (c). Navigation buttons allow for stepwise switching back and forth between answered questions (d). For not diverting users from the core input process, this example includes the solution display only at session end.

b. Grouped Interview

Users: Laymen, experienced, (experts) | one-time, frequent

Knowledge: Questions should be grouped, e.g., by topical questionnaires, as without groupings this style rather degrades to a form of Strict Interview. Auxiliary information regarding the interview items—either for each single question or for an entire group—are beneficial for automated integration.

Solution Specifics: Grouped Interview imitates more loose conversations by presenting always a group of related questions simultaneously. Thus, only the sequence of the groups is prescribed, whereas questions within a group can be explored and answered at will. Also, the default answering interaction offers more user-control: Selecting an answer does not automatically induce the presentation of the next question group but only a corresponding highlighting of the respective question/answer. For proceeding to the next question group, users need to navigate there manually using a dedicated navigation button, c.f., Figure 4.8, d. Yet again as a variant, Grouped Interview may be implemented as to present the next suitable group auto-



EINFRAGE-DIALOG: AUSWAHL STATISTISCHE BERECHNUNGSMETHODE

BEDIENUNGSHINWEISE ÖFFNEN

Fragenverlauf:

Anzahl Stufen der unabhängigen Variable? : Mehr als 2 Stufen
Wieviele unabhängige Variablen? : 1 unabhängige Variable
Wieviele abhängige Variablen? : 1 abhängige Variable
Gewünschte Prüfart? : Mittelwerte, zentrale Tendenz
Was möchten Sie tun? : Hypothesen auf Unterschiede prüfen

Skalierung und Verteilung der abhängigen Variable bei > 2 Stufen?

○ **Abhängige Variable intervallskaliert und normalverteilt bei > 2 Stufen**
○ **Abhängige Variable ordinalskaliert und verteilungsfrei bei > 2 Stufen**

ERLÄUTERUNG ZUR FRAGE:

intervallskaliert und normal verteilt:
Messungen auf dem Niveau einer Intervallskala liegen dann vor, wenn die Ausprägungen einer Variablen grössenmassig geordnet werden können und eine genaue Differenz zwischen zwei Ausprägungen angegeben werden kann. Intervallskalierte Daten sind beispielsweise für das arithmetische Mittel eine Voraussetzung. Beispiele für intervallskalierte Variablen sind: Alter, Zeit, Einkommen (Geldbetrag pro Monat), Temperatur, Körpergrösse oder Gewicht.
Die Normalverteilung ist ein Verteilungsmodell für statistische Kennwerte: Die Stichprobendaten ergeben in einem Koordinatensystem abgebildet eine symmetrische und glockenförmige Kurve, für die gilt: Median = Modus = arithmetisches Mittel.
Eine Normalverteilung liegt dann vor, wenn das zu untersuchende Merkmal in der Population, aus der die Stichprobe gezogen wird, normalverteilt ist. Falls Zweifel bestehen, ob eine normalverteilte Grundgesamtheit vorliegt, lässt sich dies mit statistischen Tests

91%

Figure 4.7: Strict Interview used with a decision tree KB for statistical methods recommendation in german. For a detailed description, see p. 58.

matically once the current group is entirely processed. The particular presentation resembles the framing UI of Strict Interview. The question groups can be included using multiple question styles, such as Questionnaire or Daily.

Consequences: Grouped Interview is less restrictive than Strict Interview and thus can result in a more efficient question processing—especially, if compact patterns are applied for presenting the question groups, such as Daily- or Check List Questionnaire. Also, Grouped Interview offers a bit more explorability regarding the interrogation sequence within the question groups. However, due to still displaying only parts of the KB—the groups—at once, Grouped Interview also is rather inappropriate regarding most debugging tasks. The other way around, the grouped presentation of this style renders it appropriate for contexts, where prescribed topical areas need to be processed in a certain sequence.

Example: Figure 4.8, p. 61, sketches a Grouped Interview documentation UI. Each question group display consists of the name of the group and the presentation of the contained questions, here, rendered in Box Questionnaire style (a). An interview history provides a little context (b) by listing the processed groups, which can be expanded as to show the answer state of the contained questions. The overall progress is estimated by a progress bar widget (c). The example starts with the init group *Common Route*, and allows users to traverse the grouping sequence via the navigation buttons (d+e).

c. Hierarchical Interview

Users: (Laymen), experienced, experts | (one-time), frequent

Knowledge: (Heuristic) Decision tree knowledge, e.g. [Puppe, 2000].

Solution Specifics: This variant offers an interactively navigable hierarchical UI. It presents answer items as clickable nodes of a tree representation that span one line of the UI, each. Corresponding question texts are omitted. This increases the compactness of this style. At the same time, it decreases users' memory load as only answer texts have to be processed instead of question-answer pairs. Solutions are represented by (distinctly highlighted) nodes—found always at the deepest nesting levels. Thus, the resulting tree structure particularly mirrors the indication paths within the KB that lead from initial questions to the solution. A separate solution panel thus is no requirement as the justification of a solution can directly be 'read' visually from the tree path. However, a solution display can nevertheless provide value—e.g., in case a non-visual, more compact justification such as a finding list (see Section 5.3.1, a.) of the solution is desired. Or in case, users demand on-the-fly information regarding which solutions become excluded during the interrogation. Auxiliary information is presented in a dedicated, fixed side panel. Those can be triggered either by clicking info icons or by simply hovering the question node.

The basic interaction in Hierarchical Interview is the selection of the respectively suitable answer node. Once clicked, such a node expands the corresponding next node level, i.e., answer options of the next appropriate follow up question. This is simply repeated until automatically a solution node is reached. As a member of the Interview pattern family, Hierarchical Interview

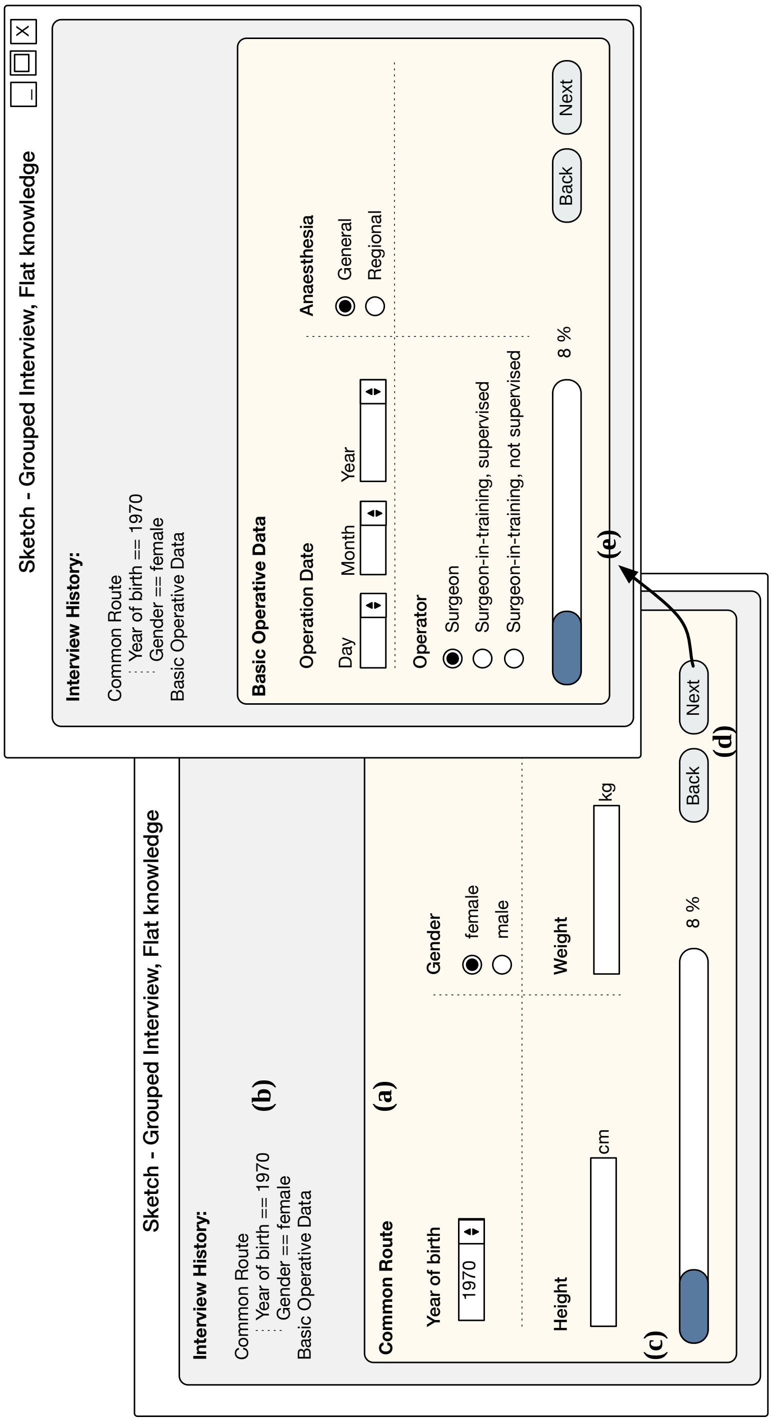
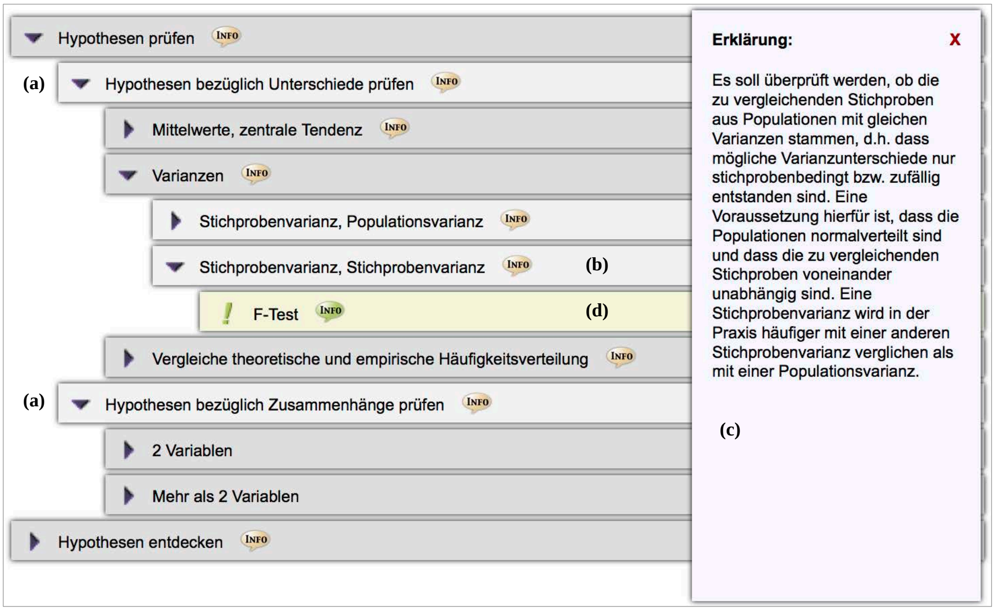


Figure 4.8: Example sketch for a Grouped Interview UI for the medical documentation use case. See p. 60 for a detailed description.


Hypothesen prüfen INFO
(a) Hypothesen bezüglich Unterschiede prüfen INFO
Mittelwerte, zentrale Tendenz INFO
Varianzen INFO
Stichprobenvarianz, Populationsvarianz INFO
Stichprobenvarianz, Stichprobenvarianz INFO (b)
F-Test INFO (d)
Vergleiche theoretische und empirische Häufigkeitsverteilung INFO
(a) Hypothesen bezüglich Zusammenhänge prüfen INFO
2 Variablen
Mehr als 2 Variablen
Hypothesen entdecken INFO

Erklärung: X

Es soll überprüft werden, ob die zu vergleichenden Stichproben aus Populationen mit gleichen Varianzen stammen, d.h. dass mögliche Varianzunterschiede nur stichprobenbedingt bzw. zufällig entstanden sind. Eine Voraussetzung hierfür ist, dass die Populationen normalverteilt sind und dass die zu vergleichenden Stichproben voneinander unabhängig sind. Eine Stichprobenvarianz wird in der Praxis häufiger mit einer anderen Stichprobenvarianz verglichen als mit einer Populationsvarianz.

(c)

Figure 4.9: Hierarchical Interview—ProKEt reference implementation with a german statistical calculation KB german. For details, see p. 63.

prescribes certain reasonable interrogation paths—leading from the initial answer options set to the solution nodes. Thereby, the single-question (i.e. here: single answer set) paradigm is adhered to by automatically closing branches that are currently not required. That is, if the user choses an alternative option on an upper level, formerly opened, non-related sub-branches are closed. Only the newly relevant branch remains displayed for reducing screen clutter and increasing clearness.

However, also a debugging variant of Hierarchical Interview is feasible that allows for comparatively exploring the interview items. Therefore, node states basically are never changed by the system. That is, opened branches remain open and collapsed branches remain collapsed until the user initiates a corresponding altering action. This enables users to open branches simultaneously—e.g., for directly comparing them. It has to be noted, that this variant provides less structure/guidance for the user, regarding the currently relevant path. Also, the screen is likely to get stuffed with nodes more quickly over time.

Consequences: In contrast to its Interview relatives, this hierarchical, space efficient tree style offers a higher compactness. Also, it exhibits skill-building ability as the coherences of interview objects are particularly visualized. In case auxiliary information is triggered by hovering the nodes, this style also offers a basic level of comprehensibility—yet not as high as Strict Interview with its automatically integrated information boxes. Also, Hierarchical Interview is not as intuitively usable as its two relatives, but in turn offers a (debugging) variant that fosters a higher explorability.

Example: Figure 4.9, p. 62, shows the reference implementation of Hierarchical Interview in ProKEt. It uses a statistical methods recommendation KB in german, see Appendix A.1, and is displayed in debugging mode—i.e., multiple branches can be opened simultaneously. Clickable nodes represent answers (a)—selecting a node expands the respective follow-up options. Auxiliary information are triggered by info icon click (b) and are displayed in the global information box (c). Solutions are also integrated in the tree as most deeply nested, distinctly styles nodes (d). Solution justification is visually represented by the expanded branches.

4.3.4 Clarifier Base Pattern

Problem Statement: A compact, transparent, descriptive, and optimally skill-building (by high comprehensibility) UI is required for a Clarification KBS. Users should further be enabled to increase efficiency in contributing their personal knowledge—e.g., by using expert shortcuts regarding the interrogation sequence (explorability/participation). Intuitive usage is no key requirement.

Solution: Clarifier characteristically uses backward knowledge, see Section 2.2. Thus, it renders the solitary target solution and all contributing questions simultaneously and offers facilities to adapt answers for immediate investigation of the consequences. Clarifier KBS can be applied for the standard consultation- or the clarification consultation use case, c.f., Section 2.1.3.

Variants: Hierarchical- and Form Add-on Clarifier.

a. Hierarchical Clarifier

Users: Experienced, experts | frequent

Knowledge: Hierarchical Clarifier exhibits its strengths optimally if the backward knowledge is refined over several abstraction levels. Thus, clarification consultation on diverse levels of expertise can be offered for lowering the hurdle of using KBS in highly expertise domains. Generally, also compactly formulated question/answer texts are favorable regarding an optimal display of the interactive tree style.

Solution Specifics: The basic presentation style of Hierarchical Clarifier is a single-line interactive tree style. In contrast to Hierarchical Interview, both the question and corresponding answers are rendered within the nodes. The default suggested question/answer presentation where answers are listed before the question, see Figure 4.10, may at first appear unusual. Yet, this enables the main answering interaction to take place in roughly the same part of the UI and thus helps to minimize interaction efforts. For alleviating problems with potentially too lengthy or too many answer options, drop-down widgets can be integrated for a more compact display. See, e.g., question *Conspicuous illness symptoms* in Figure 4.10. The target solution is presented as uppermost node of the tree, see Figure 4.10, a. It is followed by the base node set, i.e., top-level questions (Figure 4.10, b) that contribute to the solution rating directly. Each question node again can be followed recursively by further questioning levels. Each such set of

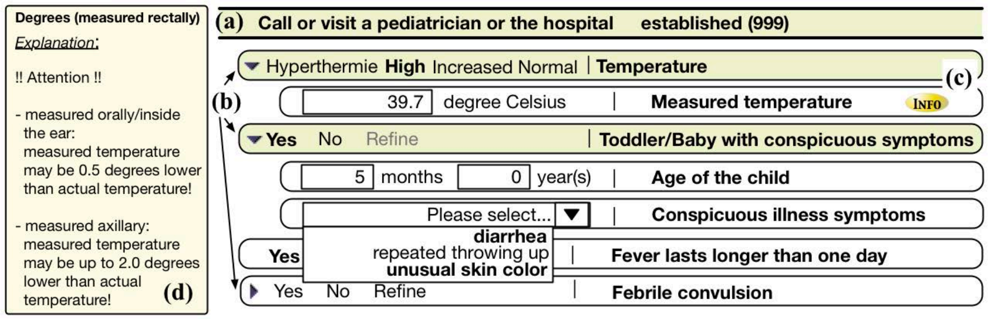

Degrees (measured rectally)
Explanation:
!! Attention !!
- measured orally/inside the ear: measured temperature may be 0.5 degrees lower than actual temperature!
- measured axillary: measured temperature may be up to 2.0 degrees lower than actual temperature!
(d)
(a) Call or visit a pediatrician or the hospital established (999)
(b)
(c)
Hyperthermie High Increased Normal | Temperature
39.7 degree Celsius | Measured temperature INFO
Yes No Refine | Toddler/Baby with conspicuous symptoms
5 months 0 year(s) | Age of the child
Please select... | Conspicuous illness symptoms
diarrhea
repeated throwing up
unusual skin color
Yes | Fever lasts longer than one day
Yes No Refine | Febrile convulsion

Figure 4.10: Sketch of Hierarchical Clarifier on the issue, whether the pediatrician/hospital should be called when specified symptoms are observed with one's child. For details, see pp. 63, ff.

child elements denotes a more fine-granular partition of its parent question (e.g., Figure 4.10, c), consisting of one or several questions. Thereby, the parent corresponds to the concept of an abstraction question and the children to those items relevant for deriving the abstraction value. Thus, questions are answered either directly; or implicitly, by expanding the refinement nodes and answering the corresponding child questions. Those answers then are propagated back to their parent elements. The control of choosing the most suitable answer/abstraction level is entirely left to the user and may be switched at will on the fly.

Descriptiveness can be further enhanced by node coloring: Abstract rating states are mapped to a certain color; question nodes (for current status mediation) and/or answer options (for providing a preview of the consequences) are colored according to their contribution to the target issue. In Figure 4.10, e.g., only the questions are colored according to their contribution to the solution due to the chosen answer. Therefore, a traffic lights scheme is used, mapping green to established, yellow to suggested, and red for excluded rating states. Overall descriptiveness can be further increased by visually representing also the node connections; i.e., the underlying knowledge that derives parent node values from their children. The ITree—i.e., specific instantiation example of Hierarchical Clarifier—in Figure 4.11 shows an example. Here, the derivation rule structures are indicated by *WENN[IF]*, *UND [AND]*, and *ODER [OR]* icons next to the respectively connected nodes. Additionally, dummy (i.e., empty) nodes exist, that serve as a meta-connection for visualizing also nested rules. A rule `questionX IF questionA AND (questionB OR questionC)`, for example, would be represented by a parent node *questionX*, followed by a node *questionA* and a dummy node for the AND connection on the same level. The dummy node in turn is followed by two nodes for *b* and *c*, c.f. Figure 4.11, p. 66.

Auxiliary information, potentially extensive in the context of highly expertise domains, are integrated in a dedicated side panel by this pattern (c.f., Figure 4.10, d). For requesting their display, both clicking on dedicated icons and automated display, when hovering the question, is feasible.

Consequences: The rather space efficient presentation of Hierarchical clarifier creates a compact presentation. Its topical refinement structure, the particular answer status mediation by color coding, and the tight integration of (potentially extensive) auxiliary information add to its overall descriptiveness and transparency. That way, Hierarchical Clarifier can be well applied in highly expertise contexts. Also, these properties support a high degree of comprehensibility, thus further fostering skill-building ability. In turn, Hierarchical Clarifier is not very intuitive to use neither regarding experts, nor laymen. However, the suitability of Hierarchical Clarifier depends more on the target users' usage frequency than on their expertise: Whereas one-time users most likely do not profit much from this style, users that use the KBS more frequently may well be trained to familiarize with the system. Yet, the efficiency of Hierarchical Clarifier again strongly depends on the users' expertise and the consequential recognition and usage of provided shortcuts.

Example: Figure 4.10, p. 64, sketches the basic Hierarchical Clarifier scheme. Schematic example of Hierarchical Clarifier—used for a rather flat abstraction structure in this case. The example clarifies the issue, whether the pediatrician/hospital should be called (a) when specified symptoms are observed with one's child by in total four questions (b). Three of them can be further refined for clarity (c) and thus denote abstractions. Abstraction nodes are expanded by clicking on the triangle handle in the front. For example, question *Temperature*, can be refined by a numerical question querying the precisely measured value in degrees (Figure 4.10, c). Auxiliary information is integrated in a dedicated side panel (d) and is triggered by clicking info icons next to the question text.

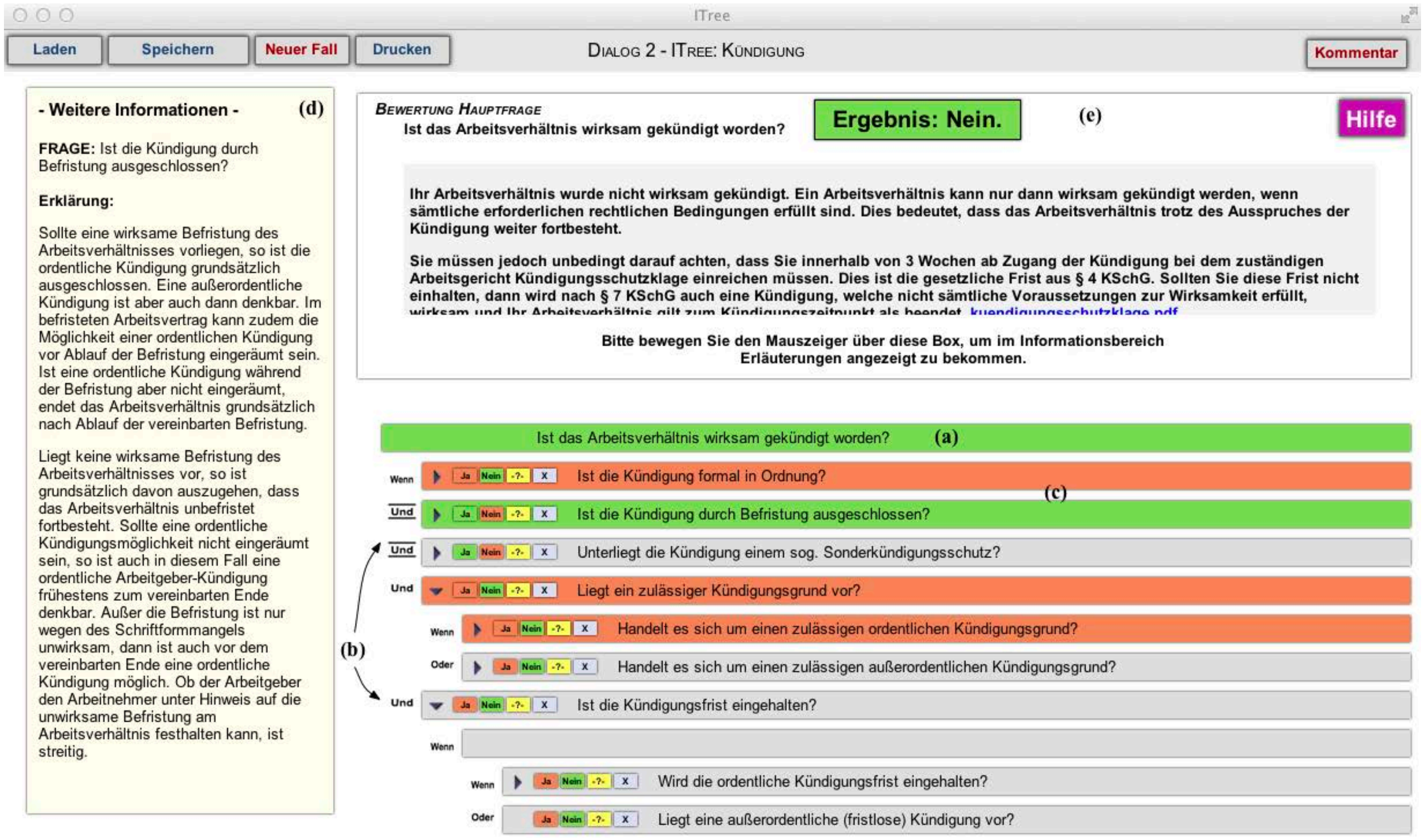
A specific instantiation of Hierarchical Clarifier—the ITree, already reported in [Freiberg and Puppe, 2012a]—is depicted in Figure 4.11, p. 66 ITree uses mostly questions with the fixed answer range *Yes, no, uncertain*, and an option to withdraw the answer: The *X* button. The target solution (= most abstracted 'question') is presented as topmost node (a). This issue is rated by a certain set of (abstract) top level questions (b). Each such question is refined recursively by a set of more fine-granular questions. When users answer refinement questions, answers are propagated towards the top of the tree. The traffic lights coloring scheme visually conveys the implications of answer options and answered questions regarding the core issue. The final state of the core issue is summarized in the header (e). Auxiliary information is shown in the dedicated side panel (d). Adhering to the basic Hierarchical Clarifier pattern, ITree uses more or less compound rules for calculating the value abstractions. For mirroring those rules, the UI presents respective AND/OR connectors in front of each node. For representing complex nested rules, dummy nodes, as described conceptually for Hierarchical Clarifier, are used.

b. Form Add-on Clarifier

Users: (Laymen), experienced, experts | (one-time), frequent

Knowledge: Only limited by the constraints, that the base justification presentations impose on the backward knowledge regarding its comprehensiveness.

Solution Specifics: Minimalistic consultation widgets are added to static base justification presentations (see Section 5.3.1). Therefore, the interview objects in the justification are implemented as triggers that invoke compact popup consultation widgets as indicated in Figures 4.12, I & II. Those present all answer options and accordingly highlight already selected answers. Additionally, also the solution contributing value may be indicated (see examples). Users then can



DIALOG 2 - ITREE: KÜNDIGUNG

- Weitere Informationen -

FRAGE: Ist die Kündigung durch Befristung ausgeschlossen?

Erklärung:

Sollte eine wirksame Befristung des Arbeitsverhältnisses vorliegen, so ist die ordentliche Kündigung grundsätzlich ausgeschlossen. Eine außerordentliche Kündigung ist aber auch dann denkbar. Im befristeten Arbeitsvertrag kann zudem die Möglichkeit einer ordentlichen Kündigung vor Ablauf der Befristung eingeräumt sein. Ist eine ordentliche Kündigung während der Befristung aber nicht eingeräumt, endet das Arbeitsverhältnis grundsätzlich nach Ablauf der vereinbarten Befristung.

Liegt keine wirksame Befristung des Arbeitsverhältnisses vor, so ist grundsätzlich davon auszugehen, dass das Arbeitsverhältnis unbefristet fortbesteht. Sollte eine ordentliche Kündigungsmöglichkeit nicht eingeräumt sein, so ist auch in diesem Fall eine ordentliche Arbeitgeber-Kündigung frühestens zum vereinbarten Ende denkbar. Außer die Befristung ist nur wegen des Schriftformmangels unwirksam, dann ist auch vor dem vereinbarten Ende eine ordentliche Kündigung möglich. Ob der Arbeitgeber den Arbeitnehmer unter Hinweis auf die unwirksame Befristung am Arbeitsverhältnis festhalten kann, ist streitig.

BEWERTUNG HAUPTFRAGE
Ist das Arbeitsverhältnis wirksam gekündigt worden?

Ergebnis: Nein.

Ihr Arbeitsverhältnis wurde nicht wirksam gekündigt. Ein Arbeitsverhältnis kann nur dann wirksam gekündigt werden, wenn sämtliche erforderlichen rechtlichen Bedingungen erfüllt sind. Dies bedeutet, dass das Arbeitsverhältnis trotz des Ausspruches der Kündigung weiter fortbesteht.

Sie müssen jedoch unbedingt darauf achten, dass Sie innerhalb von 3 Wochen ab Zugang der Kündigung bei dem zuständigen Arbeitsgericht Kündigungsschutzklage einreichen müssen. Dies ist die gesetzliche Frist aus § 4 KSchG. Sollten Sie diese Frist nicht einhalten, dann wird nach § 7 KSchG auch eine Kündigung, welche nicht sämtliche Voraussetzungen zur Wirksamkeit erfüllt, wirksam und Ihr Arbeitsverhältnis gilt zum Kündigungszeitpunkt als beendet. kuendigungsschutzklage.pdf

Bitte bewegen Sie den Mauszeiger über diese Box, um im Informationsbereich Erläuterungen angezeigt zu bekommen.

Ist das Arbeitsverhältnis wirksam gekündigt worden?
- Wenn: Ist die Kündigung formal in Ordnung?
- Und: Ist die Kündigung durch Befristung ausgeschlossen?
- Und: Unterliegt die Kündigung einem sog. Sonderkündigungsschutz?
- Und: Liegt ein zulässiger Kündigungsgrund vor?
  - Wenn: Handelt es sich um einen zulässigen ordentlichen Kündigungsgrund?
  - Oder: Handelt es sich um einen zulässigen außerordentlichen Kündigungsgrund?
- Und: Ist die Kündigungsfrist eingehalten?
  - Wenn:
    - Wenn: Wird die ordentliche Kündigungsfrist eingehalten?
    - Oder: Liegt eine außerordentliche (fristlose) Kündigung vor?

Figure 4.11: ITree variant of Hierarchical Clarifier, as implemented in the JuriSearch project (see Section 7.1.4). For details, see page 65.

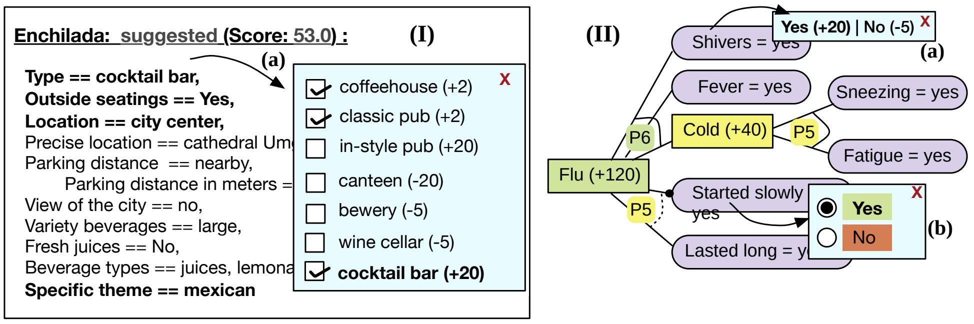


Figure 4.12: Form Add-on based on a finding list (I) and on a rule graph (II) base justification variant.

hypothetically explore the consequences of certain answers by investigating the answer popups; or by actually selecting other answer option(s). In the latter case, the value is set in the KBS session and the justification view is re-rendered, now including the newly selected finding and corresponding knowledge.

Consequences: Such flexibly explorable, familiarly styled consultation widgets in a compact presentation enable a basically more intuitive usage than Hierarchical Clarifier and thus are apt for a more diverse user range. Based on a solution justification, Form Add-on Clarifier further is well comprehensible, and when highlighting the answer consequences, it further is highly descriptive. The general popup interaction may be too inefficient for modifying/exploring large KBs, yet is perfectly suitable for investigating distinct cause–effect aspects.

Example: Figure 4.12, p. 67, depicts two exemplary Form Add-on sketches: Based on a finding list justification (I), and based on a rule graph (II). Variant I displays the Form add-on for the question *Typ [Type]* in Box Questionnaire style where the contributing value is indicated as precise number in parentheses. The option *Cocktailbar [cocktail bar]* had been originally selected during the KBS session. Thus, it is highlighted bold in the popup. The further ticked options have been selected subsequently in the clarification session. Variant II shows a OC popup in Daily style (a) with no indication regarding the solution contribution, and a Box style OC popup that indicates the (abstracted) contribution by color.

4.3.5 Clarifier Hybrid Patterns

Problem Description: A more intuitively usable and easily comprehensible UI representation for using clarification knowledge is desired.

Solution: Hybrid patterns merge intuitively, easily comprehensible UI patterns with clarification knowledge for enabling clarification also for one-time or laymen users. In contrast to pure Clarifiers, those hybrids only support the consultation targeted clarification variant, c.f., Section 2.2, p. 19, well.

Variants: Clarifier Interview and Clarifier Questionnaire.

a. Clarifier Interview

Users: Laymen, experienced, (experts) | one-time, (frequent)

Knowledge: Backward knowledge, i.e., targeting only one selected solution.

Solution Specifics: The basic UI solution is similar to Strict- or Grouped Interview, c.f., Sections 4.3.3.a., and 4.3.3.b.—with the major difference, that only targeted backward knowledge is processed sequentially. In case gradually refined knowledge is used, as for Hierarchical Clarifier, the respective refinement questions are inserted into the interrogation sequence as soon as they are invoked. For the Strict Interview variant, one refinement question after another is posed by the system. In contrast, the Grouped Clarifier Interview displays all refinement questions as one coherent group. Once all refinement questions are answered, the KBS automatically switches back to the previous abstraction level to continue the original interrogation. Additionally, Clarifier Interview can provide an extra button for leaving the refinement level without answering any or all questions there. For providing feedback regarding the current refinement level, a widget in the form of a Hierarchical Clarifier presentation can be added for presenting the rough questioning structure and marks the current state of the interrogation.

Consequences: The intuitively usable, better comprehensible UI supports also one-time users or laymen in using clarification modules. The tradeoff is, that efficiency, descriptiveness, compactness, and skill building ability are reduced in these variants as only a smaller portion of the KB—and respective coherences—are visible at a time. Thus, this style may be less attractive for trained/frequent users.

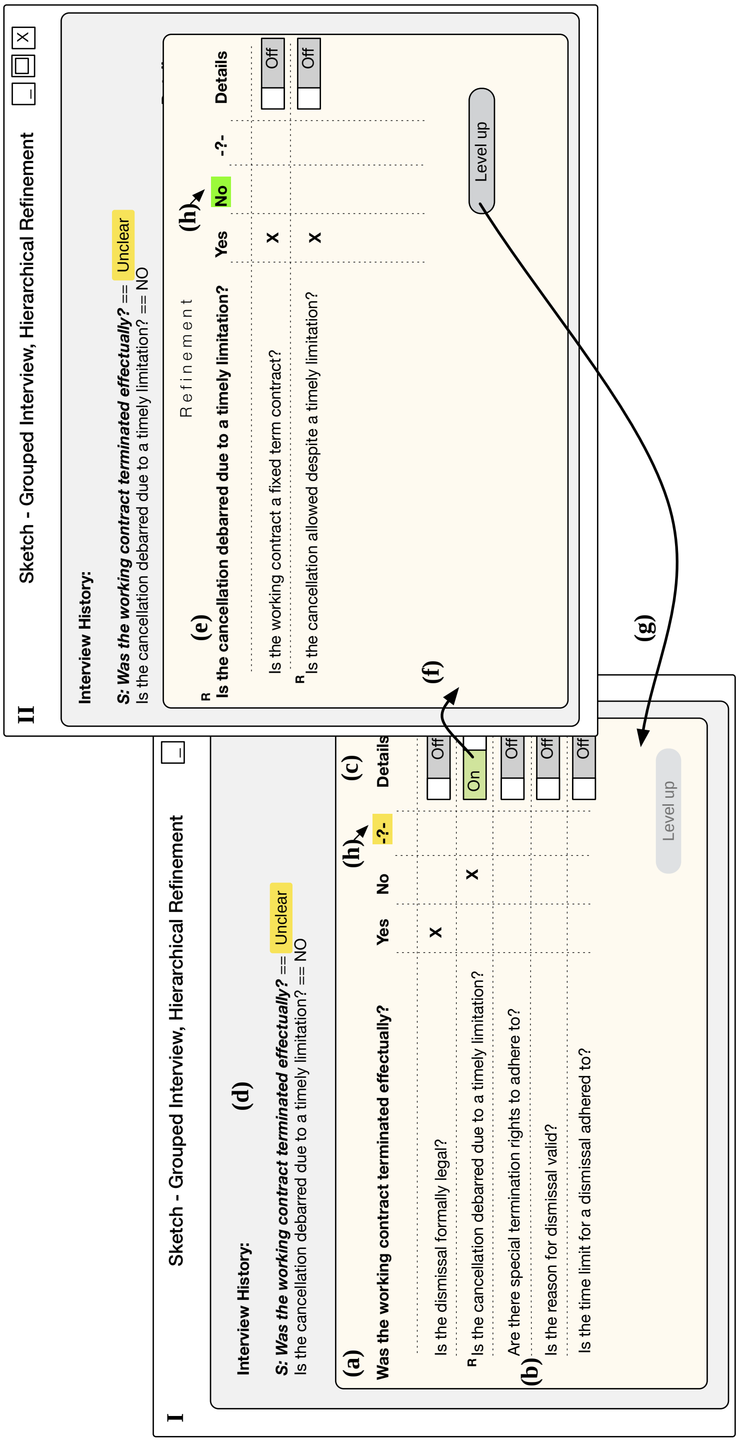
Example: Figure 4.13, p. 69, sketches a Grouped Clarifier Interview. The figure sketches the topmost (I) as well as the first refinement level (II) of such a KB. Each refinement level constitutes a display group (b). Each group display repeats the respective abstract parent question for context—which on the topmost level corresponds to the target solution (a), and on refinement levels to the abstract parent question. Refinement levels are indicated by an activated *Level up* button (Figure 4.13, g) for switching back to the parent level. The current answer states are highlighted directly within the adapted Check List display, see Figure 4.13, h. Further context is provided by an interview history widget (d) which in this case also indicates the final state of the solution/topmost question. Question groups are rendered in an adapted Check List style. Additionally, a column with switch widgets is added (c) which allows the navigation into a refinement level for a question (f).

b. Clarifier Questionnaire

Users: Laymen, experienced, experts | one-time, frequent.

Knowledge: Backward knowledge, i.e., targeting only a single solution.

Solution Specifics: The basic UI solution is similar as proposed for the Questionnaire variants, c.f., Sections 4.3.2.a. to 4.3.2.c.—with the major difference, that only targeted backward knowledge is presented. In the case of gradually refined knowledge, such a KBS extends the base


II
Sketch - Grouped Interview, Hierarchical Refinement
(e)
(f)
(g)
(h)
Refinement
Level up
I
Sketch - Grouped Interview, Hierarchical Refinement
(a)
(b)
(c)
(d)
(h)
Level up

Window II

Interview History:
S: Was the working contract terminated effectually? == Unclear
Is the cancellation debarred due to a timely limitation? == NO

	Yes	No	-?-	Details
R **Is the cancellation debarred due to a timely limitation?**				
Is the working contract a fixed term contract?	X			Off
R Is the cancellation allowed despite a timely limitation?	X			Off

Window I

Interview History:
S: Was the working contract terminated effectually? == Unclear
Is the cancellation debarred due to a timely limitation? == NO

	Yes	No	-?-	Details
Was the working contract terminated effectually?				
Is the dismissal formally legal?	X			Off
R Is the cancellation debarred due to a timely limitation?		X		On
Are there special termination rights to adhere to?				Off
Is the reason for dismissal valid?				Off
Is the time limit for a dismissal adhered to?				Off

Figure 4.13: Example sketch for a Grouped Interview implementation for refinement clarification knowledge.

question presentation by the respective refinement questions flexibly once invoked—potentially grouped as coherent *refinement questionnaire*.

Consequences: Clarifier Questionnaire mostly exhibits the same advantages and downsides as Clarification Interview, see also Section 4.3.5.a.. However, Clarifier Questionnaire provides an even more intuitive interaction regarding refinement knowledge that resembles the standard follow up question mechanism: Inserting such question(s) right in-place when triggered, and equally removing them as soon as the triggering question is changed (from the refine option to a normal answer). Yet, in the case of highly abstracted knowledge—as for example used with the ITree variants in the JuriSearch project, c.f., Section 7.1.4—the resulting UI presentation becomes rather unsteady, and users may loose tracks regarding the current level they operate on and which questions are to be answered when. Also in this case, a structural overview, that displays the rough questioning structure in an Hierarchical Clarifier resembling manner and marks the current state of the interrogation, can be highly valuable.

4.4 Synopsis

We motivated, that the core input module denotes the key essential module of any user input driven KBS. In this chapter, we introduced a collection of basic interaction styles and presentation configuration options for that module. Therefore, we proposed general input presentation dimensions that are applicable independently from specific question- or input data types—e.g., whether or not to number interview objects. On the other hand, we also summarized particular, question type dependent UI widgets, such as radio buttons.

Based thereupon, we defined an initial collection of four basic KBS UI patterns: Questionnaire, Interview, Clarifier, and Clarifier Hybrid. Each of those patterns further can be fine-tuned into several distinct variants. Each such variant then exhibits the same core properties as its base pattern, yet differs from other variants regarding more fine-grained aspects—e.g., the particular UI presentation. As a foundation for a clear delimitation and characterization of such patterns we used the set of tailored, usability-related KBS classification criteria (see Section 3.1.2).

The proposed UI and interaction styles are mainly experience-based. That is, the intention is not to provide an all-encompassing, universal, and finalized collection. Rather, we want to offer a starting point and inspirational source regarding KBS-related UI- and interaction design options as well as some reasonable KBS UI pattern templates. This equally accounts for the *KBS UI Context* presentation options, which are described in the following Chapter 5.

Chapter 5

KBS Context Presentation

Having discussed UI options and dedicated KBS UI patterns for the core input module in the previous chapter, we proceed by describing presentation options for the KBS context. As KBS context, we understand all core KBS aspects that require an own UI representation but that are not the KBS core input module itself. That is, the results, auxiliary information, and justification module.

In the following, we describe various forms of presenting the *results* of a KBS in Section 5.1. This ranges from plain input data summaries to more comprehensive and expressive solution display panels. In Section 5.2 we summarize different forms of integrating *auxiliary information* with a KBS UI. Finally, we consider presentation options for *solution justification* in Section 5.3.

5.1 KBS Results and Solution Presentation

Regarding the results of a KBS session, we distinguish two basic types dependent on the KBS use case (c.f., Section 2.1.3): For documentation systems, mostly more or less sophisticated *summarizing presentations of the entered data* suffice. We introduce some variants in Section 5.1.1. In contrast, consultation systems naturally require the display of derived solutions, typically in the form of a more expressive *solution (presentation) panel*. Configuration options thereof are presented in Section 5.1.2. Yet, depending on the particular application scenario, plain input data summaries can constitute a valuable, additional feature also of consultation KBSs.

5.1.1 KBS Input Data Summary Presentation

For presenting KBS input data summaries, we suggest three basic variants: *Plain summary*, *visually enhanced summary*, and *statistically redacted data analysis*.

a. Plain Summary

Plain summaries are the most basic form of presenting KBS input data. In the simplest variant only the entered findings are listed. Figure 5.1, a, p. 73, shows the plain summary as implemented in the EuraHS project (see also Section 7.1.3). Such a representation can be tweaked by indicating some more information regarding *abstraction questions*, *the input sequence*, and the *input coherence*.

- *Abstraction questions*: As those questions are not answered explicitly by the user but are derived by the KBS, it might be helpful in some cases to indicate that special status for immediate recognition. For example, by using a slightly different presentation style. Or by summarizing such questions within a particular own section of the listing as depicted in Figure 5.1, a, p. 73, *Abstractions*.

- *Input sequence*: Plain finding listings can be sequenced either according to the original sequence as defined in the KB, shown in Figure 5.1, a, p. 73. Or according to the actual answering sequence of the user—which particularly in explorative KBS types can vary drastically from the original defined sequence.
- *Input coherence*: Especially in the case of rather extensive documentation systems, where large amounts of data need to be entered, an input session may not be finished at once but be spread over several separate input sessions. There, it might be valuable to indicate, which questions have been entered in which input session—e.g., by grouping the findings according to the entry date.

b. (Visually) Enhanced Summary

For increasing its comprehensiveness, firstly the plain summarizing listing as introduced in Section 5.1.1, a., can be enhanced by additional features, such as *search/sorting* facilities, or *derivation justifications*:

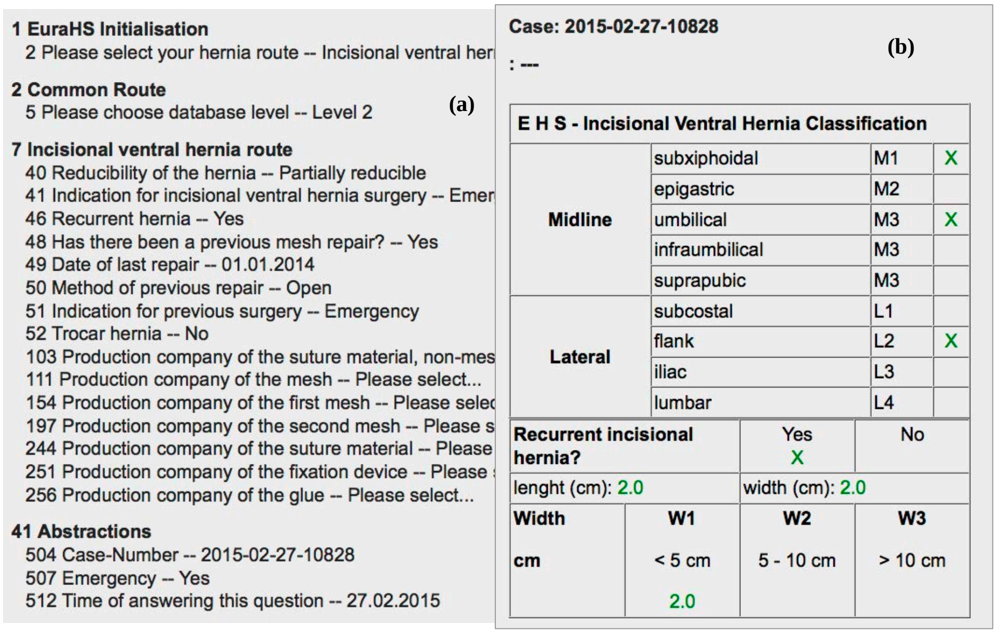
- Especially when large amount of input data are entered, *search and sorting facilities* can alleviate the task of quickly identifying specific items, such as particular questions.
- *Justification facilities* enable users to immediately investigate how a certain interview object—such as a follow up- or abstraction question—has been derived by the KBS. Such items, for example, can be distinctly highlighted and offer an explanation popup on click or mouseover.

Instead of finding lists, also generally more visual representation forms can be used. One example, is shown in Figure 5.1, b. Here, a grid-based presentation frame is used, that is filled only with selected input data. Those grids are oriented at paper-based forms, the users—in the presented case: Medical doctors—are familiar with from their daily routines. Thus, those compact visual summaries ease the perception of the most relevant information regarding particular framing conditions at a single glance.

c. Statistically Redacted Analyses

Statistically redacted analyses provide the most potential of presenting comprehensive data and corresponding interrelations. Therefore, the input data is first preprocessed by more or less complex statistical calculation methods—according to the requirements of the desired output—and subsequently made available to the user. As nearly unlimited possibilities of interesting analyses—and corresponding, reasonable presentation forms, such as spreadsheets, graphs, or other visuals—exist, a detailed description is out of the scope of this work. To give just an intuitive example, one such analysis could simply cover the mean value evaluation of all interview objects. This provides users with an overview, with which frequency choices of OC or MC questions were provided, or which is the mean value for a numerical question, etc.

An example of statistically enhanced results presentation was implemented in the EuraHS project (see also Section 7.1.3). There, the data analyst and representatives of the target users



(a)

1 EuraHS Initialisation
2 Please select your hernia route -- Incisional ventral her

2 Common Route
5 Please choose database level -- Level 2

7 Incisional ventral hernia route
40 Reducibility of the hernia -- Partially reducible
41 Indication for incisional ventral hernia surgery -- Emer
46 Recurrent hernia -- Yes
48 Has there been a previous mesh repair? -- Yes
49 Date of last repair -- 01.01.2014
50 Method of previous repair -- Open
51 Indication for previous surgery -- Emergency
52 Trocar hernia -- No
103 Production company of the suture material, non-mes
111 Production company of the mesh -- Please select...
154 Production company of the first mesh -- Please selec
197 Production company of the second mesh -- Please s
244 Production company of the suture material -- Please
251 Production company of the fixation device -- Please
256 Production company of the glue -- Please select...

41 Abstractions
504 Case-Number -- 2015-02-27-10828
507 Emergency -- Yes
512 Time of answering this question -- 27.02.2015

(b)

Case: 2015-02-27-10828

: ---

E H S - Incisional Ventral Hernia Classification			
Midline	subxiphoidal	M1	X
	epigastric	M2	
	umbilical	M3	X
	infraumbilical	M3	
	suprapubic	M3	
Lateral	subcostal	L1	
	flank	L2	X
	iliac	L3	
	lumbar	L4	

Recurrent incisional hernia?		Yes X	No
lenght (cm): 2.0		width (cm): 2.0	
Width	**W1**	**W2**	**W3**
cm	< 5 cm 2.0	5 - 10 cm	> 10 cm

Figure 5.1: Two presentation options for summarizing KBS input data: Plain summary (a) and visually enhanced summary, grid-based implementation (b). Both examples are current implementations from the EuraHS project, see also Section 7.1.3.

(medical doctors) defined a collection of relevant statistical analyses. This collection is easily accessible in the EuraHS system, so the user can choose to investigate any such analysis anytime. The corresponding data processing is entirely done in the background. The user is finally provided with a downloadable spreadsheet containing the respective analysis results.

5.1.2 KBS Solution Panel Configuration Options

In the following, we present some foundational presentation dimensions that can be combined flexibly for fine tuning the appearance of dedicated solution panels. This covers exclusively options regarding the solutions themselves. Justifications, i.e., explanations why and how the KBS derived and rated the solutions, are discussed separately in Section 5.3.

Table 5.1 summarizes the suggested general presentation dimensions with corresponding display options for solution display. The most basic configuration—*Solution Range*—regards the amount of presented solutions.

- *Multiple Solutions*: More than one solution is presented simultaneously.
- *Single Solution*: Typically, the solution with the *best* rating is displayed. Yet, it is not always feasible to design the KB such that only a single solution is derived unambiguously. Thus, additional conventions are required how to decide for *a best* solution out of a set of equally rated items. Examples include:

- The solution which attained the respective rating *first*.
- The solution with the strongest *rating coherence*: A solution that received its entire rating by a single knowledge item—e.g., one single rule—is regarded more relevant than a solution that was derived by many different knowledge items (potentially, with weaker scorings).
- The solution which *covers the most symptoms*: A solution that covers many different symptoms is rated more relevant than a solution derived only by one or two particular symptoms.

Presentation Dimension	Display Options
Solution Range	Single Solution — Multiple Solutions
Both Multiple & Single Solution	
KBS Integration	Final Display — Anytime Display
Rating Granularity	Abstract Rating — Precise Rating — Full Rating
Auxiliary Information Integration	Anytime All — None
Multiple Solutions	
Deduction Procedure	Differential Diagnosis — Diagnosis of Exclusion
Solution Granularity	Show All — Show Specified
Solution Ordering	Categorical — Alphabetical — Numerical — Mixed

Table 5.1: Basic solution presentation options. *Solution Range*, the most basic aspect, differentiates the number of solutions to be displayed. *KBS Integration*, *Rating Granularity*, and *Auxiliary Information Integration* can be applied to both Single Solution and Multiple Solutions display. *Deduction Procedure*, *Solution Granularity*, and *Solution Ordering* are only reasonable for multiple solutions display.

Presentation Dimensions—Single & Multiple Solution Range The dimensions *KBS Integration*, *Rating Granularity*, and *Auxiliary Information Integration* can be applied both to single solution and multiple solution display.
KBS Integration refers to the point in time, when the solution panel is available to the user.

- *Final Display*: Solutions are shown not before the end of the KBS session. Providing the results only at session end is especially valuable for rather inexperienced users that need more support from the KBS overall. Those might be easily distracted by a steadily changing anytime solution panel, and thus profit better from concentrating exclusively on data entry first and exploring corresponding results afterwards.
- *Anytime Display*: The solution panel is displayed anytime during the session in a fixed UI part and presents solutions always with their most recent rating. This variant is especially valuable in case users want to explore the system more freely and require immediate results or at least tendencies. This is the case particularly in the debugging use case, where anytime feedback on the consequences of answering a question is a key demand. Anytime solution display is in principle applicable for all different KBS UI types; yet, Hierarchical Interview (Section 4.3.3.c.) denotes (partly) an exception. There, a final solution panel is not required at all as solutions are displayed directly within the core

UI as additional, distinct nodes. An anytime solution panel, however, can help in case users desire an anytime update regarding the remaining possible or entirely excluded solutions.

Rating Granularity addresses the precision for presenting the current state of a solution.

- Precise Rating: An exact number correspondent of the solution state that is computed internally by the KBS. For example, heuristic rules of [d3web, n.d.] KBs internally use fixed numerical values for adding up the rating (class) of each solution—e.g., the most positive abstract rating *P7* maps to the numerical value of +999. The display of that precise number—alone, or within a *Full Rating* presentation, see below—is probably most interesting for knowledge engineers who use a KBS UI application for debugging the KB. That allows for investigating whether all rules apply the correct values to a solution state and thus lead to valid final ratings. Apart from that, also common users might prefer a precise number over abstract rating classes: The semantics of numbers (high, low, negative) are intuitively clear and thus probably always understandable. In contrast, the understandability of abstract rating categories, such as *N5*, depends on their particular naming, presentation, and explantation.
- *Abstract Rating*: The precise solution scores often are additionally mapped to an abstract category. The mapping for [d3web, n.d.] KBs encompasses the solution rating categories *established*, *suggested*, *excluded*, or *unclear*. Displaying such an abstract rating state can accelerate and ease the perception of the final results. There exist several ways for presenting abstract ratings: To explicitly name the category in textual form, or to exploit visual means. Examples for the latter are varying text size, using tailored background coloring, or adding icons for the categories—see also Figure 5.2, p. 76.
- *Full Rating*: Combines both precise rating value and abstract rating category. This is shown in Figure 5.2, p. 76, where both circular icons represent the basic rating class, and the precise rating value is added in parentheses.

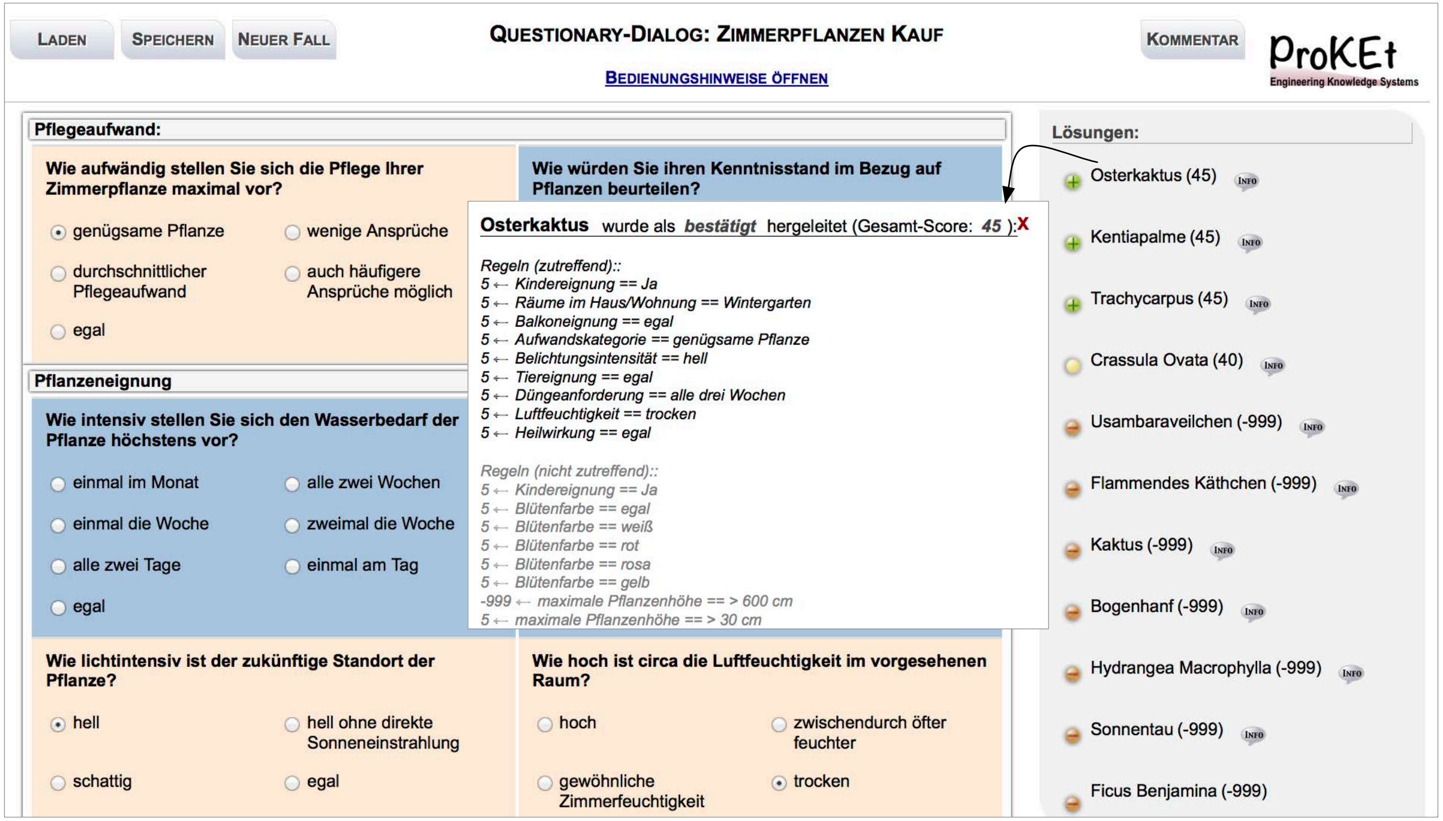
Auxiliary Information Integration refers to which add-on information are included in what way within the solution panel.

- *Anytime All*: Auxiliary information are made available directly from the solution panel at anytime. The implementation shown in Figure 5.2, p. 76, for example, provides the respective information when the info-icons next to a solution are clicked.
- *None*: In theory, it is also possible to provide none additional information. Yet, especially regarding complex KBS domains we recommend to provide the *anytime all* variant for solution presentation per default. Here, those are highly valuable for clarifying the meaning of the respective solution or of potential further action in more detail—and thus strongly add to the overall KBS comprehensibility.

Presentation Dimensions—Multiple Solutions Only For multiple solution display, also the additional presentation dimensions *Solution Granularity*, and *Solution Ordering* apply.

Solution Granularity specifies the solution classes included in the presentation.

- *Show All* specifies, that all solutions—including those with rating state *unclear*—are to be included in the solution display, see Figure 5.2, p. 76.



QUESTIONARY-DIALOG: ZIMMERPFLANZEN KAUF

BEDIENUNGSHINWEISE ÖFFNEN

Pflegeaufwand:

Wie aufwändig stellen Sie sich die Pflege Ihrer Zimmerpflanze maximal vor?

[✓ genügsame Pflanze] ○ wenige Ansprüche ○ durchschnittlicher Pflegeaufwand ○ auch häufigere Ansprüche möglich ○ egal

Wie würden Sie ihren Kenntnisstand im Bezug auf Pflanzen beurteilen?

Pflanzeneignung

Wie intensiv stellen Sie sich den Wasserbedarf der Pflanze höchstens vor?

○ einmal im Monat ○ alle zwei Wochen ○ einmal die Woche ○ zweimal die Woche ○ alle zwei Tage ○ einmal am Tag ○ egal

Wie lichtintensiv ist der zukünftige Standort der Pflanze?

[✓ hell] ○ hell ohne direkte Sonneneinstrahlung ○ schattig ○ egal

Wie hoch ist circa die Luftfeuchtigkeit im vorgesehenen Raum?

○ hoch ○ zwischendurch öfter feuchter ○ gewöhnliche Zimmerfeuchtigkeit [✓ trocken]

Osterkaktus wurde als ***bestätigt*** hergeleitet (Gesamt-Score: ***45***):X

Regeln (zutreffend)::
5 ← Kindereignung == Ja
5 ← Räume im Haus/Wohnung == Wintergarten
5 ← Balkoneignung == egal
5 ← Aufwandskategorie == genügsame Pflanze
5 ← Belichtungsintensität == hell
5 ← Tiereignung == egal
5 ← Düngeanforderung == alle drei Wochen
5 ← Luftfeuchtigkeit == trocken
5 ← Heilwirkung == egal

Regeln (nicht zutreffend)::
5 ← Kindereignung == Ja
5 ← Blütenfarbe == egal
5 ← Blütenfarbe == weiß
5 ← Blütenfarbe == rot
5 ← Blütenfarbe == rosa
5 ← Blütenfarbe == gelb
-999 ← maximale Pflanzenhöhe == > 600 cm
5 ← maximale Pflanzenhöhe == > 30 cm

Lösungen:

- Osterkaktus (45)
- Kentiapalme (45)
- Trachycarpus (45)
- Crassula Ovata (40)
- Usambaraveilchen (-999)
- Flammendes Käthchen (-999)
- Kaktus (-999)
- Bogenhanf (-999)
- Hydrangea Macrophylla (-999)
- Sonnentau (-999)
- Ficus Benjamina (-999)

Figure 5.2: An exemplary solution panel configuration, ProKEt reference implementation: Multiple solutions of all solution classes are shown in the anytime solution side panel. Each solution offers both an abstract rating (icon) and a precise score. Descending strict numerical ordering, i.e. also implicit categorical ordering of the solution listing, is applied. Integrated auxiliary information features use the anytime all variant: Additional information—e.g., explaining specialist terms—for solutions are retrieved via the info-icon.

- *Show Specified* displays only the set of solutions that falls into the specified one or several categories. Such categories may be abstract, such as *established solutions only*, or precise numbers, such as *solution value* ≥ *50*. Depending on the particular configuration, this option exhibits different semantics: On the on hand, displaying solutions that become rated positively during the session means, that those become potential candidates for the final solution(s). On the other hand, presenting those solutions that become excluded implies, that those become excluded from the set of potential final solution candidates. The latter variant can be included as anytime status update in the Hierarchical Interview UI style (Section 4.3.3.c.) for indicating clearly that choosing a certain branch consequently excludes the displayed items.

Solution Ordering refers to the way solutions can be sequenced or grouped.

- *Numerical*: Solutions are ordered according to the precise rating value. The ordering sequence can be both ascending or descending. The descending variant, that presents the solution with the highest rating first, probably is the most common one. An example is shown in Figure 5.2, p. 76. This is due to the fact, that often, users naturally are interested in identifying the *best* solution(s) in terms of the highest score, which is easily achieved by a numerical, descending order.
- *Categorical*: Groups the displayed solutions by their abstract rating category, e.g., established/suggested/excluded/unclear in [d3web, n.d.]. Again, the ordering sequence can be both descending (starting with *established solutions*) or ascending. As it might confuse users if a set of solutions, that are all rated by the same basic rating category, are displayed in an implicitly mixed-up manner (alphabetical- or scoring-based), it is highly advisable, to second-order the abstract categories: As one variant, simply the fact can be exploited that a strict numerical order leads to a score-based 'ordered categorical' display anyhow. Another *mixed* variant is *Cat-Alphabetical*, as described below.
- *Alphabetical*: Displays the solutions in alphabetical order. The sequence again can be both ascending (starting with 'A/a') or descending. This variant is appropriate, if users demand a rather fixed display. That is, if they do not want solution items to change their place during the interrogation and corresponding rating process. The latter is the normal case for numerical ordering, where solutions might be re-ordered after each entered interview item.
- *Cat-Alphabetical*: Displays the solutions basically grouped by their (abstract) rating state, but further orders each such group alphabetically. This form is appropriate, when users are not so much interested in precise rating values, but more in the abstract rating class, yet require to spot out solutions quickly based on their naming and thus demand an alphabetical sub-ordering.

5.2 Auxiliary Information Types

We propose two types of auxiliary information:

- *Local*: Enhancing the understandability of KBS core interview objects
- *Global*: Enhancing the entire KBS usage experience

5.2.1 Presenting Local Auxiliary Information

Several forms of auxiliary information can increase the understandability of core KBS interview objects, such as questions, answer options, or solutions. Regarding specifically solutions, this exclusively concerns additional information on the solution object itself, but *not* its justification (which is discussed in Section 5.3). The respective auxiliary information naturally is different for each interview object, depending on the particular content type. As example, one question might require an image for visual explanation whereas other questions can be sufficiently explained textually. In contrast, the display integration mode typically is equal for all interview objects of the same class. For example, popup display on mouseover for all questions, but popup display on mouse click for all solutions.

Auxiliary information types for core interview objects encompass:

- *Textual explanations*: Those are beneficial for better addressing the language of non-expert users in case many special terms are used. When integrated systematically for all special terms, this can result in an interactive lexicon. This not only can drastically enhance the correct understanding and usage of the KBS, but additionally foster the system's learnability and skill-building ability.
- *Hyperlinks*: Integrating hyperlinks to dedicated webpages conforms to [Duan et al., 2005], who already stated that explanation can be enhanced by linking to further relevant internet sources.
- *Supporting Media*, including:
  - *Documents*: Those may contain further official or non-official information, and can be incorporated to be viewed in a static manner or be offered for download. As example, the legal consultation system implemented in the JuriSearch project (see Section 7.1.4) provides form templates for certain legal claims, both for immediate in-browser viewing and downloading.
  - *Visuals*: E.g.,images, graphs etc. Sometimes, the integration of more comprehensive image questions may not be practicable. Then, particularly in highly specialist domains, such as medicine or engineering, the provision of a static image leastwise can help clarify answer options or solutions along with advice for further actions.
  - *Videos*: Those can provide information where static images do not suffice.

Auxiliary information display can be configured with respect to the particular integration form within the UI and its triggering mode.

- *Fixed display, show automatically*: As soon as any auxiliary information is available for an interview object, it gets displayed automatically in a dedicated, fixed part of the UI. This is used as the default variant in the reference implementation of the Strict Interview pattern, see Figure 4.7, p. 59.
- *Fixed display, show on request*: Auxiliary information is per default shown in a fixed, dedicated part of the UI if—and only if—requested by the user. The request can be triggered by click or mouseover (both, either on the interview object itself or a dedicated icon). The ITree implementation, described in Section 4.3.4.a., uses a variant where hovering the question text triggers the auxiliary information presentation in a particular side panel,

see Figure 4.11, p. 66. Alternatively, the fixed display in this variant could also be faded in/out when triggered. Yet in contrast to popup display, here still only one single auxiliary information widget is used for presentation, whereas in popup display, each interview object 'owns' its own auxiliary information popup.

- *Popup display, show on request*: Similar to the above variant, the request is triggered by the user. Yet, the information is not presented in a dedicated, fixed part of the UI, but in a popup that is placed relatively to the corresponding triggering element. The Box Questionnaire implementation in the EuraHS project (see Section 7.1.3), for example, applies hovering dedicated icons as trigger for opening the corresponding popup. Yet, explicitly clicking info icons is equally well feasible as trigger, here. It has to be noted, that regarding popup display in general an automated display of auxiliary information is not reasonable. Then, all popups for each interview object would be displayed simultaneously, leading to an illegible, unintelligible, and probably non-usable UI.

⟶ Regarding the display of *auxiliary information particularly for solution objects*, the automated presentation in a fixed display field is best feasible for single solution configurations. An example is the solution panel style implemented for the ITree KBS, see Figure 4.11, p. 66. However, as soon as multiple solutions are derived (and presented) simultaneously, an automated insertion of auxiliary information within one single widget would lead to visual clutter and an unintelligible, and thus useless, presentation. Thus, in multiple solution configurations, the popup display variant that is triggered only on request is most appropriate.

Auxiliary information related to core interview objects are basically valuable for all KBS UI types alike. This is due to the fact, that most often several (equally reasonable) ways exist to name or explain objects, and chances are rather low that the form chosen by the KBS developer will satisfy all potential users likewise. There, auxiliary information serves as additional clarifier, fostering a better understanding of the meaning of the interview objects and thus increasing the KBS comprehensibility. In turn, this supports a more correct result of the KBS session, as users most likely answer more precisely the better they understand all corresponding items.

5.2.2 Presenting Global Auxiliary Information

Several forms of auxiliary information can help to enhance the entire user experience with a KBS UI. As some basic valuable features, we suggest *Interview Objects History*, *Interview Progress Info*, *Instructional Summary*, and *Interaction Affordances*.

Interview Objects History *A list of all currently processed interview objects.*
Such a history should not only plainly list the questions, but optimally the entire findings, i.e., questions together with the selected answer. That way, a quick overview is provided, not only with respect to which questions are answered but also in what way. At the same time, this can serve as a rough, anytime justification. A sample implementation of such a findings-based interview objects history can be viewed in Figure 4.7, c, p. 59. In case questionnaires or other groupings are used, those can be additionally displayed for clarity and overview.

In an interactive variant, the list entries can serve as back-links to the respective interview objects—valuable, e.g., in case the user notices an input mistake and quickly wants to navigate back to correct it. This can be highly beneficial especially for conversational UIs, such as the Interview pattern, where only a single question is shown at a time. The same accounts for Questionnaire- or hierarchical style implementations using comprehensive KBs, where the available UI space only can present a smaller portion of the KB.

Interview Progress Info: *A textual and/or visual representation regarding the overall progress of the current KBS usage session.*

Progress information is a valuable add-on for nearly any kind of KBS. An exception are entirely explorative KBS types, such as Hierarchical Clarifier (Section 4.3.4.a.). There, fixed interrogation paths are deliberately not defined. Also, optimally many to all items can be investigated at a glance, thus also mirroring the input progress Thus, an additional, explicit progress calculation is neither feasible nor required.

Regarding strict decision tree knowledge, the progress calculation is quite straightforward: It corresponds to all interview items that are not contained in or are dependent from the currently active questioning branch. In other cases, however, the appropriate progress estimation can be more intricate. This is due to the fact, that not necessarily clearly exclusive interrogation paths exist as in decision tree knowledge. For example, questions may exist, that potentially are follow-up questions of a question in a currently active branch—yet, that have nevertheless become deactivated (=irrelevant) due to already provided answers.

Two basic values are easily retrievable from a KBS: The number of already answered questions, and the number of all questions in the KB. Deriving the progress only based thereupon poses the problem, that oftentimes not all remaining (= non-answered) questions are required for finishing a session. Thus, the calculated progress based on answered items is often too low. In contrast, we suggest to approximate the KBS progress by adding up all answered questions with those questions, that currently definitely are irrelevant for the session.

This is formally expressed by the following formula:

$$Q_{processed} = \frac{Q_{answered} + Q_{nonindFUs} + Q_{contra}}{Q_{totalKB}} \tag{5.1}$$

where:

- $Q_{totalKB}$ → The total number of all questions in the KB.
- $Q_{answered}$ → The number of all answered questions in the session so far.
- $Q_{nonindFUs}$ → The number of questions that are follow up questions of already answered questions but are *not* indicated (e.g., due to specific rules).
- Q_{contra} → explicitly contra-indicated questions (e.g., explicitly deactivated by rules). In case questionnaires are used for grouping, this includes all questions of contra-indicated questionnaires except those questions have been directly indicated by further knowledge.

Figure 5.3 sketches some variants for displaying KBS progress information. Variant (a) displays only a single value: The plain number of already answered items. As already described, the number of answered questions can vary quite drastically from the actual progress, if answered questions exclude larger parts of the remaining KB. Thus, this straightforward solution is feasible only in case the number of answered items is more relevant to the user than the relative overall progress. Alternatives (b) and (c) in contrast present both the number of answered items and the estimated overall progress, derived by formula (5.1).

A final factor regarding the appropriateness of any progress display are the user characteristics: Whether or not they accept only an entirely precise forecast, including the possibility that this value might decrease at some point—e.g., when large KB branches are re-activated by certain input that had been excluded beforehand. Or whether users prefer an approximated, yet potentially slightly inaccurate forecast that steadily increases.

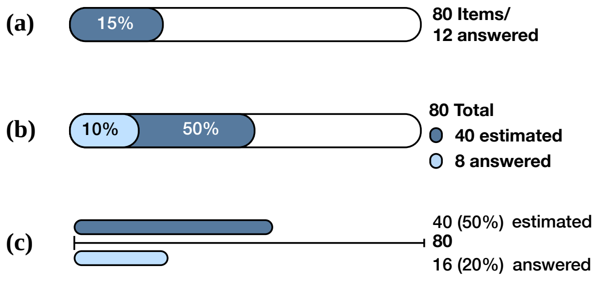


Figure 5.3: Progress information variants: Answered items only (a) and both answered items & estimated total progress (b & c).

Instructional Summary: *A pointed explanation of the KBS's usage and core functionality.*

The correct usage and data input is of core importance for achieving appropriate results with a KBS. There, the provision of an instructional summary, that explains the KBS target intention and its core features in a pointed, easy to understand manner, is essential. Regardless of whether a comprehensive, detailed documentation, tutorials, or other materials exist, such a summary should be easily and anytime accessible as a shorthand reference. This can be realized by including a corresponding button widget or link prominently into the main KBS UI, that can be clicked anytime during the KBS session and triggers such information. The first variant (button) was used in the ITree implementation, see, e.g., Figure 4.11, p. 66, *Hilfe [Help]*-Button on the upper right. Other reference implementations in this work include a dedicated link underneath the header, see, e.g., Figure 5.9, p. 91.

Interaction Affordances: *Specific interactive presentation forms of what to do / where to click.*

In case, special UI representations are used or non-standard interactions are required, interaction affordances can lower the overall, particularly initial, hurdle of coping with the system. A good example is the Hierarchical Interview style—or likewise, the Hierarchical Clarifier. For one-time or first-time users, it might not be obvious that the presented lengthy nodes actually are clickable—which however denotes the key interaction for using those systems. There, a recognizable change of the mouse cursor can indicate this behavior in an unobtrusive but straightforward way. Similarly, the importance of the instructional summary can be highlighted by changing the respective link or button when hovering with the mouse and thus catching users' attention.

5.3 KBS Justification Presentation

With an ever-growing complexity of KBS and their reasoning abilities, the seminal statement of [Hendler, 1988]—basically saying that an ES must not only present conclusions, but must also be able to explain how those were reached—is more relevant than ever. As summarized in Section 2.1.1, Figure 2.1, we also motivated a dedicated explanation module as a core part of a KBS. In the following, we refer to this as *justification* module, as to differentiate clearly from the plain auxiliary information explanations that may be integrated additionally for any interview object. A justification module provides the relevant information about *how* the current state of a KBS object was derived. Naturally, such information is relevant for solutions, yet it concerns follow up questions and abstraction questions/values likewise—i.e., every interview item or KBS state that is derived by the KBS in some way.

Traditionally, KBS modules are considered as rather self-contained units with separate responsibilities. That is, the core input module handles the data input interaction, a solution panel displays the currently derived solutions, and a justification module creates and presents justifications for desired solutions. Regarding basic justification presentation forms that are only invoked in the end of a session for rather static investigation, we present a selection of interesting variants in Section 5.3.1. Furthermore, we argue that users can strongly benefit from interactively enhanced justification presentation forms where they can immediately and easily investigate the consequences of changing selected input regarding the target solution. We discuss several such consultation-justification mash ups as *interactive justification presentation forms* in Section 5.3.2.

5.3.1 Static Base Justification Presentation

Verbalizations—i.e., a textual, more or less formal description of the relevant knowledge—are the most straightforward justification technique. Yet, with an increasing number of presented elements and corresponding interrelation complexity, more visual techniques denote equal, if not better, presentation variants.

Information Visualization for KBS Justification In general, *information visualization* is known to reveal insights regarding the represented data. Consequently, it supports discovery, decision making, and explanation, c.f., [Card et al., 1999, p. 6]. Chances are, that visualization can help to display large data sets in a more intuitively explorable manner, to ease the perception and the formation of a mental model of the data, and thus to foster its understanding. Therefore, we suggest that tailored, adapted visualization techniques offer a wonderful chance to enhance the understandability of KBS justifications.

Visualization has been successfully applied in several sub-disciplines of AI research. Most 'traditionally', it is used in the field of data analysis and data mining. Examples include the foundational works of [Tufte, 1986], or [Card et al., 1999]. More recent publications that apply visualization in practice include [Blanchard et al., 2007], or [Subasic and Berendt, 2010]. An approach for creating visualizations for rule bases has been proposed in [Zacharias, 2007]. Also, visualization has proven valuable for supporting the validation of KBs, see [Baumeister and Freiberg, 2010].

Regarding particularly the KBS justification presentation, however, visualization still is rather unexplored. Even though it is partly related to the above research efforts, a major difference concerns the fact, that KBS justifications are not solely intended for experts that develop and validate KBs. In point of fact, especially also end users with little domain knowledge can profit from justifications that explain the KBS's way of operation—in case those are presented appropriately. Therefore, justification visualizations need to unite both the core, necessary information as well as an intuitively understandable, clear presentation.

In general, a vast diversity of techniques exists that can reasonably be adapted for justification presentation. Thus, an encompassing overview and categorization is out of scope of this work. Rather, we present a selection of some interesting techniques—evolved based on the requirements of actual projects—as a foundation and inspirational source. It has to be noted that the suitability of any justification presentation technique always also depends on the underlying knowledge formalism. Due to the focus of our research, the presentation forms proposed subsequently are mostly apt for rule-based or set covering knowledge.

Table 5.2 classifies the proposed static justification techniques regarding their basic type (textual or visual) and the suitable knowledge type (rule-based or set covering knowledge). Sections 5.3.1, a.–e. then introduce the justification presentation techniques in more detail. Two more options, that can be applied for all presentation variants in this section, are discussed in the end of this section, p. 89, *Static Justification Presentation—General Considerations*.

	Presentation Type		Knowledge Type	
	Textual	Visual	Rule-based	Set Covering
a.. Finding List	X	-	X	X
b.. Rule Verbalization	X	-	X	X
c.. FS Tree	-	X	X	(X)
d.. Justification TreeMap	-	X	X	(X)
e.. Rule Graph	-	X	X	-

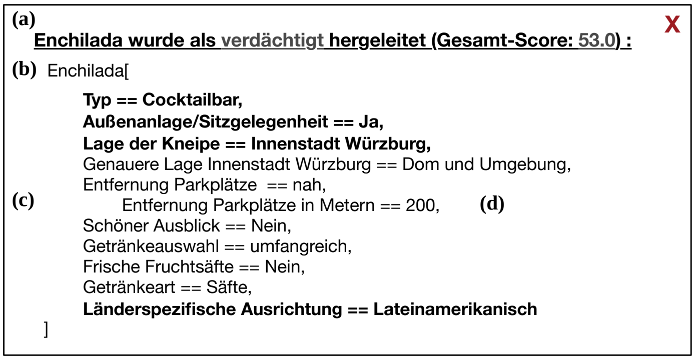
Table 5.2: Overview: Static justification presentation techniques.

a. Finding List (Textual)

The most simple yet intuitively understandable form of justification presentation probably is a finding list. The plain variant simply lists all findings related to the respective solution rating. Such an example is shown in Figure 5.4, p. 84, which sketches the reference implementation of finding lists in ProKEt. A general header (a) underlines the current state of the justified solution. Therefore, it lists the solution as well as its abstract and precise rating. The justification itself repeats the solution shortly again (b) before all relevant findings as contained in the KB are listed (c). The example shows findings based on the set covering knowledge from [d3web, n.d.] which implements *diagnostic scores* as defined in [Puppe, 2000, p. 10]. Assembling a finding list based on rules is equally feasible. Therefore, relevant findings simply need to be extracted from the contributing rules in a preprocessing step.

As default variant—depicted in Figure 5.4—we propose to include all potentially contributing findings, and to highlight those findings that were selected and thus actually contributed to the investigated rating in bold face. As one general alternative, it is also possible to list exclusively those findings, that currently contribute to the solution and omit non-selected but potentially contributing items completely. A third variant is to display all findings that are contained in the KB and to highlight the actually contributing ones. As such a latter listing can easily grow complex and lengthy in larger KBs, additionally the potentially contributing items may be marked separately.

The presentation can be further enhanced by adding some information regarding *how* the respective findings contribute to the solution. This can be indicated textually, by adding corresponding icons, or background-coloring of the finding. Depending on the underlying knowledge, potentially even further information can be integrated: For example, when working with covering knowledge from the [d3web, n.d.] toolkit, also the relative importance i.e. weighting of a finding regarding the solution could be included. Or information whether a finding is sufficient to exclude or establish a solution.


(a)
Enchilada wurde als verdächtigt hergeleitet (Gesamt-Score: 53.0) :

X

(b) Enchilada[

Typ == Cocktailbar,
Außenanlage/Sitzgelegenheit == Ja,
Lage der Kneipe == Innenstadt Würzburg,
Genauere Lage Innenstadt Würzburg == Dom und Umgebung,
Entfernung Parkplätze == nah,
(c) Entfernung Parkplätze in Metern == 200, (d)
Schöner Ausblick == Nein,
Getränkeauswahl == umfangreich,
Frische Fruchtsäfte == Nein,
Getränkeart == Säfte,
Länderspezifische Ausrichtung == Lateinamerikanisch
]

Figure 5.4: Finding list example of ProKEt, based on covering knowledge: The header (a) displays basic information regarding solution and rating. All findings potentially related to the solution are listed (c), the ones that contributed to the current solution state are highlighted in bold print.

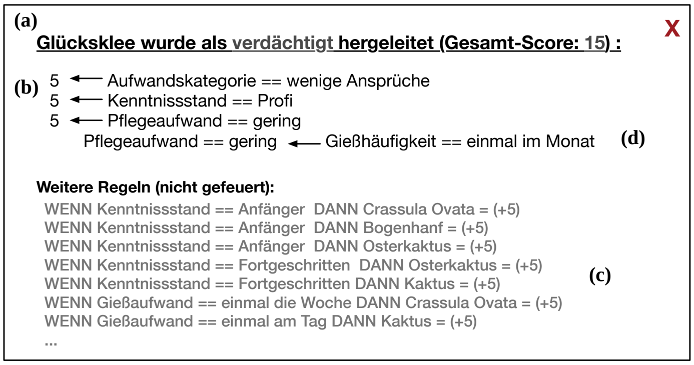
Further, also the integration of either the abstract, or the precise solution rating value, or both, denotes a presentation option. And finally, Finding Lists can vary regarding whether or not they apply an indented presentation of abstraction sources—i.e., those findings that firstly derive an intermediate (abstraction) question which in turn contributes to the investigated solution. Figure 5.4, d, shows an example, where the finding *Entfernung Parkplätze in Metern = 200* [distance to parking space in meters = 200] derives the abstraction value *Entfernung Parkplätze = nah* [distance to parking space = nearby]; thus, the abstraction source—the former question—is indented.

b. Rule Verbalization (Textual)

Another textual presentation form, originally created for rule-based knowledge but also adaptable to other knowledge types, is the rather straightforward textual rule verbalization. Rules are presented either exactly as formalized in the KB or in a slightly adapted, more intuitively readable manner. Thereby, it is possible to start by listing the rule action—e.g., rating of the respective interview object—and to list the corresponding rule condition(s) afterwards, c.f., Figure 5.5, b, p. 85. This assigns a bit more weight to the rule action part, thus highlighting, what actually is the outcome of that rule. However, the other way round—starting with the conditions and adding the consequential action at the end—is equally well possible, see Figure 5.5, c.

Figure 5.5 sketches basic presentation options: The header part again displays basis information—the name of the solution, its abstract rating, as well as its precise rating score. In the simple base variant, the section beneath provides only a reduced verbalization of the rules (Figure 5.5, b). This mainly corresponds to a rule-based finding list, see also previous sections, when equipped with additional information (in this case, the rule action). However, for supporting also expert users or knowledge engineers optimally (e.g., in validating the KB), also a more verbose presentation of the rules—e.g., strictly mirroring their actual syntax as defined in the KB—may be required.

As general alternatives, similar considerations as for the Finding List apply: Either only actually contributing rules an be listed for a quick, compact justification. Alternatively, also potentially contributing but currently not yet selected rules may be included, e.g. required, when using the rule verbalization as foundation for a Form Add-on Clarifier KBS UI. And finally, potentially also the display of the remaining rules, which do not and will never contribute to the solution is possible. For clarity, currently non-contributing (or also never-contributing) rules may be grouped in separate section(s), shown in Figure 5.5, c. Also, such sections should allow its interactive removal/integration on user request, e.g., by clicking a button or similar.


(a)
Glücksklee wurde als verdächtigt hergeleitet (Gesamt-Score: 15) : X

(b)
5 ← Aufwandskategorie == wenige Ansprüche
5 ← Kenntnissstand == Profi
5 ← Pflegeaufwand == gering
Pflegeaufwand == gering ← Gießhäufigkeit == einmal im Monat (d)

Weitere Regeln (nicht gefeuert):
WENN Kenntnissstand == Anfänger DANN Crassula Ovata = (+5)
WENN Kenntnissstand == Anfänger DANN Bogenhanf = (+5)
WENN Kenntnissstand == Anfänger DANN Osterkaktus = (+5)
WENN Kenntnissstand == Fortgeschritten DANN Osterkaktus = (+5)
WENN Kenntnissstand == Fortgeschritten DANN Kaktus = (+5) (c)
WENN Gießaufwand == einmal die Woche DANN Crassula Ovata = (+5)
WENN Gießaufwand == einmal am Tag DANN Kaktus = (+5)
...

Figure 5.5: Rule verbalization example in ProKEt. It consists of a header with basic information regarding the solution and its rating (a), a reduced rule verbalization (b), and a listing of all remaining, currently non-contributing rules (c). The source/reason for derived items is displayed indented (d).

Further, rule verbalizations can be fine-tuned regarding: The *Rating Granularity* concerning both the presented solution and its state, yet concerning also the rule action presentation. Here, all variants—exact score, abstract rating or full rating—are equally well feasible. Similarly to the finding list, also the rule verbalization can indicate abstraction questions a bit more prominently for an immediate recognition of to their special status. One possibility is sketched in

Figure 5.5, d, where the abstract question *Pflegeaufwand* [need of care] is derived automatically by the question *Gießaufwand [watering intensity]*, displayed a bit indented.

Apart from naturally fitting rule-based knowledge, it is possible to generate rule verbalizations also based on other knowledge types, such as set covering models: By extracting the relevant corresponding KB objects—solutions and findings—from the knowledge representation and assembling own rule verbalization formulations.

c. FSTree Display (Visual)

The first proposition for a more visual justification presentation is to exploit the file system (FS) tree metaphor: Solutions then correspond to top-level folders and findings, as well as intermediate solutions, become their subfolder correspondents. This is sketched in Figure 5.6, where *Influenza* and *Cold* are a solution and an intermediate solution node, and the remaining nodes represent findings. Basic presentation options include: The usage of background coloring, icons, or text (as in Figure 5.6) for indicating the solution's state. Different base renderings of solution objects vs. findings for eased visual differentiation. And Indentation—similar to the previous justification variants—for representing *abstraction/derivation* relationship.

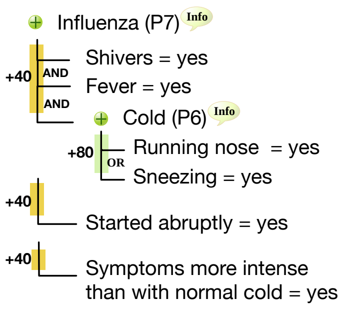


Figure 5.6: FSTree for *Influenza*, rated by four rules. The first (compound) rule contains an intermediate solution *Cold*, displayed in *integrated* mode.

Further options for rule-based knowledge are: Coloring the branches as to indicate the action type and corresponding value. See Figure 5.6, where the branch that represents the derivation rule for *Cold* is colored green as it rates *Cold* as established. Additionally either adding exact numbers—as in Figure 5.6—or the abstract rating in text form to the branches for a clearer differentiation is possible. And finally for compound rules, adding the connector type between the findings contained in that rule denotes an option. For example, the first rule represented in Figure 5.6 is a compound rule where all conditions must apply (connected by AND). However, regarding covering knowledge, the latter branch decoration naturally is not feasible due to missing information. Yet there, the findings might be additionally enhanced by information of the relative importance regarding the solution. Thus basically, FSTrees are applicable to all kinds of knowledge, that allow for extracting (rated) solution objects as well as findings—only the possibilities regarding the fine granular display options may vary a bit as explained.

d. Justification TreeMap (Visual)

TreeMaps [Shneiderman, 1998] are a well-known visualization technique today. [TreeMap Shneiderman, n.d.] provides an interesting historical overview of their development up to 2013. A TreeMap most basically represents hierarchical relationships. A prominent example is the visualization of a computer's hard disk usage. Thereby, a frame/outer slice represents a *container object*—such as a folder on a computer's file system—and slices within that container represent its *content*—corresponding to sub-folders and files within the container folder. Further, normally the size of items is mirrored by the relative size of the slices.

We propose to assemble a TreeMap for justification presentation as follows: The outermost frame represents the final top-level solution. Slices within those outer frames represent either (final) findings or derived interview objects. Figure 5.7, a, sketches the TreeMap display for a single solution, using popup display for further clarifying derived items. In the example, the intermediate solution *Cold*, that contributes to the main solution *Flu*, is rendered in a clickable manner. Thus, when clicked its justification is displayed in an own TreeMap popup (Figure 5.7, a.1). In contrast, Figure 5.7, b, shows a sketched prototype for the statical display of multiple solutions—here, *Flu*, and *Migraine*. Flu, in turn is derived by an intermediate solution *Cold*, and two further findings. The display in example (b) is entirely static, i.e., the recursive justification for *Cold* is integrated directly within the main TreeMap. In both examples, the abstract solution state is indicated by the background color of the TreeMap slice—Flu (established) is colored green, Cold (suggested) is colored yellow, and Migraine (excluded) is colored red. Additionally, the precise score of the solutions is provided in parentheses for reasons of clarity.

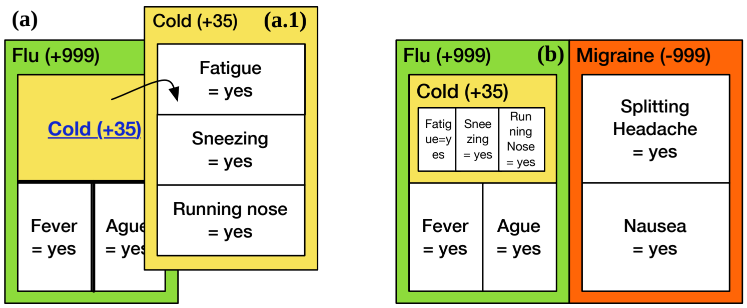


Figure 5.7: Two sketches of TreeMap solution justifications: (a) explains the solution *Flu*, derived by two findings and one intermediate solution *Cold*. The recursive justification for *Cold* is shown in an interactive popup (a.1). In (b), the two solutions Flu and Migraine are justified in a static manner. Thus, the recursive justification of *Cold* is presented directly integrated in the main TreeMap.

Further options for enhancing comprehensibility include: Varying the *Rating Granularity* for displaying solutions and derived/abstract questions, i.e., presenting the abstract rating, the precise value, or both. Indicating the particular rule type in case compound rules exist. For example, in Figure 5.7, a, the bold borders between the findings that contribute to *Flu* indicate, that this is a compound AND-rule, where all contained findings are required for deriving the solution. In contrast, normal borders as between the findings of the TreeMap for *Cold* indicate a compound OR-rule. In case covering knowledge is visualized, also information such as relative importance of a finding regarding the solution, or whether one single finding suffices, may be integrated in a TreeMap—e.g., within the findings' slices.

As implied above, TreeMaps are feasible both for rule-based and scoring knowledge, or any other knowledge type where respective KBS objects and knowledge-based actions can be extracted.

e. Rule-Graph (Visual)

Rule-graphs are a visualization specifically suitable for rule-based knowledge. The *Derivation Graph* as one possible implementation of a rule-graph visualization has already been subject to prior research. We refer to [Baumeister and Freiberg, 2010, pp. 9 ff.] for detailed information.

Rule-graphs contain distinguishable nodes for interview objects, i.e., solutions and findings, and optionally for connector objects. The latter represent which connection a compound rule provides (AND / OR / NOT connection). Then, arcs are drawn between finding nodes and the target solution, or, depending on the particular presentation style, between findings and respective connector nodes. Slightly different rule-graph schemes are depicted in Figure 5.8, a–c, visualizing the derivation of the solution *Flu* by two compound rules. The compound AND-rule further contains an intermediate solution, *Cold*. The second rule is a compound OR rule; i.e., as soon as one of the two connected conditions is true, that rule fires. Further, one condition is negated and thus represented by a NOT-connection; i.e., such a condition evaluates to true if the final finding does not apply. Variant (a) sketches an interactive display, that means, clicking on the *Cold*-node brings up a separate graph display, depicting the derivation path of the solution Cold, see Figure 5.8, a.1. In contrast, variants (b) and (c) base on a static presentation where all intermediate paths/objects are entirely integrated.

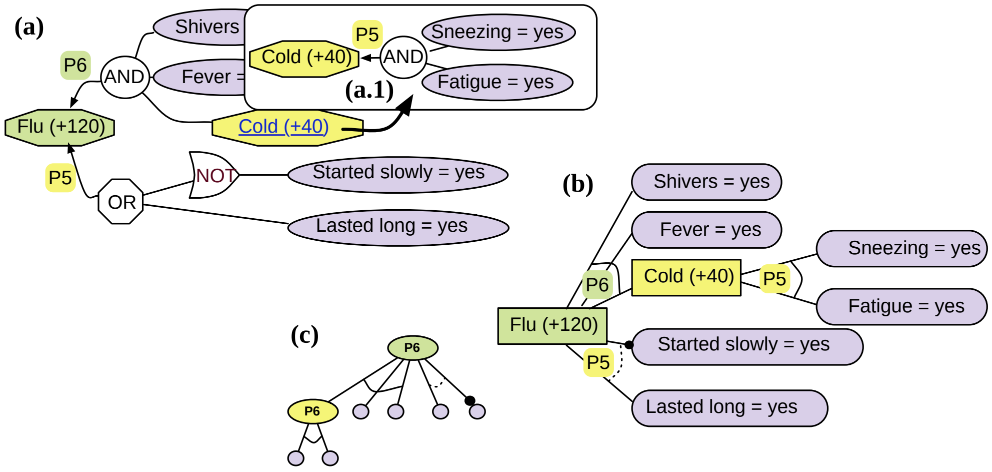


Figure 5.8: Rule-Graph sketch that visualizes two (compound) rules that rate the solution *Flu* interactively. Variant (a) is the most comprehensive presentation, also explicitly integrating rule connector nodes. Variant (b) as a more compact alternative uses distinct bow connections between the arcs for the rule connection representation. Variant (c) is the most condensed presentation and provides mostly a structural overview. This latter variant reveals details—e.g., solution/findings name/value—only on request, such as displaying an information popup when hovering with the mouse.

Basic styling options for rule-graphs encompass: Distinguishable rendering for solution-, finding-, and connector nodes (or, in the compact variants, connector arcs, see below). Indicating abstract rating states of solutions by background coloring (c.f., Figure 5.8). In the example, the solution *Flu* is derived as established and thus colored green, whereas the solution *Cold* is derived as suggested, and thus colored yellow. Decorating the rule-representing arcs by abstract or exact rating values as to indicate the rule action. For example, see Figure 5.8, a & b, where the represented rules add a positive value (P5 & P6) to the main solution *Flu*. And indicating the rule action additionally or exclusively by arc coloring.

As Figure 5.8 indicates, several levels of elaborateness can be chosen for the presentation. Variant (a) depicts the most comprehensive one, with separate nodes for compound rules connectors and all required information such as interview item texts, ratings, etc. This variant is the most space intensive one, yet it contains all information—except, potentially, recursive derivation paths—directly within the core presentation. Variants (b) and (c), in contrast, are more compact by indicating the rule type only by a distinct additional bow that connects all finding arcs that belong to the compound rule: Solid bow = AND, dotted bow = OR. The NOT-connection (borrowed from the representation of NOT-switches in plugging diagrams) is represented by a black dot. Variant (c) is the most compact presentation variant due to only slightly indicating interview items, ratings for derived objects, and connection types. On the one hand, this emphasizes the overall structure of the presented knowledge. Thus, also more complex interconnection structures can be well presented by this variant in an entirely integrated presentation, i.e., without using popups. The downside, however, is that interview items are not immediately recognizable, as no solution/finding text is provided directly, but only when requested, e.g. when hovering the respective node.

Static Justification Presentation—General Considerations With regards to the static presentation techniques presented in this section, there are two more general configuration options that can be considered for all presented alternatives: *Direct/Popup Integration* of (recursive) derivation knowledge, and *Flexible Presentation Switching*.

Generally, the intermediate justifications of derived knowledge items can be integrated directly in the main presentation—most examples in the preceding section sketched this case. Thereby, one can consider further whether to show them anytime like all other items. Or to integrate them in the respective suitable place of the main display once a triggering item is clicked. The advantage of the former variant is, that all relevant items for a justification are contained in a single display. The downside, on the other hand, is that in case of more comprehensive justifications a direct integration may render the entire presentation too complex and hard to perceive. Alternatively, such part-justifications also can be displayed separately—e.g., opening them within a popup- or new window. This, however, bears the main disadvantage, that in the case of repeatedly interconnected part-justifications, many new popups may open up which also disturbs the main presentation and the general understandability of the respective coherences. Overall, we regard finding list, rule verbalization, and FSTree as generally more suitable for directly integrating derived items as those present more items in a more compact manner. In turn, the proposed TreeMap variant seems the least appropriate candidate for static direct integration, especially with comprehensively nested and interconnected derivation paths. This is due to the fact, that such justification TreeMaps integrate lots of additional information—in contrast to the original TreeMap visualization—and thus easily can grow too complex as to intuitively convey all necessary information.

An alternative general display option is, to enable users to switch the particular presentation type. For example, the presentation may basically include all contributing knowledge items but highlight the currently inactive items distinctly as sketched in Figure 5.5, p. 85. Yet if desired, users can deactivate/hide those additional, non-contributing items for more clarity. Thus, users that are potentially interested in more deeply exploring the coherences of the KB, such as experienced users or domain experts, are per default enabled to do so. Laymen, however, that might be overwhelmed or annoyed by this comprehensive presentation, can switch to the simple view.

5.3.2 Interactive Justification Presentation

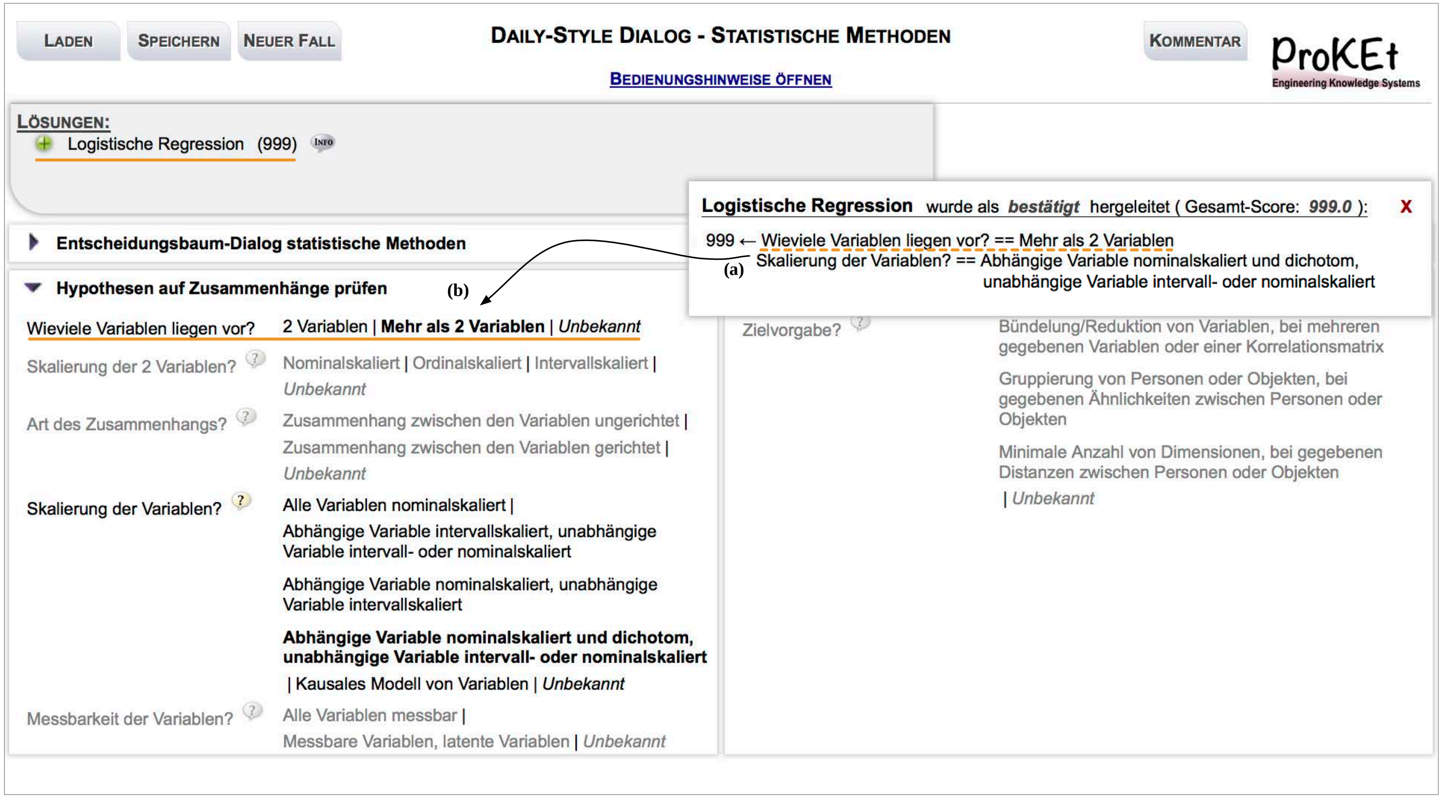
In the last section, we discussed static base justification presentation forms. *Static* thereby refers to the fact, that those presentations only mirror the current justification of a solution, but allow no further input interaction. However, users oftentimes do not simply want to review a static justification, but moreover wish to adapt answers for quickly exploring the corresponding consequences on the solution. Therefore, we propose *consultation-justification mashups* as exciting opportunity for rendering highly interactive KBS justification forms.

One simple idea for a rather loosely connected mash up, described in Section 5.3.2, a., is to provide interactive back-links from within the justification presentation back into the core KBS UI. Alternatively, the novel Clarification KBS type, introduced in Section 2.2, denotes an alternative and way more comprehensive option for realizing highly interactive justification-consultation mashups. We summarize its usage as interactive clarification element in Section 5.3.2, b..

a. Interactive Back-link into the Core Input UI

The first simple yet already effective interactivity enhancement is to provide targeted back-links from the justification presentation to the core input UI. The idea is, to design question and answer options in the justification as tailored links. When clicked, such a link directly switches back to the respective KBS UI, automatically moving the clicked object into the viewport of the user. For highly compact UI types, such as the Daily style (c.f., Section 4.3.2.b.), simply highlighting the corresponding question can already suffice. Alternatively, also an automated scrolling mechanism can be realized, that centers the relevant part of the UI—which may be additionally highlighted—on the screen. This is especially valuable in less compact UI styles, such as (comprehensive) Box Questionnaire- (c.f., Section 4.3.2.a.) or Interview (c.f., Section 4.3.3.a.) implementations as to ease the immediate recognition of the altered interview object.

The main advantage of back-linking is the flexibility regarding further consultative exploration. As users thereby switch back into the original KBS UI, additionally to the highlighted question naturally all other KB items are also available and can be further investigated at will. At the same time, this entire back-switch also induces the main downside: Therewith, the originally investigated justification view is closed and thus the focus—regarding the solution that was originally clarified—is lost. This can be alleviated by suitably highlighting not only the respective question, but furthermore also the originally inspected solution once the back-switch has been performed. Figure 5.9 sketches back-linking for a Daily KBS UI implementation. There, the user originally inspected the solution *Logistische Regression [Logistic Regression]* and wanted to explore changes regarding the question *Wieviele Variablen liegen vor? [How many variables]*. Thus, as the figure indicates, once the user clicks on that item in the justification view (a), the core UI is re-rendered. Both the originally justified solution as well as the selected object are highlighted, thus reminding users of their original intention. Back-linking is suitable



DAILY-STYLE DIALOG - STATISTISCHE METHODEN

BEDIENUNGSHINWEISE ÖFFNEN

LÖSUNGEN:
Logistische Regression (999)

Logistische Regression wurde als ***bestätigt*** hergeleitet (Gesamt-Score: ***999.0***):

999 ← Wieviele Variablen liegen vor? == Mehr als 2 Variablen
Skalierung der Variablen? == Abhängige Variable nominalskaliert und dichotom, unabhängige Variable intervall- oder nominalskaliert

Entscheidungsbaum-Dialog statistische Methoden

Hypothesen auf Zusammenhänge prüfen

Wieviele Variablen liegen vor?	2 Variablen | **Mehr als 2 Variablen** | *Unbekannt*
Skalierung der 2 Variablen?	Nominalskaliert | Ordinalskaliert | Intervallskaliert | *Unbekannt*
Art des Zusammenhangs?	Zusammenhang zwischen den Variablen ungerichtet | Zusammenhang zwischen den Variablen gerichtet | *Unbekannt*
Skalierung der Variablen?	Alle Variablen nominalskaliert | Abhängige Variable intervallskaliert, unabhängige Variable intervall- oder nominalskaliert Abhängige Variable nominalskaliert, unabhängige Variable intervallskaliert **Abhängige Variable nominalskaliert und dichotom, unabhängige Variable intervall- oder nominalskaliert** | Kausales Modell von Variablen | *Unbekannt*
Messbarkeit der Variablen?	Alle Variablen messbar | Messbare Variablen, latente Variablen | *Unbekannt*
Zielvorgabe?	Bündelung/Reduktion von Variablen, bei mehreren gegebenen Variablen oder einer Korrelationsmatrix Gruppierung von Personen oder Objekten, bei gegebenen Ähnlichkeiten zwischen Personen oder Objekten Minimale Anzahl von Dimensionen, bei gegebenen Distanzen zwischen Personen oder Objekten | *Unbekannt*

Figure 5.9: Interactive justification—back-linking. Clicking the question *Wieviele Variablen liegen vor? [How many variables?]* in the justification popup display (a) brings back up the core KBS UI—in this case, Daily style. Thereby it highlights the selected item (b) for instant recognition.

regarding all base justification presentation forms described earlier in Section 5.3.1. Therefore, only the rendering of the interview objects needs to be adapted as to integrate the link-style with additional switch back functionality. Basically, we assume this extension to exhibit the most benefits when used in combination with a UI type that displays many to all questions in a compact manner intended for explorative usage—such as the Questionnaire KBS UI variants proposed in Section 4.3.2.

b. Clarification KBS with Justification Focus

Apart from back-linking, the usage of the novel Clarification KBS type (introduced in Section 2.2) denotes the more sophisticated possibility for interactive justification presentation. Thereby, only the *justification-oriented* initialization variant, as also described in Section 2.2, is appropriate. Starting point in this case is an already derived and rated solution for which the Clarification KBS is invoked as justification module—i.e., it highlights contributing, potentially contributing but not selected, and non-relevant answer options accordingly. Due to the inherent interactivity of clarification KBS, users then can freely continue to explore alternative answer options in the further course—and investigate the results regarding the target issue, respectively. As the pattern collection demonstrates, in general various KBS UI types are feasible for Clarification KBS. Main requirement for using them in a justification-focussed way is a compact presentation of optimally all (or a greater part) of the relevant questions and answers within the clarification presentation. This supports and eases a reasonable overview and exploration. Thus, suitable patterns for the use case sketched here encompass the *Hierarchical Clarifier* and the hybrid variant *Clarifier Questionnaire*. Regarding *Form Add-on*, its feasibility for using it as justification-focussed module depends on the chosen, underlying static base justification presentation and on the size/complexity of the KB: In case the base justification is able to include all relevant knowledge items for the target solution, it can—together with appropriate Form Add-on widgets—well be used as justification module. *Clarifier Interview*, is not at all suitable here, as it presents only one single question at a time, thus not providing any reasonable justification view at all.

5.4 Synopsis

In this chapter, we discussed KBS context presentation options, encompassing the results, auxiliary information, and justification modules. Regarding the display of KBS results, we distinguished *pure summarizing* presentations and more comprehensive *solution display panels*. We also elaborated possibilities, how to integrate auxiliary information for various KBS components: For *single interview objects*, and for globally useful widgets. Concerning justification, we finally proposed both *static* and *interactive* variants. Also the KBS context presentation collection arose mainly due to practical project requirements. Thus it is not intended to constitute an exhausting, finalized collection, but rather an initial library of inspirational ideas.

Having presented conceptions regarding the presentation of both KBS core input and KBS context, we next introduce the tailored KBSE tool *ProKEt* in Chapter 6. ProKEt supports the realization of KBS adhering to the suggested development approach and presentation options.

Chapter 6

ProKEt—Tailored & Encompassing KBS Engineering

It is not sufficient, to approach the river wishing to catch some fish.
One also needs to bring a suitable fishnet.

Saying

Similarly, it is not sufficient to only plan for encompassing KBSE in theory—also appropriate tool support is required. This is especially due to the already motivated fact, that KBS UI layouts typically cannot be easily designed optimally using standard GUI builders. Rather, tailored interaction forms and design options need to be considered—e.g., for appropriately taking into account the flexible presentation and integration of non-indicated- or abstraction questions.

In this chapter, we introduce *ProKEt*, a tailored prototyping and knowledge systems engineering tool. It is targeted towards the development of participatory, browser-based KBS. Thereby, it specifically supports the previously introduced key components for encompassing KBSE. Preliminary research on ProKEt was already published in [Freiberg and Puppe, 2013, Freiberg et al., 2012, Freiberg and Puppe, 2012b, Freiberg et al., 2010].

In Section 6.1, we first provide a brief summary of the technical details of ProKEt, as well as of its artifacts. Section 6.2 then explains in what regards ProKEt supports encompassing KBSE activities. As motivated in Chapter 3, this entails extensible prototyping, template/pattern-based development, KA extension points for intertwined UI and KA development, and usability activities.

6.1 ProKEt—Technical Introduction & Artifacts

The ProKEt Framework The web application framework ProKEt integrates Java as back-end language for coupling the web application front-end with the KB and reasoning framework [d3web, n.d.]. Thus, ProKEt can be used on any computer with a current functional [Java Oracle, n.d.] installation. It is optimally used from within an IDE such as [Netbeans Oracle, n.d.] or [Eclipse, n.d.]. Those support not only the management of complex programming projects, but also provide fundamental for web-based development. For actually using the ProKEt artifacts (details see below), additionally a web server environment supporting Servlet-based web applications is required, such as [Tomcat, n.d.]. The tool is developed as open source project and can be obtained from [ProKEt, n.d.].

ProKEt Artifacts KBS developed with ProKEt most generally are Servlet-based web applications. In the following, we refer to them as *ProKEt artifacts*. ProKEt supports the creation of both pure prototypes and fully functional KBS. Both types of artifacts equal in their basic framework and build on the same base technologies. Yet, they differ with respect to their knowledge representation, and potentially, some specific properties. Figure 6.1 summarizes the main architecture of ProKEt artifacts. Based on a browser (i.e., user) request, Servlets residing

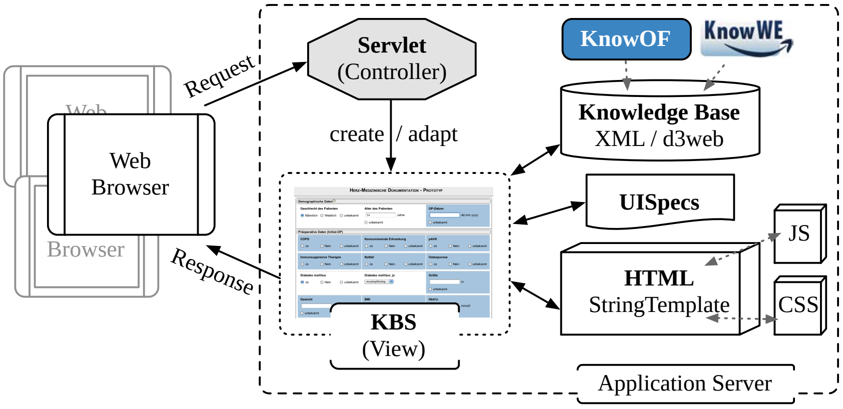


Figure 6.1: Basic framework and components of ProKEt artifacts: Servlets assemble the artifact based on the respective knowledge specification (tailored XML for pure prototypes, d3web KBs for productive KBS) and the corresponding UI specification (tailored XML, also). Artifacts build on HTML templates (realized with StringTemplate) and integrate CSS and JS/AJAX technologies for the styling and interactivity.

on the application server assemble and/or adapt the respective KBS artifact. Those artifacts are composed of (modular) HTML templates that are defined using the [StringTemplate, n.d.] technique. The specific look and feel of the UI, as defined in the ProKEt UISpecs, is fine-tuned via [W3C CSS, n.d.]. Interactivity is provided by [W3Schools JavaScript, n.d.] / [jQuery Foundation, n.d.]. Examples are answer value submission, updating the UI display regarding the current question/solution states, handling multilingualism, etc. To avoid multiple re-rendering of the entire system, [W3Schools AJAX, n.d.] techniques are applied for processing only the relevant parts of the artifact. This allows for an overall smoother responsiveness of the system. In case, a productive KB is integrated with the extensible prototypes, those can originate from the KnowOF tool (see Section 6.2.3, a.), or from KnowWE [Baumeister et al., 2011].

Pure Prototypes Pure prototypes mainly intend to serve the quick and dirty assessment, whether one of the supported basic KBS styles is potentially suitable for a given context. They are, however, not directly extensible into functional systems, and thus do *not directly* support the extensible prototyping use case. Yet, as once developed templates and styles are reused in ProKEt also for extensible prototypes, developing pure prototypes and corresponding widgets/stylings implicitly supports also the capabilities of extensible prototyping. Details on pure prototyping can be found in [Freiberg et al., 2012].

Extensible Prototypes In contrast to their pure prototype relatives, extensible prototypes are deliberately intended for the extensible prototyping use case. That is, for directly, quickly, and effortlessly extending them into functional KBS core prototypes and fully-fledged productive KBS, respectively. The corresponding extensible prototyping process is described in more detail in Section 3.1.1.

Regarding the knowledge representation, extensible prototypes integrate functional [d3web, n.d.] KBs. Thereby, d3web supports the specification of the terminology and required reasoning mechanisms. Additionally, it allows for defining auxiliary information, such as additional explanatory texts for interview objects, images, documents, etc. Currently, the d3web toolkit is the only supported KB specification format. However, d3web has grown to an extensive knowledge engineering toolset over the years, that supports various knowledge representations, such as (production) rules, decision trees, or set covering knowledge, along with the correspondingly required reasoners. Thus, it provides a sound basis for vast application scenarios that we found sufficient for realizing all the different KBS visions we had in mind so far.

6.2 ProKEt—An Encompassing KBSE Tool

The following sections elaborate, in what regards ProKEt practically supports *extensible prototyping, pattern-based development, intertwined KB & UI development*, and *usability activities*—the key components for lifting KBSE approaches to a more encompassing level.

6.2.1 Extensible Prototyping

As proposed in Section 3.1.1, extensible prototyping bases on the creation of a mature and interactive front-end core KBS prototype and two extension steps, for adding both the productive KB and framing functionality. This is explicitly supported by ProKEt.

Basic specification format Extensible prototyping in ProKEt strongly founds on the ProKEt UI Specification file (*UISpecs*, in the following). The <dialogOpts> tag, see Figure 6.2, a, specifies global KBS options, i.e., properties that are not specifically UI-related, but relevant to the future KBS as a whole. Those options mostly correspond to KBS framing functionality.

The tags <globalUIOpts> and <localUIOpts> define UI-related configuration options, corresponding to the core KBS UI. Global UI options thereby concern the UI presentation of the entire KBS, e.g., the basic UI type. Also, question- or answer-related options, that are to be used as a global default, are defined here, e.g., a default 3-column style. Local UI options, in contrast, denote specific options only for the defined individual KBS UI component(s), i.e., they overwrite the global default UI property. Thus, ProKEt offers a powerful inheritance and substitution mechanism for UI options.

The <dataOpts> tag specifies the KB to be used, as shown in Figure 6.2, c. Finally, also the ProKEt usability extension (*PUE*, see Section 6.2.4) is configured within the UISpecs, using the <ueOpts>-tag (see Figure 6.2, d). The UE thereby does not support extensible prototyping itself directly; rather, it denotes the add-on for integrated usability-support, postulated as third key component of encompassing KBSE.

```
 <dialogOpts
(a) header="BoxQuestionnaire: Statistics" login="" required="" languages="de;;;enGB" debug="false">
    <globalUIOpts
       dialogtype="Questionary" showNonIndicated="HIDE" css="basicQuestionary" dialogcolumns="1"
       questionnairecolumns="2" questioncolumns="1" solutionsAnytime="TRUE" showAbstractions="True"
       unknownVisible="True"
(b) />

    <localUIOpts
       <dropdown TRUE="ChoiceQuestion3" />
    </localUIOpts>

(c) <dataOpts kb="statisticsRecommender.d3web" />

(d) <ueOpts logging="FALSE" feedback="FALSE" questionnaire="NONE" />
 </dialogOpts>
```

Figure 6.2: Extensible Prototyping with ProKEt: Using an adapted specification that integrates a d3web KB as data source (c). General framing properties are specified in the *dialogOpts* tag (a). Question-specific properties can be defined globally, i.e., for all questions likewise, or locally, i.e., for specified questions (b). Additionally, the ProKEt usability extension is configured via the *ueOpts* (d).

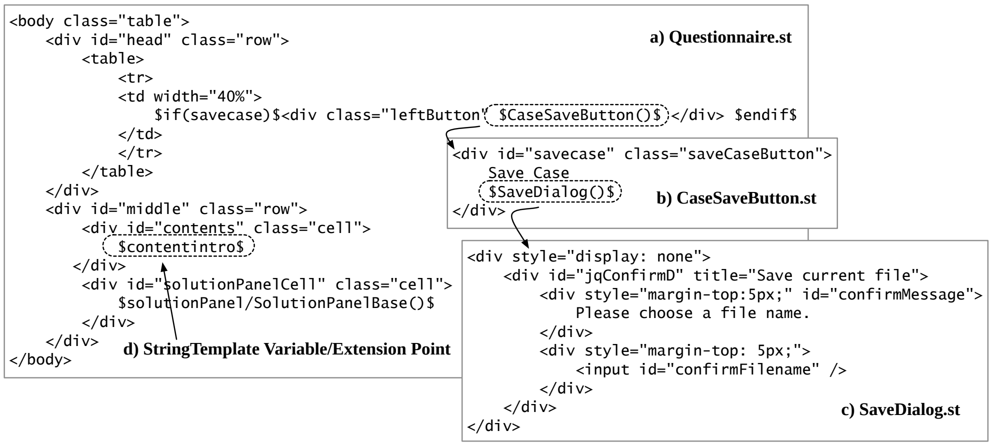
Extensible Prototyping—Practically Applied with ProKEt The three extensible prototyping steps, introduced in Section 3.1.1, are supported by ProKEt as follows:

- *EP1—KBS Front-end Prototype:* Realizing extensible prototypes in ProKEt requires a functional d3web KB and an UISpecs. Thereby, ProKEt deviates slightly from the prescription, that in EP1 creates a pure front-end prototype. However, several default KBs are offered by ProKEt out of the box, so there is no need for catering with KA at all initially—which again conforms to EP1. For adapting the front-end prototype, ProKEt offers diverse options, discussed in more detail in Section 6.2.2. Regarding EP1, mainly property-based configuration using the UISpecs, and an extension of style- or template files is suitable.

- *EP2—KBS Core Prototype:* Transforming a front-end prototype into a KBS core prototype, only requires the addition of the desired KB by defining the KB in the *dataOpts* part of the UISpecs. When not using a ProKEt KA extension, the KB has to be inserted in a specified ProKEt folder so it is found at runtime. When using the ProKEt KA extensions, this is not necessary as the KB then is automatically deployed correctly with the KBS UI artifact. At runtime, the required contents are directly fetched from the KB and mapped to the corresponding UI elements.

- *EP3—Fully Functional KBS:* Migrating from a functional KBS core prototype to a fully fledged KBS solution then only requires the definition of global dialog properties for specifying the desired framing functionality. Examples are an instructional text, the languages to be supported, or the general KBS header. Those properties all are defined via the *dialogOpts* tag of the UISpecs.

As both adaptions needed for the extension steps are defined in the same single KBS UI specification, ProKEt equally well enables a direct progression from front-end prototype to fully-fledged system.



```
<body class="table">
    <div id="head" class="row">
        <table>
            <tr>
            <td width="40%">
                $if(savecase)$<div class="leftButton"> $CaseSaveButton()$ </div> $endif$
            </td>
            </tr>
        </table>
    </div>
    <div id="middle" class="row">
        <div id="contents" class="cell">
          $contentintro$
       </div>
        <div id="solutionPanelCell" class="cell">
            $solutionPanel/SolutionPanelBase()$
        </div>
    </div>
</body>
```

```
<div id="savecase" class="saveCaseButton">
    Save Case
    $SaveDialog()$
</div>
```

```
<div style="display: none">
    <div id="jqConfirmD" title="Save current file">
        <div style="margin-top:5px;" id="confirmMessage">
            Please choose a file name.
        </div>
        <div style="margin-top: 5px;">
            <input id="confirmFilename" />
        </div>
    </div>
</div>
```

Figure 6.3: StringTemplate-based, nested pattern/KBS UI assembly in ProKEt: A framing template defines the base structure—here, in Questionnaire style (a). A save-button widget template (b) is called from within the frame, and a custom template for a save case dialog widget (c) again is called from within the save button template. Further, (d) shows the definition of plain StringTemplate variables that are filled with content by the respectively used (Java-based) renderer classes.

6.2.2 Pattern-based UI Development

In Section 3.1.2, we motivated the (re)use of proven KBS UI solutions in the form of both modular widget-based and system-wide patterns as second key ingredient for encompassing KBSE. A basic collection of core KBS UI patterns was introduced in Section 4.3, and the most relevant ones—according to their application in actual projects—are supported by ProKEt out of the box.

Patterns are implemented in ProKEt by using the [StringTemplate, n.d.] technique. This allows for a highly modular definition of both the framing pattern (UI) structure and singular widgets such as certain button variants. Thus, fine-grained widget templates can be assembled as to gradually form more and more complex widgets, which finally are integrated in the framing pattern template.

Figure 6.3 illustrates the modular, nested template definition: The template frame for the Questionnaire pattern is shown in Figure 6.3, a, *Questionnaire.st*. This contains—amongst other elements—a button for opening a dialog for saving KBS cases/sessions. The template for that button is shown in Figure 6.3, b, *CaseSaveButton.st*, and the template for the dialog widget itself in Figure 6.3, c, *SaveDialog.st*. The basic framing template then defines extension points. Those are addressed either by StringTemplate itself—resembling a function call, e.g., *$Case Save Button()$*—or by Java-based renderers that fill property definitions such as *$childrens$* in Figure 6.3, d. The button template then in turn contains the extension point for the save case dialog widget.

Adapting and extending the base solutions ProKEt offers several mechanisms for configuring or extending the available pattern realizations. With respect to the adaption efforts and (programming) expertise those operate on various different levels.

- *UISpecs Properties:* The simplest variant is to use predefined UI configuration options of ProKEt by adding them in the UISpecs. For extensible prototypes, those mostly comply to those described in Sections 4.1 and 4.2.
- *StringTemplate Restructuring/Combination:* StringTemplate files define the basic HTML structure of the KBS and the contained widgets. An example is, whether the KBS contains a solution display panel or not, and where this is to be placed. Thus, general modifications of the structure, i.e., widget sequence, of a pattern can be achieved by adapting the respective StringTemplate files. Thereby, the modular StringTemplate specification and assembly as illustrated above strongly fosters the (re)use of already existing templates, as well as the easy integration of new modules.
- *CSS Restyling:* Fine-grained styling issues, such as particular color schemes or spacing, are defined via CSS. Thereby, ProKEt provides default styles for each pattern as to offer examples for at least one feasible UI representation. Thus, for fine-tuning a given pattern styling, one can either adjust existing CSS files or, preferably, introduce new style files that overwrite default styles and include them simply as property in the UISpecs.
- *Sophisticated Code-base Extensions*: The most comprehensive form of extension is the implementation of entirely new patterns. This mostly also induces the introduction of corresponding new templates, functionality, and renderers, or of additional general UISpecs configuration options. This variant naturally requires an adaption of the ProKEt code base.

6.2.3 Knowledge Acquisition Extension Points

Regarding the realization of productive KBS artifacts, ProKEt specifically works with [d3web, n.d.] KBs and supports their KA. By KA, we mean the specification and evaluation of the KB itself. As motivated beforehand, ProKEt itself fore-mostly focusses on KBS UI design and usability evaluation support as to promote the relevance and tighter integration of those activities in KBSE. However, for fostering the novel intertwined KB–UI development paradigm, ProKEt further provides for the seamless and effortless integration of KA activities: By offering extension points, that allow for easily and tightly coupling (external) KA tools with ProKEt. One such tool is the KA environment KnowOF that processes specifically formatted standard office files into d3web KBs, see Section 6.2.3 a.. Another tool is the semantic wiki KnowWE [Baumeister et al., 2011] for collaborative KB development, see Section 6.2.3 b.. That way, ProKEt bundles all necessary activities in one toolset. This in turn strongly fosters the intertwined KB & UI development and immediate review of both aspects simultaneously without much additional efforts.

a. KnowOF—Standard Office-based KA

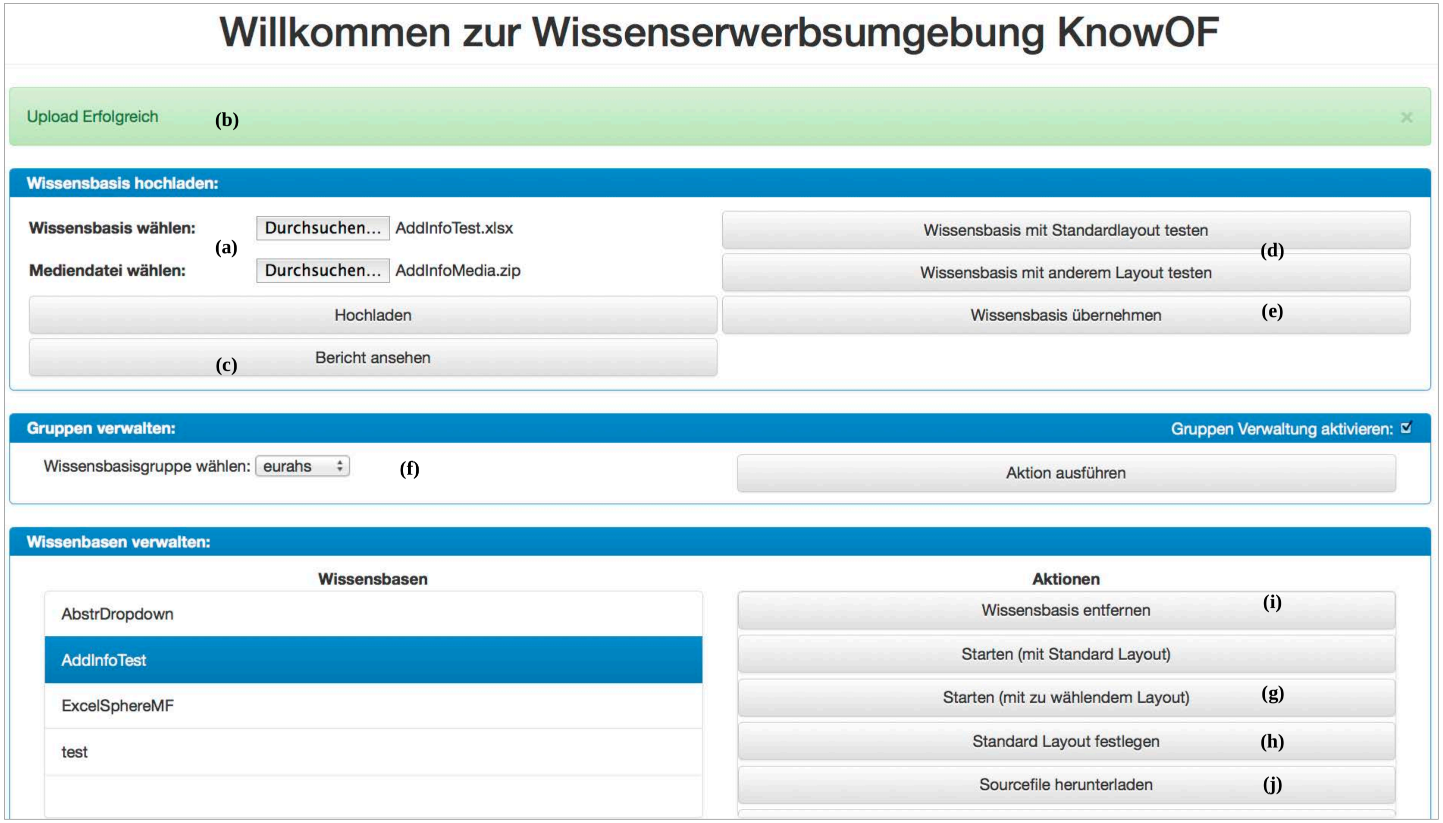
As already argued in Section 2.1.5, the active participation of domain experts in the KA process highly beneficial. Experiences from ongoing projects showed, that domain experts—as, e.g., medical doctors, or legal specialists—often refrain from learning to handle specialist KA tools. Rather, they demand to use their familiar standard software as condition for any participation efforts. Excel and Word are two established standard office tools applied in various domains. Using those for KA, domain experts still need to learn a specific knowledge definition markup which is to be used for defining the office documents. Yet now, experts can create the required documents in their familiar environments. This drastically reduces the barrier of active KA participation. Further, reviewing such a KB in a functional KBS UI requires as little as a single button click in KnowOF. This offers an even more intuitive way for checking and refining the KB. Which in turn increases the trust towards the correctness/quality of the KB, the trust in the tool itself, as well as in the own ability to handle such a tool.

The current implementation state of KnowOF (in german) is depicted in Figure 6.4, p. 100. Its key functionality is the upload of standard office files and their transformation into d3web KBs, see Figure 6.4, *Wissensbasis hochladen [Upload Knowledge Base]*. Additionally, KnowOF offers some user and KB management features, c.f., the lower half of the figure.

For the upload, the desired document needs to be specified, as well as optionally media files for auxiliary information—e.g., images, or encompassing archive files such as ZIP (a). Once those contents are parsed, KnowOF displays the status—success or failure—prominently in the header (b). It then allows for investigating a KB parse report which lists, e.g., parse errors as well as the formal KnowWE syntax for representing the knowledge (c). In the case of parsing success, the KB can be tested with the parser-generated UI configuration or one of the further provided default KBS UI variants (d), and—if approved of—can be stored in the KB repository (e). For storing the KBs, KnowOF offers both a default common repository as well as the possibility to group KBs, e.g., topically (f). The latter can be helpful in larger projects, where different KA teams need to specify many different KBs, but where one team is not interested into or allowed to access the work of the other team. The contents of the current KB repository/group are listed in the bottom part of the UI. There, also further KBS UI types can be chosen (g) for testing the KB in alternative UI layouts. Here, enabled by the direct coupling of KnowOF and ProKEt, the chosen KB and the selected KBS UI are always assembled into a functional KBS on the fly. This allows for immediate testing of both newly uploaded and existing KBs from the repository with alternative KBS UI variants in a straightforward manner. Also, the presumedly most suitable KBS UI type can be fixed for the respective KB (h). Finally, KBs can also be removed from the repository (i).

Currently, KnowOF provides support for Microsoft Excel and Word. Thereby, the Excel-variant offers a higher diversity regarding the target knowledge type, e.g., rule-based or set covering knowledge. Word, in contrast, is mainly used for specifying tailored hierarchical refinement knowledge as used, for example, by the ITree KBS UI.

KnowOF—Excel-based KA Regarding Excel-based KA, a flexibly adaptable reference specification template builds the foundation. This defines on the one hand the structure of the KA spreadsheet, i.e., *what* to enter in *which* columns. On the other hand, the syntax regarding certain certain fixed properties, such as the short-hand for the question type. That way, the particular properties and appearance of the Excel input format are easily adaptable for diverse users. Examples include changing the sequence or the descriptive header of the columns. This possibility to tailor the input format strongly fosters user satisfaction and active participation in the process. Also, such a tailored specification format can reduce KA errors whilst increas-



Willkommen zur Wissenserwerbsumgebung KnowOF

Upload Erfolgreich (b)

Wissensbasis hochladen:

Wissensbasis wählen: AddInfoTest.xlsx (a)

Mediendatei wählen: AddInfoMedia.zip

Gruppen verwalten:

Gruppen Verwaltung aktivieren: ☑

Wissensbasisgruppe wählen: eurahs (f)

Wissenbasen verwalten:

Wissensbasen	Aktionen
AbstrDropdown	Wissensbasis entfernen (i)
AddInfoTest	Starten (mit Standard Layout)
ExcelSphereMF	Starten (mit zu wählendem Layout) (g)
test	Standard Layout festlegen (h)
	Sourcefile herunterladen (j)

Figure 6.4: Current state of the standard office KA tool KnowOF (in german), providing functionality for uploading, managing, and testing KBs with specified UI variants. For details, see pp. 99 ff.

ing efficiency and reducing efforts and costs. This renders Excel-based KA a highly flexible, universally applicable and beneficial KA method in diverse contexts.

Once an Excel-based knowledge specification is uploaded into KnowOF, the corresponding parse tools are invoked. The verifier EXUP first checks the spreadsheet knowledge specification against the reference specification regarding syntactical correctness. In case errors are found, the parse and upload process is cancelled and an error report is assembled. KnowOF then displays the *failure* status and makes the report available in the UI. In case no errors are found, EXUP creates an intermediate representation using the comma separated value file format (.csv). Based thereupon, the semantical parser SEMPA first creates a text file that adheres to the general KnowWE formalization syntax. Then, the KnowWE headless app—basically, a KnowWE instance that requires no wiki/UI but runs in the background—is invoked and compiles a d3web KB. Also, SEMPA creates a ProKEt UI specification file *layout.xml* in case UI-related configuration options were provided in the spreadsheet. This parser-provided ProKEt UI specification then is set as the default layout specification for the initial review of the KB in KnowOF, c.f., Figure 6.4, c. However, it is also possible to directly transfer the parsed KB to the repository and investigate it using other, predefined layout specifications, e.g., for the Questionnaire or Daily style.

KnowOF—Word-based KA of Hierarchical Knowledge Word-based KA was originally requested during the JuriSearch project (see also Section 7.1.4). There, the experts were eager to participate in the KA task, yet they demanded to be able to use Microsoft Word. Thus, Word-based KA was particularly tailored towards knowledge for the ITree KBS UI style, required in JuriSearch—that is, hierarchical clarification knowledge that potentially can be abstracted over several levels and corresponding value propagation mechanisms that are realized by certain rules. Therefore, the built-in structuring view of Word allows for intuitively mirroring the hierarchical (abstraction) relations between questions and their refining child-questions. However, apart from that project, the demand for Word-based KA was not as strong as for Excel-based KA. Also, as described above, the Excel-based solution offers a great general flexibility regarding most diverse knowledge types. Thus, Word-based KA was not further generalized and therefore still addresses exclusively ITree clarification knowledge.

Basically, Word-based KA is similar to the Excel-based variant regarding its internal transformation. The core difference is, that it is less flexible regarding the input format adaption. That means, the Word-parser also checks the input document against certain characteristics, yet those are fixed and not easily—i.e., not without adapting the entire parser—extensible. Further, the explicit intermediate .csv representation is missing. That means, the Word-parser firstly performs the syntax check. In the case of failure, the process is cancelled and an error report is generated and made available via KnowOF. In the case of success, the Word-parser directly generates the text-based KnowWE syntax file which is then used by the KnowWE headless app to generate a d3web KB. This again is available via KnowOF, can be transferred to the repository, and then be investigated using the predefined ITree KBS UI style.

b. ProKEt–KnowWE Coupling

Due to the highly flexible KA options, regular quality checks, and a lively developer community, the semantic wiki KnowWE occurred as a natural alternative to KnowOF. For details on basic

KA with KnowWE, see [Baumeister et al., 2011]. Conforming to the demand for intertwined KB and UI development, also a seamless integration mechanism between the KnowWE wiki and ProKEt was investigated in the course of the EuraHS project (see also Section 7.1.3). Therefore, KnowWE was extended as to provide a *direct deploy* feature. There, clicking a deploy-button induces the compilation of the KB from the specified current KnowWE contents. Then, the resulting KB is directly merged with a—previously specified—ProKEt KBS front-end prototype which resides on the same server. Thus, experts can formalize the required knowledge in KnowWE, and—by one simple click—investigate the KB within their desired target UI representation directly and effortlessly.

The direct coupling and thus live preview feature basically was well approved of during the EuraHS project. The domain expert responsible for KA stated that it was much easier to review the potentially highly comprehensive EuraHS KBs within the target KBS UI than based on the KnowWE markup format (basically: Specifically formatted text). Yet, despite an extensive KA workshop we had conducted with the expert for familiarizing with KnowWE, diverse problems kept being reported continuously. This manifested our assumption, that oftentimes the hurdle of learning a new formalization language in an entirely new, unknown environment (such as KnowWE, compared to Excel) is too high and has a rather obstructive effect on the KA process. Thus, even if the EuraHS expert finally managed to mostly autonomously perform KA with KnowWE after some training and workshops, the KnowWE–ProKEt coupling was not yet further generalized. This decision further based on the experiences, that so far Excel-based KA with KnowOF was better accepted and adopted in all other KBSE projects. Thus, in contrast to KnowOF—that allows for parsing KBs and reviewing them within various ProKEt artifacts in a generalized manner—the KnowWE–ProKEt coupling needs to be setup and adapted manually in a project- and target UI specific way if it is to be used for other projects.

Subsumed, we regard both KA extensions of ProKEt as basically highly beneficial. Yet, with respect to a highly autonomous participation of the domain expert, that requires as little additional support by the KBS developer as possible, the standard office based variants seem more appropriate. In case office tools are not desired, however, and especially when a dedicated knowledge engineer can be incorporated and trained in using KnowWE, also the wiki-based KA offers a valuable alternative.

6.2.4 The ProKEt Usability Extension (PUE)

Regarding usability evaluation in general, there exists a vast range of software to date. Examples are tools for collecting and evaluating click-log data or qualitative, e.g., questionnaire-based, data. Such tools are provided both as stand-alone software solutions and as online services. However, most of them need to be installed, configured, or integrated with the website or program to be inspected with at least some additional efforts. Often, also specific additional software or frameworks become required. Finally, many such tools—specifically more comprehensive commercial software or services—also plainly cost (lots of) money. Therefore, ProKEt explicitly and seamlessly integrates selected usability evaluation mechanisms, particularly tailored to the KBS context. Those features can be activated for any ProKEt artifact by simply adding the desired properties to their XML specification. As there are no further installation or other efforts required, this renders the application of usability activities at any point in time a straightforward, affordable task. ProKEt thereby provides the means for both implicitly and explicitly supporting usability.

a. Implicit usability support

Many potential usability flaws can be detected and eliminated by simply applying an iterative development process. As described previously in Section 3.2, this is supported by specifically fostering the development activities according to the EAM, based on extensible prototyping.

b. Explicit Usability Support

Explicit usability support implies the application of denoted usability evaluation techniques as provided by the PUE. This encompasses quantitative and qualitative data collection mechanisms with ProKEt artifacts. All collected usability data—quantitative as well as qualitative—is gathered as a [JSON, n.d.]-based log file for an eased further processing by the PUE analysis module or by manual inspection. Further, the log file data can be exported into a CSV file format. This enables, for example, an eased import of the data into more comprehensive, external statistical tools or spreadsheet programs for further inspection/analysis.

Quantitative Data Collection Quantitative data is collected by a tailored *click logging* mechanism that captures all keyboard action relevant in the context of a KBS session. Those encompass potentially interesting, global actions, such as session management actions. Mainly, however, all activities related to characteristic KBS elements and interactions, e.g., answering a question, are logged with the corresponding timestamp. In the case of consultation sessions, further the respective results of the session are logged, e.g., the proposed diagnose(s).

Qualitative Data Collection The PUE further offers the integration of *(usability) questionnaires* as well as of an *anytime feedback mechanism* for collecting qualitative data.

Questionnaires thereby can be configured either as an additional button or link that opens the corresponding survey on click. Or as to display the questionnaire automatically right after the session has been completed by the user. Corresponding query form widgets are again implemented based on the StringTemplate mechanism. Thus, also such user query forms are easily exchangeable or adaptable for different needs. As examples of standard usability questionnaires, ProKEt currently offers the System Usability Scale/SUS [Brooke, 1996] and the NASA Task Load Index/NASA TLX [Hart, 2006] as out of the box available templates. Also, some more tailored questionnaires are included that were designed to specifically fit the needs of certain KBS case studies—e.g., as performed during the JuriSearch project (see also Section 7.1.4).

Additionally, also an *anytime feedback* mechanism for collecting informal user feedback regarding any desired aspect(s) of the KBS anytime during a KBS session is available. In ProKEt, this is per default implemented as a specific web form for entering respective feedback. That form is invoked by a dedicated button and sends the entered feedback data to a previously defined mail address—e.g., of the developer or main evaluator.

ProKEt Basic Data Analysis Module Apart from collecting several forms of evaluation data, the PUE also seamlessly integrates a simple data analysis module. This calculates selected usability measures automatically based on the quantitative data collected during ProKEt evaluation sessions. Examples are metrics as *average task duration*, *success rate*, or *number of unused widgets*, as introduced by various usability experts, such as [Bevan and Macleod, 1994, Constantine and Lockwood, 1999, Nielsen, 1993a]. Those metrics are in parts slightly adapted as to fit the specific requirements of KBS. For example, *success* is defined as *correctly derived solution compared to the reference/test case* in the context of consultation systems. A listing of the currently supported metrics and their calculation formulae is provided with the ProKEt documentation.

6.3 Encompassing KBSE with ProKEt—Synopsis

In this Chapter, we introduced the tailored prototyping and knowledge systems engineering tool ProKEt. Apart from its technical framework, we explained in what regards ProKEt specifically supports the key activities of encompassing KBSE as proposed earlier in Section 3. Namely: Extensible prototyping, pattern-based development, intertwined development of KB and UI, and denoted usability activities. ProKEt offers a sound, fundamental collection of readily available KBS UI styles and basic UI configuration options out of the box. On the other hand, ProKEt is easily extensible regarding several levels of (programming) expertise and efforts.

Having conceptualized diverse UI presentation options and core KBS UI patterns, and having presented the corresponding development tool ProKEt, we conclude the practical part of this work by reporting the evaluation of both the approach and the tool in the next Chapter 7.

Chapter 7

Evaluation

The evaluation of the proposed research subdivides into three parts, according to the main contributions of this work. First, we discuss the benefits of the tailored KBSE tool ProKEt in Section 7.1. Therefore, we estimate some general benchmarks. Additionally, we report experiences of applying both ProKEt and the proposed agile encompassing process model (EAM), motivated in Section 3.2, in several current KBS projects.

For the ITree UI style, as a specific instantiation of the novel Hierarchical Clarifier pattern, we report the key evaluation activities and the corresponding results, as well as the consequential evolution of ITree itself in Section 7.2.

Apart from ITree, we further have evaluated reference implementations of a selected set of the KBS UI patterns proposed in this book. There, we investigated their overall UI/interaction design and usability, and also their utility and suitability regarding several application contexts. The results thereof are reported in Section 7.3.

In Appendix C of this work, we provide exemplary material for the discussed evaluations. The entire materials, consisting of all applied questionnaires, problem descriptions, task descriptions, and the unabridged tables containing the findings and statistical measures, are made available free of charge on the publication server of the online version of this thesis. The respective link as well as an overview of those materials is provided in Appendix E.

7.1 Evaluation Part I—KBSE with ProKEt

The first part of the evaluation concerns the benefits of using ProKEt as tailored KBSE tool. Therefore, we estimate some development benchmarks in Section 7.1.1, using selected seminal KBS projects as reference implementations.

Further, in Sections 7.1.2–7.1.4 we report three case studies that required the development of tailored KBS solutions: *Mediastinitis*, *EuraHS*, and *JuriSearch*. Those already also have been introduced in previous publications, see [Freiberg et al., 2012, Freiberg and Puppe, 2012a]. We discuss the detailed course of each project with a focus on which activities regarding the EAM have been performed. As it can be recognized, the focus of activities has shifted in the reported projects. This is attributed to the fact, that all projects were started at other points in time and thus the proposed research and tool were evolved differently.

7.1.1 General Benchmark Estimation for ProKEt

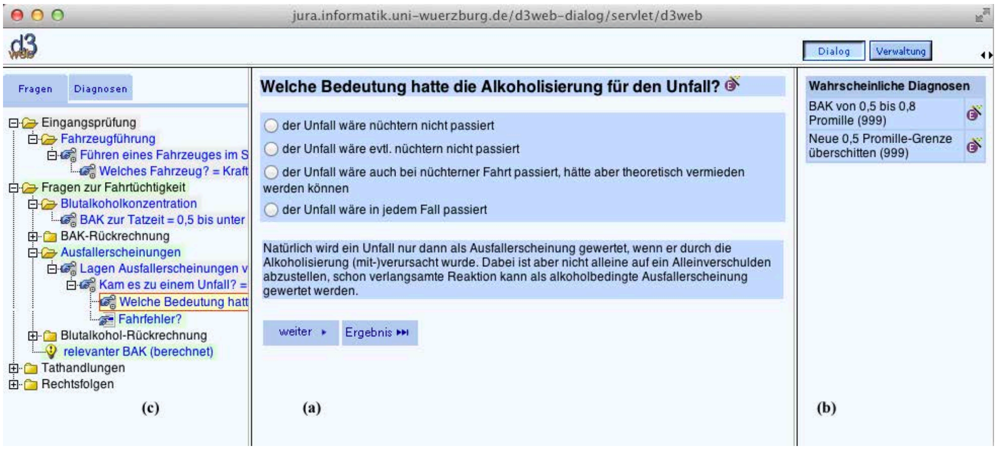
For providing some benchmarks regarding ProKEt-based KBSE, we compare reference values of selected KBS solutions, developed formerly by our group, with the estimated values when implementing comparable solutions with ProKEt today. We investigated: A *Legal Consultation Web Application* (Figure 7.1); *Dialog2* (Figure 7.2); and a *Statistical Folding Dialog*, (Figure 7.3). Where possible, we interviewed the former developers or other persons tightly involved and/or familiar with the project, directly. For assessing the benchmarks, we focussed on the core data input-, results-, and explanation presentation facilities. That is, we deliberately excluded efforts concerning the framing functionality, such as particularly tailored case- or KB management functionality, as well as KA itself. We felt, that this could have easily blurred the overall benchmarks as those tasks exhibit too much potential for variations. It further has to be noted, that those seminal benchmarks denote estimations only. This is due to the fact, that the projects mostly were not precisely documented regarding their development efforts, tasks, or time—e.g., in man hours. Also, the development was mostly specifically targeted for a certain domain or usage context in all cases This consequently resulted in a rather specific, monolithic implementations without the potential of reuse. Regarding the ProKEt benchmarks, those are based on its current development state and available functionalities (June/2014).

Table 7.1 summarizes the key benchmarks for each system in its seminal implementation (prev) and the current estimation based on ProKEt. This regards: The average time in person hours (ph) of implementing and re-implementing the KBS, respectively. The relative ease of adaption and maintenance on a scale from 1 (very easy) to 6 (very hard). And the optimally required expertise for performing adaptions or continuous maintenance (B = beginner, E = expert).

	Legal (Fig.7.1)		**Dialog2 (Fig.7.2)**		**Folding (Fig.7.3)**	
Benchmark	Prev	ProKEt	Prev	ProKEt	Prev	ProKEt
1. Average time	300ph	100/5ph	320 ph	100/5ph	8ph	1ph
2. Ease of adaption	5–6	1–3	4–6	1–3	4–5	1–3
3. Developer expertise	E	B–E	E	B–E	E	B–E

Table 7.1: Benchmark comparison of seminal self-contained (Prev) versus ProKEt-based (ProKEt) KBSE for three reference KBS.

First, an increased efficiency regarding ProKEt-based KBSE can be seen (*Benchmark 1*). Main reason is the strong support for reusing existing solutions in the form of singluar and complexly nested UI modules.



Eingangsprüfung
Fahrzeugführung
Führen eines Fahrzeuges im S
Welches Fahrzeug? = Kraft
Fragen zur Fahrtüchtigkeit
Blutalkoholkonzentration
BAK zur Tatzeit = 0,5 bis unter
BAK-Rückrechnung
Ausfallerscheinungen
Lagen Ausfallerscheinungen v
Kam es zu einem Unfall? =
Welche Bedeutung hatt
Fahrfehler?
Blutalkohol-Rückrechnung
relevanter BAK (berechnet)
Tathandlungen
Rechtsfolgen

Welche Bedeutung hatte die Alkoholisierung für den Unfall?

- der Unfall wäre nüchtern nicht passiert
- der Unfall wäre evtl. nüchtern nicht passiert
- der Unfall wäre auch bei nüchterner Fahrt passiert, hätte aber theoretisch vermieden werden können
- der Unfall wäre in jedem Fall passiert

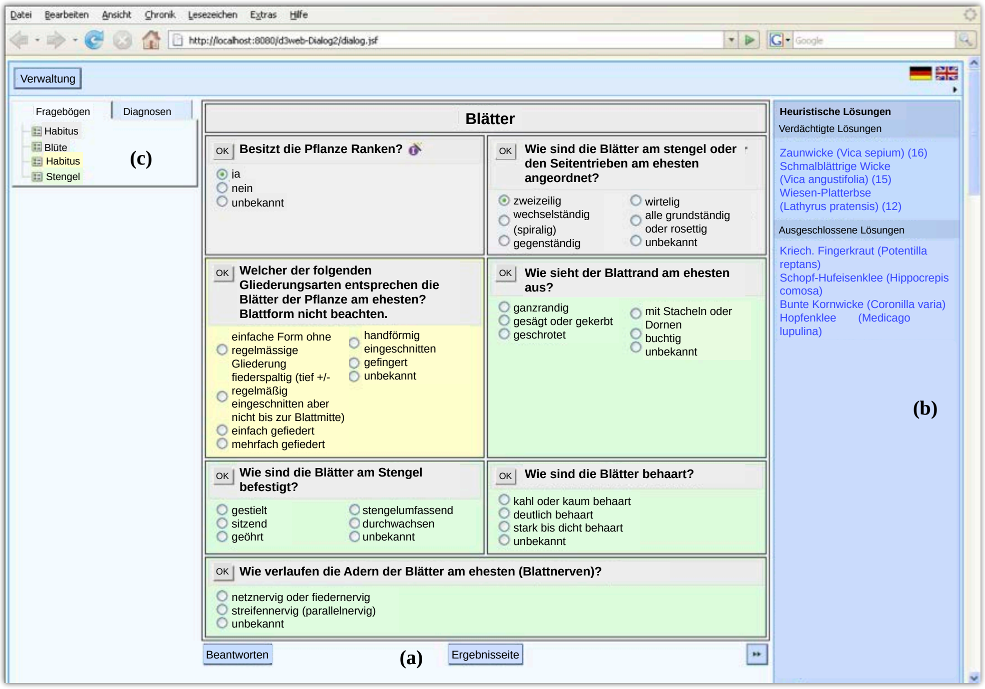
Natürlich wird ein Unfall nur dann als Ausfallerscheinung gewertet, wenn er durch die Alkoholisierung (mit-)verursacht wurde. Dabei ist aber nicht alleine auf ein Alleinverschulden abzustellen, schon verlangsamte Reaktion kann als alkoholbedingte Ausfallerscheinung gewertet werden.

Wahrscheinliche Diagnosen

BAK von 0,5 bis 0,8 Promille (999)

Neue 0,5 Promille-Grenze überschritten (999)

Figure 7.1: Legal web consultation KBS (in german), seminal version; basically conforming to the *Grouped Interview* variant (Section 4.3.3.b., pp. 58 ff.). One question or a question group is displayed at a time (a). Solutions are shown anytime on the right (b). An integrated interview item navigation and -history panel are offered (c).



Habitus
Blüte
Habitus
Stengel

Blätter

Besitzt die Pflanze Ranken?
[✓ ja]
nein
unbekannt

Wie sind die Blätter am stengel oder den Seitentrieben am ehesten angeordnet?
[✓ zweizeilig]
wechselständig (spiralig)
gegenständig
wirtelig
alle grundständig oder rosettig
unbekannt

Welcher der folgenden Gliederungsarten entsprechen die Blätter der Pflanze am ehesten? Blattform nicht beachten.
einfache Form ohne regelmässige Gliederung
fiederspaltig (tief +/- regelmäßig eingeschnitten aber nicht bis zur Blattmitte)
einfach gefiedert
mehrfach gefiedert
handförmig eingeschnitten
gefingert
unbekannt

Wie sieht der Blattrand am ehesten aus?
ganzrandig
gesägt oder gekerbt
geschrotet
mit Stacheln oder Dornen
buchtig
unbekannt

Wie sind die Blätter am Stengel befestigt?
gestielt
sitzend
geöhrt
stengelumfassend
durchwachsen
unbekannt

Wie sind die Blätter behaart?
kahl oder kaum behaart
deutlich behaart
stark bis dicht behaart
unbekannt

Wie verlaufen die Adern der Blätter am ehesten (Blattnerven)?
netznervig oder fiedernervig
streifennervig (parallelnervig)
unbekannt

Heuristische Lösungen

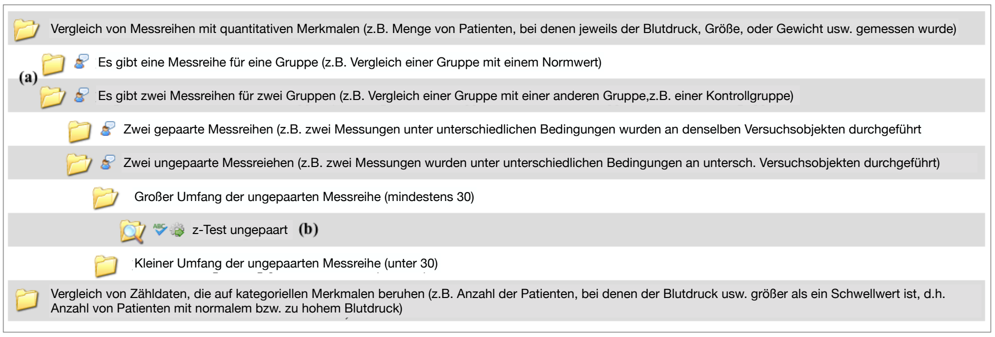
Verdächtigte Lösungen

Zaunwicke (Vica sepium) (16)
Schmalblättrige Wicke (Vica angustifolia) (15)
Wiesen-Platterbse (Lathyrus pratensis) (12)

Ausgeschlossene Lösungen

Kriech. Fingerkraut (Potentilla reptans)
Schopf-Hufeisenklee (Hippocrepis comosa)
Bunte Kornwicke (Coronilla varia)
Hopfenklee (Medicago lupulina)

Figure 7.2: Dialog2 consultation KBS (in german), resulting from a master's thesis [Krawczyk, 2007]. Basically similar to the legal consultation KBS (Figure 7.1), yet using more recent Java Server Faces technologies.


Vergleich von Messreihen mit quantitativen Merkmalen (z.B. Menge von Patienten, bei denen jeweils der Blutdruck, Größe, oder Gewicht usw. gemessen wurde)

Es gibt eine Messreihe für eine Gruppe (z.B. Vergleich einer Gruppe mit einem Normwert)

(a)

Es gibt zwei Messreihen für zwei Gruppen (z.B. Vergleich einer Gruppe mit einer anderen Gruppe,z.B. einer Kontrollgruppe)

Zwei gepaarte Messreihen (z.B. zwei Messungen unter unterschiedlichen Bedingungen wurden an denselben Versuchsobjekten durchgeführt

Zwei ungepaarte Messreiehen (z.B. zwei Messungen wurden unter unterschiedlichen Bedingungen an untersch. Versuchsobjekten durchgeführt)

Großer Umfang der ungepaarten Messreihe (mindestens 30)

z-Test ungepaart (b)

Kleiner Umfang der ungepaarten Messreihe (unter 30)

Vergleich von Zähldaten, die auf kategoriellen Merkmalen beruhen (z.B. Anzahl der Patienten, bei denen der Blutdruck usw. größer als ein Schwellwert ist, d.h. Anzahl von Patienten mit normalem bzw. zu hohem Blutdruck)

Figure 7.3: Folding Dialog for statistical calculation recommendation (in german)—seminal interactive prototype. This implementation basically corresponds to the *Hierarchical Interview* pattern (Section 4.3.3.c., pp. 60 ff.). Question nodes are expanded by clicking the folder icons (a), solutions thus are reached at an inner tree levels (b) automatically.

This further induces the way more positively estimated efforts for adapting and maintaining ProKEt-based KBS solutions (*Benchmark 2*). Especially beneficial is the fact, that ProKEt already supports a collection of diverse interesting KBS UI patterns along with simple to comprehensive possibilities of their fine-tuning and adaption/extension. Admittedly, when realizing an entirely novel KBS type, or not yet available framing functionality, the initial efforts increase also for ProKEt. Then, naturally, the tool first has to be extended itself. Yet, regarding each further, similar implementation, where those extensions then can be (re)used, the increase in efficiency and ease of development accumulates quickly (see, e.g., the EuraHS project, Section 7.1.3, pp. 114 ff.). This strongly contrasts the remarks of the interviewed former developers, who rather dissuaded from reviving and/or adapting the seminal systems today at all—due to partly outdated technologies, and due to rather weak, not easily extensible application architectures. That fine-tuning of ProKEt artifacts on diverse levels also leads to the improvement regarding the required developer expertise (*Benchmark 3*). In contrast, all interviewed developers of the former KBS stated the need for expert developers, due to the overall complexity and little modularity of those projects.

For justifying specifically Benchmark 1, average KBSE time, we now summarize the main tasks for re-implementing each system with ProKEt in more detail. The *legal consultation web application* (Figure 7.1) and the *Dialog2 web application* (Figure 7.2) basically are quite similar. Both systems integrate a self-contained d3web KB into the consultation application UI. Thereby, the legal web application is constrained to a fixed set of predefined KBs for which it is also optimized. In contrast, Dialog2 basically allows for loading arbitrary KBs. Both implementations comply to the *Grouped Interview* pattern variant (Section 4.3.3.b.): They always retrieve and display only the single next active question, c.f., Figure 7.1, or questionnaire/group, c.f., Figure 7.2. Both KBS offer an interview item- and a solution-based navigation panel on the left. When clicking on an interview item, the KBS displays the respectively chosen item(s) in the centered core input module for immediate processing—and thus allows users to deviate from the suggested interrogation sequence. Clicking on a solution in turn brings up the current explanation for the solution. The interview item navigation also denotes a form of interview history, as questions there are highlighted according to their answer status. Yet, this is a rather simple variant thereof, as only questions and not the particular answers are listed. Finally, both implementations integrate an anytime solution panel on the right-hand side.

For re-implementing those systems, we estimate a ProKEt-based value of about 100 and 5 person hours (ph), respectively. The first value relates to the fact, that ProKEt currently offers a ready-to-use solution for strict single-question interview KBS, but not yet fully supports the described grouped interview style. Thus, the tool would have to be expanded with respect to suitable HTML templates, renderers, specific interactions, and style adaptions first, thus inducing the (initially) high number of hours. Also, widgets for the interview item and solution navigation menus are not yet available and require some additional work. In contrast, a solution panel implementation, that can be configured for anytime display by property, is already available in ProKEt out of the box and can easily be integrated. However, once the *Grouped Interview* pattern variant and the navigation widgets are fully integrated with ProKEt, re-implementing a similar system would require far less efforts. We estimate about 5ph, as in this case several more configuration options—e.g., number of columns, anytime solution display, detail configuration of the solution/explanation presentation, etc.—need to be considered than, e.g., for a re-implementation of the Folding Dialog.

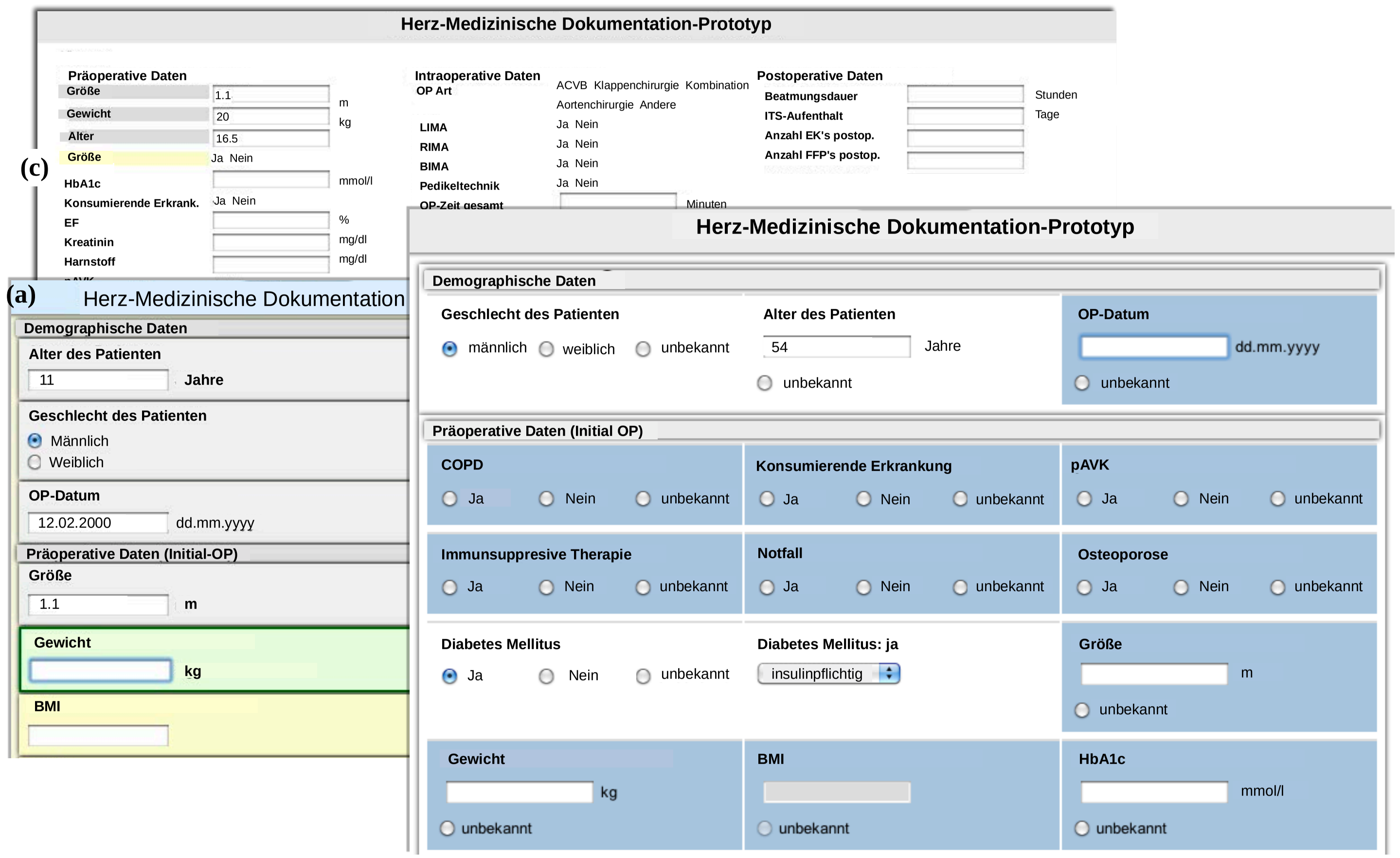
Regarding the statistical Folding Dialog, the ProKEt-based average time benchmark is very low, as the corresponding base pattern *Hierarchical Interview* (Section 4.3.3.c., pp. 60 ff.) is already fully supported by the tool. Thus, the creation of another instance of that system is as simple as defining the respective (new) KB. Concerning the UI, at most little efforts would be required. Examples could be a further fine-tuning of the presentation style, used icons, coloring, etc.

7.1.2 Mediastinitis—Intelligent Medical Documentation

General Information Mediastinitis [Mediastinitis DGTHG, n.d.] is a german national project started in 2011. The goal of the project is to improve patient care in the cardiac medical context. Therefore, data of patients, that are affected by a wound healing process disorder after a cardiac surgery in 26 participating clinics in Germany, are collected into a central database. At regular distances—currently, once a year—that data is statistically evaluated for developing appropriate future treatment strategies.

The data is collected by the medical doctors themselves, which poses several important objectives: Data entry needs to be straightforward, smooth, and efficient. Flexibly resuming a data entry session—e.g., previously discontinued due to probable interruptions during stressful daily medical routine—is mandatory. Ensuring a high data quality regarding correctness and completeness is another key requirement as future treatment strategies are developed based on the data analyses. The basic development activities conformed to the proposed EAM (Section 3.2, pp. 34 ff.). Yet, this excluded usability-related activities, which at that point in time were neither integrated into the development model, nor fully supported by in ProKEt.

Course of the Project—Front-End Prototyping A medical doctor, who was both reference person of the project as well as future end-user of the system, created a preliminary knowledge specification: A rather informal spreadsheet listing of an exemplary test question set. Based thereupon, three initial KBS prototypes were developed, as shown in Figure 7.4. Those basically all denoted variants of the *Questionnaire base pattern*.



(c)

Herz-Medizinische Dokumentation-Prototyp

Präoperative Daten

Größe	1.1	m
Gewicht	20	kg
Alter	16.5	
Größe	Ja Nein	
HbA1c		mmol/l
Konsumierende Erkrank.	Ja Nein	
EF		%
Kreatinin		mg/dl
Harnstoff		mg/dl

Intraoperative Daten

OP Art	ACVB Klappenchirurgie Kombination Aortenchirurgie Andere	
LIMA	Ja Nein	
RIMA	Ja Nein	
BIMA	Ja Nein	
Pedikeltechnik	Ja Nein	
OP-Zeit gesamt		Minuten

Postoperative Daten

Beatmungsdauer		Stunden
ITS-Aufenthalt		Tage
Anzahl EK's postop.		
Anzahl FFP's postop.		

(a)

Herz-Medizinische Dokumentation

Demographische Daten

Alter des Patienten
11 Jahre

Geschlecht des Patienten
[✓ Männlich] Weiblich

OP-Datum
12.02.2000 dd.mm.yyyy

Präoperative Daten (Initial-OP)

Größe
1.1 m

Gewicht
kg

BMI

Herz-Medizinische Dokumentation-Prototyp

Demographische Daten

Geschlecht des Patienten: [✓ männlich] weiblich unbekannt	Alter des Patienten: 54 Jahre; unbekannt	OP-Datum: dd.mm.yyyy; unbekannt

Präoperative Daten (Initial OP)

COPD: Ja Nein unbekannt	Konsumierende Erkrankung: Ja Nein unbekannt	pAVK: Ja Nein unbekannt
Immunsuppresive Therapie: Ja Nein unbekannt	Notfall: Ja Nein unbekannt	Osteoporose: Ja Nein unbekannt
Diabetes Mellitus: [✓ Ja] Nein unbekannt	Diabetes Mellitus: ja: insulinpflichtig	Größe: m; unbekannt
Gewicht: kg; unbekannt	BMI: unbekannt	HbA1c: mmol/l; unbekannt

Figure 7.4: The three initial prototypes of the Mediastinitis-Registry (in german): One-column Box Questionnaire style (a), three-column Box Questionnaire style (b), three-column Daily Questionnaire style (c).

The *first prototype*, Figure 7.4, a, is a Box Questionnaire (Section 4.3.2. a.) variant in an entire single-column style. That is, with regards to both the global UI frame as well as to the questionnaire/question configuration. This results in a strict top-down listing, resembling to paper-based questionnaires. Status mediation in this variant is provided by a distinctive coloring scheme: Gray for answered questions, green for the current active question, and yellow for yet unanswered questions. This adheres to our proposition regarding suitable coloring schemes for KBS status mediation support, see Section 4.1, pp. 39. Further, non-indicated items are hidden per default and only inserted into the interrogation sequence once activated. Abstraction questions, in contrast, are displayed anytime, yet distinctly greyed for highlighting their state. This prototype did not use any default answer option. However, the strict one-column style led to a lot of wasted UI space, see Figure 7.4, a, question *COPD*—especially considering today's increasing monitor sizes and display resolutions. Consequentially, the overall UI grew very lengthy. This required lots of scrolling and decreased the overall compactness. That in turn reduced the overall status mediation ability of the KBS, which again deteriorated the self-descriptiveness of the system.

Status mediation and self-descriptiveness, however, are generally postulated as important usability characteristics of a system, e.g., by [Nielsen, 1994] (1st heuristic) or the [ISO 9241–110, 2006] (norm 2). Therefore, we experimented with a more space efficient variant of Box Questionnaire for the *second prototype*: A three-column style regarding both questionnaires and questions. That is, three questions per questionnaire, and three answer options per question in a row, as shown in Figure 7.4, b. This seemed especially feasible as most questions contained only two to three answers at maximum. Some particular questions with more than three answer options were represented by drop-down lists, thus additionally saving display space, see Figure 7.4, b, question *Diabetes mellitus: ja* in the center of the figure. Overall, this led to a very consistent and structured appearance, as all choice questions either displayed three options, or a drop-down widget—thus conforming to [Lidwell et al., 2010, p. 56] who see both aesthetic and functional consistency as important usability fostering characteristics. Also, a different coloring scheme was applied, now only indicating already answered (white) and still remaining (blue) questions, excluding the highlighting of the currently active question. This mainly intended to reduce the extraneous cognitive load of users, c.f., [Whitenton, 2013], by removing the distinction between the next suggested and further remaining questions—unnecessary in Questionnaire anyhow where users deliberately are free to answer questions in any desired sequence. Another reason was to simply experiment with an overall more decent coloring scheme, yet still adhering to suitable KBS coloring schemes as proposed in Section 4.1, p.39. Further, a default answer alternative *unknown [unbekannt]* was added in this prototype, see Figure 7.4, b. Regarding non-indicated- and abstraction items, the previous configuration was preserved.

As a *third alternative* for an even more effective exploitation of available UI space, we experimented with a three-column Daily variant (Section 4.3.2.b.), as depicted in Figure 7.4, c. That resulted in three questionnaire columns next to each other, which in this case was feasible as all the question and answer texts were rather short. Questions were displayed in the juxtaposed, flat Daily manner. The applied alignment of question/answer elements thereby supported a better, visual structure, as suggested by [Lidwell et al., 2010, p. 24]. Again, a different coloring scheme was tested, adhering to the suggestions of Lidwell [Lidwell et al., 2010, p. 48]. This encompassed indicating answered questions (grey, for unobtrusively grouping done items), the currently active question (yellow, for highlighting the active one with a warm, foreground color), and leaving not yet processed questions completely uncolored (again for reducing overall cognitive load). Indication- and abstraction questions were handled as in the other two variants.

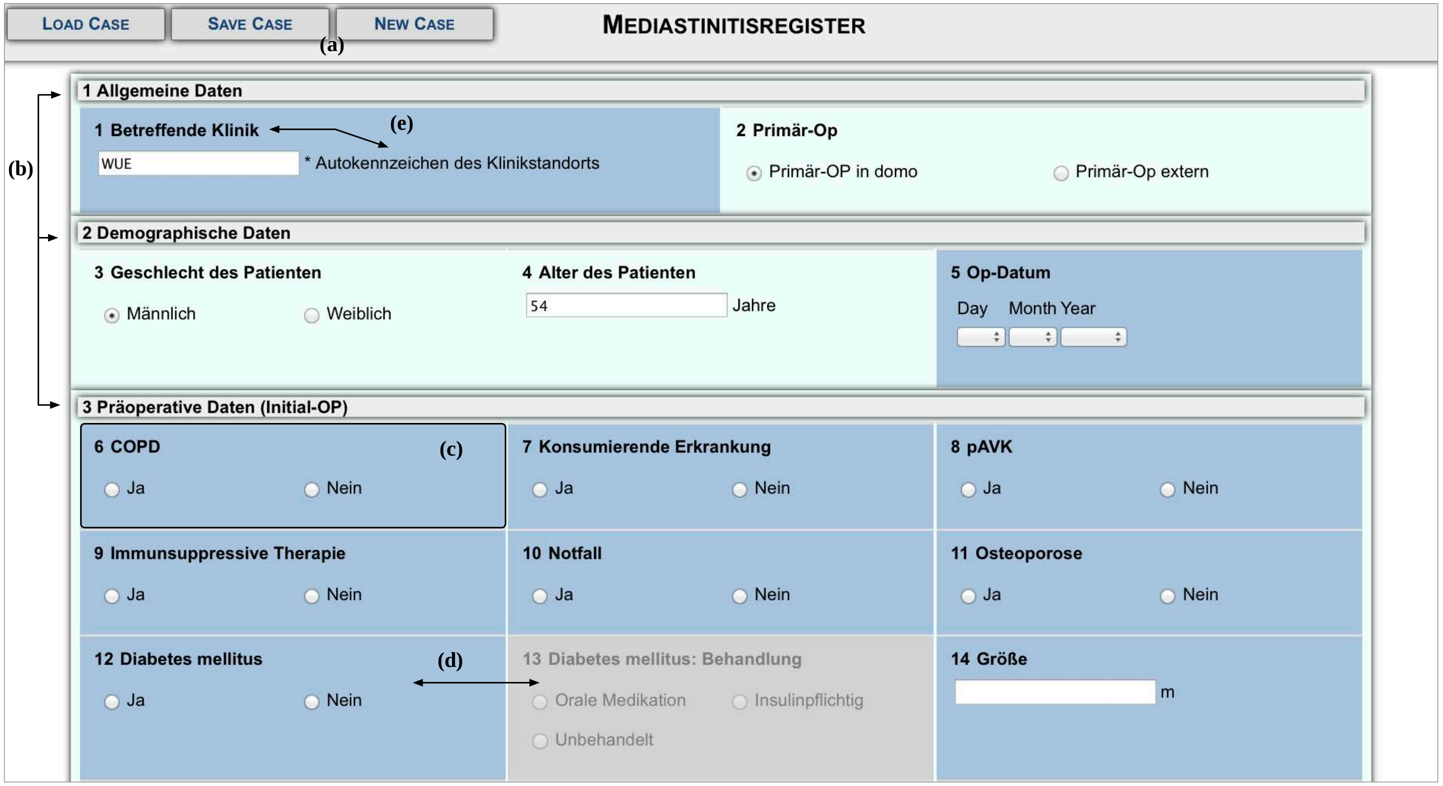
Experimenting with those three different UI alternatives was greatly facilitated by ProKEt: Once the initial prototype was specified with respect to its knowledge, configuring the pattern presentation—e.g., regarding the number of columns—was as easy as changing the corresponding UI properties for the prototype in ProKEt. The three prototypes finally were made available on a demo-server, where they were reviewed by the expert—by actively using them for entering exemplary patient data. In the end, the three-column Box Questionnaire variant—Figure 7.4, b—was decided to match the requirements best. With the determination of the KBS use case *documentation* and the basic KBS UI pattern *(3-column) Questionnaire*, the System Metaphor according to our definition in Section 3.2 was completely specified.

Course of the Project—Extensible Prototyping Based on the chosen prototype, iterative extensible prototyping began—corresponding to the cyclic part of the EAM (see Section 3.2). In the first iteration, the initial informal knowledge specification was formalized into a d3web KB using the semantic wiki KnowWE [Baumeister et al., 2011]. Regarding the UI, the main request were basic session- and user management facilities—to that date not yet contained in the ProKEt toolkit as ready-to-use framing functionality. Thus, as a first step, button dummy widgets were added to the dialog header (c.f. Figure 7.5, a) as to make an initial proposition *where* and *how* to integrate those features without having to fully implement the entire mechanisms. Those dummies demonstrated a basic look and feel regarding the button triggers and the corresponding dialog windows, yet they still were dysfunctional. The developed KB and UI solution were then merged, and the resulting core KBS prototype was made available on the server to be reviewed by the expert again.

In the second iteration, we experimented with omitting the default *unknown* option again. Also, non-indicated questions were adapted to be displayed in an anytime greyed manner (c.f., Figure 7.5, b)—for avoiding an oftentimes changing UI when flexibly integrating follow-up questions, and thus for preserving more presentation constancy. Concerning the KB, also a more extensive refactoring was requested: On the one hand, it was extended by additional questions and questionnaires, as well as by auxiliary information for various elements (e.g., Figure 7.5, e); other, already existent, parts of the KB were renamed or restructured.

Hereafter, several more development cycles followed, during which both UI and KB were further refined step by step. The UI-related tasks consisted of gradually implementing and extending the session-/user management mechanisms and corresponding widgets, of fine-tuning the grid-based styling of further experimenting with the color scheme, and finally of completely hiding not yet indicated questions instead of displaying them in a disabled manner. Regarding the KB, mostly a reformulation of existing content or an adaption of the questioning sequence was required. As before, each iteration was finished by creating a new version of the productive KB, installing it on the server, and requesting the expert review.

The final implementation of the Mediastinitis-Registry is presented in Figure 7.5. The core dialog consists of diverse question types: Textual-, numerical-, date-, OC-, and MC questions. Each question is rendered as one coherent box of the dialog grid (Figure 7.5, c), which applies a three-/two-column style (question-/answer-based). Regarding the coloring scheme for sta-



LOAD CASE | SAVE CASE | NEW CASE

MEDIASTINITISREGISTER

1 Allgemeine Daten

1 Betreffende Klinik
WUE * Autokennzeichen des Klinikstandorts

2 Primär-Op
[✓ Primär-OP in domo] Primär-Op extern

2 Demographische Daten

3 Geschlecht des Patienten
[✓ Männlich] Weiblich

4 Alter des Patienten
54 Jahre

5 Op-Datum
Day Month Year

3 Präoperative Daten (Initial-OP)

6 COPD
Ja Nein

7 Konsumierende Erkrankung
Ja Nein

8 pAVK
Ja Nein

9 Immunsuppressive Therapie
Ja Nein

10 Notfall
Ja Nein

11 Osteoporose
Ja Nein

12 Diabetes mellitus
Ja Nein

13 Diabetes mellitus: Behandlung
Orale Medikation Insulinpflichtig
Unbehandelt

14 Größe
m

Figure 7.5: Final implementation of the Mediastinitis-Registry (in german)—see pp. 112 f., for a detailed description.

tus mediation, the experts demanded an adaption of the proposed blue-white scheme of the second prototype: By coloring answered questions in a light green hue. This might be not optimal theoretically, c.f. [Lidwell et al., 2010, p. 48], as a darker blue and a light green do not well conform to any of the proposed, aesthetic color combinations. However, it was explicitly demanded by the experts, thus in accordance with the aesthetic-usability effect, see [Lidwell et al., 2010, p. 20], we complied to the experts wishes according to their perception of aesthetic colors. Questionnaires provide additional structuring (Figure 7.5, b). In contrast to the foundational prototype (Figure 7.4, b), the answer alternative *unbekannt [unknown]* was finally omitted. Additional functionality regarding session- and user management was fully implemented and is available via button widgets (Figure 7.5, a) placed prominently in an always visible header part of the UI. Non-indicated questions are per default displayed in a greyed manner as to indicate their special status (Figure 7.5, d)—those become activated, and thus displayed in standard style—as soon as the respective triggering questions are answered. Regarding specific questions, auxiliary information in the form of explanatory text is available via an icon trigger, that opens an overlaying popup for displaying named information, or integrated directly next to the respective object (Figure 7.5, e).

Mediastinitis has been successfully used in practice for more than two years at the time of this writing. Also, since about late 2012, its implementation has not changed much anymore—except minor, general updates due to basic ProKEt updates. Once a year, typically at the beginning, a statistical base analysis is generated regarding the average answer distributions for all questions. That is, the frequency each choice of choice questions is chosen, the average values for numerical questions, and so on. This is presented and discussed at a national congress on cardiac medicine.

7.1.3 EuraHS—Intelligent Medical Documentation

General Information [EuraHS, n.d.] is a project cooperation with the European Hernia Society (EHS). Its main goal is to improve patient care and increase knowledge regarding the practice of abdominal wall hernia surgery. Therefore, relevant data are collected and statistically evaluated as to gain insights regarding hernia surgery practice in the participating mostly european countries. Similarly to the Mediastinitis-Registry—see previous section—the data are entered by the medical doctors and other medical staff themselves, thus posing the same, basic requirements: Straightforward, smooth, and efficient data entry, easily resumable data entry sessions due to likely interruptions, and assurance of a high data quality regarding correctness and completeness. Compared to Mediastinitis, however, there were some additional requirements.

- *Image Questions:* Questions that display an image with clickable areas that correspond to the answer alternatives. Those were anticipated particularly beneficial in the medical context where answers can often be stated more precisely by an appropriate image than by plain text.
- *Input Summaries:* For enhancing the overall KBS status mediation regarding already entered data, both plain finding lists, and more compact grid-based, visual tables were requested. The latter adhered to tabular forms, the medical doctors also used (paper-based, offline) in their daily routine.

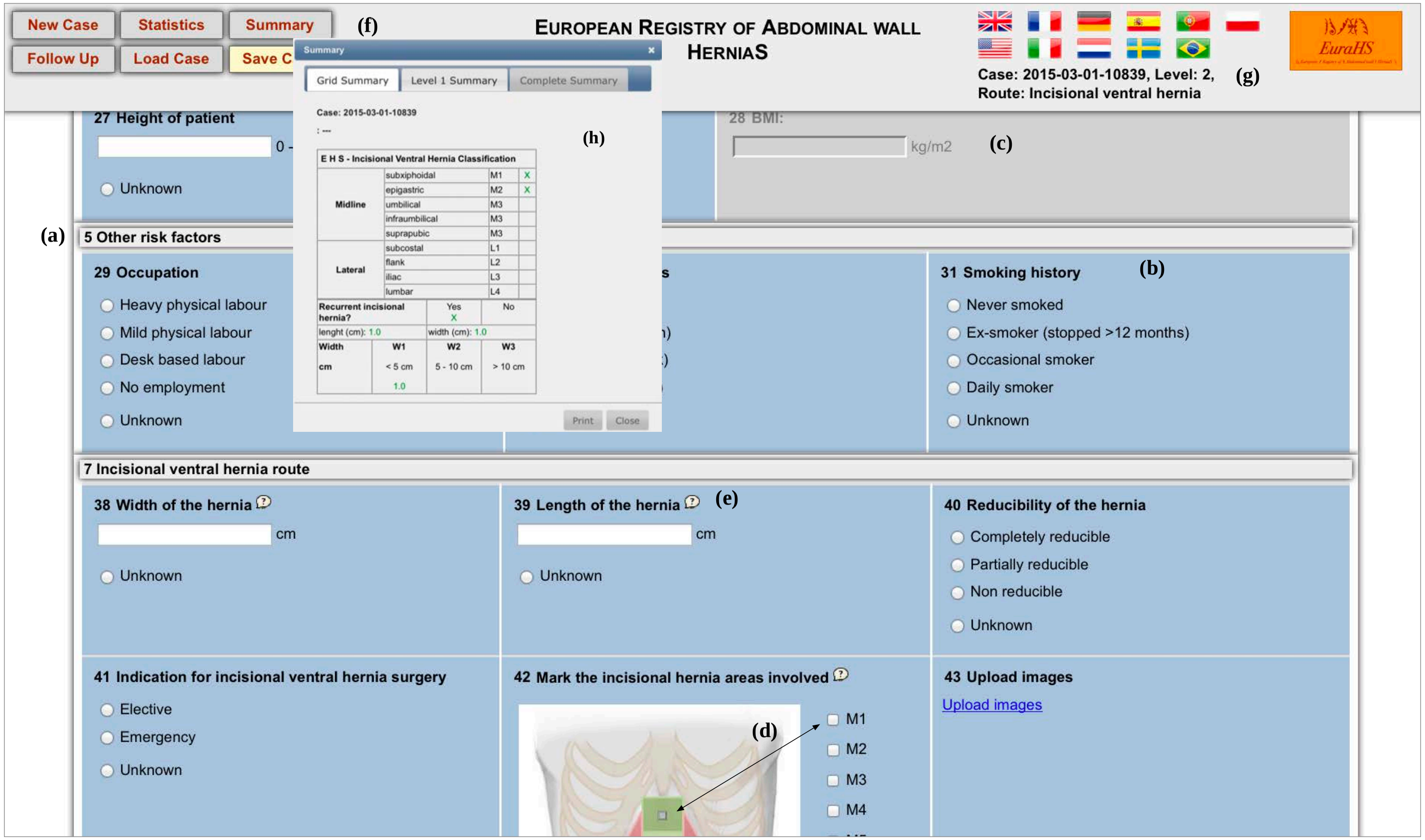
- *Extended Auxiliary Information:* Apart from plain pop-ups with textual explanations, also the pop-up display of images, tables, or PDF files was demanded.
- *Multilingualism:* For EuraHS as a european-wide project, the availability of diverse languages was another key requirement. It was decided, that this should concern only the core dialog contents, i.e., question, answer, and auxiliary information texts. Framing UI elements—such as a *Save* button—should be entirely kept in english as we assumed that such base functionalities and corresponding widget labels would be easily understood by all participating nationalities.
- *Follow-up Reminder Mechanism:* Notifying users that a defined timespan, e.g., one year, has passed since last entering data in the respective case. Then, users can add further data that had not been relevant in the previous session(s)—e.g., regarding follow-up examinations of the healing process.
- *Dynamic KBS Features:* In EuraHS, the indication mechanisms are much more complex when compared to Mediastinitis. EuraHS basically consists of multiple fundamental examination paths—*routes*—that differ regarding the core topic. Those are activated by answering a dedicated starting question (i.e., question indication). Each route in turn contains several topically related questionnaires. This mechanism ensures a basic reduction of the displayed questions to only the relevant set.
- *Two-level concept:* Required for all routes, the level concept mainly affects the number of questions included in the respective questionnaires. Thereby, the short level is intended to persuade also doctors with little spare time to participate in the clinical studies based on EuraHS by entering their operation data.

EuraHS Prototype and Front-End Prototyping EuraHS basically resembled Mediastinitis in many aspects: Concerning the base application type (documentation KBS) or its usage context/target users (in the hospital during or after operations, used by medical staff). As the familiar Box Questionnaire pattern had proven valuable for Mediastinitis, this naturally was our first proposition also for EuraHS. Thus, we adapted a prototype, that basically was very similar to the second Mediastinitis prototype (blue-white coloring scheme, default unknown answer option), with regards to initial exemplary data and presented this to the corresponding expert. Again, this was a medical doctor and future user of the system. Having actively tested the prototype on a dedicated demo-server, the proposed solution was agreed on as suitable foundation. This settled the same basic UI style (*Box-Questionnaire*) and KBS type (*documentation KBS*)—and with it, the System Metaphor—as in Mediastinitis.

Course of the Project—Extensible Prototyping In the first, following development cycle, again an initial version of the d3web KB was created by manually transferring the semi-formal Word specification into KnowWE [Baumeister et al., 2011]. Thereby, only a small part of the first route was included for testing reasons. Also the level concept was skipped temporarily as to get a working version of the system with some real knowledge—and thus a realistic impression of the future system—as soon as possible. Additionally, some minor requests regarding the UI were already considered. For example, omitting the unknown answer option and listing the answer alternatives of questions vertically. The resulting productive KBS was updated on the demo server for expert review. The feedback once more positively confirmed the suitability of the targeted KBS. In the next iteration, the KB was expanded as to fully cover the first route of the system. Further, the header of the system was adapted as to include additionally required button widgets, e.g., regarding the follow-up reminder mechanism. Image questions still were only roughly mimicked by displaying the desired image next to a multiple choice question. That way, the future appearance of image questions could be demonstrated without having to fully implement the mechanisms instantly. After renewed confirmation by the expert, image questions were fully implemented in the next iteration. Also, there was some fine-tuning of the UI. Examples are the continuous numbering of questionnaires and questions, the styling/arrangement of the buttons in the header, the inclusion of a logo and of flag icons for the future multilingualism feature, and a display for the current route and level within the header. In subsequent iterations, also the mentioned header widgets—e.g., for displaying the summaries, the follow-up reminder, and for linking to a statistics evaluation submodule—were entirely implemented. Further, the KB was gradually extended as to cover the remaining routes as well as the two-level concept.

In the further course of the project, a dedicated project manager was inaugurated. Apart from organizational stuff, she was mainly intended to support further KA tasks and to evaluate and cross check the respectively updated KBS. Thus, we offered her a two-days workshop at our institute where she was taught basic KA with KnowWE. Furthermore, we implemented a direct coupling between the EuraHS KnowWE instance (containing exclusively the EuraHS KB) and ProKEt. This allowed for transferring an updated KB directly into the EuraHS KBS front-end implementation by simply clicking a button within the wiki. That way, the project manager was provided valuable tools for refining the KB mostly autonomously. This mainly included adaptions such as: Renaming questions, altering the question order, introducing auxiliary information texts, or specifying the language counterparts for the KB. Regarding highly complex KB issues—e.g., re-structuring the levels-mechanism—we still provided knowledge engineering support. Yet overall, the active involvement of the project manager allowed us to mainly concentrate on UI refinement and implementation of additional software features (e.g., encrypted login). This consequently led to a more efficient, overall development process.

The current state of the EuraHS documentation system is depicted in Figure 7.6. The KBS exhibits a basic similarity to Mediastinitis, applying also the Box Questionnaire pattern. In contrast, it uses the originally proposed blue-white coloring scheme that was basically motivated in Section 4.1, a default answer option unknown, and a one column style for answer options. The latter was mostly due to the overall lengthier answer texts in EuraHS, which did not render well with a multi-column style for answers. The header additionally consists of widgets for presenting the more comprehensive framing functionality widgets (summary, statistics, session management, follow up, see Figure 7.6, f) and multilingualism support (Figure 7.6, g). Regarding the core dialog, questionnaires (Figure 7.6, a) are used for grouping related questions (e.g., *Other risk factors* in the figure). Thereby, up to at most three questions—each contained in a framing box along with their answer alternatives (Figure 7.6, b)—are displayed in one row of the questionnaire, adhering to the three-/single-column (question/answers) style. For offering more overview and orientation, questionnaires and questions are numbered continuously. Abstraction questions are displayed anytime greyed (Figure 7.6, c), and non-indicated objects remain hidden until particularly activated by indication knowledge. In addition to the basic question types—textual, numeric, date, one choice, multiple choice—also image questions are



EUROPEAN REGISTRY OF ABDOMINAL WALL HERNIAS

Case: 2015-03-01-10839, Level: 2,
Route: Incisional ventral hernia

Summary

Grid Summary | Level 1 Summary | Complete Summary

Case: 2015-03-01-10839

: ---

E H S - Incisional Ventral Hernia Classification			
Midline	subxiphoidal	M1	X
	epigastric	M2	X
	umbilical	M3	
	infraumbilical	M3	
	suprapubic	M3	
Lateral	subcostal	L1	
	flank	L2	
	iliac	L3	
	lumbar	L4	

Recurrent incisional hernia?	Yes X	No	
lenght (cm): 1.0	width (cm): 1.0		
Width	W1	W2	W3
cm	< 5 cm	5 - 10 cm	> 10 cm
	1.0		

27 Height of patient
0 -
Unknown

28 BMI:
kg/m2

5 Other risk factors

29 Occupation
- Heavy physical labour
- Mild physical labour
- Desk based labour
- No employment
- Unknown

31 Smoking history
- Never smoked
- Ex-smoker (stopped >12 months)
- Occasional smoker
- Daily smoker
- Unknown

7 Incisional ventral hernia route

38 Width of the hernia
cm
Unknown

39 Length of the hernia
cm
Unknown

40 Reducibility of the hernia
- Completely reducible
- Partially reducible
- Non reducible
- Unknown

41 Indication for incisional ventral hernia surgery
- Elective
- Emergency
- Unknown

42 Mark the incisional hernia areas involved
- M1
- M2
- M3
- M4

43 Upload images
Upload images

Figure 7.6: EuraHS, current state—for a detailed description, see pp. 116, f.

realized (Figure 7.6, d). Those allow for either selecting the clickable image regions or for selecting the choices out of the list next to the image. Also, both the requested compact grid view (Figure 7.6, h—Grid Summary) and a plain findings listing were realized as summary widgets. Auxiliary information is included via automatically triggered popups when hovering the info-icons (Figure 7.6, e).

EuraHS is an ongoing, highly active project. In May 2012, a launch symposium was given for officially introducing the EuraHS system to the european hernia medical community (and thus, future users). So far, both in 2013 and in 2014, two dedicated clinical studies were started—the *class of 2013/2014*. Thus adaptions of the system now are only been undertaken very carefully on the live server, as to avoid any incidents or system faults that may obscure the results of the current study. Also, we have set up an identical test server for continuing the development in a safe manner.

Currently, the *KB is expanded* quite extensively again as to broaden the topics covered by the registry and thus to attract even more end-users. This in turn can help to render any derived statistics and evaluation of the entered data even more meaningful. Therefore, the *KA method is switched* from the wiki KnowWE towards the spreadsheet-based parse tool KnowOF. Thus, the (trained) project manager is enabled to actively support KA; however, now also interested medical doctors themselves who most likely are familiar with spreadsheet software are enabled to perform basic KA tasks. Regarding those additional KB modules, we further investigate the *suitability of a tailored Daily Questionnaire UI*. Another request is the timely *extension of the multilingualism* support, also with the goal to open the registry for additional potential users within Europe (and beyond). So far, EuraHS supports english, french, german, spanish, polish, and italian; portuguese and dutch are under development at the time of this writing, and several more languages such as czech and chinese are requested. Other issues are the realization of a *more secure data encryption* (based on anonymized storage of the entered cases), or the further *refinement and extension of the summaries*.

7.1.4 JuriSearch—Legal Clarification Consultation

General Information In early 2012, the JuriSearch project was initiated as a cooperation between the university of Würzburg and the RenoStar legal counseling corporation. The project is partly founded by the Free State of Bavaria. JuriSearch aims at building a web-based consultation framework for the legal domain. Potential target users are diverse. Examples include legal laymen, searching for a basic understanding/estimation of their case; fresh lawyers that seek for guidance regarding legal (sub)domain(s) that are not their special field of work; or trained legal service support staff that want to quickly estimate legal cases regarding the appropriateness of further steps and potential profitability. Thus, a careful and universal UI/interaction design—in the sense of satisfying a diverse user group—and a high level of usability were prime important factors.

The envisioned internet platform intends to provide an entrance module that supports interested users in locating and specifying the respective concrete legal topic—e.g., by offering topical listings, or mind maps. As each such legal topic in itself is rather comprehensive, separate clarification modules for each topic are provided. JuriSearch thereby covers manifold legal (sub-)domains, e.g., the right of cancellation, the law of tenantry, or the social laws.

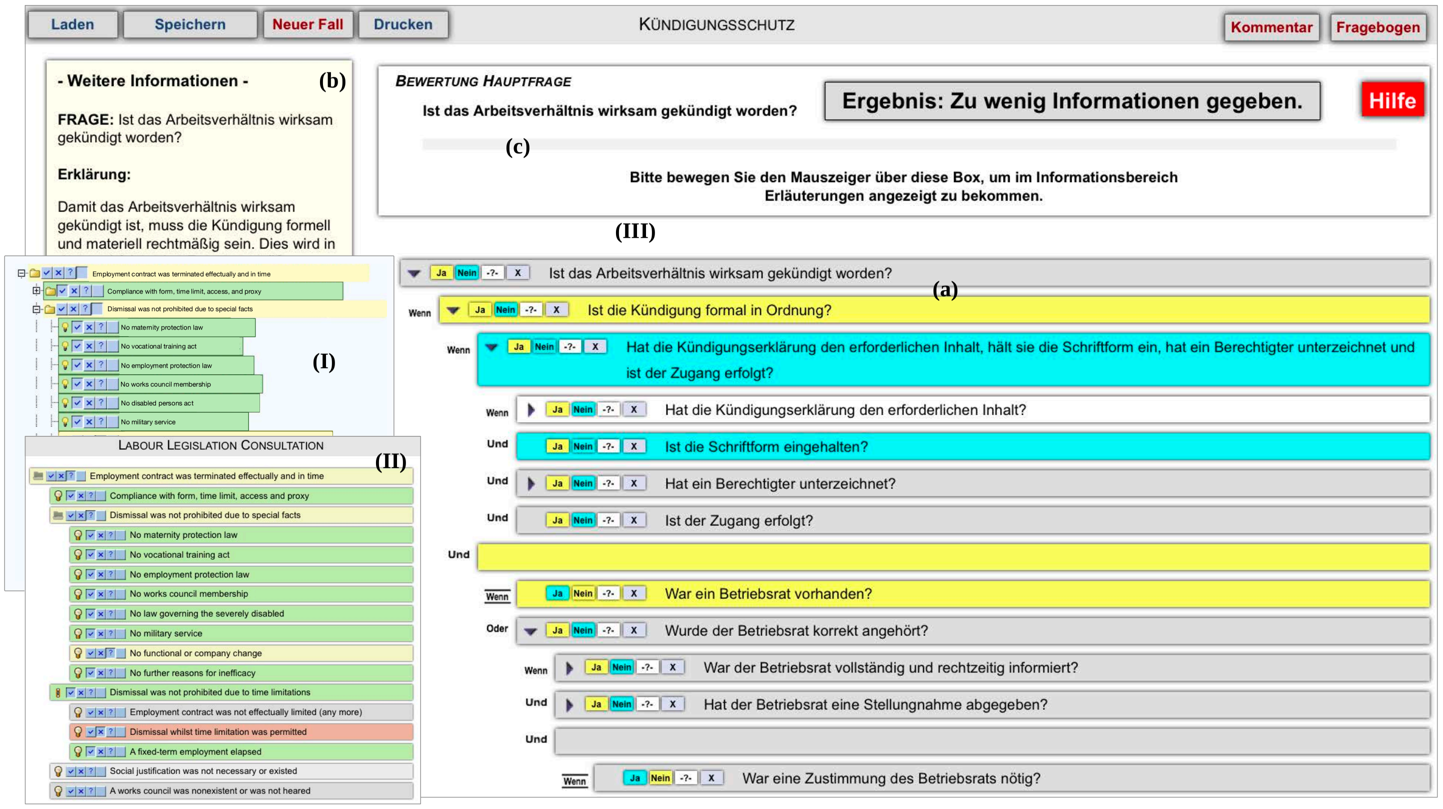
Front-End Prototyping and the Final JuriSearch Prototype The RenoStar partners basically were open to novel approaches regarding UI and interaction design. Thus, this project offered the opportunity to investigate the realization, applicability, and usability of our novel vision of clarification KBS, as the intended KBS modules specifically targeted single solutions. Therefore, a particular instantiation of the Hierarchical Clarifier (c.f., Section 4.3.4.a.) base pattern—the *ITree* variant—was developed. Based on some prior sketches and mock-ups and while continuously extending ProKEt regarding the respectively required renderers, a first prototype was developed. Figure 7.7 shows both the initial, computer-based mock-up (I) as well as the first ProKEt prototype (II). The knowledge at that time consisted of some more or less invented questions—the specification of the real juristic knowledge with its complex interrelations denoted the main project contribution of RenoStar.

This first prototype was demonstrated in an initial kick-off meeting. After discussing several possibilities for refining and extending such a system, the prototype was approved of as appropriate solution for the intended purpose. However, it was also suggested, to evaluate whether actual users would be able to easily understand and use such a (firstly rather unfamiliar) UI type. As potential alternative, an Interview type UI was proposed for evaluation. Thus, a second solution, employing the hybrid Clarification Interview base style (c.f., Section 4.3.5.a.), was implemented as an interactive prototype with ProKEt. Two evaluation studies later—one comparing ITree and the Interview type (see Section 7.2.1, pp. 123 ff.), and one regarding two different ITree variants (see Section 7.2.2, pp. 128 ff.) in spring 2012—it was decided to stick with the novel ITree conception.

As a matter of fact, the tailored realization of the ITree clarifier required a particular knowledge representation: Questions with fixed answer sets *yes*, *no*, and *uncertain*; and specific rules for ensuring the correct value propagation from child- to parent questions. Thus, we first experimented with appropriately formalizing such knowledge in d3web yet without integrating those test KBs with the prototype at that point. Therefore, the first development iterations mainly concerned a refinement and extension of the base UI. Examples include:

- Reworking *the basic styling*, e.g., design, labeling, coloring, and size of buttons and icons.
- Adding a *dedicated, fixed side panel for displaying auxiliary information*. This was one key aspect, as in a comprehensive context such as the legal domain, the availability of extensive additional information is crucial.
- Adapting the *presentation of the core issue rating/justification*: First, this was represented solely by the topmost tree node and corresponding coloring, c.f., Figure 7.7, II. Gradually, it received a more prominent representation in a clearly recognizable top panel which was redesigned several times up to the current state, depicted in Figure 7.7, IIIc.

Regarding KA, which was accounted the major responsibility of the legal experts during that project, a familiar software tool was strongly requested. Thus, both the XML-based ProKEt prototype specification as well as wiki-based KA approaches opted out. Therefore, we thought of a standard office representation based on Microsoft Word that made strong use of the built-in structuring/indexing view of Word for representing the hierarchical relationships in the KB (parent/abstract questions vs. their children/refinement questions). Additionally, a tailored parser was implemented that firstly created a ProKEt front-end prototype specification. Then during the course of several iterations, RenoStar staff sent us the formalized Word-files. As the completely integrated upload and parse tool was not yet available we then parsed those files



- Weitere Informationen -

FRAGE: Ist das Arbeitsverhältnis wirksam gekündigt worden?

Erklärung:

Damit das Arbeitsverhältnis wirksam gekündigt ist, muss die Kündigung formell und materiell rechtmäßig sein. Dies wird in

BEWERTUNG HAUPTFRAGE

Ist das Arbeitsverhältnis wirksam gekündigt worden?

Ergebnis: Zu wenig Informationen gegeben.

Bitte bewegen Sie den Mauszeiger über diese Box, um im Informationsbereich Erläuterungen angezeigt zu bekommen.

Ja Nein -?- X Ist das Arbeitsverhältnis wirksam gekündigt worden?
Wenn Ja Nein -?- X Ist die Kündigung formal in Ordnung?
Wenn Ja Nein -?- X Hat die Kündigungserklärung den erforderlichen Inhalt, hält sie die Schriftform ein, hat ein Berechtigter unterzeichnet und ist der Zugang erfolgt?
Wenn Ja Nein -?- X Hat die Kündigungserklärung den erforderlichen Inhalt?
Und Ja Nein -?- X Ist die Schriftform eingehalten?
Und Ja Nein -?- X Hat ein Berechtigter unterzeichnet?
Und Ja Nein -?- X Ist der Zugang erfolgt?
Und
Wenn Ja Nein -?- X War ein Betriebsrat vorhanden?
Oder Ja Nein -?- X Wurde der Betriebsrat korrekt angehört?
Wenn Ja Nein -?- X War der Betriebsrat vollständig und rechtzeitig informiert?
Und Ja Nein -?- X Hat der Betriebsrat eine Stellungnahme abgegeben?
Und
Wenn Ja Nein -?- X War eine Zustimmung des Betriebsrats nötig?

Employment contract was terminated effectually and in time
Compliance with form, time limit, access, and proxy
Dismissal was not prohibited due to special facts
No maternity protection law
No vocational training act
No employment protection law
No works council membership
No disabled persons act
No military service

LABOUR LEGISLATION CONSULTATION

Employment contract was terminated effectually and in time
Compliance with form, time limit, access and proxy
Dismissal was not prohibited due to special facts
No maternity protection law
No vocational training act
No employment protection law
No works council membership
No law governing the severely disabled
No military service
No functional or company change
No further reasons for inefficacy
Dismissal was not prohibited due to time limitations
Employment contract was not effectually limited (any more)
Dismissal whilst time limitation was permitted
A fixed-term employment elapsed
Social justification was not necessary or existed
A works council was nonexistent or was not heared

Figure 7.7: Evolution of the ITree clarification KBS, a tailored instantiation of the Hierarchical Clarifier base pattern (Section 4.3.4) developed during the JuriSeach project. Lo-fi prototype in the form of a computer mock-up (I), extensible prototype created with ProKEt (II) using invented questions, and current (Mai 2014) productive system (III) including a correct, comprehensive legal KB (in german).

manually, created corresponding ITree prototypes, and deployed them on a test server for their interactive assessment by the RenoStar staff.

Extensible Prototyping After figuring out an appropriate d3web knowledge representation, the Word-parser was rewritten as to create d3web KBs. ProKEt on the other hand was extended as to integrate that new format with the extensible prototypes. Then, the parser was integrated with a tailored front-end—the predecessor of KnowOF—on the server which served as connector between the Word KBs and ProKEt: Thus, the upload tool parsed the Word-file, transferred it into the d3web format, and copied the KB into a ProKEt instance automatically. This enabled the RenoStar staff to upload their Word-files to the test server and instantly invoke an ITree clarification KBS with that KB. Thus, the experts were enabled to concentrate on their main tasks independently from us: Formalizing the KBs in Word, and checking the results in a productively running ITree implementation.

Similarly as in EuraHS, outsourcing the (greater part of) the KA tasks eased and accelerated the overall development efforts on the side of the university members. This in turn strongly helped to concentrate on UI/interaction activities as well as to conduct several evaluation studies. Developing the suitable d3web representation, the parser and upload tool took several months of time. Thus, at the time extensible prototyping started, many UI- and basic KBS features had already been implemented. Therefore, the main efforts were committed to updating and fine-tuning existing KBs, as well as creating several entirely new ones for the many diverse legal issues—thereby always following the intertwined KB–UI development paradigm. One issue that was implemented regarding the UI in that phase, however, was the experimental integration of additional question types, such as numerical, date-based, or scoring questions. Therefore, also own renderers for representing the respective, slightly differently styled tree nodes and an extension of the value processing interactions were required.

At the end of 2013, another evaluation was preformed that comparatively assessed three ITree KBS—details are reported in Section 7.2.3, pp. 131 ff. Based on the insights there, both the ITree UI and partly also again the KBs were refined. Two further studies followed in early 2014 that—amongst other KBS UI patterns—targeted also ITree: An expert assessment, reported in Section 7.3.1, and a large-scale user study, reported in Section 7.3.2. Each of those studies again revealed several issues both regarding the UI and the KB solution. Thus, over time this led to great improvements of the particular ITree implementation for JuriSearch. This is also mirrored in the respective results of the studies, especially the success rates—i.e., the percentage of correctly solved problems. There, ITree was able to increase an initial (low) success rate of about 42% to about 90–100% in the last studies, depending on the KB topic and particular problem.

JuriSearch also is a still ongoing and active project. Current efforts are mainly focussed on formalizing knowledge regarding many further legal (sub-)domains as to finally cover all topics required for the encompassing, legal consultation platform that is aimed at as overall goal.

7.1.5 Subsumption: KBSE with ProKEt (Evaluation)

Regarding the reported case studies Mediastinitis, EuraHS, and JuriSearch, a certain shift regarding the focus of activities is evident. This was naturally induced by the different evolvement states of the tool ProKEt, but also manifested by the project requirements themselves.

Mediastinitis as the first ProKEt-based KBS project started at a time where ProKEt mostly supported the pure prototype construction and configuration. Back then, not yet many KBS patterns/UI presentation alternatives had been investigated in general. Thus, there was some experimentation regarding the general UI configuration options, as reported earlier, but no experimentation regarding entirely different KBS UI styles. On the other hand, the latter was not necessarily required, as the project partners were quite convinced of a representation that basically resembles standard web questionnaires. Thus, in Mediastinitis, the activities focussed on a steady expansion of ProKEt—especially regarding the integration of productive d3web KBs.

At the time *EuraHS* was initiated, it already faced a state of ProKEt that allowed for the basic integration of functional formalized d3web KBs. Due to the high complexity of the EuraHS domain, KA was one of the major tasks during that project. The quick approval of the basic, Mediastinitis-resembling KBS solution for EuraHS by the experts, allowed to quickly set on with KA. Apart from that, another major challenge were the new framing requirements imposed by EuraHS, encompassing the extension of existing and addition of new base functionality. Examples are the multilingualism, or user/case management. Thus, ProKEt itself was greatly extended regarding both the KBS core input presentation and interaction as well as the framing functionality support.

In contrast to those two projects, *JuriSearch* demanded a tailored own UI type, due to the highly expertise legal domain and the goal to particularly support also power users from that domain. Thus, the overall focus was shifted much more towards UI experimentation and repeated evaluation. As by then ProKEt already offered a bunch of general framing functionality for KBS, the main efforts in JuriSearch concerned the realization of the novel UI and interaction types in ProKEt as well as their iterative evaluation.

Despite those differing foci, ProKEt as development tool exhibited its powerful benefits in each project. Basically, ProKEt supports a sound collection of different KBS patterns out of the box. Those can easily be fine-tuned along several configuration dimensions, see Section 4.1, pp. 39 ff., rendering the reuse of existing solutions, that require only minor adaptions, highly efficient. That quick and easy property-based configurability as well as the possibility to adapt ProKEt artifacts nearly unlimitedly via exchanging selected CSS commands/files further fosters in a way participative prototyping—which, adhering to the *participatory design* definition in [Beaudouin-Lafon and Mackay, 2003, p. 1010] implies prototyping, where users are actively involved in the process. This was realized, for example, in the course of the EuraHS project. During a meeting between developers and an expert representative, several ideas regarding the structuring and/or (re)styling were discussed at the example of a functional core KBS prototype—moreover, most aspects were immediately checked in a live-programming / adaption session, which was perceived as especially beneficial by the expert who thus could propose and investigate his ideas right away.

In case more extensive adaptions become necessary, the highly modular internal realization of UI elements in ProKEt, which allow for an easy adaption and reuse, are profitable. The EuraHS initiation was the first time, that those benefits became strongly evident: Founding on the Mediastinitis prototype, a first functional artifact was quickly available. This was in the further course refactored, adapted, and extended, both regarding its UI-based fine-tuning and additional (framing) functionality.

Admittedly, adding entirely new functionality takes its time also with ProKEt. Nevertheless, we are convinced that also there its modular architecture best supports such tasks. Once a certain functionality or widget has been added to the framework, however, the reusability of ProKEt modules for all artifacts makes up for the initial development efforts as soon as the respective feature is required in another KBS again. Take for example the multilingualism feature, originally was implemented for EuraHS—meanwhile, a default intuitive language selection widget can be integrated out of the box with all ProKEt artifacts. The only prerequisites are a property in the KBS UI specification that defines the required languages, as well as the translations of the KB contents in the d3web KB. Other examples include the default session management, the summary presentation, or the feedback and logging facilities for ProKEt artifacts. Consequently, the more efficiently the base KBS can be implemented, the more resources remain for deliberate UI/styling based experimentation and evaluation.

Another strength of ProKEt is the explicit support for intertwined KB–UI development and the possibility to merge the resulting core KBS prototype with functionality into fully functional KBS anytime. This excelled during the EuraHS and the JuriSearch projects: Despite the different applied KA methods, the basic separate-and-merge paradigm was adhered to—in both projects this proved highly profitable as it accelerated the overall course of the project. Also, it allowed us to concentrate on required extensions of the KBS UI/frame whilst the KB was refined by the project partners themselves.

7.2 Evaluation Part II—ITree Evolution

In this work, we have introduced *clarification KBS* as a novel KBS type. ITree, as a particular instantiation of Hierarchical Clarifier that originated from the JuriSearch project (see Section 7.1.4), has been subject to several evaluations during the last years.

7.2.1 Study I—ITree vs. One-Question—March 2012

A first usability evaluation was performed in March 2012, still in the prototyping phase of the JuriSearch project. Parts of the results/discussion were already published in [Freiberg and Puppe, 2013, Freiberg and Puppe, 2012a]. Here, the main objective was to evaluate whether the novel ITree style or a more established Interview/One-question (oneQ) style constitutes the more appropriate UI type for the clarification modules required in the project. The two styles mainly differ in their degree of freedom/guidance: Whereas ITree allows an entirely explorative questioning sequence, oneQ—adhering to *Clarifier Interview* hybrid in Section 4.3.5, pp. 68 ff.—basically suggests a strictly guided sequence.

As Figure 7.8 shows, ITree (a) presents questions in its characteristic, hierarchical tree-style. OneQ (b) in contrast, presents only the one suitable next question expanded at a time, thus imitating a user–system conversation. Both variants apply the same KB, i.e., the same hierarchical refinement questioning structure. There, ITree allows its users to freely navigate any refinement level of arbitrary questions anytime. In contrast, oneQ basically follows the questioning sequence prescribed by the respective question level and allows users to refine only the current active question at hand. Further, the respective previous question is always folded as soon as the next—refinement or next sequenced—question is presented. This destroys much of the contextual knowledge that ITree contrastingly facilitates by always presenting all questions of the current hierarchy level in addition to the surrounding, further structure (limited by display size only). It has to be noted, that both variants were implemented as pure prototypes at that point in time. That is, they did not integrate a functional d3web KB. Due to the evaluation results, see below, the hybrid Clarifier Interview variant was not further pursued. Thus regarding fully functional KBS artifacts, ProKEt offers only a Strict Interview reference implementation to date, but no Clarification Interview variant, so far.

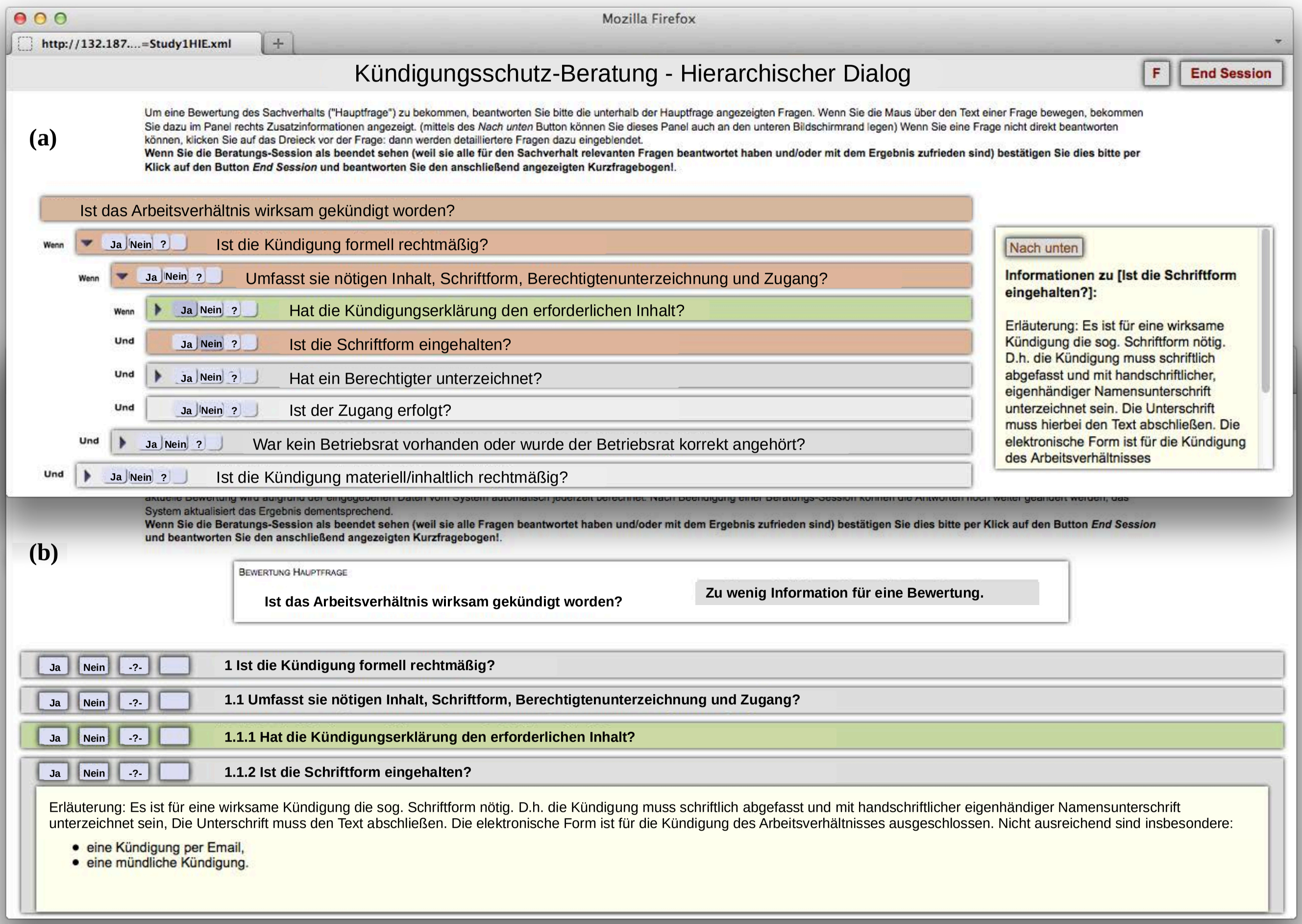
a. Framing Conditions

Preliminary Test: For checking the feasibility of the evaluation setup, a pre-test was conducted. The basic technical feasibility could be confirmed, yet the test resulted in in a refinement of the task description and problem statements.
Main Evaluation: 21 members from our department—roughly between 25 and 35 years—participated in this study. As computer scientists, they all were familiar with general computer and web system usage. Yet, they mostly had little to no experience regarding the specific KBS types under evaluation, and no experience regarding the target domain (labour legislation). Two exemplary problem descriptions from the labour legislation domain were created—see Appendix C.1.2 for an example. The entire materials of this work are available online, see Appendix E. The participants were asked to solve one problem with ITree and the other with oneQ. That is, to rate whether the dismissal in the described scenario is legally correct or whether legal claims can be taken. To avoid biased results due to the sequence of using the UIs, that sequence, as well as which problem to solve with which UI, was altered between participants. This lead to a 2 X 2 setting for the study. For enabling a remote study, the two described KBS variants were deployed on a dedicated server. All required instruction material was provided to the participants per email. Both quantitative (log) data as well as qualitative (questionnaire) data were collected with the help of the ProKEt usability extension. Based on the log data, both the average task time as well as the success rate were calculated. Thereby, the task time encompassed the time from initially loading the KBS until finishing the session by answering and submitting the usability questionnaire. The questionnaire was available on button click within the respective test KBS and users were instructed to answer it once they were satisfied with the result of the consultation session. Therefore, seven questions were assembled, see Table 7.2. Additionally, also the free feedback mechanism was activated , providing us with several more, valuable remarks from users collected right during the evaluation sessions.

b. Results & Discussion

The results of this first ITree evaluation study are summarized in Table 7.2. First, they led to the assumption that the task time depends not only on the actual UI type, but further on:

a) the participants' reading speed vs. the length of questions, problem- and task description.
b) the understandability of interview items.
c) the understandability of the problem description.
d) the potentially already existing knowledge regarding the problem at hand.
e) the usage conditions—during daily job routine vs. after end of work.



Kündigungsschutz-Beratung - Hierarchischer Dialog

Um eine Bewertung des Sachverhalts ("Hauptfrage") zu bekommen, beantworten Sie bitte die unterhalb der Hauptfrage angezeigten Fragen. Wenn Sie die Maus über den Text einer Frage bewegen, bekommen Sie dazu im Panel rechts Zusatzinformationen angezeigt. (mittels des *Nach unten* Button können Sie dieses Panel auch an den unteren Bildschirmrand legen) Wenn Sie eine Frage nicht direkt beantworten können, klicken Sie auf das Dreieck vor der Frage: dann werden detailliertere Fragen dazu eingeblendet.
Wenn Sie die Beratungs-Session als beendet sehen (weil sie alle für den Sachverhalt relevanten Fragen beantwortet haben und/oder mit dem Ergebnis zufrieden sind) bestätigen Sie dies bitte per Klick auf den Button *End Session* und beantworten Sie den anschließend angezeigten Kurzfragebogen!.

Ist das Arbeitsverhältnis wirksam gekündigt worden?
Wenn ▼ Ja Nein ? Ist die Kündigung formell rechtmäßig?
Wenn ▼ Ja Nein ? Umfasst sie nötigen Inhalt, Schriftform, Berechtigtenunterzeichnung und Zugang?
Wenn ▶ Ja Nein ? Hat die Kündigungserklärung den erforderlichen Inhalt?
Und Ja Nein ? Ist die Schriftform eingehalten?
Und ▶ Ja Nein ? Hat ein Berechtigter unterzeichnet?
Und Ja Nein ? Ist der Zugang erfolgt?
Und ▶ Ja Nein ? War kein Betriebsrat vorhanden oder wurde der Betriebsrat korrekt angehört?
Und ▶ Ja Nein ? Ist die Kündigung materiell/inhaltlich rechtmäßig?

Informationen zu [Ist die Schriftform eingehalten?]:

Erläuterung: Es ist für eine wirksame Kündigung die sog. Schriftform nötig. D.h. die Kündigung muss schriftlich abgefasst und mit handschriftlicher, eigenhändiger Namensunterschrift unterzeichnet sein. Die Unterschrift muss hierbei den Text abschließen. Die elektronische Form ist für die Kündigung des Arbeitsverhältnisses

aktuelle Bewertung wird aufgrund der eingegebenen Daten vom System automatisch jederzeit berechnet. Nach Beendigung einer Beratungs-Session können die Antworten noch weiter geändert werden, das System aktualisiert das Ergebnis dementsprechend.
Wenn Sie die Beratungs-Session als beendet sehen (weil sie alle Fragen beantwortet haben und/oder mit dem Ergebnis zufrieden sind) bestätigen Sie dies bitte per Klick auf den Button *End Session* und beantworten Sie den anschließend angezeigten Kurzfragebogen!.

BEWERTUNG HAUPTFRAGE

Ist das Arbeitsverhältnis wirksam gekündigt worden? **Zu wenig Information für eine Bewertung.**

Ja Nein -?- **1 Ist die Kündigung formell rechtmäßig?**
Ja Nein -?- **1.1 Umfasst sie nötigen Inhalt, Schriftform, Berechtigtenunterzeichnung und Zugang?**
Ja Nein -?- **1.1.1 Hat die Kündigungserklärung den erforderlichen Inhalt?**
Ja Nein -?- **1.1.2 Ist die Schriftform eingehalten?**

Erläuterung: Es ist für eine wirksame Kündigung die sog. Schriftform nötig. D.h. die Kündigung muss schriftlich abgefasst und mit handschriftlicher eigenhändiger Namensunterschrift unterzeichnet sein, Die Unterschrift muss den Text abschließen. Die elektronische Form ist für die Kündigung des Arbeitsverhältnisses ausgeschlossen. Nicht ausreichend sind insbesondere:

- eine Kündigung per Email,
- eine mündliche Kündigung.

Figure 7.8: Screenshots of the interactive pure prototypes that were comparatively evaluated in the first ITree evaluation study in March 2012: ITree (a) and One-Question (oneQ, b) style.

For example, task time values varied between users regarding the same UI quite drastically—ITree exhibited values between 6 and 26 minutes. We interpret this mainly as indicator for item a (reading speed), as the other aspects did not vary greatly in this evaluation.

Also, aspect (c) was confirmed by several subjective feedback. In particular, it was stated that the better results (and confidence in the system) mainly were due to an easier problem description. Or the other way around, that a bad understandability of the problem description leads to overall uncertainty regarding how to proceed and the correctness of the results.

With respect to the baseline difference between the task times of ITree and oneQ, however, we assume that (a)–(d) most probably had no influence: The KBs integrated with both test systems were exactly the same, thus reading speed/length of the questions (a) and question/problem understandability (b) & (c), should not have impacted the results. Regarding the prior knowledge (d), the domain of the core issue was also equal, thus also opting out as influential factor. The observed average task times for ITree and oneQ are, by a narrow margin, statistically not significant on a one-sided unpaired t-test, p=0,068. In retrospect, we account that difference most likely to the more explorative character of ITree.

Evaluation Item	**ITree**	**oneQ**
Log-Data: Usability Metrics		
Average Task Time (minutes +- SD)	**13m38s** +- 6m49s	**10m39s** +- 5m49s
Success Rate (correctly solved/all cases in %)	**42.86**	**38.1**
Questionnaire results with rating scale: 0 (disagree completely)–6 (agree completely)		
Q1: The interaction with the KBS was intuitive	**4.05** +- 1.20	**2.90** +- 1.79
Q2: The system (re)actions were understandable	**4.43** +- 1.54	**2.76** +- 1.45
Q3: The final solution rating was understandable	**4.52** +- 1.54	**3.33** +- 1.85
Q4: I could solve the problem correctly	**3.67** +- 1.53	**2.52** +- 1.83
Q5: I gained further knowledge using the KBS	**4.05** +- 1.32	**2.95** +- 1.72
UI Preference in %	**81**	**14**

Table 7.2: ITree Evaluation Study I: Itree versus One-question UI. Results of the log-data-based metrics, of the usability questionnaire on a scale from 0 (disagree completely) to 6 (agree completely) with both the mean value +- SD and of one additional question regarding the *UI Preference* with the three options ITree, oneQ, and no Preference. There, 'no Preference' is the difference of the listed values to 100%. Original wording of the questions was german, see Appendix C.1.3.

The success rate exhibited no statistical significance on a one-sided binomial test with p=0.11 and p=0.16, respectively. Thereby, ITree overall produced less incorrect, but then, slightly more no-solution results than oneQ. This indicates, that in case ITree can be used, it produces better results, but that the hurdle to use that UI type might be higher. Nevertheless, the rather low success rates in both cases were a clear indicator for the need to refine the KB contents/structure. This was also affirmed by subjective user feedback: In 11 cases (52%) the wording of the questions was perceived as incomprehensible/cumbersome due to often used duplicate negations and legal specialist language. This was even more aggravating as the participants were entirely legal laymen and thus not at all familiar with legal terms and language. Also, the hierarchical structure and representation of the KB—that followed the legal subsumption logic—was perceived unfavorable. It seemed to impose difficulties regarding the top level (entry) questions for the users and thus posed a high entrance hurdle for using the system.

User feedback further reported some rather fundamental problems with both UIs. This concerned the button representation and consequential interaction:

- Not understanding the real meaning of the -?- button as answer option ’unclear’ (4 cases / 19 %). Rather, the KBS was expected to display more elaborate explanations or to automatically display the next refinement level.
- Not understanding the empty button. Which was designated to reset the question by removing the answer (3 cases / 14%).
- Quite controversial rating of the answer button design. On the one hand as well and supportive (regarding an indication of the consequences), on the other hand, as confusing.

As one remarkable finding, the clear UI preference for ITree is statistically significant on a χ square test with p<0,05 and with an anticipated distribution of 50% (ITree), 30% (oneQ), and 20% (no preference). This might on the one hand be due to the fact that the participants—as computer scientists—are well accustomed to tree-like UI representations and thus perceived ITree as naturally intuitive to use. This free explorability of ITree might have been another reason for that type of test users to prefer that UI form.

Regarding the questionnaire, ITree scored better in all requested items. Ratings were provided on a scale from 0 (disagree completely) to 6 (agree completely). The values are all statistically significant using an unpaired, one-sided t-test with $p \leq 0{,}05$. Thereby, particularly Q1 and Q2 indicate an overall better usability of the ITree style compared to oneQ with regards to the legal target domain. The rather weak values in Q4 regarding both UI types, however, show a general uncertainty (or distrust) in the correct functioning of the KBS. The main reasons probably were the comprehensibility issues with the KB, as well as the problems with the actual UI implementations. A more extensive, explicit final explanation of the solution might have leveraged the distrust, and thus was scheduled as important feature for the next development iteration. Nevertheless, Q3 suggests that ITree as an UI style seems to exhibit basic explanatory skills by itself, as anticipated of such deliberately explorative Clarifier implementations. Q5, finally, especially affirmed our assumption that ITree additionally evinces some skill-building abilities.

c. Insights from the study

As the major outcome of that first study, it was decided to stick with the envisioned, novel ITree UI style instead of refining the Clarification Interview variant. Another important insight was the urgent need to completely rethink and adapt the contents and structure of the KB, so that also legal laymen users are supported the best possible way in working with the system. This led to both reworking the *legally oriented* version of the KB as well as to creating a completely new *laymen user oriented* version of the KB. Those were evaluated in the second user study, see Section 7.2.2. Further, some UI-related issues were revealed. Examples are a more careful overall design and, implicitly or explicitly, communicating the meaning of UI widgets such as the answer buttons.

For revealing issues that have not been anticipated beforehand—no matter how careful RE and system/UI design have been performed—the power of usability testing/experiments is consented on by the majority of usability practitioners. This first study also clearly affirmed this, especially with respect to specialist KBS software. One example was the misunderstood meaning of the -?- (answer option 'undecided') button, which was not at all anticipated or thought about as potential problem beforehand. As another insight from performing the pre-test of this evaluation, we also learnt the strong benefits of testing the evaluation setup itself for revealing particularly problems with the task/problem statements.

7.2.2 Study II—ITree vs. ITree (KB Refinement)—May 2012

Based on the outcome of the first ITree study, a second evaluation was conducted during the JuriSearch project soon after. This time, with the main goal to compare two different alternatives of structuring the KB contents and the consequences on the perception and rating of the ITree UI. Therefore, a KB oriented more towards legal experts (ITreeExpert) and one intended for legal laymen (ITreeLaymen) were used with the ITree base UI that had been improved regarding the basic functionality flaws found in study 1.

a. Framing Conditions

In this study, 23 members from the RenoStar corporation were recruited as participants. However, only the results of 18 members could be evaluated correctly, as the remaining members reported problems with both the logging and feedback mechanisms (most probably due to local/personal settings of the firewall). The 18 participants exhibited a mixed expertise in the legal domain, as 6 of them were legal specialists, whereas the remaining persons were of mixed professions, such as accountants, or secretaries. Neither of the participants, however, were explicitly trained computer science experts, thus they can be assumed to be generally little to not experienced particularly with KBS and tailored UI forms. Similar to the first study, however, those test users had varying expertise in general computer and web system usage.

The basic evaluation setup was very similar to study I (see Section 7.2.1): Two exemplary problem descriptions from the domain of cancellation were created. Based on the experiences in Study I, those were formulated more precisely and restructured as to be more easily and quickly understood. The participants then were asked to solve one problem with the ITree-Expert and the other with the ITreeLaymen variants. To avoid biased results, again a 2 X 2 setting for the study as described before was applied—i.e., not only altering the association between problem and KBS variant, but also altering the sequence of using the KBS. An exemplary problem description and instruction sheet from this study are provided in Appendix C.2.2, the entire materials are available online, see Appendix E. The study again was conducted remotely. Thereby, one RenoStar staff member was set in charge for answering questions/solving problems and communicating to the coordinating member of the university when required. The two test systems were deployed on the test server. The required instruction material was provided to the RenoStar contact person per email. However, the material was also handed out to the participants in a print version. This intended to relieve them from switching between different computer programs for reading the instructions/problems and using the KBS.

Regarding the evaluation, both quantitative (log) data as well as qualitative (questionnaire) data were collected with the help of the ProKEt usability extension. Based on the log data, both the average task time as well as the success rate were calculated. The questionnaire consisted of seven items in total, c.f., the questions listed in Table 7.3. Also the free feedback mechanism was activated during this study.

b. Results & Discussion

The results of the second usability study with ITree are summarized in Table 7.3. For both KBS

Evaluation Item	ITreeLegal	ITreeLaymen
Log-Data: Usability Metrics		
Average Task Time (minutes +- SD)	**19m39s** +- 17m21s	**19m14s** +- 11m28s
Success Rate (correctly solved/all cases in %)	**61**	**72**
Questionnaire results with rating scale: 0 (worst)–6 (very positive)		
Q1: The basic interaction was intuitive	**3.89** +- 1.41	**3.89** +- 1.23
Q2: The KBS contents were understandable	**4.11** +- 1.60	**4.22** +- 1.11
Q3: The solution rating was understandable	**4.11** +- 1.91	**4.22** +- 1.77
Q4: I could solve the problem correctly	**3.61** +- 1.94	**3.17** +- 1.46
Q5: I gained further knowledge using the KBS	**3.89** +- 1.88	**3.67** +- 1.75
UI Preference in %	**35.3**	**29.4**

Table 7.3: ITree Evaluation Study II: Comparative KB assessment of a legal expert- versus a laymen-oriented variant, both used in the ITree UI. Results of the log-data-based metrics, of the usability questionnaire on a scale from 0 (disagree completely) to 6 (agree completely) with the mean value +- SD, and of the additional questionnaire item *UI Preference* with the three options ITree expert, ITree laymen. Thereby, 'no Preference' is the difference of the listed values to 100%. The original wording of the questions was german and can be found in Appendix C.2.3.

variants, the average task time as well as the questions from the final questionnaire were very similar. The biggest differences concerned the self-estimation of the users whether they were able to solve the case correctly (Q4), and the perception whether the system fostered to gain knowledge regarding the domain (Q5). In both cases, ITreeLegal scored better that the laymen variant. This strongly indicates that the original legal structuring is not as bad after all.

This assumption, however, as well as the UI preference values, are contradicted by the results from Q2/Q3 and the success rate: Both regarding the KB (Q2) and the understandability of the results (Q3), the laymen variant scored slightly better with a rating of 4.22 versus 4.11 (ITreeLegal) in both questions. Also, the success rate was somewhat higher regarding ITreeLaymen (72 % vs. 61 % for ITreeLegal). Nevertheless, the explicit preference was stronger for ITreeLegal (35.3 % vs. 29.4 % for ITreeLaymen), but not very strong overall, as also 35.3 % of the users stated an indifference between both variants.

Overall, not even the preference regarding *ITreeLaymen by actual laymen* was as evident as assumed: The laymen variant was preferred in five cases—thereof, four times by laymen, and once by a legal expert. The ITreeLegal variant, in contrast, was preferred in six cases—thereof, three times by legal experts and three times by laymen. Most remarkable, however, was the fact that six laymen users stated a complete indifference between both variants. This, together with the overall preference for the ITreeLegal regarding several aspects (Q4, Q5, UI Preference), leads to the assumption that the reformulated laymen KB still was not user-oriented enough—or in turn that the legal variant is still better. Also, the subjective feedback suggested, that there was no big difference between both test systems with regards to their KB and that both alternatives were somehow too juristic. Further, the KB suitability also seems to be an extremely subjective perception, thus attaining a generally well-rated, user-focussed variant is not a trivial task—if possible at all due to a high diversity of 'general potential users' and the absence of a formal, user-oriented structuring scheme. In contrast, the structure of the ITreeLegal variant is based on commonly taught legal schemes, and thus can more easily be unified for a broader range of legal experienced and expert users. Thus, it was decided to concentrate only on the legal variant in the future and dump the laymen KB variant.

In this study, the success rates exhibited a statistical significance on a one-sided binomial test with p=0.0098 (ITreeLegal) and p=0.00064 (ITreeLaymen). Compared to the success rate of the ITree UI gained in Study I (42.86 %), the results have improved regarding both KB structuring alternatives. Given the assumption, that the chance of 'guessing' the correct solution for a problem is 33.3 % (possible guesses: correct, incorrect, no idea), values of about 60 % and 70 % can be seen as a good success of the system. Yet, as subjective feedback also underlined, there still existed room for improvement. The wording of the questions, especially of the additional explanations, the frequent usage of (non-translated) legal specialist terms, and still also the general structure, were still perceived as too domain specialist and thus not suitable for legal laymen.

User feedback additionally highlighted the following major issues:

- A general difficulty of comprehending the semantics of the red/green color scheme. Especially, when contradicting the western cultural semantics of yes /no /positive /negative. For example, if an answer 'yes' leads to a negative valuation of the solution and thus to a red answer button 'yes'.
- Problems with retrieving the auxiliary information by hovering the entire question text. There, unintentional touching of another question immediately displayed the information for that question and removed the originally intended information.
- Difficulties with understanding the solution justification.

c. Insights from the study

As key outcome from this study it was decided to dump the user-oriented variant of the KB entirely, based on the reasons mentioned above. However, due to some subjective remarks also the suspicion manifested, that the perception of the usability and suitability of the ITree KBS is not exclusively dependent on the KB and the UI—but additionally also on the formulation of the respective case/problem that is to be clarified. The concrete investigation of this aspect was set as main goal for the subsequent evaluation, reported in Section 7.2.3.

7.2.3 Study III—ITree Laymen Suitability—December 2013

According to the insights of the preceding studies, the ITree base implementation was reworked concerning the major findings of the preceding study. Also, the legal counseling corporation *RenoStar* refined the existing labour legislation KB and created several new legal KBs.

a. Evaluation Goal & Framing Conditions

Main objective of this third ITree study was to assess whether the general suitability of this KBS type had increased due to the most recent adaptions. On the one hand, concerning the new implementation of the ITree UI and interaction. Yet also concerning an assessment of the further refined and novel legal KBs.

The evaluation was performed in December 2013 as part of a lecture on medical informatics at the University of Würzburg. In total, 19 students participated in the study. This implies that they denoted topical laymen users but exhibited an experienced to advanced proficiency regarding computer and interactive web interface usage in general. They were equipped with three problem descriptions. Two concerned the legal valuation of a dismissal, one the valuation of a hit and run accident. Students were asked to retrieve a legal estimation of each of the cases with the help of the provided ITree implementations. Those were made available on a remote test server, thus each student could access the KBS using their own computer. Afterwards, the participants were asked to fill in a tailored questionnaire. This contained two general, problem independent questions (Q1 & Q2 in Table 7.4), and six questions that were to be answered for each case separately. Regarding those latter questions, we assumed a dependency on the respective KB contents/structure, which is why they were to be answered separately. For the original questionnaire, we refer to Appendix C.3.2. The entire materials again are available online, see also Appendix E.

b. Results

Table 7.4 summarizes the (objective) success measures. Also, the ratings from the conclusive (subjective) questionnaire are provided on a scale from 1 (very good) to 6 (very bad). The major, more informal remarks concerning both the UI as well as the KB are provided in Table 7.5.

The general rating value of 2.1 for the ITree KBS (Q1) is a quite good signal regarding an overall evolution and amelioration of that UI type. Q2, the second general question, concerned the detectability of the KBS on the framing webpage. This question mainly was of interest for the RenoStar staff as the basic ITree clarification modules are offered embedded in an overall framing web page. There, the average rating was also very good, implying that the chosen way of embedding and linking the ITree modules denoted an appropriate solution.

The next, notable result was a perceptible increase of the success rate to former evaluations. In Case 1, a similar KB and problem description as in previous evaluations (e.g., ITree Study II, pp. 128 ff., *ITree Legal Variant*) were used, yet those had been refined a bit regarding the wording and add-on information. Still, this evaluation exhibited an increase of the success rate from 61% to 100%. However, also the rates of Case 2 (89.5%) and Case 3 (100%) are remarkably good. This is statistically significant on a one-sided binomial test with p=1,90735E-06 (Case 1 & Case 3) and p=0,000326157 (Case 2). Thereby, we founded on an expected success value of 0.5—when 'guessing', either the wrong or the correct solution.

Basically, Case 1 and 2 used the same UI and KB, only the problem description varied. There, the less good success rate of Case 2 confirms our assumption that KBS results are not only dependent of the respectively applied UI and used KB, but that additionally also the problem description itself is a relevant influencing factor. This is also mirrored by the rationale, the students were asked to provide for each solution. For Case 1 and 3, mostly the correct explanation was given, however, sometimes not entirely complete (Case 1). For Case 2, however, mostly no rationale was provided at all.

	Case 1	**Case 2**	**Case 3**
Success Measures			
Success Rate (correctly solved/all cases in %)	**100%**	**89.5%**	**100%**
Wrong solution	/	**2**	/
Rationale (average)	incomplete	no	correct
Questionnaire results with rating scale: 1 (very good)–6 (very bad)			
Q1. Rating of the overall utility of the KBS	**2.1** (+-0.71)		
Q2. Detectability on the webpage?	**1.5** (+-0.70)		
Q3. Quality of the knowledge base contents	**2.3** (+-0.87)	**2.6** (+-1.35)	**1.8** (+-1.01)
Q4. Ease of use	**2.5** (+-0.96)	**2.4** (+-1.12)	**2.0** (+-0.82)
Q5. Efficiency of the KBS	**2.5** (+-0.61)	**2.5** (+-0.90)	**2.2** (+-0.83)
Q6. Visual representation of the result(s)	**2.4** (+-0.98)	**2.5** (+-1.07)	**2.4** (+-1.06)
Q7. Belief in the result(s) correctness?	**2.1** (+-1.03)	**3.2** (+-1.40)	**1.6** (+-1.01)
Q8. Assistance how to proceed with the case	**1.7** (+-0.67)	**1.9** (+-0.85)	**2.7** (+-1.49)

Table 7.4: Summary of the conclusive questionnaire in the third ITree study, december 2013. Average values are provided with standard derivation in parentheses. Three problem descriptions—Case 1: Summary dismissal. Case 2: Orderly dismissal. Case 3: Hit & run accident.

Concerning Q1–Q8, the ratings of Case 3 stick out. They were better regarding all but the last question. We assume this to be due to the fact that the hit & run topic overall was far less complex than the dismissal problem area. This theory is supported by the far more positive rating of Q3, the KB quality.

Further, this again is an indication of the theory regarding the influence of the problem description. This is seconded by the fact, that KB quality (Q3) was rated a bit worse in Case 2 compared to Case 1—despite the fact, that those both cases used the exact same ITree implementation and KB, and only differed in the problem description. One reason could be, that Case 2 did not use the exact same special terms as mirrored in the KB, which made it harder for users to find the relevant items—leading them to the conclusion that this might be a KB design issue. Also, the results of Q7 point in that direction, as there similarly a better overall rating for Case 1 was provided, even if the only difference actually was the problem description.

Finally outstanding is the fact that the provided assistance of the KBS of how to proceed in the interrogation (Q8) received the worst ratings for the otherwise best rated Case 3. This, we again attribute to the knowledge/topical complexity. The lesser complexity of the hit & run problem area in consequence led to a lesser abstracted KB with fewer refinement levels. In turn,

#	HE	ISO	Statement
1	1,7	1	KBS initialization takes too long, unstructured/confusing text is displayed meanwhile. Answering a question always induces a complete reload; this slows the KBS down too much.
2	1,8	2,4	Auxiliary information side panel is too small.
3	1	2	Entirely answered sub trees should be folded or highlighted otherwise for providing more overview/consultation status feedback.
4	1,4	2,4	Bug: And/Or markers sometimes vanish after toggling a question node.
5	3,5,7	3	Bug: When opening the KBS in different browser tabs simultaneously, the systems interfere with each other.
6	4	4	English menu buttons are inconsistent with the overall german KBS design.
7	1	2,7	The result should provide a more detailed explanation as to which reasons are necessary / sufficient for a legal claim.
8	1,8	2,4	With smaller screens (1024x600) the auxiliary information field partly vanishes.

Table 7.5: ITree Evaluation Study III. Subjective user remarks, mapped to the heuristics of [Nielsen, 1994] (HE) and [ISO 9241–110, 2006] (ISO).

this may have caused users to rate the perceived assistance worse, as not as much fine-grained question (levels) were offered than in the dismissal problem area.

c. Insights

As a basic insight from this study, once again the value of iterative evaluation excelled; especially regarding highly expertise domains such as the targeted legal domain. For example, the steady gradual refinement of the comprehensive cancellation topic led to a remarkable increase in the success rate of that system. Thereby, it became clear that actual users of such a system most often are able to find other / additional shortcomings that the responsible experts cannot envision beforehand—probably caused by their expertise and implicit background knowledge.

The subjective remarks again revealed the need for improving several aspects of the implementation. The main point of critics was the overall slow system response time. There, we worked on the (re)loading mechanisms as to achieve a better performance. Another critical issue was the lack of understandability of the solution justification by legal laymen—despite having refined and adapted it. However, this is not an overly critical issue, as we anyhow suspect (and thus targeted) Clarifier KBS to be particularly apt for experienced to expert users. Also, the particular ITree implementation probably will be used mostly by trained service support staff anyhow. Thus, we stuck with that existing solution. Apart from that, several minor findings regarding the interaction and overall design were reported. Most of them were considered and respectively adapted before the subsequent evaluations in 2014, see Section 7.3—which were not targeted exclusively towards ITree-targeted, yet also considered this style amongst others.

7.2.4 Further ITree Evaluations and Subsumption

Apart from the focussed ITree studies reported in the preceding sections, ITree also was part of the general pattern assessment. The respective results of both the expert evaluation in February/March2014 and the user study in April/May 2014 are reported in Sections 7.3.1 and 7.3.2. Recapitulating the overall ITree evolvement, the repeated and numerous evaluations during the last two years clearly improved ITree regarding both the UI/interaction design as well as the KB contents. Especially, the success rate increased steadily, reaching very good values of about 90–100% in the last studies in december 2013 and april 2014.

The overall ITree evolvement was also clearly mirrored in the respective, subjective evaluation findings. Whereas in the initial evaluation mainly general issues emerged—e.g., unnoticed implementation bugs—the focus switched later on. Subsequent iterations exhibited more and more also fine-grained issues with the KB contents and structure. Also, the remarks concerning particular design decisions and interaction requirements grew more specific over time. Figure 7.7, p. 120, visually subsumes the ITree evolution from the first conceptual mock-up (I), over the first ProKEt-based ITree prototype (II) for the dismissal clarification system, to the current implementation state (III).

7.3 Evaluation Part III—Selected KBS UI Patterns

Regarding the KBS UI patterns, introduced in Section 4.3, pp. 48 ff., our main objective was to not only provide an assembly of several theoretical pattern specifications, but also to offer a collection of readily available reference implementations with the tool ProKEt. Thereby, those implementations should not only denote an arbitrary realization of the pattern specification, but particularly also guarantee a certain level of quality regarding their design and usability. In the course of the subsequently reported study, the following patterns have been implemented with ProKEt and evaluated with respect to their overall usability:

- Questionnaire—in the Box- and Daily variant
- Interview—in the Strict- and Hierarchical variant
- Hierarchical Clarifier—specifically the legal ITree variant

The evaluation thereby was twofold: Firstly, gaining insights regarding the inherent usability of each pattern realization on its own, thus based on expert evaluations of each implementation—reported in Section 7.3.1. And secondly, regarding the comparative perception of the applicability and suitability of the KBS solutions in specified contexts—addressed in a large-scale user study, reported in Section 7.3.2. Evaluation subject were the particular pattern implementations. Yet, we hoped for being able to draw also some implementation-independent conclusions regarding the general suitability and quality of the patterns.

7.3.1 KBS UI Patterns—Expert Evaluation

Evaluation Goal & Framing Conditions The main goal of this evaluation was an assessment of the general usability and design of each of the selected pattern implementations. A preliminary test run, with only one of the five target KBS UI styles, was performed for validating the applicability and suitability of the evaluation setup before the main evaluation. The evaluation focus particularly was laid on the basic UI and interaction design of the core input module. Thus—after some remarks in the preliminary evaluation—framing functionality such as session management facilities was deactivated as to foster the concentration on the main data input task.

As evaluation techniques, *heuristic evaluation (HE)* based on Nielsen's ten heuristics [Nielsen, 1994] and *cognitive walkthrough (CW)* [Wharton et al., 1994] were applied. The evaluation has not been performed by usability professionals with years of experience in their field, but by HCI students of our institution. Thus, we are well aware of the fact, that this evaluation cannot be accounted as a fully informed expert review. However, the foundation for the reviews were established usability methods. Moreover, we accounted it as particularly interesting to perform the evaluation with unbiased student experts. There, we assumed that this could result in fresher, rather uninfluenced, more spontaneous feedback and thus potentially in some interesting non-traditional insights.

Results Presentation The setup and results of the preliminary- and the main evaluation are reported in separate sections—one for each assessed KBS implementation. The basic outcome, i.e., 'raw data', of the applied expert evaluation techniques were usability reports for each assessed KBS UI style. Those have been processed, summarized, and unified into finding tables for providing overview of all relevant findings for each implementation. Thereby, the findings are mapped to the matching heuristics of [Nielsen, 1994].

a. Preliminary Evaluation—December 2013

In total, eight evaluators participated in the pre-test in 12/2013. This targeted only the ITree UI style, with the main objective to validate the evaluation setup.

UI Solution, KB, & Task The UI of the pre-evaluated ITree corresponds to the example depicted in Figure 7.13, p. 146. A description of the functionality and interaction of ITree, as particular instantiation of the Hierarchical Clarifier pattern, is provided in Section 4.3.4.a. (pp. 63 ff.). For the preliminary evaluation, a *lodging deficiencies KB* was used—a KB for clarifying potential lodging defects that could justify a rent reduction or a legal claim, see Appendix A.6. The task description both for the heuristic evaluation and for the cognitive walkthrough are provided in Appendix C.4.1.

Insights from the preliminary evaluation As the preliminary evaluation run rather was intended as a feasibility check, Table 7.6 only summarizes the most critical problems exemplarily. The original spreadsheet listing that contains all findings in an unabridged manner is provided online—see Appendix E for an overview. Anyhow, most findings were reported in the main evaluation in a more precise manner. The most critical issues of this pre-test—e.g., concerning the auxiliary information display, general system response time, or basic UI ambiguities—have been adapted before conducting the main study.

Regarding the evaluation setup itself, we gained two key insights. To *prescribe more strictly which evaluation technique to apply*. In the test run, participants were free to chose HE or CW. As a result, some evaluators neither adhered strictly to the one or the other, but rather

	Topic	Problem
1.	Auxiliary Information Box	a. Presentation of option 'show at bottom': Faulty scroll behavior, panel covers important questions thus UI not entirely usable anymore b. Automated content update when hovering questions: irritating/annoying c. Contents vanish when mouse moves over question buttons
2.	Color Scheme Red/Green	a. No double coding of information; may be difficult for color-blinds b. Changing/reverse color coding for selected questions is confusing c. Confusing, when color semantics are used controversially to known cultural semantics, e.g., when red=yes. d. Missing explanation of color semantics or reverse questions
3.	Basic UI Intention / Understandability	a. Does one need to open all child questions (was unclear) b. No expert shortcuts (answering top level items as shortcut was not understood) c. Value propagation from children to parents sometimes not working
4.	General UI Design Issues	a. Question toggle triangle too small and unobtrusive b. Markers for 'when', 'and', 'or' connections are confusing c. Automatic scroll-jump to the top is confusing/annoying d. Confusing first question: Why is it answerable if it is derived anyways?
5.	General Interaction Issues	a. Slow initialization and system response after answering a question b. No warning before deleting old data when starting a new case c. Meaning of answer option *uncertain*: Often sets the entire dialog state to unclear (too much weight of this option?); child questions are not automatically expanded when answering with unclear

Table 7.6: Summary of the main findings (problems only, topically grouped) of the preliminary evaluation in the course of the KBS UI pattern assessment. Implementation target was an ITree implementation (see Figure 7.13, p. 146) for the lodging deficiencies KB (see Appendix A.6).

provided only rough finding listings. Also, we noticed the requirement of *explicitly demanding finding severity/relevance ratings according to fixed scales*. Here, some evaluators provided ratings, some others did not, others again made up some rating scales of their own. However, especially regarding more neutrally formulated statements such as *all questions can be expanded* an unmistaken rating is essential as otherwise such findings are practically useless due to the unknown intention of the evaluator.

b. KBS UI Styles—Main Evaluation

The main evaluation—which took place in February to March 2014—was performed by in total 30 HCI students of the University of Würzburg. During that timespan, the target systems were made available on a dedicated test server. Thus, the participants were free to choose when and where to perform each evaluation remotely. Basically, the participants were divided in six groups of five members. Thereof, three groups performed a HE, the other three groups a CW. Each participant was instructed to perform the respective evaluation for each UI independently. However, participants were allowed to summarize all their findings into one usability report per group. The detailed task descriptions are provided in Appendix C.4.2.

Evaluation target were all of the previously selected five KBS UI pattern reference implementations:

1 *Box Questionnaire*—House Plants Recommender KB (App. A.2)
Screenshot in Figure 7.9, major findings in Table 7.7.

2 *Daily Questionnaire*—Pub Recommendation KB (App. A.3)
Screenshot in Figure 7.10, major findings in Table 7.8

3 *Strict Interview*—Statistical Calculation Recommender KB (App. A.1)
Screenshot in Figure 7.11, major findings in Table 7.9

4 *Hierarchical Interview*—Statistical Calculation Recommender (App. A.1)
Screenshot in Figure 7.12, major findings in Table 7.11

5 *ITree*—Labour Legislation (App. A.4) & Hit and Run Accident (App. A.5) KBs. Screenshot (same UI used for both KBs) in Figure 7.13. Major findings in Table 7.10

The screenshots show the evaluated pattern implementations. For details on the basic design and interaction we refer to the KBS UI pattern specifications in Section 4.3, pp. 48 ff. As the participants were all german students, all KBs were used in a german variant.

For each assessed KBS UI style, we summarize the major evaluation findings (i.e., mostly problems) in a condensed table. This further lists the assigned average severity rating 0 (no problem)–4 (highly severe problem) as well as the associated heuristics according to [Nielsen, 1994]. Table 7.12 further provides a subsumption of common findings for all assessed systems. The evaluation focussed on the core input facilities, thus we did not include remarks concerning the framing functionality (e.g., resetting cases) or the KB contents. The original spreadsheet listing that contains all findings in an unabridged manner is provided with the encompassing materials available online—see Appendix E for an overview.

Insights from the main evaluation Fore-mostly, the expert findings served as valuable foundation for enhancing the reference implementations in ProKEt, thus we do not discuss all issues in detail. Yet, at the end of this section, pp. 150 ff., we subsume an overall estimation of the current usability and design state of all five reference implementations and the underlying KBS UI patterns, based on both the joint evaluation results from expert evaluation and user study. The most severely rated issues were immediately fixed in the reference implementations as to provide an enhanced foundation for the subsequent comparative user study. This exemplarily included:

- Adding a *New Case* warning before overwriting data (all impl.)
- Unifying labels/languages regarding the feedback form (all impl.)
- Improving the line spacing and fixing the issue with resetting answers on repeated click (Daily Questionnaire)
- Fixing the bugs with the interview history/progress bar/navigation buttons (Strict Interview)
- Introducing a *New Case* button and functionality (Hierarchical Interview)
- Adapting the instructions / initial short-info regarding the particular coloring scheme and the required interaction, and further tweaking the system response/reloading behavior (ITree)

The basic tendency (as no precise overall rating was required) arising from the results was, that Box Questionnaire was the overall winner due to an overall familiar intuitive UI metaphor. This was followed by Strict- and Hierarchical Interview. Thereof, Hierarchical Interview overall received the better remarks. Yet, for Strict Interview overall more potential (given, that certain issues are fixed in future implementations) was predicted. Daily Questionnaire basically was also well-perceived, yet received many remarks concerning specifically an inappropriate KB, which seemed to have deterred its overall perception. Finally, ITree was rated worst in this expert assessment.

	Topic	Problem	S	HE
1.	Solution Presentation	a. Displaying excluded solutions is not necessary/counterintuitive	2	8
		b. Do not display *infinity* as solution score but an actual number	1	2
		c. Solution justification box not visible when triggered for a solution at the bottom of the list	2	1
2.	General UI Design	a. Once all questions are answered, returning to dialog and adapting single questions has no effect on dialog state / solutions	3	1,3
		b. Image scaling differs—some are too large, some too small	3	/

Table 7.7: Box Questionnaire KBS, expert evaluation findings, grouped topically.

	Topic	Problem	S	HE
1.	Solution Presentation	a. Justifications are not (easily) understandable	4	2,5,10
		b. Icon for abstract rating and score are not intuitive	2.3	1,2,10
		c. Justification popup hides other popups or main UI sometimes	2	1,3,7,8
		d. *Ergebnis* label overlays solution and makes it unusable	3	3
		e. Solutions cannot be filtered regarding certain weighted aspects	4	/
2.	General UI Issues	a. Browser window size change can lead to unusability of the UI	3	1,8
		b. Question/answer display blocks too broad, not easily readable (from question to final answer) on large displays	2	5,8
		c. UI rather cluttered, too many items visible simultaneously	/	/
		d. Link to instructions too unobtrusive, instructions not optimal	2	10
		e. Linespaces should be optimized, indicate a line between questions	/	8
		f. Label *neuer Fall [new case]* is too formal for pub domain	1.5	2,8
3.	Answer Status / -Options Design	a. No differentiation of OC and MC questions	2.5	2,3,4,6,8
		b. No clear differentiation of answer options and question	2.5	2,3,4,6,8
		c. Not all answers can be easily reset by repeated click	2.5	2,3,4
		d. No global feedback if all answers of a questionnaire are done	1	1,5
4.	General Interaction	a. Input of *price* questions (OC) is cumbersome, would be better with slider widget or plain input field	1	/
		b. No weighting of question importance is possible	2	/
		c. Folding triangles for questionnaires need affordance (hover action)	1	4,6

Table 7.8: Daily Questionnaire, expert evaluation findings, grouped topically.

	Topic	Problem	S	HE
1.	Interview History	a. Basic behavior is faulty as it displays and allows question (re)answering that have already become non-indicated	3.5	1,3
		b. Interview history is displayed empty initially	1	8
2.	Progress Bar	a. Estimation of open questions is difficult as progress bar does not increase steadily	1	1
		b. Progress bar mediates that there is more work left than it actually is in tested UI, thus additional visual overhead	1	2,4,8
3.	Navigation Buttons	a. Sometimes not functional, should be deactivated when they have no effect (e.g., first and last question)	2.6	1,2,3
4.	Interaction	a. Click area of radio buttons is too small	3	3,5
		b. Answering a question automatically brings up next question	2.6	3,4,5
5.	Aux.Info Field	a. Fixed auxiliary info. field does not scale with UI automatically	1	8
		b. Explanation text hard to read: Black on (too dark) blue	4	5,7,8
6.	Results Presentation	a. End of session not clearly marked	1	/
		b. Abstract and precise score rating are superfluous for decision tree knowledge—either solution is found (then established) or not	3	2,8
		c. Design of final screen too different from previous design	3	4,7,8
		d. Overlaying presentation of solution justification and -explanation	3	1
7.	General UI Issues	a. Question/answer different styling basically good, but should not use bold print but rather size	1	2,5
		b. Radio button tooltips are obscure	1	1,2,8
		c. UI too broad overall, partly reaches over display borders	1	4,8
8.	Help / Instructions	a. Instructions partly redundant yet also partly missing; too much continuous text	2	1,8
		b. Link to instructions too unobtrusive	/	10

Table 7.9: Strict Interview, expert evaluation findings, grouped topically.

	Topic	Problem	S	HE
1.	Color Scheme	a. Question/solution state single-coded by coloring, potential problem for color-blinds	2	1,5
		b. Switching usage of red and green when answering yes or no is confusing	3	1,4,8
2.	General UI Design Issues	a. Knowledge connector markers vanish sometimes	2	1,4
		b. Help button label 'jurisearch' confusing, UI is called ITree	4	1
		c. Design of results header module not clear	1	8
		d. Hovering text for buttons partly contradicts instructions regarding button action/conseuqences	/	2,(4)
3.	Tree/Node Design	a. Overview is lost w.r.t. large KBs and deep nesting	/	1
		b. Initial partly automatic expansion is confusing	2.5	1,5,7,8
		c. ?-Button meaning not intuitive	4	1,5
		d. Buttons overall too small	3	5,7
		e. Knowledge connector markers (IF, AND, OR) rather confusing and intruding	2	6,8
		f. Empty nodes (for compound rule indication) are not understandable	1.5	2
			1	/
		g. No clear indication of unanswered potentially relevant questions		
4.	Auxiliary Information Box	a. Problem to reach info box correctly as it updates content as soon as other question is hovered	1.5	1,3
		b. Scrolling in auxiliary information box is difficult	2	1,7
		c. Enumerations are confusing (e.g., 1,1,1 instead of 1,2,3)	/	1,(4)
		d. Explanations: Meaning of 'inverse' is unclear	/	1
5.	General Interaction Issues	a. Contradicting answering possible: User entered parent overwrites derived child question answers	3.5	1,2,3, 5,9
		b. Folding triangle buttons not easily recognized	2.5	
		c. (Re)loading/system response time too long	1.5	1,4,8
		d. Expert shortcuts—answering only top questions—not recognized	2	1,3
		e. Blocked CMD/STRG-key: Should be blocked solely for reload, not	2	7
		for e.g. copy/paste		4,7
		f. Hierarchical structure suggests too strongly to start with top ques-	2	
		tions → child questions might be overseen		2

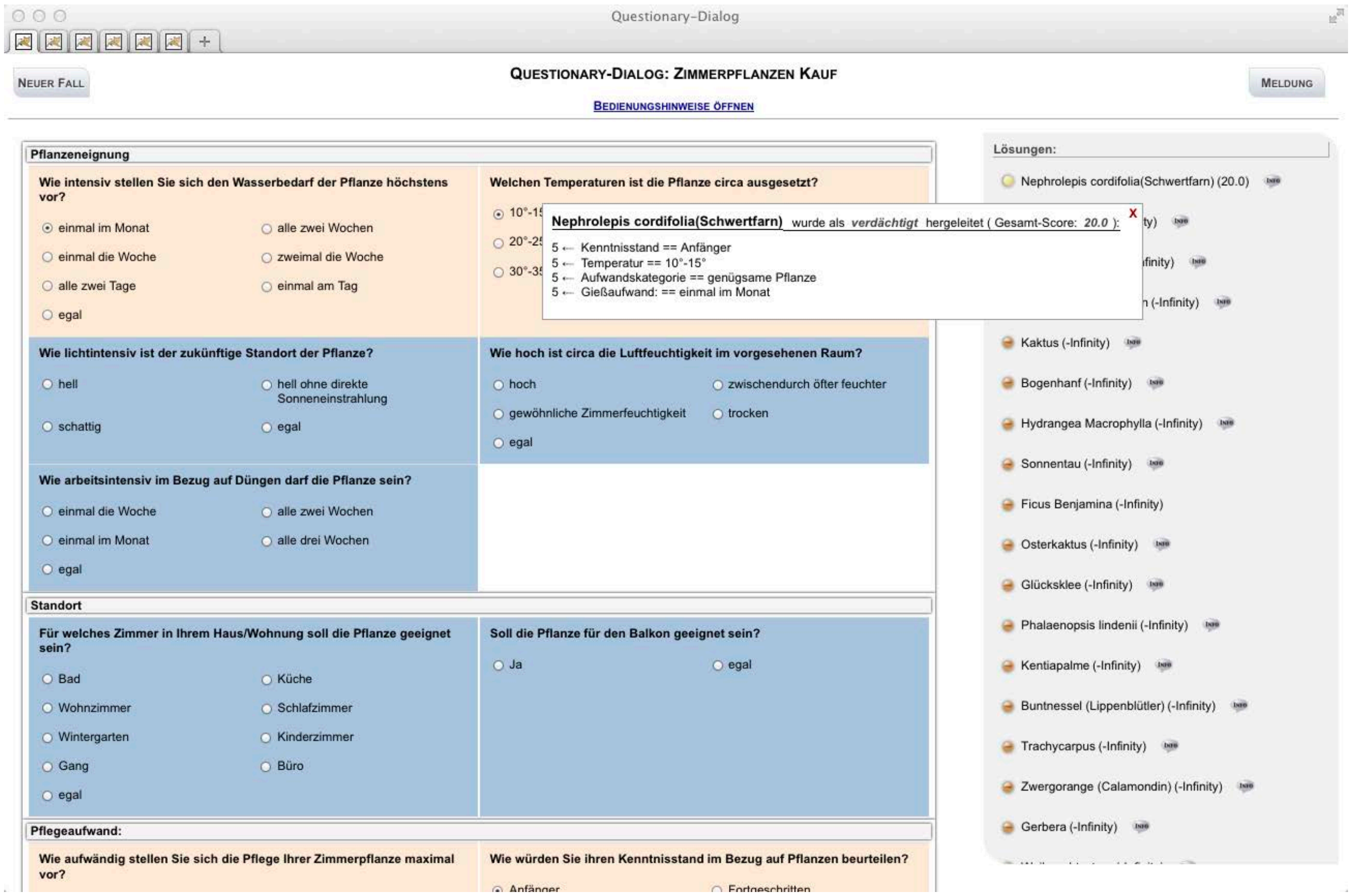
Table 7.10: ITree, expert evaluation findings, topically grouped.

	Topic	Problem	S	HE
1.	General UI Issues	a. Visual overhead / lacking overview once multiple paths are opened	1	8
		b. Font size overall too small, illegible	2	1,2
		c. Tree nodes too broad, are overlaid by auxiliary information box	2	5,6,8,10
		d. Contrast between light nodes and background is not chosen well	1	8
		e. Tree nodes ('buttons') are not clearly bounded w.r.t. click interaction	2	8
2.	Folding Nodes	a. Child nodes are not automatically closed if their parent is closed	1	1,3,6,8
		b. No automated irrelevant-path collapsing	3	1,3,6,8
3.	Auxiliary Info. Box	a. Info box does not highlight the currently explained question	1	5,6
		b. Info box overlays questions partly	3	1,3,7,8
		c. Not possible to open multiple info boxes simultaneously to compare	1	/
4.	General Interaction Issues	a. Missing button for starting new case (needs webpage reload)	2	3
		b. No quick start affordance that tree nodes are to be clicked	2	2,10
		c. Solution nodes do not react on click	2	4
		d. No (expert) interaction shortcuts, e.g. search functionality for retrieving nodes far in the tree, solution listing etc.	2	7
5.	Help / Instructions	a. Link to instructions too unobtrusive	1	10
		b. Instructions are incomplete	/	10
		c. Instructions partly too lengthy, formulated as continuous text	2	1,8
		d. Closing button vanishes when scrolling within instructions	2	3,4

Table 7.11: Hierarchical Interview, expert evaluation findings, grouped topically.

	Topic	Problem	S	HE
1.	General Interaction	a. No warning that *New Case* deletes all entered data irretrievably	2	3,5
		b. No initial check whether JavaScript is enabled	4	1,5,9
		c. No quick start affordances for new users	2	6,8
2.	Feedback Form Issues	a. Label *Meldung* unclear	1	4
		b. No mail address validation	2	4,5
		c. English warning when sending empty fields, german UI	2	4
		d. No success message	2	1
3.	General UI Issues	a. Obscure page title	1	2,8
		b. URL too technical/confusing	1	2

Table 7.12: Listing of the most relevant findings common to all KBS implementations. Findings are grouped topically, and list the average severity rating (S) and the concerned heuristics (HE) of Nielsen (Appendix D.2).



Questionary-Dialog: Zimmerpflanzen Kauf

Pflanzeneignung

Wie intensiv stellen Sie sich den Wasserbedarf der Pflanze höchstens vor?

[✓ einmal im Monat] · alle zwei Wochen · einmal die Woche · zweimal die Woche · alle zwei Tage · einmal am Tag · egal

Welchen Temperaturen ist die Pflanze circa ausgesetzt?

[✓ 10°-15°] · 20°-25° · 30°-35°

Nephrolepis cordifolia(Schwertfarn) wurde als *verdächtigt* hergeleitet (Gesamt-Score: *20.0*):

5 ← Kenntnisstand == Anfänger
5 ← Temperatur == 10°-15°
5 ← Aufwandskategorie == genügsame Pflanze
5 ← Gießaufwand: == einmal im Monat

Wie lichtintensiv ist der zukünftige Standort der Pflanze?

hell · hell ohne direkte Sonneneinstrahlung · schattig · egal

Wie hoch ist circa die Luftfeuchtigkeit im vorgesehenen Raum?

hoch · zwischendurch öfter feuchter · gewöhnliche Zimmerfeuchtigkeit · trocken · egal

Wie arbeitsintensiv im Bezug auf Düngen darf die Pflanze sein?

einmal die Woche · alle zwei Wochen · einmal im Monat · alle drei Wochen · egal

Standort

Für welches Zimmer in Ihrem Haus/Wohnung soll die Pflanze geeignet sein?

Bad · Küche · Wohnzimmer · Schlafzimmer · Wintergarten · Kinderzimmer · Gang · Büro · egal

Soll die Pflanze für den Balkon geeignet sein?

Ja · egal

Pflegeaufwand:

Wie aufwändig stellen Sie sich die Pflege Ihrer Zimmerpflanze maximal vor?

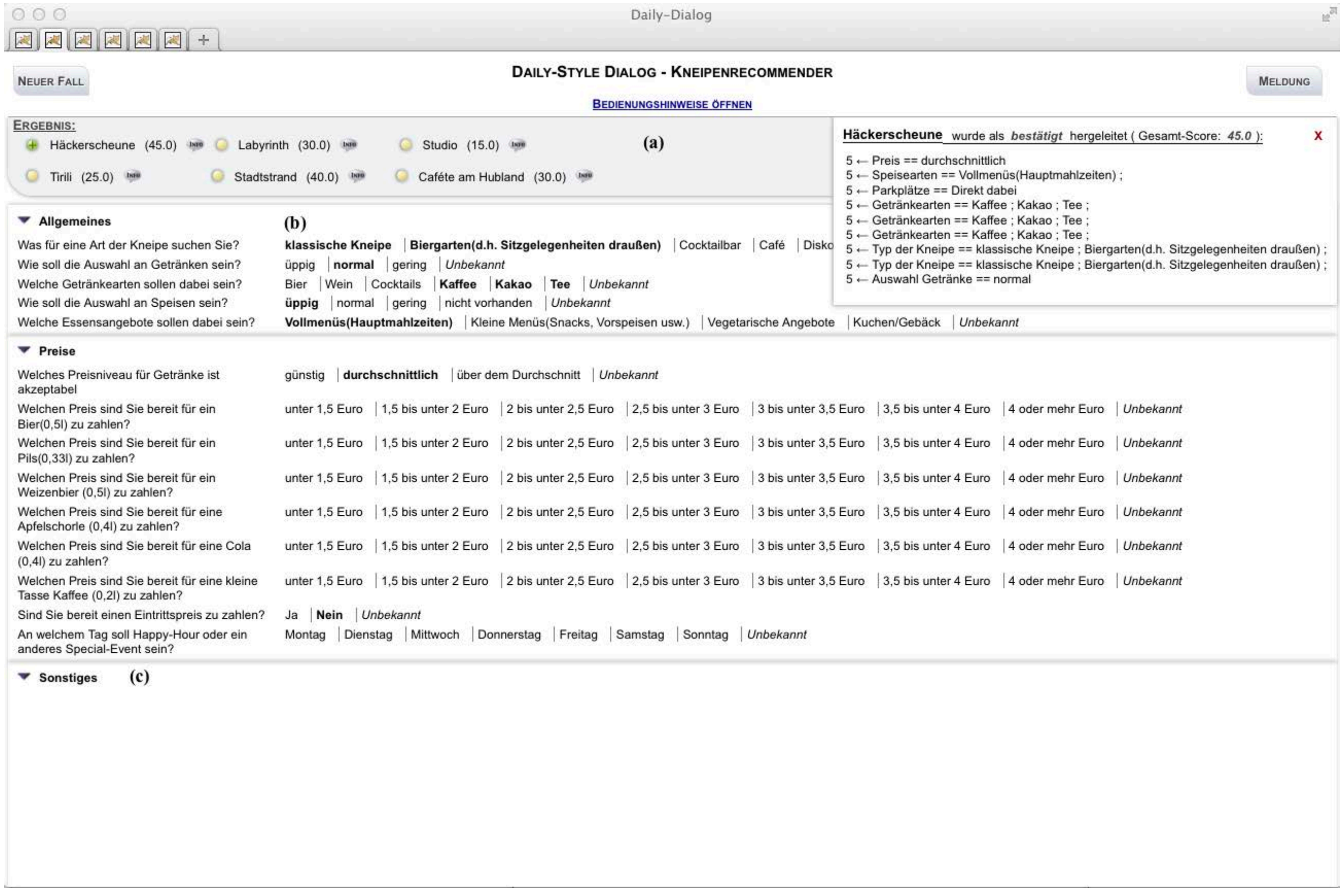
Wie würden Sie ihren Kenntnisstand im Bezug auf Pflanzen beurteilen?

[✓ Anfänger] · Fortgeschritten

Lösungen:

Nephrolepis cordifolia(Schwertfarn) (20.0)
ty)
finity)
n (-Infinity)
Kaktus (-Infinity)
Bogenhanf (-Infinity)
Hydrangea Macrophylla (-Infinity)
Sonnentau (-Infinity)
Ficus Benjamina (-Infinity)
Osterkaktus (-Infinity)
Glücksklee (-Infinity)
Phalaenopsis lindenii (-Infinity)
Kentiapalme (-Infinity)
Buntnessel (Lippenblütler) (-Infinity)
Trachycarpus (-Infinity)
Zwergorange (Calamondin) (-Infinity)
Gerbera (-Infinity)

Figure 7.9: Box Questionnaire implementation with plant recommender KB (german): Anytime solution panel, status mediation coloring, and 1-2-2 columns style—i.e., 1 questionnaire/page, 2 questions/questionnaire, 2 answer columns/question.



Daily-Style Dialog - Kneipenrecommender

Ergebnis:
Häckerscheune (45.0) Labyrinth (30.0) Studio (15.0)
Tirili (25.0) Stadtstrand (40.0) Caféte am Hubland (30.0)

Häckerscheune wurde als *bestätigt* hergeleitet (Gesamt-Score: *45.0*):
5 ← Preis == durchschnittlich
5 ← Speisearten == Vollmenüs(Hauptmahlzeiten) ;
5 ← Parkplätze == Direkt dabei
5 ← Getränkearten == Kaffee ; Kakao ; Tee ;
5 ← Getränkearten == Kaffee ; Kakao ; Tee ;
5 ← Getränkearten == Kaffee ; Kakao ; Tee ;
5 ← Typ der Kneipe == klassische Kneipe ; Biergarten(d.h. Sitzgelegenheiten draußen) ;
5 ← Typ der Kneipe == klassische Kneipe ; Biergarten(d.h. Sitzgelegenheiten draußen) ;
5 ← Auswahl Getränke == normal

Allgemeines

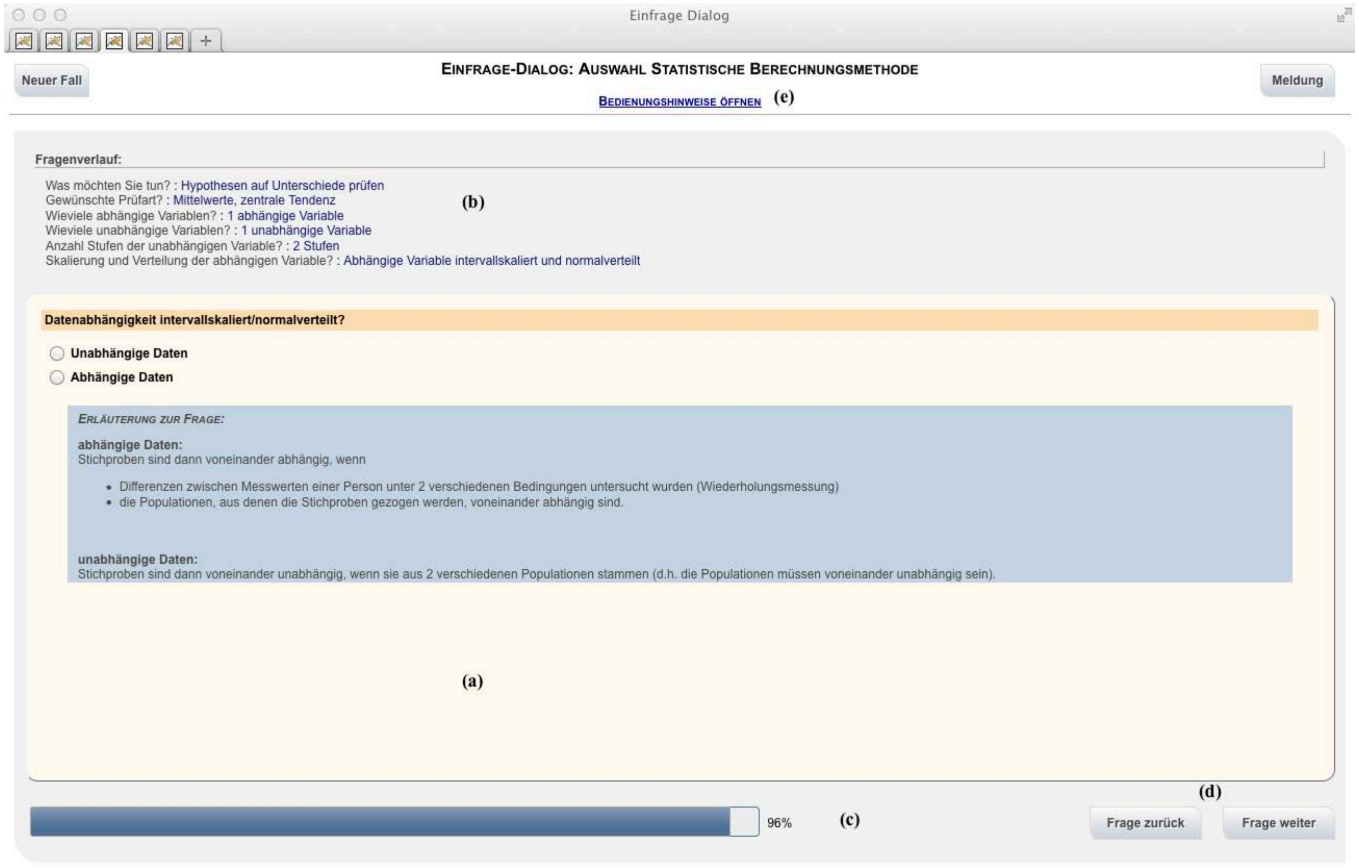
Was für eine Art der Kneipe suchen Sie?	**klassische Kneipe** | **Biergarten(d.h. Sitzgelegenheiten draußen)** | Cocktailbar | Café | Disko
Wie soll die Auswahl an Getränken sein?	üppig | **normal** | gering | *Unbekannt*
Welche Getränkearten sollen dabei sein?	Bier | Wein | Cocktails | **Kaffee** | **Kakao** | **Tee** | *Unbekannt*
Wie soll die Auswahl an Speisen sein?	**üppig** | normal | gering | nicht vorhanden | *Unbekannt*
Welche Essensangebote sollen dabei sein?	**Vollmenüs(Hauptmahlzeiten)** | Kleine Menüs(Snacks, Vorspeisen usw.) | Vegetarische Angebote | Kuchen/Gebäck | *Unbekannt*

Preise

Welches Preisniveau für Getränke ist akzeptabel	günstig | **durchschnittlich** | über dem Durchschnitt | *Unbekannt*
Welchen Preis sind Sie bereit für ein Bier(0,5l) zu zahlen?	unter 1,5 Euro | 1,5 bis unter 2 Euro | 2 bis unter 2,5 Euro | 2,5 bis unter 3 Euro | 3 bis unter 3,5 Euro | 3,5 bis unter 4 Euro | 4 oder mehr Euro | *Unbekannt*
Welchen Preis sind Sie bereit für ein Pils(0,33l) zu zahlen?	unter 1,5 Euro | 1,5 bis unter 2 Euro | 2 bis unter 2,5 Euro | 2,5 bis unter 3 Euro | 3 bis unter 3,5 Euro | 3,5 bis unter 4 Euro | 4 oder mehr Euro | *Unbekannt*
Welchen Preis sind Sie bereit für ein Weizenbier (0,5l) zu zahlen?	unter 1,5 Euro | 1,5 bis unter 2 Euro | 2 bis unter 2,5 Euro | 2,5 bis unter 3 Euro | 3 bis unter 3,5 Euro | 3,5 bis unter 4 Euro | 4 oder mehr Euro | *Unbekannt*
Welchen Preis sind Sie bereit für eine Apfelschorle (0,4l) zu zahlen?	unter 1,5 Euro | 1,5 bis unter 2 Euro | 2 bis unter 2,5 Euro | 2,5 bis unter 3 Euro | 3 bis unter 3,5 Euro | 3,5 bis unter 4 Euro | 4 oder mehr Euro | *Unbekannt*
Welchen Preis sind Sie bereit für eine Cola (0,4l) zu zahlen?	unter 1,5 Euro | 1,5 bis unter 2 Euro | 2 bis unter 2,5 Euro | 2,5 bis unter 3 Euro | 3 bis unter 3,5 Euro | 3,5 bis unter 4 Euro | 4 oder mehr Euro | *Unbekannt*
Welchen Preis sind Sie bereit für eine kleine Tasse Kaffee (0,2l) zu zahlen?	unter 1,5 Euro | 1,5 bis unter 2 Euro | 2 bis unter 2,5 Euro | 2,5 bis unter 3 Euro | 3 bis unter 3,5 Euro | 3,5 bis unter 4 Euro | 4 oder mehr Euro | *Unbekannt*
Sind Sie bereit einen Eintrittspreis zu zahlen?	Ja | **Nein** | *Unbekannt*
An welchem Tag soll Happy-Hour oder ein anderes Special-Event sein?	Montag | Dienstag | Mittwoch | Donnerstag | Freitag | Samstag | Sonntag | *Unbekannt*

Sonstiges

Figure 7.10: Daily Questionnaire implementation with Würzburg's pub consultation KB (german): Anytime top solution panel and 1 column global style—i.e., one questionnaire column, 1 question column/questionnaire.



EINFRAGE-DIALOG: AUSWAHL STATISTISCHE BERECHNUNGSMETHODE

Fragenverlauf:

Was möchten Sie tun? : Hypothesen auf Unterschiede prüfen
Gewünschte Prüfart? : Mittelwerte, zentrale Tendenz
Wieviele abhängige Variablen? : 1 abhängige Variable
Wieviele unabhängige Variablen? : 1 unabhängige Variable
Anzahl Stufen der unabhängigen Variable? : 2 Stufen
Skalierung und Verteilung der abhängigen Variable? : Abhängige Variable intervallskaliert und normalverteilt

Datenabhängigkeit intervallskaliert/normalverteilt?

- ○ Unabhängige Daten
- ○ Abhängige Daten

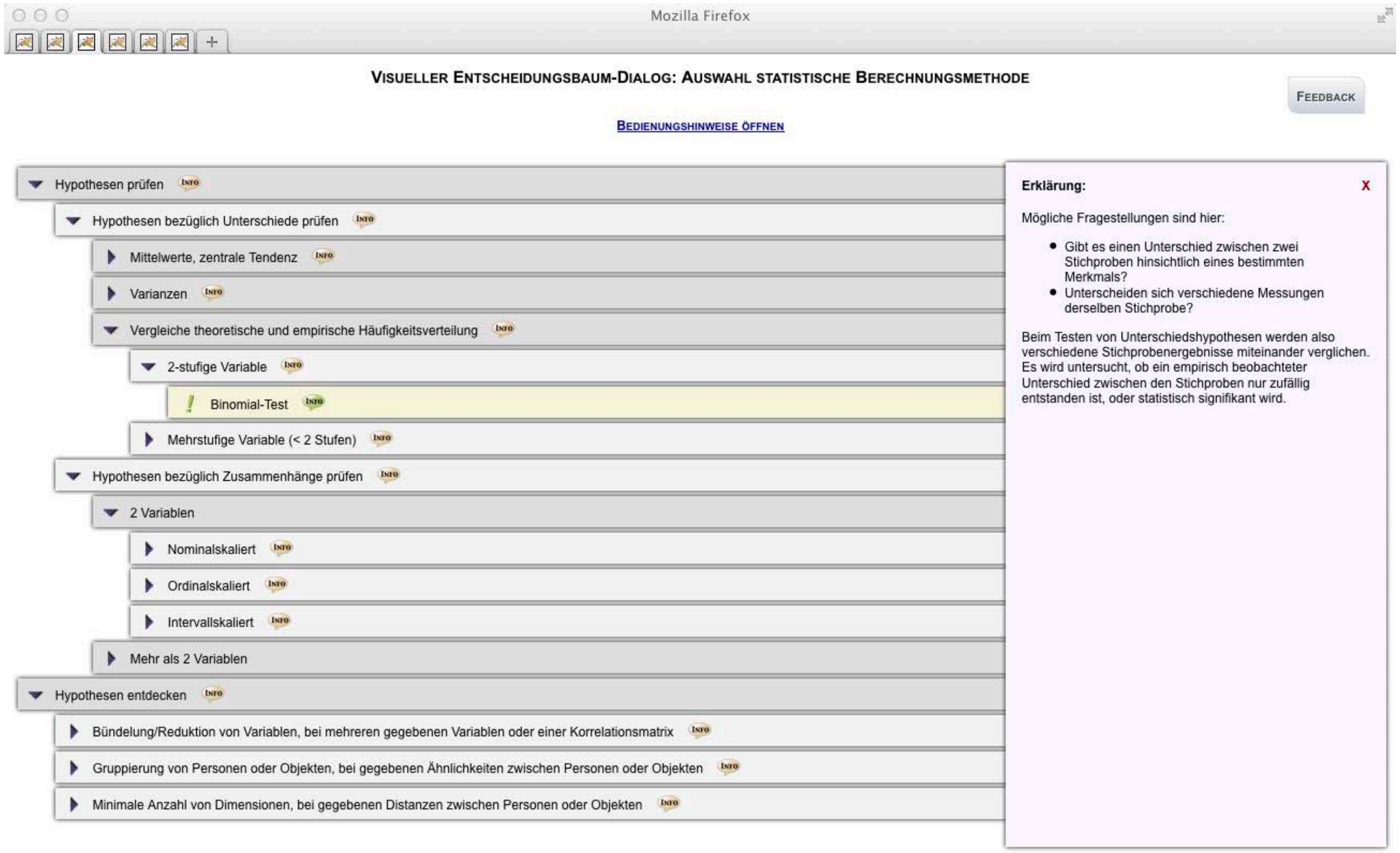
ERLÄUTERUNG ZUR FRAGE:

abhängige Daten:
Stichproben sind dann voneinander abhängig, wenn

- Differenzen zwischen Messwerten einer Person unter 2 verschiedenen Bedingungen untersucht wurden (Wiederholungsmessung)
- die Populationen, aus denen die Stichproben gezogen werden, voneinander abhängig sind.

unabhängige Daten:
Stichproben sind dann voneinander unabhängig, wenn sie aus 2 verschiedenen Populationen stammen (d.h. die Populationen müssen voneinander unabhängig sein).

Figure 7.11: Strict Interview implementation with a statistical recommender KB (german): Additional interview history, progress information, and navigation buttons.



Mozilla Firefox

VISUELLER ENTSCHEIDUNGSBAUM-DIALOG: AUSWAHL STATISTISCHE BERECHNUNGSMETHODE

FEEDBACK

BEDIENUNGSHINWEISE ÖFFNEN

- Hypothesen prüfen INFO
  - Hypothesen bezüglich Unterschiede prüfen INFO
    - Mittelwerte, zentrale Tendenz INFO
    - Varianzen INFO
    - Vergleiche theoretische und empirische Häufigkeitsverteilung INFO
      - 2-stufige Variable INFO
        - ! Binomial-Test INFO
      - Mehrstufige Variable (< 2 Stufen) INFO
  - Hypothesen bezüglich Zusammenhänge prüfen INFO
    - 2 Variablen
      - Nominalskaliert INFO
      - Ordinalskaliert INFO
      - Intervallskaliert INFO
    - Mehr als 2 Variablen
- Hypothesen entdecken INFO
  - Bündelung/Reduktion von Variablen, bei mehreren gegebenen Variablen oder einer Korrelationsmatrix INFO
  - Gruppierung von Personen oder Objekten, bei gegebenen Ähnlichkeiten zwischen Personen oder Objekten INFO
  - Minimale Anzahl von Dimensionen, bei gegebenen Distanzen zwischen Personen oder Objekten INFO

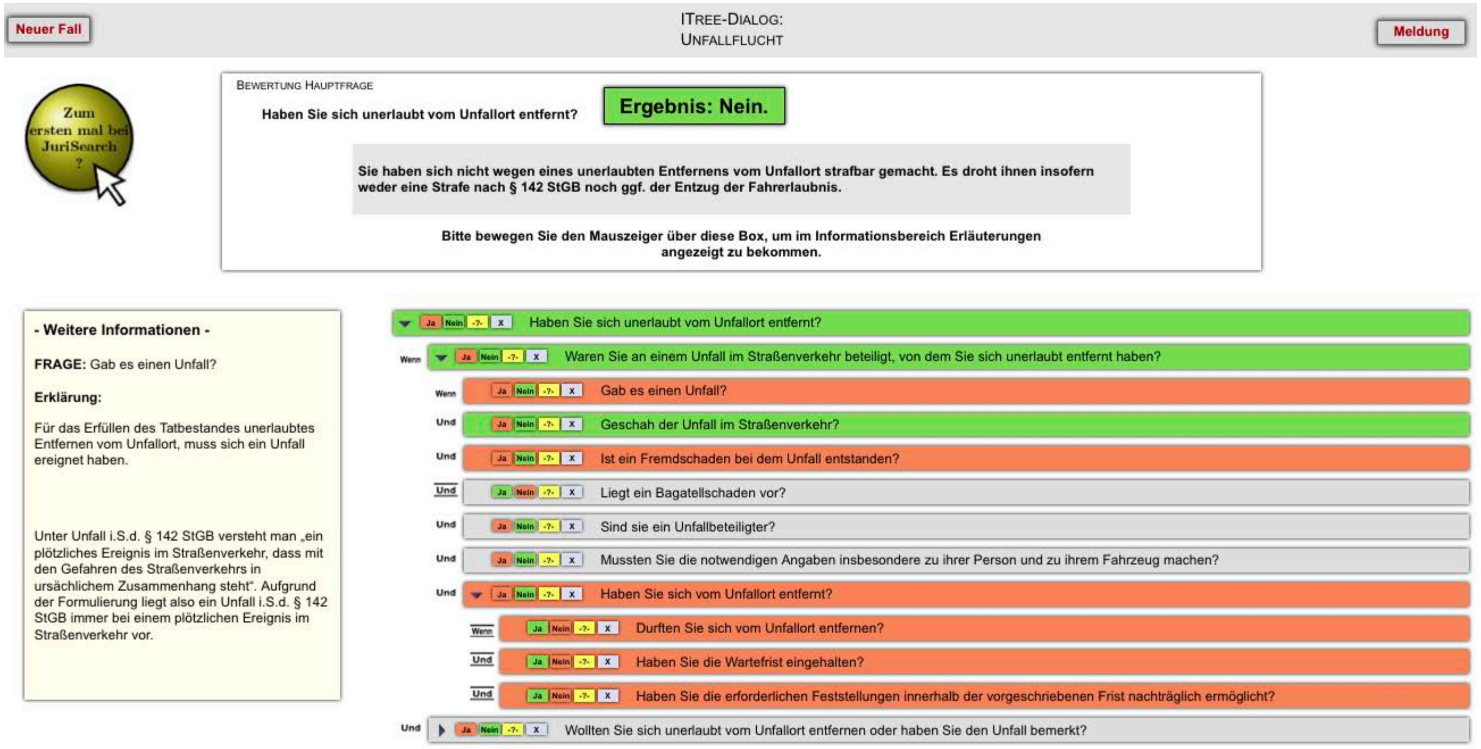
Erklärung: X

Mögliche Fragestellungen sind hier:

- Gibt es einen Unterschied zwischen zwei Stichproben hinsichtlich eines bestimmten Merkmals?
- Unterscheiden sich verschiedene Messungen derselben Stichprobe?

Beim Testen von Unterschiedshypothesen werden also verschiedene Stichprobenergebnisse miteinander verglichen. Es wird untersucht, ob ein empirisch beobachteter Unterschied zwischen den Stichproben nur zufällig entstanden ist, oder statistisch signifikant wird.

Figure 7.12: Hierarchical Interview implementation with a recommender KB (german) on statistical calculation techniques.



ITREE-DIALOG:
UNFALLFLUCHT

BEWERTUNG HAUPTFRAGE

Haben Sie sich unerlaubt vom Unfallort entfernt?

Ergebnis: Nein.

Sie haben sich nicht wegen eines unerlaubten Entfernens vom Unfallort strafbar gemacht. Es droht ihnen insofern weder eine Strafe nach § 142 StGB noch ggf. der Entzug der Fahrerlaubnis.

Bitte bewegen Sie den Mauszeiger über diese Box, um im Informationsbereich Erläuterungen angezeigt zu bekommen.

- Weitere Informationen -

FRAGE: Gab es einen Unfall?

Erklärung:

Für das Erfüllen des Tatbestandes unerlaubtes Entfernen vom Unfallort, muss sich ein Unfall ereignet haben.

Unter Unfall i.S.d. § 142 StGB versteht man „ein plötzliches Ereignis im Straßenverkehr, dass mit den Gefahren des Straßenverkehrs in ursächlichem Zusammenhang steht“. Aufgrund der Formulierung liegt also ein Unfall i.S.d. § 142 StGB immer bei einem plötzlichen Ereignis im Straßenverkehr vor.

```
Haben Sie sich unerlaubt vom Unfallort entfernt?
Wenn  Waren Sie an einem Unfall im Straßenverkehr beteiligt, von dem Sie sich unerlaubt entfernt haben?
      Wenn  Gab es einen Unfall?
      Und   Geschah der Unfall im Straßenverkehr?
      Und   Ist ein Fremdschaden bei dem Unfall entstanden?
      Und   Liegt ein Bagatellschaden vor?
      Und   Sind sie ein Unfallbeteiligter?
      Und   Mussten Sie die notwendigen Angaben insbesondere zu ihrer Person und zu ihrem Fahrzeug machen?
      Und   Haben Sie sich vom Unfallort entfernt?
            Wenn  Durften Sie sich vom Unfallort entfernen?
            Und   Haben Sie die Wartefrist eingehalten?
            Und   Haben Sie die erforderlichen Feststellungen innerhalb der vorgeschriebenen Frist nachträglich ermöglicht?
Und   Wollten Sie sich unerlaubt vom Unfallort entfernen oder haben Sie den Unfall bemerkt?
```

Figure 7.13: ITree implementation with the hit & run accident clarification KB (german). The second assessed ITree used the exact same UI/KBS framework but included a labour legislation KB (german).

7.3.2 KBS UI Patterns—Comparative User Study

Evaluation Goal & Framing Conditions The main objective of this large-scale comparative user study was to investigate the overall perception of the different KBS and their suitability regarding the provided tasks. For this study, three different KBs were used with the five KBS UI variants from the preceding expert evaluation. Thus, they basically conform to the examples depicted in Figures 7.9 to 7.13, pp. 142 ff. In contrast, partly different or refined KBs were used:

- House plants decision support KB—Box- / Daily Questionnaire
- Statistical calculation recommender KB—Strict- / Hierarchical Interview
- Labour legislation clarification KB—ITree

By using two KBs within two different UI solutions each, we hoped for insights regarding which KBS UI type was perceived better suitable for a fixed task. In total, roughly 300 computer science students of the university of Würzburg participated in this study in April/May 2014. Thereby, two tasks were assigned:

- **Task 1**: Solving the given problems with the designated KBS solutions. For details on the tasks, see Appendix C.5.1. Afterwards answering a tailored evaluation sheet, targeting base characteristics of KBS, such as rating the *overall utility of the KBS* on the scale 1 (very good)-6 (very bad).
- **Task 2**: Assessing each KBS implementation with regards to the ISO 9241–110 [ISO 9241–110, 2006] norm.

a. Task 1: Success, Utility/Usability Questionnaire, & Free Comments

Table 7.13 presents the results of Task 1 based on the questionnaire and the calculated success rates. In total, we received 248 utilizable contributions. Box Questionnaire exhibited the highest success rate, closely followed by Daily Questionnaire and ITree. Thereby, Box- and Daily style better supported finding any solution at all, which in turn also resulted in a higher rate of wrong solutions (relatively to Hierarchical Clarifier). Also confirmed by the free comments, this was due to the fact that both the KB and the problem description were more easily understandable for Box- and Daily style than for the more comprehensive statistical calculation domain used with Hierarchical Clarifier. More precisely, this confirmed the assumption, already manifested during the ITree studies: That basically the quality of the KB is an important influencing factor on the overall KBS perception. This is affirmed by the observed correlation between the KB quality Q4, and each of overall utility Q1, belief in the results correctness Q3, efficiency Q7, and success rate. For all those items, strong correlation values exist—Q4/Q1: 0.9986; Q4/Q3: 0.9571; Q4/Q7: 0.9813; Q4:SuccessRate: 0.8325. The slightly weaker correlation value for the success rate can be interpreted as an indicator that, regarding the overall success of a KBS, the KB quality actually is one important, but not the sole, influencing factor. Rather, also the general ease of use/interaction, UI design, as well as the respective problem descriptions add to the overall perception. The fact, that again Box- and Daily Questionnaire exhibit the best ratings regarding the ease of use (Q6) and (Box only) regarding the UI solution also confirms this assumption.

Despite scoring best overall, the aspect of knowledge mediation (Q5) was rated worst for both Daily and Box Questionnaire. We attribute this to the fact, that the Questionnaire styles

Evaluation Item	**Daily** Plants	**Box** Plants	**Hier.Intv.** Statistics	**StrictIntv.** Statistics	**ITree** Legal
Success Measures					
Success Rate (#/%)	220/**88.71**	227/**91.53**	67/**27.02**	50/**20.15**	219/**88.31**
No solution (#/%)	2/**0.81**	5/**2.02**	30/**12.10**	28/**11.30**	17/**6.85**
Wrong solution (#/%)	26/**10.48**	16/**6.45**	151/**60.88**	170/**68.55**	12/**4.84**
Utility & Usability Questionnaire Items					
Q1. Overall utility	**2.04**, 0.86	**1.93**, 0.80	**3.63**, 1.26	**3.06**, 1.16	**2.72**, 1.25
Q3. Belief correctness	**1.68**, 0.89	**1.76**, 0.86	**4.13**, 1.30	**3.86**, 1.31	**3.03**, 1.41
Q4. Knowl. quality	**2.24**, 0.83	**2.16**, 0.84	**3.73**, 1.28	**3.08**, 1.23	**2.82**, 1.22
Q5. Knowl. mediation	**3.79**, 1.40	**3.77**, 1.35	**3.64**, 1.52	**3.18**, 1.43	**2.78**, 1.38
Q6. Ease of use	**1.95**, 1.04	**1.57**, 0.81	**3.30**, 1.54	**2.28**, 1.24	**3.03**, 1.29
Q7. Efficiency	**2.01**, 0.91	**1.84**, 0.77	**3.45**, 1.35	**2.86**, 1.17	**2.83**, 1.11
Q9. Rating UI design	**3.58**, 1.14	**2.60**, 1.03	**3.84**, 1.20	**2.57**, 1.00	**3.12**, 1.29

Table 7.13: Results of the comparative user study regarding five selected KBS UI types, April/May 2014. Not explicitly listed are Q2 which concerned the acquired solution (mirrored in the success rate item), and Q8 which concerned the resolution of the screen. Ratings are provided on a scale from 1 (very good) to 6 (very bad), rating in bold-print, standard deviation in normal print afterwards. Original wording of the questions was german, see Appendix C.5.2.

do not offer an as seamless integration of add-on information as the other styles. Furthermore, the plants domain per se might basically have been already quite familiar to many users, thus the perceived mediation of (new) knowledge might not have been as clear as regarding the more comprehensive statistical and legal domains. Also, despite its overall less good rating than Box, Daily received the best ratings regarding the belief in the correctness of the results (Q3). There, we suspect the highly compact presentation a key factor as this allows for an anytime overview of the selected answers—and thus, also of the potential reasons for the solution. In contrast, Box style requires a lot more scrolling and thus less overview, in turn complicating an immediate estimation which answers caused the solution state(s) and whether this seems reasonable.

The worst results regarding the success rate were achieved by Strict Interview, which partly contradicts the impression from the subjective feedback—where Hierarchical Interview received the worst overall rating. This may be due to the fact, that the overall more intuitively usable UI of Strict Interview can misguide users to decide too quickly for an answer option. In contrast, Hierarchical Interview requires users to explicitly request add-on information for questions by click, which in turn might have led users to more carefully considering those and thus choosing the appropriate options more often. Despite being rated second worst, the basic UI style of Strict Interview received the best ratings of all KBS solutions, which was also strongly confirmed by the free feedback. This basically conforms to the expert assessment, see Section 7.3.1, pp.134 ff., where Strict Interview was amongst the better rated KBS solutions. This once again underlines the tight interplay of KB and UI in a KBS, as here the UI itself was seen as beneficial, yet the negative impression of the KB led to an overall negative rating.

The overall worst rating received Hierarchical Interview. This was somewhat surprising, as also Hierarchical Interview was amongst the better rated solutions in the expert assessment, c.f., Section 7.3.1, p.137, where it had been assessed with the exact same UI and KB. Yet, this reflects the suspicion, that the perception as well as the type and severity of findings may well vary between targeted expert assessments according to fixed guidelines and (potentially rather naive) usage during a plain user study. Nevertheless, this style confirmed to our expectations in exhibiting a better perceived knowledge mediation (Q5) than the two Questionnaire styles. We attribute this to the fact, that Hierarchical Interview, in contrast to the Questionnaire variants, mirrors the knowledge coherences and derivation path(s) of solutions visually.

Despite also addressing a highly complex domain, ITree with the labour legislation KB scored way better than Hierarchical- and Strict Interview. This most probably is due to the fact that the named legal KB has been refined by legal experts over many months—also confirmed by the good rating of the knowledge quality (Q4). Also, several previous studies (c.f., Section 7.2, pp.123 ff.) had detected diverse issues which already had been reworked at that time. However, the basically rather unfavorably rated ease of use (Q6) as well as the additional free comments further conform to our assumption that ITree is specifically suitable for frequent- or trained users with a certain degree of expertise, and not so much for laymen/one-time users.

The additionally provided free comments are subsumed in Table 7.14, p. 150. The unabridged findings spreadsheet is available online, see Appendix E. Thereby, for each KBS UI style, the topical group, the major relevant aspects (Details) and the frequency (F) with which a topic has been addressed are listed. This feedback basically further undermines the general tendency as described above: That the Questionnaire styles Box- and Daily scored best, especially for one-time/laymen users. Those are followed by Hierarchical Clarifier, given some introduction/familiarization. A bad overall rating, but a good forecast regarding its potential when eliminating certain bugs, attained Strict Interview. The worst rating received Hierarchical Interview, which was neither entirely convincing regarding the UI, nor regarding the KB.

b. Part 2: Assessment according to ISO 9241–110

The results of rating the KBS UI styles according to [ISO 9241–110, 2006], subsumed in Table 7.15, further mostly confirm the already described tendency regarding Box and Daily being the best-rated variants and Hierarchical Interview receiving the worst overall results. The unabridged findings are available online, see Appendix E.

One remarkable finding concerns the aspect, that the learnability of ITree exhibited the worst rating. This contradicts the result of Q5 from part 1 of the evaluation, where ITree received the best rating regarding the mediation of knowledge (see Table 7.13, p. 148, Q5). Further, users also mentioned in several comments, that they had been able to gain some insights regarding the legal topic at hand. This conforms to our original assumption, that this style offers a high degree of learnability and skill-building ability. In contrast, the high value regarding a potential individualization (IN6) of ITree complies to our intention that it especially fosters users to process the questions regarding their personal proficiency level—and thus to use ITree in an individual manner. Strict Interview achieved a basically better overall score than in the subjective results (questionnaires) of the user study.

	Topic	Details	F
1. Daily	Unfavorable answer design	Answers hardly recognizable; too many answers too near to each other; no well recognizable feedback regarding answer state; no difference between OC and MC question rendering, no affordances	42
	Well designed	Provides overview; compact presentation; no scrolling required; well structured	27
	Intuitive	Intuitive, simple, and efficient interaction	20
	Unfavorable UI design	UI design not appealing, boring; too little use of colors; partly overlaying widgets when resizing (sometimes rendering some items unusable)	19
	Score/result incomprehensible	Presentation of the score value was not intuitively understood; justifications too formal/specialist	14
2. Box	Well structured	Clearer structured than daily, due to using box design, and color coding answer states	63
	Intuitive	Intuitive, easy to use, self-descriptive	14
	Score/result incomprehensible	Presentation of the score value was not intuitively understood; justifications too formal/specialist	13
	Unfavorable UI design	Too much colors used; UI too lengthy for large KBS; boxes generally too large; multi-column styles with juxtaposed presentation of answer options is confusing	12
3. Hier.Interv.	Expert orientation	Only usable with training or prior knowledge (yet then, potentially efficient); not for laymen/one-time users; too many special terms	44
	Unfavorable structuring	No clear structure/overview; too many node levels; coherent paths not highlighted, thus may be missed; optically not very appealing	42
	Unintuitive	Not self-descriptive; no affordances to indicate click-ability of nodes/interaction; no prominent first-time instructions	20
	Overall favorable perception	Provides overview and clear structure; intuitively usable; automated justification by visual representation	12
4. Strict Interv.	Clear Structure	Tidy, clear layout; less complex than Hierarchical Interview due to single question presentation	38
	Expert orientation	Only usable with prior knowledge; not for laymen/one-time users, rather for experts; too many special terms; too many (and too comprehensive) explanations	23
	Aux. Info. good	Allows for directly comparing options; auxiliary information helpful	18
	Intuitive	Basically intuitive and self-descriptive	9
5. ITree	Unintuitive	Badly structured; too complex; too many node levels; non-intuitive UI; confusing	43
	Well structured	Clear overview; effective and (potentially after initial training) efficient	31
	Red-green probl.	Partly counterintuitive/inconsistent usage of red/green is confusing	24
	Good Add-on Info	Good interaction; information are informative	19
	System response	UI is slow; overall bad responses; reloading/initial render problems	17
	Expert orientation	Too many special terms; prior knowledge required	15
	Laymen suitability	Also suitable for laymen; offers enough information	14
	Good UI design	Fine color scheme; automatically available add-on info is valuable	13
	Unfavorable UI design	Small buttons; UI not appealing; empty nodes are confusing; logical operator icons are confusing	11

Table 7.14: KBS UI Patterns, comparative user study, May 2014—subjective feedback. The table lists the most relevant remarks, both negative and positive, regarding the five assessed KBS UIs, topically grouped and provided with the frequency (F) the topic was addressed.

It scored the second-best ratings in several questions, namely regarding self-descriptiveness (IN2), expectation conformity (IN4), and learnability (IN7). This undermines the general tendency of the previous questionnaire, and the basic impression from the expert assessment, that Strict Interview in general is a favorable, intuitive UI variant.

7.3.3 Overall Insights of the KBS UI Pattern Assessment

Regarding the five assessed KBS UI pattern implementations, *Box Questionnaire* gained the best overall results. As example, its 'structured composition' as well as its 'unobtrusive design' was acknowledged in the expert assessment. Additional feedback from the comparative study further confirmed this perception. An important factor surely is its familiar design resembling

Evaluation Item	**Daily** Plants	**Box** Plants	**Hier.Intv.** Statistics	**StrictIntv.** Statistics	**ITree.** Legal
IN1. Task Adequacy	**2.07**, 1.10	**1.82**, 0.99	**3.10**, 1.45	**2.22**, 1.29	**2.38**, 1.40
IN2. Self-Descriptiven.	**2.24**, 1.12	**1.82**, 1.01	**2.64**, 1.32	**1.98**, 1.13	**2.21**, 1.24
IN3. Controllability	**1.77**, 0.95	**1.78**, 0.88	**2.77**, 1.34	**2.48**, 1.31	**2.10**, 1.17
IN4. Expectance Conf.	**2.19**, 1.16	**1.80**, 0.83	**2.33**, 1.20	**1.99**, 1.13	**2.66**, 1.33
IN5. Error Tolerance	**2.27**, 1.33	**2.24**, 1.40	**2.59**, 1.48	**2.60**, 1.47	**2.59**, 1.50
IN6. Individualization	**3.88**, 1.45	**4.03**, 1.39	**4.26**, 1.15	**4.26**, 1.23	**3.80**, 1.40
IN7. Learnability	**2.54**, 1.39	**2.31**, 1.33	**2.56**, 1.36	**2.40**, 1.31	**3.50**, 1.35

Table 7.15: Results of the comparative rating of five selected KBS UI types, April/May 2014, according to the ISO 9241–110 norm. Ratings are provided on a scale from 1 (very good) to 6 (very bad), the rating value printed bold, the standard deviation afterwards in normal print. For the original wording (german) of the norms, see Appendix D.1.

standard web- and paper-based forms. Also, the applied house plants KB and, consequently, also the tasks to be solved, were not as complex and topically demanding as the legal- or the statistical KBs. Yet, also in general the least basic flaws were reported for this style in the expert assessment.

Daily Questionnaire basically was rated controversially in the expert assessment. On the one hand, it was valued as a very minimalistic and structured style that can present many fine-grained options in a highly compact, clear manner. On the other hand, particularly this compactness simultaneously was suspected a potential source for UI overload. This aspect may be ameliorated in the future by further refining the styling with respect to a better visual indication of its structure. For example, by distinguishing more clearly between question- and answer presentation, or adapting the highlighting of selected items. This was basically also the consenting outcome of the comparative user study—yet there, Daily received an overall more positive rating.

Both *Hierarchical Interview* and *Strict Interview* exhibited the greatest discrepancy between expert- and user study assessment. In the former, both variants received similarly good base ratings. Hierarchical Interview even was remarked to be 'easy, efficient, and functional', up to the level where 'operating errors seem impossible except if induced by a mis-/non-understanding of the knowledge/information'. Yet, this mostly positive valuation of the experts was not mirrored in the comparative study afterwards, where Hierarchical Interview was the (quite clear) overall loser—even though the same UI solutions and KBs were used. One critical point, found in both assessments for Hierarchical Interview, was the basic interaction with the nodes. First, child nodes are not closed automatically with their parents, thus when expanding the parent again, all previously opened children are immediately expanded again, thus cluttering the screen. Second, non-indicated paths in the tree are not collapsed automatically when opening another node—here, however, some comments contrastingly valued this even an advantageous feature as thus several paths could be compared simultaneously. Yet it was remarked consentingly, that a visual indication of coherent paths/nodes could be highly beneficial.

Strict Interview similarly was rated quite good in the expert assessment: As clearly structured, minimalistic, and intuitive, resembling 'talking to a person intending to help you on your problem'. Most evaluators saw high potential in this UI type, especially regarding novice or one-time users. Yet its overall perception was spoilt by some critical flaws that concerned the interview history, the progress estimation, and the navigation buttons (all related to the fact that users could navigate back to already deactivated items and answer them despite their irrelevance for the interrogation). This conforms quite well to the impression gained during the user study. There, Strict Interview was basically also valued particularly regarding its UI and interaction design, yet—similar as with Hierarchical Interview—severe problems with the KB and problem description became evident and seemed to deter the overall rating.

As KB problems were reported both for Hierarchical and Strict Interview in the user study, we suspect this being one of the key factors for decreasing their rating that much, there. This was also confirmed by a quite obvious correlation we found between the KB quality and KBS utility, KBS efficiency, and the belief in the result's correctness (see Section 7.3.2, a., p. 147). Thus, an additional reason for the non-compliance to the expert evaluation could be a differing expertise of the evaluators/users regarding statistical methods and their application contexts.

For the *ITree* style, the assumption manifested that this is not well appropriate for one-time or laymen users, but exhibits its strengths rather for frequent or trained users. In the expert assessment, ITree was assessed with two KBs, one more comprehensive example, and one less extensive example. However, independent from the respectively assessed KB variant still often the basic way of operation, particularly regarding when and with which intention to answer refinement questions, was not intuitively understood—thus supporting the initial assumption regarding suitable target users. Further key points that seemed to decrease the perception of ITree were the coloring scheme, especially in combination with the concept of inverse questions. Those expert findings were quite well confirmed also by the results of the comparative study. Also there, ITree received good overall results, mirrored particularly by its high success rate and most of the queried usability/utility aspects. Yet, subjective feedback of users again reported similar flaws, e.g., with respect to the color scheme. Also, again the assumption was confirmed that ITree requires at least some domain knowledge and familiarization with the UI/interaction.

Based on the results of the KBS UI assessment, we cautiously draw some more general conclusion for the patterns themselves. Box Questionnaire, Daily Questionnaire, and Strict Interview overall seem best suitable for laymen and one-/first-time users due to their basically intuitive usage and design. The concrete decision regarding one or the other style mostly depends on the degree of strictness required from the interrogation and on the desired compactness of the presentation. In contrast, Hierarchical Interview and especially ITree seem better appropriate in the context of trained, frequent, or domain expert users. There, they offer a highly efficient operation and additionally provide a directly integrated base justification. In case, a Clarification KBS is required in the context of rather inexperienced or laymen users, we suspect the hybrid patterns Clarifier Questionnaire and Clarifier Interview (see Section 4.3.5, pp. 67) most favorable—this assumption, though, still is open to verification in future studies.

7.4 Evaluation Synopsis and Lessons Learned

The different case studies and evaluations reported in this chapter underline the strength of the development approach and of the tool ProKEt proposed in this thesis.

First of all, ProKEt was able to live up to its anticipated benefits: To leverage overall KBS development while allowing to focus on so far rather neglected activities such as UI/interaction design and usability evaluation of KBS. This was on the one hand shown by the estimated benchmarks, reported in Section 7.1.1, where former self-contained KBS development was contrasted with todays ProKEt-based development. On the other hand, the reported case studies of actual projects from the medical and legal domain, see Sections 7.1.2, 7.1.3, and 7.1.4 exemplified in detail the activities supported by ProKEt. Each time, this led to a highly agile, iterative development process with positive consequences on the overall course of the project and resulting implementation. As a positive side effect, this further enabled visual interactive KB debugging. For example, the project manager in EuraHS as well as the reference experts in the JuriSearch project profited greatly from the possibility to review their formalized/updated knowledge directly in a fully functional KBS instance instead of debugging plain-text wiki markup or extensive Word documents. Finally, ProKEt also leveraged the setup and conduction of highly iterative KBS usability studies.

The Hierarchical Clarifier UI style itself has been invented and proposed in this work as a novel UI style specifically targeted towards *Clarification KBS*. Repeated evaluations steadily refined the ITree style, a specific instantiation of Hierarchical Clarifier, regarding both the UI-/interaction based solution as well as regarding the inner, KB-related evolution. The corresponding assessments and the gradual evolvement of ITree were thoroughly reported in Section 7.2.

The two assessments (expert- and user study-based) of the five selected KBS UI pattern variants Box-/Daily Questionnaire, Hierarchical/Strict Interview, and ITree, see Section 7.3, showed a quite clear tendency: Overall, Box Questionnaire received best ratings, followed by Daily Questionnaire, Hierarchical Clarifier, Strict Interview, and Hierarchical Interview. Looking at the results more precisely, further the assumption manifests, that Box Questionnaire, Daily Questionnaire, and Strict Interview are especially suitable for one-/first-time users and domain laymen. In contrast, Hierarchical Interview and particularly ITree are more apt for frequent, trained users. Further, especially ITree also could live up to the assumption to provide for a strong explainability and skill-building ability.

During those studies, we often observed also strongly contradicting subjective user opinions. For example, *Daily is well-structured* versus *Daily is ill-structured*. This particularly also underlines that the suitability of a given base style not only strongly depends on the user and usage context. Also, the complexity and domain of the knowledge, and the type of problems/cases to be solved and their description are strongly influencing factors. Thus, it needs to be carefully reconsidered each time anew, which solution to apply. There, a collection of foundational patterns, as proposed in this work, provides strong support in narrowing down potential solutions to one or few most appropriate ones that in the further course can be evaluated against each other in more detail. This was done, for example, in the JuriSearch project, where initially the novel ITree- and a more familiar Interview style had been compared before deciding on the base UI and discussing its further evolvement.

As a fundamental insight, we learned that expectations and assumptions regarding potential target users, their requirements, abilities, and the resulting suitability of a certain KBS solution most often are *not* (entirely) correct—both concerning the overall system design as well as the KB. Consequentially, iterative evaluation and adaption cycles regarding both components are indispensable. Also, the benefits of conducting preliminary evaluations ('assessing the evaluation method') for checking the feasibility of a chosen approach and materials became evident. For example, we learned during the first JuriSearch evaluation that providing the problem and task description additionally in a printed version is more likely to please users and to support the evaluation process. Likewise, the preliminary evaluation before the main expert assessment of the KBS UI patterns showed the need for highly precise task descriptions and fixed rating scales. This again requires appropriate tools for efficiently setting up and—when required—easily modifying evaluation scenarios, including test KBS with different KBs or slightly adapted configurations. As the reported projects and evaluation activities show, ProKEt provides powerful support regarding all those aspects.

Part III

Conclusion

Chapter 8

Summary

The field of knowledge-based systems is still highly active with respect to development methodologies and tools. Yet, the actual engineering of knowledge-based systems (KBSE) remains a challenging task, Due to the complexity of correctly acquiring the relevant knowledge, particularly the aspects of UI design, (usability) evaluation, and user-centered development are rather neglected. In this thesis, we introduced an encompassing powerful, yet straightforward and affordable, KBSE approach as well as corresponding tools for its practical application. As an innovation, we further proposed an entirely novel KBS type—*Clarification KBS*—as a mashup of consultation and justification base interactivity. Also, we delimitated and characterized a basic set of UI design options and, based thereupon, introduced KBS UI patterns both for 'traditional' as well as for the novel KBS type.

8.1 Towards Encompassing KBSE

We motivated three key components for encompassing KBSE: *Tailored KBS UI patterns* as a foundation and inspirational source, that strongly promote the reuse of proven solutions. *Extensible prototyping*, that fosters an UI-oriented, user-centered development process, as well as the seamless integration of KA activities. And *selected usability instruments* for integrating both implicit and explicit usability activities into overall KBSE. Due to the broad range of existing KBSE approaches, mostly focussing on the KB, we did not propose 'yet another methodology'. Rather, we demonstrated the flexibility that arises from combining the proposed key development activities with existing approaches. As an example, we described the integration with the Agile Process Model [Baumeister, 2004]. The resulting encompassing agile process model *EAM* was successfully applied in several current KBS projects, as reported in this work.

8.2 KBS UI Design and Patterns

We distinguish modules for representing four UI-related KBS core parts: The *core input module*, a *results module*, a *justification module* (in case solutions are derived), and *auxiliary information*. Thereby, the core input module denotes the key essential element. In this work, we proposed a collection of KBS UI patterns that describe variants of in total three basic (interaction-focussed) core patterns, as well as one additional, hybrid variant. Those found on the experiences in former KBS projects and adhere to known usability and design principles. Further, we specified several basic configuration dimensions for implementing KBS solutions. The ProKEt toolset comprises reference implementations of both those KBS patterns as a whole and of the named presentation configuration options. However, we aimed not only at providing a plain collection of KBS UI patterns. Particularly, we wanted to ensure a certain quality regarding the basic specification and its practical realization in ProKEt. Therefore, a first selection of pattern reference implementations has been evaluated using both expert evaluation techniques and a large-scale user study: Box Questionnaire, Daily Questionnaire, Strict Interview, Hierarchical Interview, and Hierarchical Clarifier. The results of those studies indicate, that both Questionnaire variants are particularly suitable regarding laymen and one-time users, as they offer a highly intuitive, familiar UI representation and interaction. Also, the Strict Interview variant was rated as particularly apt for this user group based on the expert findings. Yet, in the user study the applied test KB—a rather comprehensive one from the statistical domain—had a deterring effect on the overall results for this style. A similar outcome concerns the Hierarchical Interview, which was rated as potentially efficient UI form by the experts, yet the concrete study results were not as positive. In general, Hierarchical Interview was rated more apt for proficient users due to its specific hierarchical interaction form. Finally, the Hierarchical Clarifier variant received very promising ratings, especially considering the fact that it was applied in (and is especially apt for) a highly expertise domain with complex knowledge and many special terms.

8.3 Clarification KBS for User Participation

Apart from classifying already (partly) known or applied KBS UI solutions in the form of patterns—such as the Questionnaire base style—we further developed a novel KBS type: *Clarification KBS* as a mashup of consultation and justification components that particularly uses backward knowledge. That is, knowledge that is specifically targeted towards a single solution and thus consists solely of questions that potentially contribute to the target solution rating. Clarification KBS further specifically foster usage-related user participation: By allowing for a highly explorative data entry task, and thus enabling users to contribute their personal expertise to the problem solving process. Finally, such systems support the idea of learnability, i.e., to convey knowledge regarding the target domain to the user by simply using the system. During the JuriSearch project, an instantiation of the *Hierarchical Clarifier* pattern, ITree, has been successfully applied for developing legal clarification consultation modules.

8.4 Encompassing KBSE with ProKEt

For practically supporting the proposed development approach we developed the tailored prototyping and knowledge systems engineering tool *ProKEt*. The most powerful features of that tool encompass:

- A *selection of predefined reference implementations of the KBS UI patterns* proposed in this work. Those basically are ready to use out of the box: By simply copying and adapting a corresponding default ProKEt UI specification file. For example, by exchanging only the reference to the desired KB. Or by a straightforward, property-based fine-tuning of the implementations according to the suggested, general KBS configuration options.

- A *basic modular template-based architecture* that fosters the easy extension and adaption of existing KBS templates on several levels of (implementation) detail and expertise. In the simplest form based on KBS UI specs adaption. Also quite straightforward is including own or tailoring existing CSS styles. Structural adaptions are enabled on the template level (HTML, StringTemplate). And finally, the most comprehensive variant denotes (re)writing renderers or further, e.g., interaction-related, modifications on the code level.
- Out of the box available *basic usability evaluation features*. Apart from basically increasing KBS implementation efficiency—and drastically reducing the efforts of setting up frequent, highly iterative evaluations—ProKEt offers two further specific features for supporting KBS usability evaluations: A tailored click logging mechanism for usage data collection and the integration of predefined (quite easily adaptable, when required) utility- and usability evaluation questionnaires. Those features can be activated easily by adding the corresponding property to the KBS UI Specs.
- Support for *intertwined development of UI and KB* by offering several KA extension points: First, the tool KnowOF, which encompasses both a *Microsoft Word parser* for hierarchical clarification knowledge and a more general *Excel/Spreadsheet parser* that flexibly supports diverse knowledge formats. Second, the exemplary coupling with the semantic wiki KnowWE [Baumeister, 2004]. ProKEt and those extension points are seamlessly coupled, thus fostering the development of KB and UI in parallel and inspecting the mutual influences of one part on the other in a direct, effortless manner—therewith strongly supporting the proposed, encompassing user-centered KBSE approach.

8.5 Practical Experiences

Various practical experiences were gained with the proposed approach and tools. *Mediastinitis* and *EuraHS* are two ongoing projects from the medical domain. In both cases, the implemented artifacts are intelligent documentation KBS. Both projects started in-midst of the work on the overall encompassing approach, the pattern collection, and corresponding support by ProKEt. Thus, the focus there lay on performing extensible prototyping in a highly iterative manner with informal expert reviews following each iteration. This basically followed the proposed EAM—yet no concrete, targeted usability studies or experiments were conducted at that time. *JuriSearch*, from the legal domain, required the implementation of separate clarification consultation modules for various legal topics. There, the research and development tools had already matured. This allowed for investigating the novel tailored instantiation of the Hierarchical Clarifier UI style—ITree—during highly iterative, usability-oriented development-and-evaluation cycles in great detail. Again, the basic activity flow closely adhered to the proposed EAM. Apart from those projects that were tied to developing specific KBS types, we evaluated selected basic KBS UI patterns with regards to their general usability and overall inherent design. This aimed at increasing the quality of the proposed KBS UI pattern collection and reference implementations provided by ProKEt.

Chapter 9

Discussion

The proposed encompassing and user-centered development paradigm, the foundational KBS pattern collection, and the tool ProKEt basically proved to be a powerful toolset for leveraging KBSE. In the following, we discuss the main contributions of the presented research to the field of KBS engineering.

9.1 Encompassing, user-centered KBSE—Flexible Ingredients & Enhanced User Participation

The proposed key components of encompassing, user-centered KBSE are deliberately not intended to describe yet another, stand-alone KBSE approach. Rather, they are easily—partly or entirely—integrable with arbitrary KBSE methodologies. The exemplary integration with the Agile Process Model [Baumeister, 2004] was shown in this work.

As motivated, the solitary application of extensible (or other forms of evolutionary) prototyping can be risky from the project management view. Thus, we recommend to embed the described activities into a basic process model—as demonstrated in this work. This helps to settle an obligatory base plan—e.g., regarding intermediate- and final goals, basic responsibilities, etc.—that can help to prevent a failure of the project or never ending development cycles, but that is enhanced by particular UI- and usability related activities.

Based on such a model, another strength of the proposed approach excels: An increased and strong user participation regarding both the development of the KBS, as well as its usage. The EAM, as proposed in this work, together with the matching tailored KBSE tool ProKEt offers room for user participation regarding the aspects of requirements engineering, live adaption/development sessions, intertwined KB/UI development, and repeated evaluation. Consequently, knowledge engineers can better concentrate on UI/interaction- and usability-related activities for working out KBS UI variants with an even stronger usability, leading optimally to an increased, usage-related user participation. As an example therefore, in this work the novel Clarification KBS type and corresponding UI representations were proposed.

9.2 A KBS Showcase of Patterns and Prototypes

Both the pattern collection, their reference implementation, and their configuration options in ProKEt serve as a valuable showcase of different basic KBS solutions. Basically, this has the potential to *foster and enhance requirements engineering*. This is due to the fact, that discussing potential solutions visually and using (static) patterns or (interactive) implementations can greatly ease the understanding of the potential future system and its basic abilities/features for the customers—as already observed by [Bloom and Chung, 2001]. Consequently, requirements then often can be formulated more precisely. This helps to reduce misunderstandings and thus, to decrease the need for future improvement or modification, efforts, and costs. The other way around, the better the intended requirements are met, the greater the chances are for an overall high user satisfaction and KBS usability. We experienced KBS showcasing as beneficial in all practically conducted projects, at the beginning of which we provided a static or interactive KBS solution for the customer: For initially convincing them of the suitability of a certain KBS solution—conforming to the suggestion that demo prototypes of KBS are of strategic importance for demonstrating the feasibility of ideas and value of the targeted technology to management, domain experts, or the own colleagues [Cupello and Mishelevich, 1988]. For accelerating and refining RE, as motivated. And for specifying the resulting target implementation more precisely so that both customer and developer had a clear idea of what to expect of the future system.

Undoubtedly, the proposed collection of UI presentation options and resulting patterns is not entirely exhaustive and there may well be more reasonably interesting presentation options. Therefore, the KBS UI pattern collection is intended rather as a starting point and foundational source of inspiration regarding the design and assembly of KBS UI core components—that can easily (and should) be extended for further contexts/domains.

9.3 Clarification KBS for Comprehensive Domains

Clarification KBS particularly target highly expertise domains—where comprehensive, and complexly interrelated knowledge regarding specific solutions exists. One example from the presented research is the clarification of the correctness of an (unlawful) dismissal. The corresponding KB created during the course of this work consists of (currently) roughly 80 base questions. Particularly in a domain such as legal rights, it is mostly sufficient to answer some few, distinct questions that are most relevant for the case, whereas a greater part often can be left aside. Yet, the introduction of abstraction questions, for grouping such base and refinement questions, likewise can easily grow quite complex itself, too. For example, in the unlawful dismissal example in our work, about 10 such abstraction levels exist.

Thus, standard UI solutions such as a Strict Interview or even a Questionnaire type are mostly rather inappropriate. This is attributed to the fact, that those typically present all interview items (with potentially many irrelevant items) strictly sequenced or within a grid-based rendering without any clear representation regarding the interdependencies of the interview items. The resulting requirement to either answer all those questions (Interview) or not to being able to figure out potentially existing expert shortcuts (Questionnaire) can easily discourage users.

In contrast, the *novel Hierarchical Clarifier* style provides an explicit visual representation of the entire (abstraction) structure of such a KB. This leads to a freely explorable and processable representation, and moreover visually indicates both the coherences between the questions and the consequences of respective answer options. Thus, users can decide autonomously, whether to take the shortcut path (not as much questions overall, but formulated rather abstract/generalized and thus requiring more expertise), or whether to work on the more detailed, base question levels (more questions, yet those are more detailed / precise).

In the course of this work, we developed the *ITree* KBS style as a particular instantiation of the Hierarchical Clarifier pattern. This already has been subject to several different (usability) evaluations. The overall results are highly promising: The basic UI design and interaction still often were perceived as unfavorable or hard to comprehend—thus, the overall rating often was not the best, and there existed more users (relative to, e.g., Box Questionnaire) that could not reasonably handle that UI style at all. Still, the objective *success rate was constantly high* during the last studies (90–100%), implying that users can profit quite well from such an UI once they overcome the initial hurdle and understand the system metaphor. This conforms to another tendency, that emerged during the studies—yet so far has not been particularly formally validated: A better familiarization with the ITree design and interaction with *repeated / frequent usage*, thus leading to a more positive valuation. For familiarized users, also another anticipated characteristic of Hierarchical Clarifier actually was confirmed in the studies: Its ability to *foster and enhance user participation* in the sense that users can contribute their personal expertise to the problem solving process, and in turn to offer a *high degree of learnability*. Another not yet formally validated but quite obvious observation was, that the perception of Hierarchical Clarifier strongly depends on the used KB and on the respective problem descriptions.

In summary, Hierarchical Clarifier denotes a promising step towards the application of Clarification KBS in highly expertise domains where using standard KBS is no option. Yet, further research efforts are necessary for decreasing the initial hurdle of using this KBS type. For reducing the familiarization efforts, we further assume the proposed hybrid KBS types, such as Clarifier Interview, to be beneficial. Yet, their particular consequences still need to be formally invested. This includes aspects such as whether such hybrids actually can live up to the pure Hierarchical Clarifier variant regarding efficiency, skill-building-ability, and user participation; or whether they really do decrease the initial usage hurdle as anticipated, or if new/further problems arise.

One idea in this regard is, to offer hybrid clarification KBS variants, such as Clarifier Interview, but extend them by an interactive Hierarchical Clarifier status view. That is, the user firstly progresses through a more guided clarification consultation, yet is provided with an overview of the current state (which question at which hierarchical level is currently inspected) by the status view widget. Additionally, this widget is clickable, and triggers a fully functional, respectively expanded Hierarchical Clarifier UI. Thus, users are enabled to switch between a guided progression (Interview/Hybrid view) and an explorative, potentially better comprehensible and explicable UI (Hierarchical Clarifier view).

9.4 *ProKEt*—Template-based, Modular KBSE Tool

On the practical side, we introduced the tailored prototyping and knowledge systems engineering tool *ProKEt* was introduced. An initial conceptualization of ProKEt founded on the outcome of a master thesis [Mitlmeier, 2010]. Major parts of this work later were re-implemented by the author for enhancing the basic architecture and ameliorating the modularity and extensibility of the tool. Further, ProKEt was extended for integrating [d3web, n.d.] KBs as well as regarding the supported KBS base patterns and configuration possibilities. Therewith, it matured from a pure prototyping framework to a comprehensive KBSE tool. Also, a bachelor thesis [Coca, 2013] investigated possibilities for integrating justification visualization in ProKEt. Further extensions, mostly related to specific project requirements (EuraHS / JuriSearch), were provided by the colleagues Albrecht Striffler and Felix Herrmann. The KA extension points were implemented by the colleagues Elmar Böhler (Microsoft Word parser for JuriSearch), Felix Herrmann (KnowOF KA tool), and Georg Dietrich (general spreadsheet/Excel parser). This encompassed transferring the informal—yet for the project partners more intuitive—office-based knowledge specification into formalized d3web KBs and ProKEt KBS UI specification files. The functionality for correctly processing the parsed KBs and propagating the presentation options to the ProKEt artifacts was extended and gradually refined for the most part by the author.

ProKEt as encompassing KBSE tool fosters several interesting use cases. The highly modular and extensible architecture of ProKEt regarding the realization of UI widgets and more comprehensive UI modules enables and eases *UI-related experimentation* in the context of KBS. Therefore, several levels of adapting existing or defining new KBS UI styles requiring various efforts and experience exist. During the Mediastinitis project, for example, the initial prototypes used in the RE and system definition phase were configured mainly using predefined properties (e.g., multi-column and single-column style) and exchanging/adapting CSS files. In contrast, the addition of tailored renderers, e.g., for the ITree implementation, required way more efforts and time. However, the modular reuse-fostering architecture of ProKEt provided strong support also here.

ProKEt basically provides all general modules and functionality in a readily available manner, so that they can be activated easily for all ProKEt artifacts likewise. Examples are session management, multilingualism features, or a default results or solution display panel. Thus, especially also the extension steps as proposed for *extensible prototyping* are fostered: On the one hand, merging KBS UI and KB into a functional core KBS. And further (or enabled by ProKEt, also simultaneously) extending them into fully productive KBS solutions. This renders ProKEt-based KBSE in general *very efficient and thus affordable*, basically fostering UI related activities to be integrated explicitly in KBSE, and further supporting user-centered development in various regards.

Finally, it can be argued that the provision of fixed patterns and respectively pre-implemented UI modules rather constrains potential creativity regarding the realization of novel solutions. We argue, however, that the patterns (and reference implementation in ProKEt) are not to be seen as strict prerequisite, but are more intended as a foundation for further development. That is, they define the most appropriate base interaction and UI KBS type for a given context and framing conditions, but their particular realization and fine-tuning still is up to the respective developer. There, the modularity and extensibility of ProKEt encourage the contribution of further adaptions or extensions for the foundational pattern collection and reference implementations.

9.5 Intertwined KB and UI development

The encompassing KBSE paradigm and ProKEt both explicitly foster the *tightly intertwined development of KB and UI* as well as the consequential *visual support for KB assessment*. This already proved to be highly beneficial in actual projects. First, the possibility of visually debugging the KB, was especially approved of by the domain experts that participated in the KA process. Examples are the EuraHS and the JuriSearch project, where experts upload and immediately review their knowledge specification within a functional KBS instance on a demo server. Apart from saving experts' traveling costs and time, c.f., [Duan et al., 2005], we recognized that that active expert participation in fact increased their identification with and belief in the final KBS. This further strengthened the overall cooperation and decreased the risk of canceling the project midways. This conforms to the claim of [Nurminen et al., 2003], that experts are the more positive towards the new system the more they are able to influence the development process. For knowledge engineers, such visual KB debugging admittedly cannot (fully) replace other, more formal methods or more comprehensive tools for KB evaluation; examples are the systematic investigation regarding redundant KB objects. Regarding some other tasks, however, the visual examination can provide clear benefits. For example, with respect to the investigation of the correctness of predefined interrogation paths, which may be rather cumbersome based on spreadsheets (KnowOF) or plain text markup files (KnowWE).

At any rate, the separate but intertwined development paradigm enables a realistic preview and intuitive access of the KB in its target environment. Thus, the *efficient investigation of the mutual effects of KB- and UI design* is eased and consequently the detection of other problem classes, arising due to the strong mutual influence of KB and UI, is fostered. As example, comprehensive question texts might not work well with diverse target UI styles/widgets. Yet, it should be taken care not to mix UI- and KB-based experimentation regarding the same issue too loosely, as this might obscure rather than support any real gain of insight. In the question formulation example, that could imply to construct two separate prototypes—one showing an exemplary adaption of the UI but using the original KB, the other one using a partly reformulated KB within the original UI—and to compare the results of both examples before adapting the entire UI or KB, respectively.

9.6 Fostering Usability in the KBS Context

The support for easily creating adapted, fully functional KBS artifacts rather than pure prototypes in a straightforward manner, as enabled by ProKEt, fosters highly iterative development cycles. It is quite an established view "[...] that user interfaces should be designed iteratively in almost all cases because it is virtually impossible to design a user interface that has no usability problems from the start" [Nielsen, 1993a]. Additionally, iterative development offers the chances to detect potential problems earlier in the course of development, which in turn can help to reduce adaption/correction costs compared to when certain problems are not discovered before the end of development. We saw this confirmed in the conducted projects, during which we experienced such frequent iterations as highly valuable: On the one hand, in fact the overall quality of the KBS increased. On the other hand, also the participating experts were satisfied with observing the continuous progress of 'their' system and thus felt been taken seriously regarding their requirements and participation. Also, those iterations helped to demonstrate both developers and experts, whether (or not) they are still pursuing the right paths according to the consented goals.

Apart from that implicit usability enhancement, the readily available functional KBS artifacts created with ProKEt offer the chance to integrate also targeted usability evaluation features at any time. This can easily be done anytime and free of charge or additional efforts. The value thereof became evident especially during the JuriSearch project, which profited strongly from the frequently conducted formal KBS evaluations—which were conducted in addition to even more frequent adaptions and resulting informal assessments with the project partners. There, especially the average task time or the logging of the derived result were important evaluation benchmarks that could be easily collected and evaluated. Overall, the frequent evaluation activity led to a notable amelioration of the ITree KBS UI style, that was developed as a novelty in this project. Also regarding the KBS UI pattern assessment, the support for easily setting up alternative KBS variants with minimal efforts leveraged to conduct comparative user studies more than once, and with slightly changing objectives.

Being enabled to both implicitly and explicitly assess KBS solutions is even the more relevant as we learned, that the quality and usability of a KBS never regards the UI alone, but also always entails the comprehensiveness of the KB as well as of the problem descriptions. Thus, the intertwined KB and UI development, as well as easily and highly iteratively assessing resulting artifacts either informally or in formal studies, can considerably increase the overall usability, user acceptance of, and user satisfaction with the KBS. We regard this as particularly essential, as KBS often are applied in critical contexts where failures of such a system can directly influence human health or lead to considerable financial losses. Thus, subsumed, the proposed approach and corresponding tool ProKEt offer great potential for ameliorating KBSE in an encompassing way and thus denote an important contribution to the current KBS research.

Chapter 10

Outlook

The novel development paradigm and corresponding development toolset for web-/browser-based KBS presented in this thesis lays a powerful foundation for a more encompassing, UI-, user-, and usability-focussed development process. Thus, this work denotes an important theoretical and practical contribution to current KBS research. Despite their general applicability, utility, and profit for the overall development process, the topics included in this work also open doors towards various exciting future research directions.

10.1 Expanding Development Tools

ProKEt and its KA extensions are proposed as an encompassing KBS prototyping and development framework that supports the intertwined development of UI and KB, as well as usability evaluation activities. Apart from a general extension of the ProKEt UI design capabilities—regarding the realization of additional or the adaption of existing KBS UI patterns—the following, more comprehensive ideas seem worth considering.

Knowledge Expansion Regarding the back-end, ProKEt currently supports [d3web, n.d.] KBs only. This decision is founded on the fact that this toolkit has been developed and refined over years, is strictly quality controlled yet open source, and supports various forms of knowledge (e.g., rule-based, set covering models) out of the box. The spreadsheet-based KA extension *KnowOF* so far exhibits considerable capabilities that satisfy the base requirements of most current KBS projects. However, the [d3web, n.d.] toolkit—and also the semantic knowledge formalization wiki KnowWE—provide a lot of additional functionality that may be interesting for future projects. Thus, a corresponding extension regarding the d3web functionality space denotes an important and valuable step. To add even more to the flexibility of ProKEt and for opening up towards potential further application contexts, the integration of further knowledge representations and reasoners is promising. Further established, well-investigated knowledge formats include case-bases, bayesian-, and neural networks. In all those areas, a vivid research community has evolved and manifold reasoning methods and corresponding tools are available. Examples include but are by far not limited to *myCBR* [Roth-Berghofer et al., 2012] or *jCOLIBRI* [Recio-García et al., 2008] for CBR. Or *The Bayes Net Toolbox for Matlab* [Murphy, 2001], or *GeNIe & SMILE* [GeNie & SMILE, n.d.] for bayesian nets. In the times of the ever-increasing evolvement of the Semantic Web, the integration of ontologies and ontology-based reasoners—examples include *Sesame* [Broekstra et al., 2002] or *Jena* [McBride, 2002]—also denotes an exciting and promising topic.

Supporting Visual Interactive Development Processes Also aiming towards more visual and interactive development processes are a prospective further direction for ProKEt. The first possibility is some kind of *drag & drop* editor. Such an editor would provide standardized widgets, default KBS UI frames, etc. as offered by ProKEt, and allow users to construct KBS UIs by simply and freely dragging and dropping the required widgets. Another vision is to provide specific *interactive KBSE or adaption views* that can be directly modified. There, a selected base UI is rendered—e.g., a default Questionnaire variant—and in contrast to the standard view provides additional facilities for directly fine-tuning the UI within itself. For example, by offering tailored menus or popups that allow for configuring the base view according to the presentation configuration options for KBS components introduced in this work.

Such visual and interactive development extensions presumedly offer the following benefits: Firstly, fostering a more intuitive overall UI development workflow as the visual manipulation of UIs / widgets is already quite common. Secondly, rendering development activities more efficient, as moving around and assembling UI widgets, or adapting them via menus or popup facilities, can probably be done more quickly than adapting or extending comprehensive templates. It has to be noted though, that such facilities most likely are not feasible to provide all fine-granular configuration options as are made available when modifying the StringTemplate files or the ProKEt code base (e.g., when implementing own, tailored renderers). However, regarding the proven, oftentimes (re)used templates and configurations, such an editor assumedly would work well. Specifically the investigation regarding the feasibility of such a tool in general, and some benchmark comparisons with the already provided ProKEt development facilities and capabilities, seems an interesting topic.

Visual Evaluation Support In some evaluation scenarios, it is required to compare the expected input regarding a targeted solution with the actually provided user input and resulting solution. This can be leveraged by a tailored, visual comparison editor: Such a tool would allow for loading both the expected target input and solution, as well as the actual user input. Inspired by modern version control tools, this input can, as example, be displayed in some split screen presentation, thereby distinctly highlighting their differences—e.g., deviations of actual user input with required input for retrieving a certain solution.

10.2 Expanding the Theories

Apart from the practical tools, also the suggested encompassing KBSE approach and the KBS UI patterns imply further interesting research opportunities.

Encompassing KBSE key activities and further KBSE approaches As demonstrated, the suggested encompassing development paradigm and corresponding key activities, such as extensible prototyping, can be well integrated with already existing, KA focussed methodologies. Apart from the Agile Process Model used in this work, to date many further KBS development methodologies are well researched and practically applied, such as CommonKADS [Schreiber et al., 2000], Mike [Angele et al., 1998], or Scrum [Beedle and Schwaber, 2002]. Therefore, the methodological integration of the named key activities with such methodologies denotes one interesting starting point for further research. Apart from such a theoretical specification of resulting composite development approaches, especially practical experiences are of interest: That is, their actual usage in further projects and a (comparative) specification of resulting experiences—for example, concerning the contexts when to apply them and consequential results for the respective project.

Towards a KBS UI Pattern language In this thesis, we suggested a basic collection of UI styles and presentation configuration options for KBS UIs, as well as a first set of core KBS UI/interaction patterns. Naturally, that proposed collection—fore-mostly based on actual project experiences—yet comprises only a small portion of the potential design and implementation space. There, further application contexts and domains may even lead to more adapted or entirely novel presentation forms. Particularly the field of justification/explanation has been researched vividly for years and still denotes a topical issue. Examples include but are not limited to [Southwick, 1991, Barzilay et al., 1998, Mao and Benbasat, 2000, Richards, 2003, Roth-Berghofer, 2004, Sørmo et al., 2005, McSherry, 2005, Pu and Chen, 2006, Tomic et al., 2012, Baumeister and Striffler, 2013]. In those works, both the final justification of a KBS (i.e., of the derived results), as well as a KBS's explainability during its usage, are investigated. This research direction should be assessed systematically and in-depth, regarding the feasibility of an encompassing classification and delimitation of results presentation/justification- and explanation (in the sense of auxiliary information) patterns.

The availability of patterns for core input-, results-, and auxiliary information display then further facilitates the specification of an all-encompassing *KBS UI pattern language* as attractive long-term goal. In his seminal work about an architectural pattern language, [Alexander et al., 1978] postulates the necessity of defining relationships and connections between the contained, singular patterns. This enables users of such a language, to recognize related solutions at one glance—that is, either related solutions at the same hierarchical level, or sub-solutions that solve only dedicated parts of the larger-scale problem. In this work, we already saw a first example regarding such an interconnection between patterns: The CheckList style, which can realize a self-contained KBS solution, but can also be integrated with Questionnaire or with Grouped Interview as sub-module. Similarly, such interrelations should to be worked out also for explanation-/justification-related patterns. For constituting an overall encompassing pattern language, however, it needs to be investigated if one or even more general pattern levels can be specified—where patterns address the entire KBS UI with all relevant sub modules and describe reasonable combinations of patterns for each contained sub-module. In case this is not feasible, the resulting separate pattern collections with fewer or weaker interconnections rather constitute a *KBS UI pattern catalogue*, a term brought up by [Gamma et al., 1994] in their foundational work regarding Software Design Patterns. Nevertheless, KBS UI patterns provide exciting research opportunities with the potential of both contributing valuably to the theoretical research foundation and to practical KBSE.

10.3 Further KBS Input and Output Dimensions

The extension of KBS input and output dimensions denotes on the one hand a necessity—due to the current and constantly ongoing development regarding computers and similar devices. On the other hand, it is a challenging and interesting field for further research efforts in itself.

Input: Text- and Speech Recognition The introduced patterns and corresponding implementations basically focused on standard form UIs for the web browser that typically require manual user input. However, especially regarding larger systems, support for text- and speech recognition and based thereupon a tailored KBS interrogation control denotes another promising extension point. The general idea thereby is, to extend KBS UIs additionally by particular input widgets that support text- or speech recognition. There the user enters (manually or by speech) either the desired question, or a complete question/answer pair. The KBS then filters the KB: In case the KBS was not able to clearly associate the input with interview objects, the potentially best matching items are displayed for further manual processing (i.e., reading and manually clicking desired answers). If on the other hand the input can be clearly matched and consists of both a question and corresponding answer option, the KBS automatically propagates that finding in the current KBS session and re-renders the KBS UI, highlighting the thus answered latest finding.

Assumed advantages of such an extension are: More efficient processing of large KBSs, where the standard manual interaction may be too cumbersome. Or the context of service support, where the operating staff needs to access certain questions (typically based on keywords/keyphrases) and corresponding answer options out of a large set of items as efficiently as possible. There, such an enhanced input facility can serve as a filter mechanism to quickly retrieve the desired items and thus shorten the interrogation to the required minimum. The in-depth analysis of existing text- and speech recognition approaches for the sketched context, as well as the practical integration with the ProKEt toolset, is an exciting future research opportunity.

Output: Going Mobile for Smartphones and Tablets So far, the presented research first and fore-mostly targets standard browser presentation. However, mobile devices, such as smartphones and tablets gain increasing presence today and oftentimes replace the usage of personal computers entirely—both in business- and in private usage contexts. One of many examples is an expert system for mobile retinal disease detection [Bourouis et al., 2014]. In that context, the UI-based optimization—required due to even fewer available UI space—poses one of the biggest challenge. Also, additional interaction types based on swiping- or touch gestures need to be considered. Also, the aforementioned enhancement by text- and speech-recognition can add to the usability of mobile KBS UIs.

10.4 Visualization for KBS Justification

Apart from core KBS object presentation options, we suggested also a first conceptual collection of KBS justification presentation forms, including some visual variants. Some of them have further been implemented as a proof of concept for ProKEt. Here, the in-depth investigation of further, appropriate visualization and presentation techniques for KBS justification, as well as their implementation and evaluation, offer ample room for exciting research.

10.5 Testing, testing, testing...

Both the approach and tool proposed in this paper foster the efficient creation, adaption, and highly frequent assessment of KBS artifacts. Therefore, encompassing and manifold evaluation activities denote a straightforward, but not less beneficial field for further research. This encompasses, but is not limited to:

- A basic utility/usability assessment of the so far visionary, sketched KBS UI patterns, similar as already reported in this work for selected base patterns.
- Comparative evaluations regarding assumedly similarly applicable patterns so far not considered—e.g., Box/Daily Questionnaire vs. CheckList style for KBs with suitable question sets. Or Box Questionnaire vs. Grouped Interview (i.e., explorability vs. part guidance).
- A thorough investigation regarding potential enhancements of the novel Clarifier KBS UI type itself—e.g., modifying the auxiliary information retrieval, assessing different base color schemes, or integrating a special terms lexicon that automatically explains such terms when hovering them.
- Investigating the tradeoffs of pure Clarifier solutions in comparison to Hybrid Clarifier solutions—where the latter assumedly do not offer as much efficiency, flexibility, skill-building ability, and user participation, yet in turn provide a much more intuitive base interaction/design.
- Investigating the value of combined Clarifier Interview–Hierarchical Clarifier KBS where the respective view can be switched by users at will—as sketched in the discussion (Section 9.3, p. 163). This regards issues, such as whether these variants denote too much overload for users, whether users are able to grasp and exploit the KBS concept, etc.
- Verifying more formally the correlation between KB quality, problem description quality, and overall KBS results. That is, with respect to whether all those aspects weigh the same on average. Or whether the current assumption, backed on first evaluation results, proves true that the KB may be the most influential factor.

We regard this aspect of thoroughly and iteratively testing, evaluating, and refining both seemingly obvious, and bravely envisioned, KBS UI solutions as essential...

...because the simple is never obvious! (Peter Rudl, *1966))

Bibliography

[Afacan and Demirkan, 2011] Afacan, Y. and Demirkan, H. (2011). An ontology-based universal design knowledge support system. *Knowledge-Based Systems*, 24(4):530–541.

[Akerkar and Sajja, 2010] Akerkar, R. A. and Sajja, P. S. (2010). *Knowledge-based Systems*. Jones and Bartlett Publishers. ISBN 13: 978-0763776473.

[Alexander et al., 1978] Alexander, C., Ishikawa, S., and Silverstein, M. (1978). *A Pattern Language: Towns, Buildings, Construction*. Oxford University Press, New York. ISBN 13: 978-0195019193.

[Altenkrüger and Büttner, 1992] Altenkrüger, D. and Büttner, W. (1992). *Wissensbasierte Systeme*. Vieweg. ISBN 13: 3-528-05244-9.

[Angele et al., 1998] Angele, J., Fensel, D., Landes, D., and Studer, R. (1998). Developing Knowledge-Based Systems with MIKE. *Automated Software Engineering: An International Journal*, 5(4):389–418.

[Arnowitz et al., 2006] Arnowitz, J., Arent, M., and Berger, N. (2006). *Effective Prototyping for Software Makers (Interactive Technologies)*. Morgan Kaufmann.

[Başçiftçi and İncekar, 2011] Başçiftçi, F. and İncekar, H. (2011). Web based medical decision support system application of Coronary Heart Disease diagnosis with Boolean functions minimization methodased Medical Decision Support System Application of Coronary Heart Disease Diagnosis with Boolean Functions Minimization Method. *Expert Systems with Applications*, 38(11):14037–14043.

[Balsamiq Studios, n.d.] Balsamiq Studios (n.d.). Balsamiq Mockups. Online (accessed May 23rd, 2014). `http://balsamiq.com/products/mockups/`.

[Barzilay et al., 1998] Barzilay, R., Mccullough, D., Rambow, O., Decristofaro, J., Korelsky, T., Lavoie, B., and Inc, C. (1998). A New Approach to Expert System Explanations. In *Proceedings of the 9th International Workshop on Natural Language Generation*, pages 78–87.

[Baumeister, 2004] Baumeister, J. (2004). *Agile Development of Diagnostic Knowledge Systems*. IOS Press, AKA, DISKI 284. ISBN 13: 978-1586034634.

[Baumeister and Freiberg, 2010] Baumeister, J. and Freiberg, M. (2010). Knowledge Visualization for Evaluation Tasks. *Knowledge and Information Systems*, 29(2):349–378.

[Baumeister et al., 2011] Baumeister, J., Reutelshoefer, J., and Puppe, F. (2011). KnowWE: A Semantic Wiki for Knowledge Engineering. *Applied Intelligence*, 35(3):323–344.

[Baumeister and Striffler, 2013] Baumeister, J. and Striffler, A. (2013). Explanation in Episodic and Continuous Decision Support Systems. In *FGWM'13: Proceedings of German Workshop of Knowledge and Experience Management (at LWA'2013)*.

[Bäumer et al., 1996] Bäumer, D., Bischofberger, W. R., Lichter, H., and Züllighoven, H. (1996). User Interface Prototyping—Concepts, Tools, and Experience. In *ICSE '96: Proceedings of the 18th international conference on Software engineering*, pages 532–541.

[Beaudouin-Lafon and Mackay, 2003] Beaudouin-Lafon, M. and Mackay, W. (2003). Prototyping Tools and Techniques. In *The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications*, pages 1006–1031, Hillsdale, NJ, USA. L. Erlbaum Associates Inc.

[Beedle and Schwaber, 2002] Beedle, M. and Schwaber, K. (2002). *Agile Software Development with Scrum*. Prentice Hall. ISBN 13: 978-0130676344.

[Ben-Zvi, 2012] Ben-Zvi, T. (2012). Measuring the Perceived Effectiveness of Decision Support Systems and their Impact on Performance. *Decision Support Systems*, 54(1):248–256.

[Beraldi et al., 2011] Beraldi, P., Violi, A., and Simone, F. D. (2011). A Decision Support System for Strategic Asset Allocation. *Decision Support Systems*, 51(3):549–561.

[Bevan and Macleod, 1994] Bevan, N. and Macleod, M. (1994). Usability Measurement in Context. *Behaviour and Information Technology*, 13:132–145.

[Blanchard et al., 2007] Blanchard, J., Guillet, F., and Briand, H. (2007). Interactive Visual Exploration of Association Rules with Rule-focusing Methodology. *Knowledge and Information Systems*, 13(1):43–75.

[Bloom and Chung, 2001] Bloom, P. and Chung, Q. (2001). Lessons Learned from Developing a Mission-critical Expert System with Multiple Experts through Rapid Prototyping. *Expert Systems with Applications*, 20(2):217–227.

[Borchers, 2001] Borchers, J. (2001). *A Pattern Approach to Interaction Design*. John Wiley & Sons, Inc., New York, NY, USA. ISBN: 0471498289.

[Bouamrane et al., 2011] Bouamrane, M.-M., Rector, A., and Hurrell, M. (2011). Using OWL Ontologies for Adaptive Patient Information Modelling and Preoperative Clinical Decision Support. *Knowledge and Information Systems*, 29:405–418.

[Bourouis et al., 2014] Bourouis, A., Feham, M., Hossain, M., and Zhang, L. (2014). An Intelligent Mobile Based Decision Support System for Retinal Disease Diagnosis. *Decision Support Systems*, 59(0):341–350.

[Broekstra et al., 2002] Broekstra, J., Kampman, A., and van Harmelen, F. (2002). Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In *International Semantic Web Conference*, pages 54–68, Sardinia, Italy.

[Brooke, 1996] Brooke, J. (1996). SUS: A Quick and Dirty Usability scale. In Jordan, P. W., Weerdmeester, B., Thomas, A., and Mclelland, I. L., editors, *Usability evaluation in industry*. Taylor and Francis, London.

[Calabrese et al., 2012] Calabrese, F., Corallo, A., Margherita, A., and Zizzari, A. (2012). A Knowledge-based Decision Support System for Shipboard Damage Control. *Expert Systems with Applications*, 39(9):8204–8211.

[Card et al., 1999] Card, S. K., Mackinlay, J. D., and Shneiderman, B., editors (1999). *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[Castellanos et al., 2011] Castellanos, V., Albiter, A., Hernández, P., and Barrera, G. (2011). Failure Analysis Expert System for Onshore Pipelines. Part-II: End-User Interface and Algorithm. *Expert Systems with Applications*, 38(9):11091–11104.

[Cebi et al., 2009] Cebi, S., Celik, M., Kahraman, C., and Er, I. D. (2009). An Expert System towards Solving Ship Auxiliary Machinery Troubleshooting: SHIPAMTSOLVER. *Expert Systems with Applications*, 36(3, Part 2):7219–7227.

[Chen et al., 2012] Chen, Y., Hsu, C.-Y., Liu, L., and Yang, S. (2012). Constructing a Nutrition Diagnosis Expert System. *Expert Systems with Applications*, 39(2):2132–2156.

[Cho, 2010] Cho, V. (2010). MISMIS–A Comprehensive Decision Support System for Stock Market Investment. *Knowledge-Based Systems*, 23(6):626–633.

[Coca, 2013] Coca, A. N. (2013). Diagnose-Visualisierung in Wissenbasierten Systemen. Bachelor Thesis (University of Würzburg, Germany).

[Constantine and Lockwood, 1999] Constantine, L. L. and Lockwood, L. A. D. (1999). *Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design*. Addison-Wesley Professional.

[Cupello and Mishelevich, 1988] Cupello, J. M. and Mishelevich, D. J. (1988). Managing Prototype Knowledge/Expert System Projects. *Communications of the ACM*, 31(5):534–550.

[d3web, n.d.] d3web (n.d.). d3web Homepage. Online (accessed June 3rd, 2014). `http://www.d3web.de/`.

[Devraj and Jain, 2011] Devraj and Jain, R. (2011). PulsExpert: An Expert System for the Diagnosis and Control of Diseases in Pulse Crops. *Expert Systems with Applications*, 38(9):11463–11471.

[Dokas, 2005] Dokas, I. M. (2005). Developing Web sites for Web based Expert Systems: A Web Engineering Approach. In *Proceedings of the Second International ICSC Symposium on Information Technologies in Environmental Engineering (Magdeburg)*, pages 202–217. Shaker Verlag.

[Duan et al., 2005] Duan, Y., Edwards, J. S., and Xu, M. X. (2005). Web-based Expert Systems: Benefits and Challenges. *Information & Management*, 42:799–811.

[Eclipse, n.d.] Eclipse (n.d.). Eclipse Homepage. Online (accessed June 3rd, 2014). Eclipse Foundation. `http://www.eclipse.org/`.

[Erdem and Göçen, 2012] Erdem, A. S. and Göçen, E. (2012). Development of a Decision Support System for Supplier Evaluation and Order Allocation. *Expert Systems with Applications*, 39(5):4927–4937.

[Erickson, n.d.] Erickson, T. (n.d.). Interaction Design Patterns Page. Online (accessed June 3rd, 2014). `http://www.visi.com/~snowfall/InteractionPatterns.html`.

[EuraHS, n.d.] EuraHS (n.d.). EuraHS Homepage. Online (accessed June 3rd, 2014). `http://www.eurahs.eu`.

[Floyd, 1984] Floyd, C. (1984). A Systematic Look at Prototyping. In Budde, R., Kuhlenkamp, K., Mathiassen, L., and Zullighoven, H., editors, *Approaches to Prototyping*. Springer-Verlag New York, Inc.

[Freiberg et al., 2009] Freiberg, M., Baumeister, J., and Puppe, F. (2009). The Usability Stack: Reconsidering Usability Criteria regarding Knowledge-Based Systems. In *FGWM'09: Proceedings of German Workshop of Knowledge and Experience Management (at LWA'2009)*.

[Freiberg et al., 2010] Freiberg, M., Mitlmeier, J., Baumeister, J., and Puppe, F. (2010). Knowledge System Prototyping for Usability Engineering. In *FGWM'10: Proceedings of German Workshop of Knowledge and Experience Management (at LWA'2010)*.

[Freiberg and Puppe, 2012a] Freiberg, M. and Puppe, F. (2012a). iTree: Skill-building User-centered Clarification Consultation Interfaces. In *Proceedings of the International Conference on Knowledge Engineering and Ontology Development (KEOD 2012)*. SciTePress Digital Library.

[Freiberg and Puppe, 2012b] Freiberg, M. and Puppe, F. (2012b). Template-based Extensible Prototyping for Creativity- and Usability-Oriented Knowledge Systems Development. In Nalepa, G. J., Cañadas, J., and Baumeister, J., editors, *Proceedings of the 8th Workshop on Knowledge Engineering and Software Engineering (KESE8)*. CEUR Proceedings.

[Freiberg and Puppe, 2013] Freiberg, M. and Puppe, F. (2013). Prototyping-based Usability-oriented Knowledge Systems Engineering. In Reiterer, H. and Deussen, O., editors, *Mensch und Computer 2012—12. fachübergreifende Konferenz für interaktive und kooperative Medien*. Oldenbourg Verlag.

[Freiberg et al., 2012] Freiberg, M., Striffler, A., and Puppe, F. (2012). Extensible Prototyping for Pragmatic Engineering of Knowledge-based Systems. *Expert Systems with Applications*, 39(11):10177–10190.

[Gamma et al., 1994] Gamma, E., Helm, R., Johnson, R., and Vlissides, J. M. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional, 1 edition. ISBN 13: 978-0-2016-3361-0.

[GeNie & SMILE, n.d.] GeNie & SMILE (n.d.). GeNIe & SMILE Homepage. Online accessed March 18th, 2014). Decision Systems Laboratory. `http://genie.sis.pitt.edu`.

[Gennari et al., 2003] Gennari, J. H., Musen, M. A., Fergerson, R. W., Grosso, W. E., Crubézy, M., Eriksson, H., Noy, N. F., and Tu, S. W. (2003). The Evolution of Protégé: An Environment for Knowledge-based Systems Development. *International Journal of Human-Computer Studies*, 58(1):89–123.

[Ghandforoush and Sen, 2010] Ghandforoush, P. and Sen, T. K. (2010). A DSS to Manage Platelet Production Supply Chain for Regional Blood Centers. *Decision Support Systems*, 50(1):32–42.

[Gottlob et al., 1990] Gottlob, G., Frühwirth, T., and Horn, W., editors (1990). *Expertensysteme*. Springer-Verlag. ISBN: 3-211-82221-6.

[Graham, 2003] Graham, I. (2003). *A Pattern Language for Web Usability*. Pearson Education. ISBN 13: 9780201788884.

[Grahovac and Devedzic, 2010] Grahovac, D. and Devedzic, V. (2010). COMEX: A Cost Management Expert System. *Expert Systems with Applications*, 37(12):7684–7695.

[Grove, 2000] Grove, R. F. (2000). Design and Development of Knowledge-Based Systems on the Web. In *ISCA Conference on Intelligent Systems'00*, pages 147–150.

[Hakim and Spitzer, 2000] Hakim, J. and Spitzer, T. (2000). Effective Prototyping for Usability. In *Proceedings of IEEE professional communication society international professional communication conference and Proceedings of the 18th annual ACM international conference on Computer documentation: technology & teamwork*, IPCC/SIGDOC '00, pages 47–54, Piscataway, NJ, USA. IEEE Educational Activities Department.

[Hart, 2006] Hart, S. G. (2006). Nasa-Task Load Index (Nasa-TLX); 20 Years Later. In *Human Factors and Ergonomics Society Annual Meeting*, volume 50, pages 904–908.

[Hasan and Isaac, 2011] Hasan, S. and Isaac, R. (2011). An Integrated Approach of MAS-CommonKADS, Model-View-Controller and Web Application Optimization Strategies for Web-based Expert System Development. *Expert Systems with Applications*, 38(1):417–428.

[He, 2013] He, W. (2013). Improving User Experience with Case-based Reasoning Systems using Text Mining and Web 2.0. *Expert Systems with Applications*, 40(2):500–507.

[He et al., 2009] He, W., Wang, F.-K., Means, T., and Xu, L. D. (2009). Insight into Interface Design of Web-based Case-based Reasoning Retrieval Systems. *Expert Systems with Applications*, 36(3, Part 2):7280–7287.

[Hendler, 1988] Hendler, J. (1988). *Expert Systems: The User Interface*. Human/Computer Interaction. Ablex Publishing, Norwood, NJ. ISBN 13: 978-0893914295.

[Holzinger et al., 2011] Holzinger, A., Kosec, P., Schwantzer, G., Debevc, M., Hofmann-Wellenhof, R., and Frühauf, J. (2011). Design and Development of a Mobile Computer Application to Reengineer Workflows in the Hospital and the Methodology to Evaluate its Effectiveness. *Journal of Biomedical Informatics*, 44(6):968–977.

[Holzinger and Slany, 2006] Holzinger, A. and Slany, W. (2006). XP+UE->XU Praktische Erfahrungen mit eXtreme Usability. *Informatik Spektrum*, 29(2):91–97.

[Huang et al., 2009] Huang, J.-M., Jou, Y.-T., Zhang, L.-C., Wang, S.-T., and Huang, C.-X. (2009). A web-based model for developing: A mold base design system. *Expert Systems with Applications*, 36(4):8356–8367.

[Huettig et al., 2004] Huettig, M., Buscher, G., Menzel, T., Scheppach, W., Puppe, F., and Buscher, H.-P. (2004). A Diagnostic Expert System for Structured Reports, Quality Assessment, and Training of Residents in Sonography. *Medizinische Klinik*, 99(3):117–122.

[İÇ and Yurdakul, 2009] İÇ, Y. T. and Yurdakul, M. (2009). Development of a Decision Support System for Machining Center Selection. *Expert Systems with Applications*, 36(2):3505–3513.

[İÇ and Yurdakul, 2010] İÇ, Y. T. and Yurdakul, M. (2010). Development of a Quick Credibility Scoring Decision Support System using Fuzzy TOPSIS. *Expert Systems with Applications*, 37(1):567–574.

[ISO 9241–110, 2006] ISO 9241–110 (2006). Ergonomie der Mensch-System-Interaktion - Teil 110: Grundsätze der Dialoggestaltung; DeutscheFassung EN ISO 9241-110:2006. International Organization for Standardization (ISO); `http://www.iso.org/`.

[Jackson, 1998] Jackson, P. (1998). *Introduction to Expert Systems*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 3rd edition. ISBN 13: 978-0201876864.

[Java Oracle, n.d.] Java Oracle (n.d.). Java Software Download. Online (accessed June 4th, 2014). `http://www.java.com/de/download/manual.jsp`.

[Jetstrap (Drifty Co.), n.d.] Jetstrap (Drifty Co.) (n.d.). Jetstrap Software. Online (accessed May 23rd, 2014). `https://jetstrap.com`.

[jQuery Foundation, n.d.] jQuery Foundation (n.d.). jQuery Homepage. Online (accessed June 4th, 2014). `http://jquery.com`.

[JSON, n.d.] JSON (n.d.). Introducing JSON. Online (accessed June 6th, 2014). `http://www.json.org/`.

[Kajiyama and Satoh, 2014] Kajiyama, T. and Satoh, S. (2014). An Interaction Model between Human and System for Intuitive Graphical Search Interface. *Knowledge and Information Systems*, 39(1):41–60.

[Keleş et al., 2011] Keleş, A., Keleş, A., and Yavuz, U. (2011). Expert System based on Neuro-Fuzzy Rules for Diagnosis Breast Cancer. *Expert Systems with Applications*, 38(5):5719–5726.

[Kieffer et al., 2010] Kieffer, S., Coyette, A., and Vanderdonckt, J. (2010). User Interface Design by Sketching: A Complexity Analysis of Widget Representations. In *Proceedings of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, EICS '10, pages 57–66, New York, NY, USA. ACM.

[Kim et al., 2009] Kim, Y.-H., Song, J.-H., and Park, J.-H. (2009). An Expert System for Fatigue Life Prediction under Variable Loading. *Expert Systems with Applications*, 36(3, Part 1):4996–5008.

[Kiriş et al., 2010] Kiriş, S., Yüzügüllü, N., Ergün, N., and Çevik, A. A. (2010). A Knowledge-based Scheduling System for Emergency Departments. *Knowledge-Based Systems*, 23(8):890–900.

[Knublauch, 2002] Knublauch, H. (2002). Extreme Programming of Knowledge-Based Systems. In *Proceedings of eXtreme Programming and Agile Processes in Software Engineering (XP2002)*.

[Kolhe et al., 2011] Kolhe, S., Kamal, R., Saini, H. S., and Gupta, G. (2011). An intelligent Multimedia Interface for Fuzzy-logic Based Inference in Crops. *Expert Systems with Applications*, 38(12):14592–14601.

[Krawczyk, 2007] Krawczyk, A. (2007). Konfigurierbarer Dialog zur Nutzung von wissensbasierten Systemen. Master's thesis, University of Würzburg, Germany.

[Kurbel, 1989] Kurbel, K. (1989). *Entwicklung und Einsatz von Expertensystemen*. Springer-Verlag. ISBN: 3-540-51013-3.

[Laasko, n.d.] Laasko, S. A. (n.d.). User Interface Design Patterns. Online (accessed March 2nd, 2014). `http://www.cs.helsinki.fi/u/salaakso/patterns/`.

[Leichtenstern and André, 2010] Leichtenstern, K. and André, E. (2010). MoPeDT: Features and Evaluation of a User-centred Prototyping Tool. In *Proceedings of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, EICS '10, pages 93–102, New York, NY, USA. ACM.

[Li et al., 2009] Li, N., Wang, R., Zhang, J., Fu, Z., and Zhang, X. (2009). Developing a Knowledge-based Early Warning System for Fish Disease/Health via Water Quality Management. *Expert Systems with Applications*, 36(3, Part 2):6500–6511.

[Lichter et al., 1993] Lichter, H., Schneider-Hufschmidt, M., and Züllighoven, H. (1993). Prototyping in Industrial Software Projects—Bridging the Gap between Theory and Practice. In *ICSE '93: Proceedings of the 15th international conference on Software Engineering*, pages 221–229.

[Lidwell et al., 2010] Lidwell, W., Holden, K., and Butler, J. (2010). *Universal Principles of Design*. Rockport Publishers Inc., 2nd edition. ISBN 13: 978-1-59253-587-3.

[Lin and Landay, 2008] Lin, J. and Landay, J. A. (2008). Employing Patterns and Layers for Early-stage Design and Prototyping of Cross-device User Interfaces. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human Factors in computing systems*, pages 1313–1322.

[Ma'aruf and Garba, 2012] Ma'aruf, L. M. and Garba, M. (2012). Design and Implementation of an Expert Diet Prescription System. *International Journal of Artificial Intelligence and Expert Systems*.

[Mao and Benbasat, 2000] Mao, J.-Y. and Benbasat, I. (2000). The Use of Explanations in Knowledge-Based Systems: Cognitive Perspectives and a Process-Tracing Analysis. *Journal of Management Information Systems*, 17(2):153–179.

[Mayhew, 1999] Mayhew, D. (1999). *The Usability Engineering Lifecycle*. Morgan Kaufmann Publishers. ISBN-13: 978-1558605619.

[McBride, 2002] McBride, B. (2002). Jena: A Semantic Web Toolkit. *IEEE Internet Computing*, 6(6):55–59.

[McCurdy et al., 2006] McCurdy, M., Connors, C., Pyrzak, G., Kanefsky, B., and Vera, A. (2006). Breaking the Fidelity Barrier: An Examination of our Current Characterization of Prototypes and an Example of a Mixed-fidelity Success. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1233–1242.

[McGee-Lennon et al., 2009] McGee-Lennon, M. R., Ramsay, A., McGookin, D., and Gray, P. (2009). User Evaluation of OIDE: A Rapid Prototyping Platform for Multimodal Interaction. In *Proceedings of the 1st ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, EICS '09, pages 237–242, New York, NY, USA. ACM.

[McGraw, 1992] McGraw, K. L. (1992). *Designing and Evaluating User Interfaces for Knowledge-based Systems*. Ellis Horwood. ISBN 13: 978-0139326745.

[McSherry, 2005] McSherry, D. (2005). Explanation in Recommender Systems. *Artificial Intelligence Review*, 24(2):179–197.

[Mediastinitis DGTHG, n.d.] Mediastinitis DGTHG (n.d.). Homepage Deutsche Gesellschaft für Thorax-, Herz-, und Gefäßchirurgie. Online (accessed June 6th, 2014). `http://www.dgthg.de/Register` → Mediastinitis Register.

[Mersmann and Dojat, 2004] Mersmann, S. and Dojat, M. (2004). SmartCare^{tm} - Automated Clinical Guidelines in Critical Care. In *ECAI'04/PAIS'04: Proceedings of the 16th European Conference on Artificial Intelligence, including Prestigious Applications of Intelligent Systems*, pages 745–749, Valencia, Spain. IOS Press.

[Milne and Nicol, 2000] Milne, R. and Nicol, C. (2000). Tiger: Continuous Diagnosis of Gas Turbines. In *ECAI 2000, Proceedings of the 14th European Conference on Artificial Intelligence*, pages 711–715.

[Mitlmeier, 2010] Mitlmeier, J. F. (2010). Ein Prototyping-Framework zur Entwicklung von wissensbasierten Dialogsystemen. Bachelor Thesis.

[Murphy, 2001] Murphy, K. P. (2001). The Bayes Net Toolbox for MATLAB. *Computing Science and Statistics*, 33:2001.

[Netbeans Oracle, n.d.] Netbeans Oracle (n.d.). Netbeans IDE Homepage. Online (accessed June 6th, 2014). `http://netbeans.org/`.

[Neumann et al., 2002] Neumann, M., Baumeister, J., Liess, M., and Schulz, R. (2002). An Expert System to Estimate the Pesticide Contamination of Small Streams using Benthic Macroinvertebrates as Bioindicators: II. The Knowledge Base of LIMPACT. *Ecological Indicators*, 2(3):239–249.

[Nghia and Puppe, 2009] Nghia, D. D. and Puppe, F. (2009). Hybrides, skalierbares Diagnosesystem für freie Kfz-Werkstätten [Hybrid and scalable diagnosis system for car garages]. *KI: German AI magazine*, 23(2):31–37.

[Nielsen, 1993a] Nielsen, J. (1993a). Iterative User Interface Design. *IEEE Computer*, 26(11):32–41.

[Nielsen, 1993b] Nielsen, J. (1993b). *Usability Engineering*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. ISBN 13: 978-0125184069.

[Nielsen, 1994] Nielsen, J. (1994). Heuristic Evaluation. In Nielsen, J. and Mack, R. L., editors, *Usability Inspection Methods*, pages 25–62. John Wiley & Sons, New York. ISBN 13: 978-0471018773.

[Noy et al., 2000] Noy, N. F., Grosso, W. E., and Musen, M. A. (2000). Knowledge-Acquisition Interfaces for Domain Experts An Empirical Evaluation of Protege-2000. In *Proceedings of the 12th Internal Conference on Software and Knowledge Engineering. Chicago, USA, July, 5-7, 2000*, pages 5–7.

[Nurminen et al., 2003] Nurminen, J. K., Karonen, O., and Hätönen, K. (2003). What Makes Expert Systems Survive over 10 Years - Empirical Evaluation of Several Engineering Applications. *Expert Systems with Applications*, 24(2):199–211.

[O'Leary Daniel E. , 1991] O'Leary Daniel E. (1991). Design, development and validation of expert systems: A survey of developers. In Ayel, M. and Laurent, J.-P., editors, *Validation, Verification and Test of Knowledge-based Systems*, pages 3–19. John Wiley & Sons, Inc.

[OmniGraffle The Omni Group, n.d.] OmniGraffle The Omni Group (n.d.). OmniGraffle Homepage. Online (accessed June 6th, 2014). `http://www.omnigroup.com/omnigraffle/`.

[Papić et al., 2009] Papić, V., Rogulj, N., and Pleština, V. (2009). Identification of Sport Talents using a Web-oriented Expert System with a Fuzzy Module. *Expert Systems with Applications*, 36(5):8830–8838.

[Plant and Gamble, 2003] Plant, R. and Gamble, R. (2003). Methodologies for the Development of Knowledge-Based Systems, 1982-2002. *The Knowledge Engineering Review*, 18:47–81.

[ProKEt, n.d.] ProKEt (n.d.). ProKEt Project Homepage. Online (accessed June 6th, 2014). `https://sourceforge.net/projects/proket/?source=directory`.

[Protégé Stanford, n.d.] Protégé Stanford (n.d.). Protégé Home, Stanford University. Online (accessed June 6th, 2014). `http://protege.stanford.edu`.

[Pu and Chen, 2006] Pu, P. and Chen, L. (2006). Trust Building with Explanation Interfaces. In *IUI '06: Proceedings of the 11th international conference on Intelligent user interfaces*, pages 93–100.

[Puppe, 1988] Puppe, F. (1988). *Einführung in Expertensysteme*. Springer-Verlag. ISBN: 3-540-19481-9.

[Puppe, 1993] Puppe, F. (1993). *Systematic Introduction to Expert Systems*. Springer-Verlag. ISBN: 3-540-56255-9.

[Puppe, 2000] Puppe, F. (2000). Knowledge Formalization Patterns. In *Proceedings of PKAW 2000, Sydney Australia*.

[Puppe et al., 2000] Puppe, F., Ziegler, S., Martin, U., and Hupp, J. (2000). *Wissensbasierte Diagnosesysteme im Service-Support [Knowledge-Based Systems for Service-Support]*. Springer. ISBN-13: 978-3540672883.

[Rahimi et al., 2007] Rahimi, S., Gandy, L., and Mogharreban, N. (2007). A Web-based High-performance Multicriteria Decision Support System for Medical Diagnosis: Research Articles. *International Journal of Intelligent Systems*, 22:1083–1099.

[Recio-García et al., 2008] Recio-García, J. A., Díaz-Agudo, B., Gómez-Berbis, J. M., and González-Calero, P. A. (2008). The WINGS of jCOLIBRI: A CBR Architecture for Semantic Web Services. *International Journal of Web Services Practices (IJWSP). Selected papers from the International Conference on Next Generation Web Services Practices Conference (NWeSP'07)*, 3(1):35–40.

[Rettig, 1994] Rettig, M. (1994). Prototyping for Tiny Fingers. *Communications of the ACM*, 37:21–27.

[Richards, 2003] Richards, D. (2003). Knowledge-Based System Explanation: The Ripple-Down Rules Alternative. *Knowledge and Information Systems*, 5(1):2–25.

[Rodríguez et al., 2010] Rodríguez, T. J., Vitoriano, B., and Montero, J. (2010). A Natural-disaster Management DSS for Humanitarian Non-Governmental Organisations. *Knowledge-Based Systems*, 23(1):17–22.

[Rogers, 2011] Rogers, Y. (2011). *Interaction Design—Beyond Human-Computer Interaction*. John Wiley & Sons, Ltd, 3rd edition. ISBN 13: 978-0470665763.

[Roth-Berghofer et al., 2012] Roth-Berghofer, T., Garcia, J. A. R., Sauer, C. S., Bach, K., Althoff, K.-D., Diaz-Agudo, B., and Calero, P. A. G. (2012). Building Case-based Reasoning Applications with myCBR and COLIBRI Studio. In Petridis, M., Roth-Berghofer, T., and Wiratunga, N., editors, *Proceedings of the 17th UK Workshop on Case-Based Reasoning (UKCBR)*, pages 71–82. School of Computing, Engineering and Mathematics, University of Brighton, UK.

[Roth-Berghofer, 2004] Roth-Berghofer, T. R. (2004). Explanations and Case-Based Reasoning: Foundational Issues. In *Advances in Case-Based Reasoning*, pages 389–403. Springer-Verlag.

[Ruckert and Klein, 1996] Ruckert, C. and Klein, S. (1996). Optimizing the User-Interface of a Knowledge-based System. In *Proceedings of the Seventh International Workshop on Database and Expert Systems Applications, 1996.*, pages 166–171.

[Ruiz-Mezcua et al., 2011] Ruiz-Mezcua, B., Garcia-Crespo, A., Lopez-Cuadrado, J. L., and Gonzalez-Carrasco, I. (2011). An Expert System Development Tool for Non AI Experts. *Expert Systems with Applications*, 38:597–609.

[Russel and Norvig, 2010] Russel, S. J. and Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Pearson. ISBN 13: 978-0136042594.

[Saadé et al., 2004] Saadé, R. G., Tsoukas, A., and Tsoukas, G. (2004). Prototyping a Decision Support System in the Clinical Environment: Assessment of Patients with Osteoporosis OSTEODSS. *Expert Systems with Applications*, 27(3):427–438.

[Sánchez-Pi et al., 2012] Sánchez-Pi, N., Carbó, J., and Molina, J. M. (2012). A Knowledge-based System Approach for a Context-aware System. *Knowledge-Based Systems*, 27(0):1–17.

[Santos et al., 2011] Santos, L., Coutinho-Rodrigues, J., and Antunes, C. H. (2011). A Web Spatial Decision Support System for Vehicle Routing using Google Maps. *Decision Support Systems*, 51(1):1–9.

[Sarma et al., 2010] Sarma, S. K., Singh, K. R., and Singh, A. (2010). An Expert System for Diagnosis of Diseases in Rice Plant. *International Journal of Artificial Intelligence and Expert Systems*, 1(2):26–31.

[Schmettow, 2006] Schmettow, M. (2006). User Interaction Design Patterns for Information Retrieval Systems. In Hvatum, L. and Zdun, U., editors, *Proceedings of the 11th European Conference on Pattern Languages of Programs, EuroPLoP 06*, pages 489–511.

[Schreiber et al., 2000] Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., de Velde, W. V., and Wielinga, B. (2000). *Knowledge Engineering and Management - The CommonKADS Methodology*. MIT Press, 2 edition. ISBN 13: 978-0262193009.

[Sebastia et al., 2009] Sebastia, L., Garcia, I., Onaindia, E., and Guzman, C. (2009). e-TOURISM: A Tourist Recommendation and Planning Application. *International Journal on Artificial Intelligence Tools*, 18(05):717–738.

[Shneiderman, 1998] Shneiderman, B. (1998). Treemaps for Space-constrained Visualization of Hierarchies. *ACM Transactions on Graphics (TOG) Volume*, 11:92–99.

[Shneiderman and Plaisant, 2004] Shneiderman, B. and Plaisant, C. (2004). *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison Wesley. ISBN 13: 978-0321197863.

[Shue et al., 2009] Shue, L.-Y., Chen, C.-W., and Shiue, W. (2009). The Development of an Ontology-based Expert System for Corporate Financial Rating. *Expert Systems with Applications*, 36(2, Part 1):2130–2142.

[Song et al., 2008] Song, G., He, Y., Chu, F., and Gu, Y. (2008). HYDES: A Web-based Hydro Turbine Fault Diagnosis System. *Expert Systems with Applications*, 34(1):764–772.

[Sørmo et al., 2005] Sørmo, F., Cassens, J., and Aamodt, A. (2005). Explanation in Case-Based Reasoning—Perspectives and Goals. *Artif. Intell. Rev.*, 24(2):109–143.

[Southwick, 1991] Southwick, R. W. (1991). Explaining Reasoning: An Overview of Explanation in Knowledge-based Systems. *The Knowledge Engineering Review*, 6:1–19.

[StringTemplate, n.d.] StringTemplate (n.d.). StringTemplate Homepage (Terrence Parr). Online (accessed June 6th, 2014). `http://www.stringtemplate.org/`.

[Subasic and Berendt, 2010] Subasic, I. and Berendt, B. (2010). Discovery of Interactive Graphs for Understanding and Searching Time-indexed Corpora. *Knowledge and Information Systems*, 23(3):293–319.

[Sutton and Patkar, 2009] Sutton, D. and Patkar, V. (2009). CommonKADS Analysis and Description of a Knowledge based System for the Assessment of Breast Cancer. *Expert Systems with Applications*, 36(2, Part 1):2411–2423.

[Tidwell, 2005] Tidwell, J. (2005). *Designing Interfaces — Patterns for Effective Interaction Design*. O'Reilly Media Inc. ISBN 13: 978-0596008031.

[Ting et al., 2010] Ting, S., Kwok, S., Tsang, A., and Lee, W. (2010). CASESIAN: A Knowledge-based System Using Statistical and Experiential Perspectives for Improving the Knowledge Sharing in the Medical Prescription Process. *Expert Systems with Applications*, 37(7):5336–5346.

[Ting et al., 2011] Ting, S., Kwok, S., Tsang, A. H., and Lee, W. (2011). A Hybrid Knowledge-based Approach to Supporting the Medical Prescription for General Practitioners: Real Case in a Hong Kong Medical Center. *Knowledge-Based Systems*, 24(3):444–456.

[Tomcat, n.d.] Tomcat (n.d.). Tomcat Homepage (The Apache Software Foundation). Online (accessed June 6th, 2014). `http://tomcat.apache.org/`.

[Tomic et al., 2012] Tomic, B., Horvat, B., and Jovanovic, N. (2012). An Explanation Facility Framework for Rule-Based Systems. *International Journal on Artificial Intelligence Tools*, 21(4).

[Tomic et al., 2006] Tomic, B., Jovanovic, J., and Devedzic, V. (2006). JavaDON: An Open-source Expert System Shell. *Expert Systems with Applications*, 31(3):595–606.

[TreeMap Shneiderman, n.d.] TreeMap Shneiderman (n.d.). Treemaps for Space-constrained Visualization of Hierarchies. Online. `http://www.cs.umd.edu/hcil/treemap-history/`; accessed June 7th, 2014.

[Tufte, 1986] Tufte, E. R. (1986). *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, CT, USA. ISBN: 0-9613921-0-X.

[UCD def. by UPA, n.d.] UCD def. by UPA (n.d.). Resources: About Usability—What is User-Centered Design? Online (accessed June 6th, 2014). `http://www.usabilityprofessionals.org/usability_resources/about_usability/what_is_ucd.html`.

[UPA, n.d.] UPA (n.d.). Usability Professionals' Association (UPA) Homepage. Online (accessed June 6th, 2014). `http://www.usabilityprofessionals.org/`; accessed June 6th, 2014.

[van Welie, n.d.] van Welie, M. (n.d.). Welie.com—Patterns in Interaction Design. Online (accessed Mar. 2nd, 2014). `http://www.welie.com/patterns/`.

[W3C CSS, n.d.] W3C CSS (n.d.). Cascading Style Sheets. Online (accessed July 12th, 2012). `http://www.w3.org/Style/CSS/`.

[W3Schools AJAX, n.d.] W3Schools AJAX (n.d.). Ajax tutorial. Online (accessed March 16th, 2013). `http://www.w3schools.com/ajax/default.asp`.

[W3Schools JavaScript, n.d.] W3Schools JavaScript (n.d.). Java script reference. Online (accessed June 4th, 2014). `http://www.w3schools.com/jsref/default.asp`.

[Wan and Lei, 2009] Wan, S. and Lei, T. C. (2009). A Knowledge-based Decision Support System to Analyze the Debris-flow Problems at Chen-Yu-Lan River, Taiwan. *Knowledge-Based Systems*, 22(8):580–588.

[Waterman, 1993] Waterman, D. A. (1993). *A Guide to Expert Systems*. Addison-Wesley. ISBN: 0-201-08313-2.

[Wharton et al., 1994] Wharton, C., Rieman, J., Lewis, C., and Polson, P. (1994). The Cognitive Walkthrough Method: A Practitioner's Guide. In Nielsen, J. and Mack, R. L., editors, *Usability Inspection Methods*, pages 105–140. John Wiley & Sons, Inc., New York, NY, USA.

[Whitenton, 2013] Whitenton, K. (2013). Minimize Cognitive Load to Maximize Usability. Online (accessed May 24th, 2014). `http://www.nngroup.com/articles/minimize-cognitive-load/`.

[Xiaoshuan et al., 2009] Xiaoshuan, Z., Zetian, F., Wengui, C., Dong, T., and Jian, Z. (2009). Applying Evolutionary Prototyping Model in Developing FIDSS: An Intelligent Decision Support System for Fish Disease/Health Management. *Expert Systems with Applications*, 36(2, Part 2):3901–3913.

[Zacharias, 2007] Zacharias, V. (2007). Visualization of Rule Bases– The Overall Structure. In *I-KNOW '07: Proceedings of International Conference on Knowledge Management*.

[Zeng et al., 2012] Zeng, Y., Cai, Y., Jia, P., and Jee, H. (2012). Development of a Web-based Decision Support System for Supporting Integrated Water Resources Management in Daegu City, South Korea. *Expert Systems with Applications*, 39(11):10091–10102.

Part IV

Appendix

Appendix A

Knowledge Bases in this Work

In the following, we subsume the core knowledge bases (KBs) mentioned in this thesis. They were implemented with the [d3web, n.d.] toolkit. The knowledge was formalized either with the semantic wiki KnowWE [Baumeister, 2004] or the standard office-based knowledge acquisition (KA) environment KnowOF; see Section 6.2.3.a. of this work for the latter. In the following, for each KB we provide a short summary (target domain, main objective), the realization (knowledge specifics, KA process), and the usage scenarios.

A.1 Statistical Calculation Recommender

- **Summary:** The statistical calculation recommender KB (in german) provides support in selecting an appropriate statistical calculation method for given contextual requirements. Thereby, the user enters symptoms that characterize (or exclude) specific statistical methods, e.g., regarding the number of dependent and independent variables.
- **Realization:** The KB was developed with the semantic wiki KnowWE. It implements strict (heuristic) decision tree knowledge and contains 22 questions and 29 solutions. Based on the symptoms, exactly one (or none) solution is derived.
- **Usage:** This KB was used with reference implementations of the Strict Interview (e.g., Figure 7.11, p. 144) and the Hierarchical Interview (e.g., Figure 7.12, p. 145) pattern during several user studies.

A.2 House Plants Recommendation

- **Summary:** The house plants recommender KB (in german) provides advice for choosing appropriate house plants for given framing conditions. The user enters symptoms related to the plants' characteristics, such as appearance, target location, care intensity, etc.
- **Realization:** The KB was developed with the KA environment KnowOF, based on a spreadsheet formalization of the knowledge. It implements rule-based, forward consultation knowledge. It contains 17 questions and 35 solutions. One or several solutions can be derived by the system simultaneously.
- **Usage:** his KB was used with reference implementations of the Box Questionnaire (e.g., Figure 7.9, p. 142) in one of the user studies and for informal tests with the Daily Questionnaire reference implementation of ProKEt.

A.3 Pub Recommendation for the city of Würzburg

- **Summary:** The pub recommender KB for the city of Würzburg (in german) provides decision support regarding which pub/restaurant/location in Würzburg to visit regarding the given framing conditions. Those conditions—such as the type of the location, availability of parking lots, average prices, the selection range of drinks and meals, etc.—are entered as symptoms.
- **Realization:** The KB was developed with the KA environment KnowOF, based on a spreadsheet formalization of the knowledge. It implements rule-based, forward consultation knowledge. It contains 26 questions and 32 solutions. One or several solutions can be derived by the system simultaneously.
- **Usage:** This KB was used with reference implementations of the Daily Questionnaire (e.g., Figure 7.10, p. 143) in the user studies and for informal tests with the Box Questionnaire reference implementation of ProKEt.

A.4 Unlawful Dismissal

- **Summary:** The clarification KB on unlawful dismissal (in german) derives, whether a dismissal was legally correct in a given context. This context is described by the entered symptoms—for example, whether the formal regulations regarding a dismissal are adhered to. Therefore, questions are recursively abstracted/refined into more fine-granular questions. This enables users to either quickly attain a result by answering abstract (summarizing) top-level questions, or by answering the refined questions, which requires a longer session.
- **Realization:** The KB was developed with the KA environment KnowOF, based on a word processor document that contains the knowledge formalization. The KB targets a single solution only—whether or not the dismissal was legally correct—and consists of about 80 questions. Thereby, 5 top-level abstract/summarizing questions derive the solution directly and are further refined on up to 10 abstraction levels.
- **Usage:** This KB was used with the reference implementation of the ITree instantiation of the Hierarchical Clarifier pattern (e.g., Figure 7.13, p. 146) in the user studies as well as in the JuriSearch project (see Section 7.1.4).

A.5 Hit & Run Accident

- **Summary:** The Hit & Run Accident clarification KB (in german) investigates, whether the legal case of a hit & run accident is given. The context therefore is described by the entered symptoms, e.g., whether the accident has taken place in the public, or whether the required period of time for waiting has been adhered to. Questions are recursively abstracted/refined into more fine-granular questions. This enables users to either quickly attain a result by answering abstract (summarizing) top-level questions, or by answering the refined questions, which requires a longer session.

- **Realization:** The KB was developed with the KA environment KnowOF, based on a word-processing document that contains the knowledge formalization. The KB targets a single solution only and consists of about 14 questions. Thereby, 2 top-level abstract/ summarizing questions derive the solution directly and are further refined on up to 3 abstraction levels.
- **Usage:** This KB was mainly used with the reference implementation of the ITree instantiation of the Hierarchical Clarifier pattern (e.g., Figure 7.13, p. 146) in the KBS UI pattern assessment.

A.6 Lodging Deficiencies Clarification

Summary The lodging deficiencies KB (in german) clarifies, whether some potential given deficiencies in a flat allow for legal claims against the landlord. The context is described by entering symptoms, e.g., whether a valid tenancy agreement exists, and whether deficiencies such as a blackout of the heaters exist. Questions are recursively abstracted/refined into more fine-granular questions; this enables users to either quickly attain a result by answering abstract (summarizing) top-level questions, or by answering the refined questions, requiring a longer session.

Realization The KB was developed with the KA environment KnowOF, based on a word-processing document that contains the knowledge formalization. The KB targets a single solution only and consists of about 90 questions in total; thereby, 18 top-level abstract/summarizing questions derive the solution directly and are further refined on up to 5 abstraction levels.

Usage This KB mainly was used with the reference implementation of the ITree instantiation of the Hierarchical Clarifier pattern (e.g., Figure 7.13, p. 146) in the KBS UI pattern assessment.

Appendix B

Literature Review—KBSE Today

Regarding the engineering of knowledge-based systems (KBSE), the impression manifested that the KBS front-end most often is implemented rather ad hoc—thus conforming to our own, former experiences. The systematic investigation of UI design alternatives, experimentation, and evaluations targeted towards usability, or the formulation and application of best practices and patterns still seemed absent in that context—at least, not much seems to get published. To confirm (or disprove) that assumption, and to gain an impression of the most current state of the art on that topic in general, an encompassing literature review was performed. Therefore, we investigated relevant journals from the domains of KBS/ES engineering and AI. We searched journal issues—from January 2009 to April 2014—for publications that contain one or more of the following keywords in their title, abstract, or that seemed basically related to the topic/keyword: *Prototyping* ; *GUI* ; *User Interface* ; *Interface Design* ; *Knowledge System* ; *Knowledge-based System* ; *Diagnosis System* ; *Expert System* ; *Decision Support System*. Table B.1 lists the reviewed journals, the actually available issues and thus investigated years, the respectively used url/method how to access the journal, as well as the number of papers that were relevant in our specified context.

The resulting findings of the literature review are summarized in Table B.3. In the table, all articles are listed that matched our investigation objective—in particular that means, that papers which deal exclusively with the KBS back-end (algorithm, KB, etc.) are *not* included in that table. Thereby, the categories denoted in the table headings imply the meaning and potential values as presented in Table B.2. **The main insights encompass:**

- The major research efforts still concern the KBS back-end, i.e., reasoning algorithms, models, knowledge formalization, etc.
- An increasing number of KBS is developed for the internet or browser-based usage
- The KBS UI is considered still sparsely (or at least, not much is published). UI-solutions are not generalized (e.g., patterns), thus reported solutions are hard to reuse. Tailored KBS UI prototyping/construction tools are not available.
- UI solutions mostly apply form-based presentation/interaction variants. Less often used are conversational solutions, and least often realized are own forms, such as visual, CAD, wiki, geospatial, or mashup types.
- Where prototyping activities are reported, those mostly concern the realization of a pilot system. Iterative or evolutionary prototyping efforts are rarely reported.
- Usability techniques are applied still too sparsely. Where applied, those encompass mostly some form of questionnaire, less often user studies.

Journal	Years	Access	#
Expert Systems with Applications (ESWA)	2014–2009	`http://www.sciencedirect.com/science/journal/09574174`	20
Knowledge-based Systems (KBS)	2014–2009	`http://www.sciencedirect.com/science/journal/09507051`	7
International Journal of Knowledge-Based and Intelligent Engineering Systems (KES)	2013–2009	Via DPLP: `http://dblp.kbs.uni-hannover.de/dblp/Search.action?search=&q=in%3A%22KES+Journal%22&page=2&appliedFilters=source_facet%7CDBLP`	/
International Journal of Artificial Intelligence Tools (IJAIT)	2014–2009	`http://www.worldscientific.com/worldscinet/ijait`	/
International Journal of Software Engineering and Knowledge Engineering (IJSEKE)	2013–2009	`http://www.worldscientific.com/worldscinet/ijseke`	/
Knowledge and Information Systems (KAIS)	2013–2009	`http://link.springer.com/journal/volumesAndIssues/10115`	1
International Journal of Artificial Intelligence and Expert Systems (IJAE)	2014–2010	OPEN ACCESS, `http://www.cscjournals.org/csc/journals/IJAE/archive.php?JCode=IJAE`	2
International Journal of Artificial Intelligence and Applications (IJAIA)	2012–2010	OPEN ACCESS, `http://airccse.org/journal/ijaia/current2012.html`	/
Decision Support Systems (DSS)	2014–2009	`http://www.sciencedirect.com/science/journal/01679236`	3
International Journal of Expert Systems	N/A current years, only up to 1994		/

Table B.1: Listing of the researched journals (**Journal**) with publication years (**Years**), the URL or access method (**Access**), and the number of relevant publications (**#P**).

One problem with this literature review was the quite ambiguous usage of the terms knowledge-based-, expert-, and decision-support system and thus the identification of suitable publications. Another problem was the fact, that relevant works have been reported not only in the mentioned domains of KBS engineering and AI, but also in more specialized domains as, e.g., agriculture or medicine. Thus, also journals such as *Computers and Electronics in Agriculture* or *Computers in Biology and Medicine* contained potentially matching publications. However, we deliberately focussed the review on the mentioned core AI/KBS journals as to investigate specifically the KBSE methods and practices in the core community. Further, we sensed that if there exist any patterns/best practices/methods of KBS UI development and evaluation those should (and probably would) be reported in the topically most closely related journals in the first place (or at least, also there).

Head	Meaning/Values
Ref	***Literature reference of the publication***
Cont	***Application context/domain of the KBS:*** Industry, environment, medicine, etc.
Type	***KBS type:*** **Web**—Browser-based; **StA**—Desktop; **Emb**—Embedded; **Mob**—Mobile.
Pro	***Prototyping variant:*** **ThA**—Throw-away; **Evo**—Evolutionary; **PI**—'Protot. implemented', e.g., pilot.
UI	***Basic KBS UI type***: **Form**—Form-based UI, corresponding to our Questionnaire; **Interv**—conversational UI, corresponding to our Interview; **Own**—tailored non-standard UI, e.g., CAD, Interactive visual UI..., **Integ**—Input from external sources, **Touch**—Touch UI for mobile devices.
Usab	***Whether and which usability techniques were applied:*** **Ques**—Questionnaire; **Iv**—Interview; **Ex**—Formal Experiment; **FF**—Feedback Form; **Test**—(Informal) user test, **(b)**—Before development, **(a)**—After development, **i**—intermediate, **Met**—Metaphor-based UI; **FSt**—Field Study; **TA**—Thinking Aloud.

Table B.2: KBS research classifiers, used in Table B.3, column header and meaning.

Reference	Context	Type	Pro	UI	Usab
2012					
[Calabrese et al., 2012]	Military: Shipboard damage control	Emb	N/A	Integ	N/A
[Chen et al., 2012]	Medicine: Nutrition diagnosis	Web	No	Form	N/A
[Erdem and Göçen, 2012]	Industry: Supplier evaluation & order allocation	StA	N/A	Form	Test (a)
[Ma'aruf and Garba, 2012]	Medicine: Diet prescription	Web	N/A	N/A	N/A
[Sánchez-Pi et al., 2012]	Individual: Context-aware service advice	Mob	N/A	Touch	N/A
[Zeng et al., 2012]	Environment: Water resources management	Web	No	Form, GIS	N/A
2011					
[Afacan and Demirkan, 2011]	Design: Universal design support tool	StA	PI	Form, Own	Test(a), Quest (a), TA(a)
[Başçiftçi and İncekar, 2011]	Medicine: Heart disease diagnosis	Web	N/A	N/A	N/A
[Beraldi et al., 2011]	Finance: Asset allocation decision support	Web	PI	Form	N/A
[Bouamrane et al., 2011]	Medicine: Preoperative clinical DSS	StA	N/A	N/A	N/A
[Castellanos et al., 2011]	Industry: Pipeline failure analysis	StA	N/A	Form	N/A
[Devraj and Jain, 2011]	Agriculture: Pulse crops disease diagnosis	Web	N/A	Interv	Ques (a): Feedback form
[Hasan and Isaac, 2011]	Agriculture: Sugarcane disease	Web	N/A	Form	Test (a), Ques (a)
[Keleş et al., 2011]	Medicine: Breast-cancer diagnosis	StA	N/A	Form	N/A
[Kolhe et al., 2011]	Agriculture: crop disease plant protection	Web	N/A	Form	Ques (a)
[Santos et al., 2011]	Transportation: Vehicle routing	Web	PI	Form & Own	N/A
[Ting et al., 2011]	Medicine: Prescription advice	StA	N/A	Form	FSt(a), Iv(a)
2010					
[Cho, 2010]	Finance: Stock market investment	StA	PI	Own	N/A
[Ghandforoush and Sen, 2010]	Medical: Platelet production supply chain management	StA	PI	Form	N/A
[Grahovac and Devedzic, 2010]	Business: Cost management	StA	N/A	Interv	N/A

Ref	Cont	Type	Pro	UI	Usab
[İÇ and Yurdakul, 2010]	Finance: Credibility scoring	StA	N/A	Form	N/A
[Kiriş et al., 2010]	Medicine: Scheduling emergency departments	StA	N/A	Form	N/A
[Sarma et al., 2010]	Agriculture: Rice plant disease diagnosis	StA	N/A	Interv	N/A
[Ting et al., 2010]	Medicine: prescription	StA	PI	Form	Met: Medical records, Iv (a)
[Rodríguez et al., 2010]	Environment: Natural-disaster management	Web	N/A	N/A	N/A
2009					
[Huang et al., 2009]	Industry: Mold base design	Web	PI	Own	N/A
[İÇ and Yurdakul, 2009]	Industry: Machining center selection	StA	N/A	Form	N/A
[Kim et al., 2009]	Engineering: Fatigue life prediction	StA	N/A	Form, Interv	N/A
[Li et al., 2009]	Environment: Fish disease early warning	Web	PI	Integ	Ques (b), Iv (b)
[Papić et al., 2009]	Individual: Sports advice	StA to Web	PI	N/A	N/A
[Shue et al., 2009]	Business: Financial rating	StA	N/A	N/A	N/A
[Sutton and Patkar, 2009]	Medicine: Breast cancer diagnosis	Web	PI	Form	N/A
[Xiaoshuan et al., 2009]	Veterinary: Fish disease/health	StA	ThA, Evo	Form	Test (i), Iv (i), FF (i)

Table B.3: Summary of the findings of the journal-based literature research regarding the current state of the art in building KBS/ES. Special focus on applied UI and usability techniques. The table lists the reference, the application context, the KBS type, the Prototyping technique applied (if any), the basic UI type, and the usability technique(s) applied. N/A denotes that no information was given.

Appendix C

Evaluations—Materials

In the following, we provide exemplary additional materials for the evaluation studies reported in this thesis. That is, of the instruction sheets, problem descriptions, and the concluding (usability) questionnaires. The original wording of those documents mostly is german, due to the characteristics of the studies groups (german participants). The unabridged and entire materials for all studies are available in a freely accessible online repository, see Appendix E.

- Appendix C.1 — ITree Study I (03/2012).
- Appendix C.2 — ITree Study II (05/2012).
- Appendix C.3 — ITree Study III (12/2013).
- Appendix C.4 — The KBS UI pattern expert evaluation (02–03/2014).
- Appendix C.5 — The KBS UI pattern user study (04–05/2014).

C.1 ITree—Study I (03/2012)

C.1.1 Exemplary Instruction Sheet

See p. 208

C.1.2 Exemplary Problem Description

Fall 1 - Arbeitsrecht / Kündigungsschutz
Siegfried Surfer erhielt am **27.02.2012** eine ordentliche Kündigung seines Arbeitgebers. Er erhielt sie **persönlich überreicht** vom **Prokuristen** der Firma, **der die Kündigung auch unterzeichnet** hat. Der **Betriebsrat wurde korrekt angehört**.
Die Firma beschäftigt **8 Vollzeit-Angestellte und 6 teilzeitbeschäftigte** Arbeitnehmer mit 20 Stunden; da Surfers **unbefristetes Arbeitsverhältnis** zudem **seit 01.91.2007 (also bereits über 5 Jahre)** lief, müsste nach Auffassung Siegfried Surfers das Kündigungsschutzgesetz für ihn gelten und somit **eine soziale Rechtfertigung** notwendig machen.
Da das Arbeitsverhältnis weiterhin **keine besonderen Regelungen zur Kündigungsfrist** umfasste, soll es nun durch diese als ordentliche Kündigung bezeichnete Kündigung **zum 30. April 2012** enden. Diese Kündigungsfrist von 2 Monaten zum Monatsende findet Siegfried Surfer bei seiner Dauer der Betriebszugehörigkeit etwas kurz. Verwunderlicher aber findet er den genannten Kündigungsgrund: Er habe unerlaubt das Internet **privat** genutzt, obwohl erst eine Woche zuvor am 20.07.2012 eine diesbezügliche Schulung zur firmeninternen Richtlinie

stattgefunden hatte, welche dies **nunmehr zulässigerweise untersagt**. Dies stimmt zwar, aber Siegfried Surfer kann sich einfach nicht vorstellen, dass der genannte Grund überhaupt zur ordentlichen Kündigung berechtigt; wenn überhaupt, so Surfers Vermutung, dann könne er wohl zumindest **nicht ohne Vorwarnung - wie beispielsweise eine Abmahnung - einfach so** gekündigt werden. Siegfried Surfer möchte nun am **03.03.2012 (innerhalb der 3-wöchigen Kündigungs-schutzklagefrist)** wissen, ob die ausgesprochene ordentliche Kündigung wirksam ist.

C.1.3 Concluding Questionnaire: Question Catalogue

Questions 1 to 5 were to be rated on a scale from 0 (completely disagree) to 7 (completely agree). Question 6 was a one choice question, and question 7 provided the possibility of entering any desired (free) comment(s).

1. Ich empfand die Interaktion mit dem System insgesamt als intuitiv.
2. Ich empfand die inhaltliche Strukturierung der Fragen als verständlich.
3. Ich konnte nachvollziehen, warum das System zu der finalen Bewertung des Sachverhalts kam.
4. Ich denke, ich konnte mithilfe des Systems den Sachverhalt zutreffend einschätzen / lösen.
5. Ich habe das Gefühl, durch die Benutzung des Systems zusätzliches Wissen bezüglich des Sachverhalts gewonnen zu haben.
6. Welcher der beiden Dialoge hat mir besser gefallen? (Nur nach Bearbeitung beider Fälle beantworten!): Dialog 1 — Dialog 2 — Beide gleich
7. Weitere Anmerkungen: Freitext

C.2 ITree—Study II (05/2012)

C.2.1 Exemplary Instruction Sheet

See p. 209

C.2.2 Exemplary Problem Description

Fall 1 - Arbeitsrecht / Kündigungsschutz
Kathrin Krank arbeitet seit 01.04.2009 bei ihrem Arbeitgeber in einem unbefristeten Arbeitsverhältnis. Besondere Kündigungsfristen wurden nicht vereinbart. Ihr Arbeitgeber beschäftigt neun Vollzeit-Angestellte und drei teilzeitbeschäftigte Arbeitnehmer mit 20 Stunden. Betriebsrat ist keiner vorhanden.
Aufgrund der Folgen eines Zeckenbisses im Juni 2010 litt Kathrin Krank an mehrfachen Kurzerkrankungen – sie war seither insgesamt in 2010 acht Wochen krank, in 2011 vier Wochen. Ende 2011 konnte die vollständige Heilung der Grunderkrankung erreicht werden. Im April 2012 erkrankte sie an einer Grippe, weshalb ihr Arzt sie für fünf Tage für arbeitsunfähig erklärte.

Während dieser krankheitsbedingten Abwesenheit erhielt sie per Übergabe-Einschreiben am 23.04.2012 wegen Krankheit eine ordentliche Kündigung ihres Arbeitgebers zum 31.05.2012. Kathrin Krank fragt sich Folgendes:

- Kann ihr eine Kündigung wirksam zugehen per Post – muss sie diese nicht persönlich erhalten?
- Kann sie wegen einer Krankheit und zudem in der Krankheit gekündigt werden?
- Stimmt die Kündigungsfrist — ist diese nicht für über drei Jahre Betriebszugehörigkeit zu kurz?

Sie möchte daher nun 27.04.2012 wissen, ob die ausgesprochene ordentliche Kündigung wirksam ist.

C.2.3 Usability Questionnaire: Question Catalogue

Questions 1 to 5 were to be rated on a scale from 0–7 (completely disagree–completely agree). Question 6 and 7 were one choice questions, question 8 was a free comment question.

1 Ich empfand die Interaktion mit dem System insgesamt als intuitiv.

2 Ich empfand die inhaltliche Strukturierung der Fragen als verständlich.

3 Ich konnte nachvollziehen, warum das System zur finalen Bewertung des Sachverhalts kam.

4 Ich denke, ich konnte mithilfe des Systems den Sachverhalt zutreffend einschätzen/lösen.

5 Ich habe das Gefühl, durch die Benutzung des Systems zusätzliches Wissen bezüglich des Sachverhalts gewonnen zu haben.

6 Welcher der beiden Dialoge hat mir besser gefallen? (Nur nach Bearbeitung beider Fälle beantworten!): Dialog 1 — Dialog 2 — Beide gleich

7 Ich bin Jurist und verfüge über juristisches Hintergrundwissen im/in...:
Arbeitsrecht – anderen Rechtsgebieten – ich bin KEIN Jurist

8 Weitere Anmerkungen: Freitext

C.3 ITree—Study III (12/2013)

C.3.1 Exemplary Problem Description & (Short) Instructions

Kündigungsschutz/Fristlose Kündigung Die 45 jährige Kathrin Krank arbeitet seit 01.04.1999 Vollzeit in einem ungekündigten Angestelltenverhältnis für die Fuchs AG, einem Hersteller von exklusiver Bekleidung und Accessoires mit Sitz in München. Zu ihrem Aufgabengebiet gehört die Pflege und Wartung der Homepage der Firma, welche auch eine sog. Kassenfunktion aufweist. Mit Hilfe dieser Kassenfunktion können die Kunden die gewünschte Ware direkt bei der Fuchs AG bestellen. Kathrin Krank macht ihre Arbeit gerne, ist mit Eifer bei der Sache und arbeitet stets überkorrekt. Daher macht es ihr auch nichts aus, regelmäßig Überstunden

zu machen. Sie verdient brutto 2.800,00 Euro. Neben Kathrin Krank sind bei der Fuchs AG noch 115 Vollzeit-Angestellte beschäftigt. Es besteht ein Betriebsrat. Mit den anderen Kollegen versteht sich Kathrin Krank sehr gut, insbesondere mit ihren beiden Zimmerkolleginnen pflegt sie ein freundschaftliches Verhältnis und arbeitet im Team sehr gut zusammen. Laut ihrem Arbeitsvertrag gelten die gesetzlichen Kündigungsfristen, besondere Kündigungsfristen wurden nicht vereinbart.
Im Juni 2012 wird Kathrin Krank von einer Zecke gebissen und hat seitdem eine unerkannte Hirnhautentzündung. Von da an wird Kathrin Krank immer verwirrter. Sie verläuft sich regelmäßig in der Firma, vergisst Termine und verwechselt die Beträge der Produkte. Weder ihr direkter Vorgesetzter, Herr Martin Geduldig, noch ihre Kolleginnen erkennen Kathrin Krank wieder. Immer häufiger kommt es zu Problemen, weil Kathrin Krank die Preise verwechselt und teilweise die Ware weit unter dem Herstellungspreis anbietet. Ihr Vorgesetzter, der Kathrin Krank als zuverlässige Arbeitnehmerin zu schätzen gelernt hatte, nimmt sie lange Zeit in Schutz und versucht ihre Fehler zu vertuschen.
Gerade an dem Tag als Kathrin Krank Überstunden nimmt, um einen Spezialisten für ihr Problem aufzusuchen wird der Geschäftsführer der Fuchs AG, Herr Stefan Grantig, darauf aufmerksam, dass die Verkaufszahlen eines Kaschmirpullovers auf der Homepage explodieren. Als er sich die Seite genauer anschaut bemerkt er, dass der Kaschmirpullover anstelle von 480,- Euro für 48,- Euro angeboten wird. Sofort veranlasst er, dass die Ware von der Homepage genommen wird. Weil er seine kaufkräftigen Kunden nicht verärgern will und keinen Imageverlust erleiden will, entscheidet er sich, die bereits bestellte Ware für diesen Niedrigpreis herzugeben. Wutentbrannt verfasst er ein Schreiben an die Kathrin Krank, in welcher er sie fristlos wegen ihres Fehlers kündigt.
Kathrin Krank erfährt von den Vorkommnissen in der Firma nichts. Stattdessen freut sie sich sehr darüber, endlich zu wissen was sie hat und dass sie nach Einnahme entsprechender Medikamente in zwei bis drei Wochen wieder die Alte sein wird. Umso erschütterter ist sie, als sie am Folgetag bei Arbeitsantritt von Stefan Grantig die fristlose Kündigung erhält.
Kann sich Kathrin Krank gegen die Kündigung wehren?

- Nein
- Ja (mit Begründung in Stichworten)

C.3.2 Concluding Questionnaire: Question Catalogue

	Note
1. Wie bewerten Sie die Nützlichkeit des Beratungssystems insgesamt?	
2. Wie bewerten Sie die Auffindbarkeit der Rechtsgebiete auf der Internetseite?	

Bitte tragen Sie in jedes Kästchen eine Schulnote ein!	**Fall 1: Fristlose Kündigung**	**Fall 2: Ordentliche Kündigung**	**Fall 3: Unfallflucht**
3. Bewertung der inhaltl. Qualität der Wissensbasis			
4. Bewertung der Einfachheit der Bedienung			
5. Bewertung der Effizienz des Systems			
6. Bewertung optische Gestaltung des Ergebnisses			
7. Wie sicher sind Sie sich, dass Ihr Ergebnis korrekt ist? (1=sehr sicher, 6=kein Vertrauen in Ergebnis)			
8. Bewertung der Hilfestellung was jetzt im Fall weiter zu tun wäre			
9. Ihr Kommentar (optional)	Freitext		

C.4 KBS UI Patterns, Expert Evaluation (03/2014)

Subsequently we provide the task descriptions for the preliminary- (see C.4.1) and the main expert evaluation (see C.4.2) in early 2014. We provide the originally worded task descriptions—i.e., in german language, as the participants were german HCI students. Due to reasons of overview, we did not reproduce common standards, e.g., the used heuristics of Jakob Nielsen, here as they are also already included in Appendix D of this work.

C.4.1 Preliminary Evaluation–Task Description

Zu evaluierendes Interface Im Rahmen dieser Aufgabe ist ein Interface zu evaluieren, welches Angestellten eines Rechtsanwaltsbüuros helfen soll, eingereichte Fälle schneller zu bearbeiten. Die Angestellten haben gute Kenntnis der rechtlichen Hintergründe und mittelmäßige Erfahrung in der Bedienung von Computern. Das zu evaluierende Interface ist unter diesem Link [ITree UI Link, 2014] erreichbar. Wählen sie eine der beiden Evaluationsmethoden *Cognitive Walkthrough* und *Heuristic Evaluation*. Die Aufgabenbeschreibung zu den Methoden finden Sie in den folgenden Abschnitten.

I. Cognitive Walkthrough Gegeben sei folgendes fiktives Beispiel:

- Ein Mieter möchte sich telefonisch darüber informieren, ob er rechtlich gegen einen Mangel in seiner von ihm bewohnten Mietwohnung vorgehen kann. Es handelt sich dabei um einen Schaden an der Dachrinne in der Nähe seines Büros im 5. Stock, welcher zu Feuchtigkeit im Büro führt. Dem Mieter ist bekannt geworden, dass der Mangel dem Vermieter schon bei Vertragsabschluss bekannt war, ihm (dem Mieter) aber verschwiegen wurde. Während der Eingabe der Daten in das zu evaluierende System kann der Mieter die Frage ob die Behebung des Mangels vertraglich ausgeschlossen wurde nicht beantworten, da er den Mietvertrag gerade nicht zur Hand hat. Der Mitarbeiter markiert daher den Punkt *Sind der Mangel bzw. die Mängel erst im Laufe der Mietzeit aufgetreten oder schließt ein bei Vertragsschluss vorliegender Mangel die Mieterrechte, z.B. auf Minderung, nicht aus?* als unsicher.

- An dieser Stelle soll der Mitarbeiter den Fall speichern und einen neuen Fall öffnen, da ein weiterer Klient anruft. Zur Sicherheit möchte er den alten Fall zusätzlich ausdrucken.
- Als der Klient erneut anruft, öffnet der Mitarbeiter den gespeicherten Fall und trägt die fehlende Information *Wurde die Behebung des Mangels bzw. der Mängel durch den Vermieter vertraglich ausgeschlossen?* mit *ja* nach, nachdem ihm der Klient selbiges gemeldet hat.

Folgende Fragen sollen dem Klienten beantwortet werden:

1 Kann der Klient rechtlich gegen den Mangel vorgehen?

2 Welche Mietminderung kann aufgrund der defekten Dachrinne erwartet werden?

3 Aus welchem Grund kommt das Ergebnis zustande? (copy & paste ist hier erlaubt)

Hinweis: Bei der Durchführung sollten möglichst detaillierte Informationen in das System eingetragen werden (d.h. nicht nur Oberpunkte auswählen).

Aufgabe: Führen sie aufgrund der gegebenen Informationen einen Cognitive Walkthrough durch. Notieren sie ihre Ergebnisse und Verbesserungsvorschläge.

II. Heuristic Evaluation Diese Aufgabe beschäftigt sich mit der Anwendung der zehn Usability Regeln von Jakob Nielsen (Appendix D.2 of this work)

Aufgabe: Evaluieren sie das Interface mit der Methode Heuristic Evaluation auf Basis der oben stehenden Regeln und bewerten sie die Schwere der Verstöße. Erarbeiten sie dann Verbesserungsvorschläge für das Interface. Notieren sie ihre Ergebnisse und Verbesserungsvorschläge.

C.4.2 Main Evaluation–Task Description

The main evaluation targeted five KBS UI patterns in total. Therefore, the evaluators received identical task descriptions as follows.

a. Heuristic Evaluation

Sites to evaluate
- ITree (Hit & Run), `http://132.187.15.40:8083/ITreeUnfall/`
- ITree (Dismissal), `http://132.187.15.40:8083/ITreeKuendigung/`
- Daily-Style (Pubs), `http://132.187.15.40:8083/DailyKneipen/`
- Questionary (Plants), `http://132.187.15.40:8083/QuestionaryPanzen/`
- Interview (Statistics), `http://132.187.15.40:8083/InterviewStatistik/`

- Entscheidungsbaum (Statistics), `http://132.187.15.40:8083/DectreeStatistik/`

Report: The report has to provide method description, the evaluation task, failures as well as improvement proposals and a conclusion. The report should be created with the help of LaTeX and may either written in german or english.

Additional material: You will be provided with additional material to ease the evaluation: In the WueCampus course you can find documents describing use cases for the respective interface (which are used for the cognitive walkthrough tasks). You may use these document to get an idea of the dialogs' purpose. In addition, an Excel sheet is available in the WueCampus course, which we kindly ask you to fill after each evaluation of an interface.

Approach: First, each group member does a Heuristic Evaluation on his/her own and gathers improvement proposals. Second, the group meets and carries the information together. In the group you decide on the seriousness of the failures and write the report.

Task Description: Your task is to evaluate the user interfaces specified above by means of Heuristic Evaluation. Conduct a Heuristic Evaluation for each interface, using Nielsen's 10 heuristics. The dialogs are meant to be viewed in the Mozilla Firefox web browser, hence you should assume that each user will use that browser. Please clearly differentiate between user interface and content related problems (i.e., classify the problem in your report).

b. Cognitive Walkthrough

Sites to evaluate
- ITree (Hit & Run), `http://132.187.15.40:8083/ITreeUnfall/`
- ITree (Dismissal), `http://132.187.15.40:8083/ITreeKuendigung/`
- Daily-Style (Pubs), `http://132.187.15.40:8083/DailyKneipen/`
- Questionary (Plants), `http://132.187.15.40:8083/QuestionaryPanzen/`
- Interview (Statistics), `http://132.187.15.40:8083/InterviewStatistik/`
- Entscheidungsbaum (Statistics), `http://132.187.15.40:8083/DectreeStatistik/`

Report: The report has to provide method description, the evaluation task, failures as well as improvement proposals and a conclusion. The report should be created with the help of LaTeX and may either written in german or english.

Additional Material: You will be provided with additional material to ease the evaluation: In the WueCampus course you can find documents describing use cases for the respective interface. You shall use these document to get an Idea of the dialogs' purpose and prepare your cognitive walkthrough. In addition, an Excel sheet is available in the WueCampus course, which we kindly ask you to fill after each evaluation of an interface.

Approach: First, each group member does a Cognitive Walkthrough Evaluation on his/her own and gathers improvement proposals. Second, the group meets and carries the information together. In the group you decide on the seriousness of the failures and write the report.

Task Description: Your task is the evaluate the user interfaces specified above with the help of the Cognitive Walkthrough. Use the provided task descriptions to prepare your evaluation. The dialogs are meant to be viewed in the Mozilla Firefox web browser, hence you should assume that each user will use that browser. Please clearly differentiate between user interface and content related problems (i.e., classify the problem in your report).

c. Additional Material—Case Descriptions

Aufgabe Zimmerpflanzen Beratung – Questionary Dialog Sie möchten Ihrer Wohnung ein wenig mehr Leben und Farbe verpassen und entscheiden sich für die Anschaffung einiger Zimmerpflanzen. Die Fülle der Möglichkeiten macht Ihnen eine spontane, für Ihre Lebenssituation sinnvolle und bestmöglichst passende, Entscheidung aber schwer. Ermitteln Sie mit dem Zimmerpflanzen Beratungs System welche Pflanzen für Sie in Frage kommen.

Aufgabe Kneipenrecommender – Daily-Style Dialog Sie möchten mal wieder in Würzburg oder Umgebung ausgehen und können sich angesichts der vielen Möglichkeiten nicht recht entscheiden. Ermitteln Sie mit Hilfe des Daily-Style Dialogs Kneipenrecommender für Sie passende Optionen.

Aufgabe Statistische Beratung - Interview Sie bekommen die Aufgabe, 2 Datensätze statistisch zu analysieren. Hierzu bekommen Sie die unten beschriebenen Informationen und Zielvorgaben vorgelegt. Entscheiden Sie sich mithilfe des Interviews zur statistischen Methodenberatung für die passendste statistische Auswertungsmethode.

Datensatz 1: Gibt es einen Zusammenhang zwischen der Muttersprache und dem Wohnort innerhalb der Stadt Zürich? Gesammelte Daten: Zu einer Person werden jeweils die Muttersprache und der Wohnort ermittelt. Gefundene Lösung bitte im Report angeben!

Datensatz 2: Ein Dozent hält die selbe Vorlesung im selben Semester aufgrund der großen Nachfrage dreimal. Bei einer der Klassen beginnt er jede Stunde mit einer Zusammenfassung der letzten Stunde, bei einer Gruppe lässt er jeweils eine Gruppe von Studierenden dies tun und bei der dritten Klasse verzichtet er auf Zusammenfassungen. Die Studierenden müssen im Verlauf des Semesters vier Aufsätze abgeben. Unterscheiden sich die Noten der drei Gruppen? Gesammelte Daten: Zugehörigkeit der Studenten zu jeweils einer der 3 Gruppen, Erzielte Noten in jedem der 4 Aufsätze. Gefundene Lösung bitte im Report angeben!

Aufgabe Statistische Beratung – Entscheidungsbaum: Sie bekommen den Auftrag 2 Datensätze statistisch zu analysieren. Hierzu bekommen Sie die unten beschriebenen Informationen und Zielvorgaben vorgelegt. Entscheiden Sie sich mithilfe des Entscheidungsbaums zur statistischen Methodenberatung für die passendste statistische Auswertungsmethode.

Datensatz 1: Setzen Sonderschul- und Regelklassen-Lehrpersonen andere Unterrichtsmethoden ein? Gesammelte Daten: Zu einer Person wird jeweils ermittelt welche Art Lehrperson er/sie ist und welche Unterrichtsmethoden er/sie einsetzt. Gefundene Lösung bitte im Report angeben!

Datensatz 2: Eine Schulpsychologin möchte wissen, welche von zwei möglichen Interventionen in einem bestimmten Typ von Situation zu besseren Ergebnissen führt. Sie lässt daher die Lehrpersonen der von ihr therapierten Kinder das Verhalten nach der Intervention auf einer dreistufigen Skala beurteilen. Gesammelte Daten: Für jedes Kind wird die angewandte Interventionsart sowie die Verhaltensbeurteilung notiert. Gefundene Lösung bitte im Report angeben!

Aufgabe ITree—Fristlose Kündigung Bitte lösen Sie den Fall mit dem ITree-Dialog Kündigungsschutz. Bitte geben Sie im Evaluationsreport an, ob sich Kathrin Krank gegen die Kündigung wehren kann, und falls ja, mit kurzer Begründung (Stichpunkte).
$\longrightarrow$ The same case description was used as in the ITree Study III, see Section C.3.1.

Aufgabe ITree—Unfallflucht Bitte lösen Sie den Fall mit dem ITree Dialog zum Kündigungsschutz. Bitte geben Sie im Evaluationsreport an, ob sich die Fahrerin des Jaguars wegen unerlaubtem Entfernen vom Unfallort strafbar gemacht hat (Stichpunkte).
Das Ehepaar Anton und Babette Fuhrmann gönnen sich jeden Sonntagnachmittag jeweils ein Kännchen Kaffee und ein Stückchen Torte im Altstadtcafé von Bamberg. Hierzu fahren sie immer mit ihrem PKW, einer einjährigen silberfarbenen E-Klasse, in die Altstadt und parken dort auf einem nahen öffentlichen Parkplatz. So auch am Freitag dem 13.12.2013. Als die beiden um ca. 17.00 Uhr zu ihrem PKW zurückkehren bemerken sie sofort, dass die Beifahrertür großflächig und tief eingedrückt ist, sowie der Lack stellenweise bis auf das Blech abgescheuert wurde bzw. stellenweise roter Lack auf dem silbernen Lack aufgerieben wurde.
An der Windschutzscheibe hängt ein Zettel von einer gewissen Frau Margarte Fuchsberger, die direkt neben dem Parkplatz im ersten Stock eines Seniorenstiftes wohnt und von ihrem Couchsessel den Parkplatz gut überblicken kann. Sofort suchen die Eheleute Fuhrmann Frau Fuchsberger auf. Diese erzählt, was sie beobachtet hat:
Eine Frau ca. Mitte vierzig mit langen blonden Locken bis zur Mitte des Rückens wollte mit in ihrem schnittigen roten Sportwagen der Marke Jaguar rückwärts einparken. Wahrscheinlich rutschte sie dabei von dem Bremspedal auf das Gaspedal, denn plötzlich machte der PKW einen Satz nach hinten. Es gab dann einen großen Knall. Da das Lenkrad eingeschlagen war, drückte sich die rechte Ecke der hinteren Stoßstange des Jaguars in die Beifahrertür des Mercedes. Dann fuhr die Fahrerin des Jaguars aus der Parklücke heraus, stieg aus und begutachtete die Schäden an dem Mercedes und an dem Jaguar. Frau Fuchsberger öffnete schnell das Fenster und fragte, ob sie helfen könnte. Die Fahrerin des Jaguars entgegnete, dass alles in Ordnung sei, sie habe sich nicht verletzt. An den Fahrzeugen seine nur Kratzer. Sie würde gleich einen Zettel mit ihren Personalien an dem Mercedes anbringen, müsste aber dann sofort weiter, weil sie zu einer Theatervorführung müsste. Frau Fuchsberger schloss daraufhin ihr Fenster und widmete sich wieder ihrem Fernsehprogramm. Als sie zwanzig Minuten später wieder hinausschaute, bemerkte sie, dass kein Zettel an dem Mercedes hing und schrieb den Eheleuten einen Zettel mit ihren Kontaktdaten. Glücklicherweise hatte sich Frau Fuchsberger die Daten des Jaguars, Typ, Farbe und Kennzeichen aufgeschrieben. Die Eheleute Fuhrmann fahren sofort zur nächsten Polizeistelle und zeigen dort den Unfall an, samt Schilderung des von der Zeugin Frau Fuchsberger geschilderten Sachverhaltes. Die Fahrerin des Jaguar zeigte auch am Folgetag nicht den Unfall bei der Polizei an.

C.5 KBS UI Patterns, User Study (05/2014)

Subsequently we list the particular task description(s), evaluation sheet, and encompassing results of the (comparative) user study performed in April 2014.

C.5.1 Task Description(s)

Assessment According to the ISO 9241–110 Bitte analysieren Sie die 5 Dialoge gemäß den sieben untenstehenden Ergonomie-Kriterien der ISO 9241-110. Geben Sie dazu jeweils das Kriterium an, eine Bewertung und eine Begründung dazu.
1) Aufgabenangemessenheit, 2) Selbstbeschreibungsfähigkeit, 3) Steuerbarkeit, 4) Erwartungskonformität, 5) Fehlertoleranz, 6) Individualisierbarkeit, 7) Lernförderlichkeit

Problem Solving Tasks Spielen Sie jeden Dialog mit der unten aufgeführten Fallbeschreibung durch und notieren die Lösung zum Fall in der Tabelle. Geben Sie auch die Bildschirmgröße in Pixel des verwendeten Bildschirms an, da ein größerer Bildschirm bei der Dialogbenutzung (insbesondere des komplizierten juristischen Falles) vorteilhaft ist. Bearbeiten Sie folgende Beispiele:

a: Fragebogen Dialog im Daily-Layout für Zimmerpflanzenauswahl: Sie wollen einer Freundin eine Zimmerpflanze zum Wohnungseinzug schenken. Da Ihre Freundin als Abteilungsleiterin der Werbebranche einen arbeitslastigen Beruf besitzt und die restliche Freizeit ihre beiden kleinen Kinder und ihr 3-jährigen Siamkater beansprucht, wollen Sie ihr eine möglichst einfach zu pflegende Pflanze mit wenig Gieß- und Düngeaufwand schenken. Weil Ihre Freundin im neuen Wohnzimmer (mit gewöhnlicher Luftfeuchtigkeit) ein großes schwach abgeschattetes Fensterbrett besitzt, sollte die perfekte Pflanze hitzeresistent sein und indirekte Sonnenstrahlen vertragen können. Um nicht das komplette Fenster zu verdecken, wollen Sie eine Pflanze, die eine maximale Höhe von 40 cm sowohl bei Kauf als auch später erreicht. Die Blütenfarbe ist Ihnen zweitrangig, da Sie ihr ebenso gerne auch eine Grünpflanze schenken würden. Sie würden für Ihre gute Freundin auf jeden Fall bis zu 20 Euro für die Pflanze ausgeben. Ihre Freundin hat keine Allergien, kennt sich aber mit Pflanzen nicht so gut aus.

b: Fragebogen Dialog im Box-Layout für Zimmerpflanzenauswahl: Sie ziehen gerade frisch mit Ihrem Freund/Ihrer Freundin zusammen. Da ihr Haus ein großes, halbschattiges Treppenhaus besitzt, welches Ihnen noch sehr kahl erscheint, sehen Sie sich in der Online-Pflanzenberatung um. Die Pflanze für das Treppenhaus sollte folgenden Ansprüchen genügen: Temperatur eher kühl, schattiges Treppenhaus, durchschnittlicher Pflege-, Gieß- und Düngeaufwand, für Anfänger geeignet, großer Wuchs, darf teuer sein und soll auch für den Balkon geeignet sein.

c: Entscheidungsbaumbasierter Dialog für Auswahl eines statistischen Tests: See Appendix C.4.2.c., *Aufgabe Statistische Beratung - Interview, Datensatz 1.*

d: Interview (1-Frage-Dialog) für Auswahl eines statistischen Tests: See Appendix C.4.2.c., *Aufgabe Stat. Beratung – Entscheidungsbaum, Datensatz 2.*

e: Klärungsbasierter Entscheidungsbaum-Dialog (Fristlose Kündigung): See App. C.3.1.

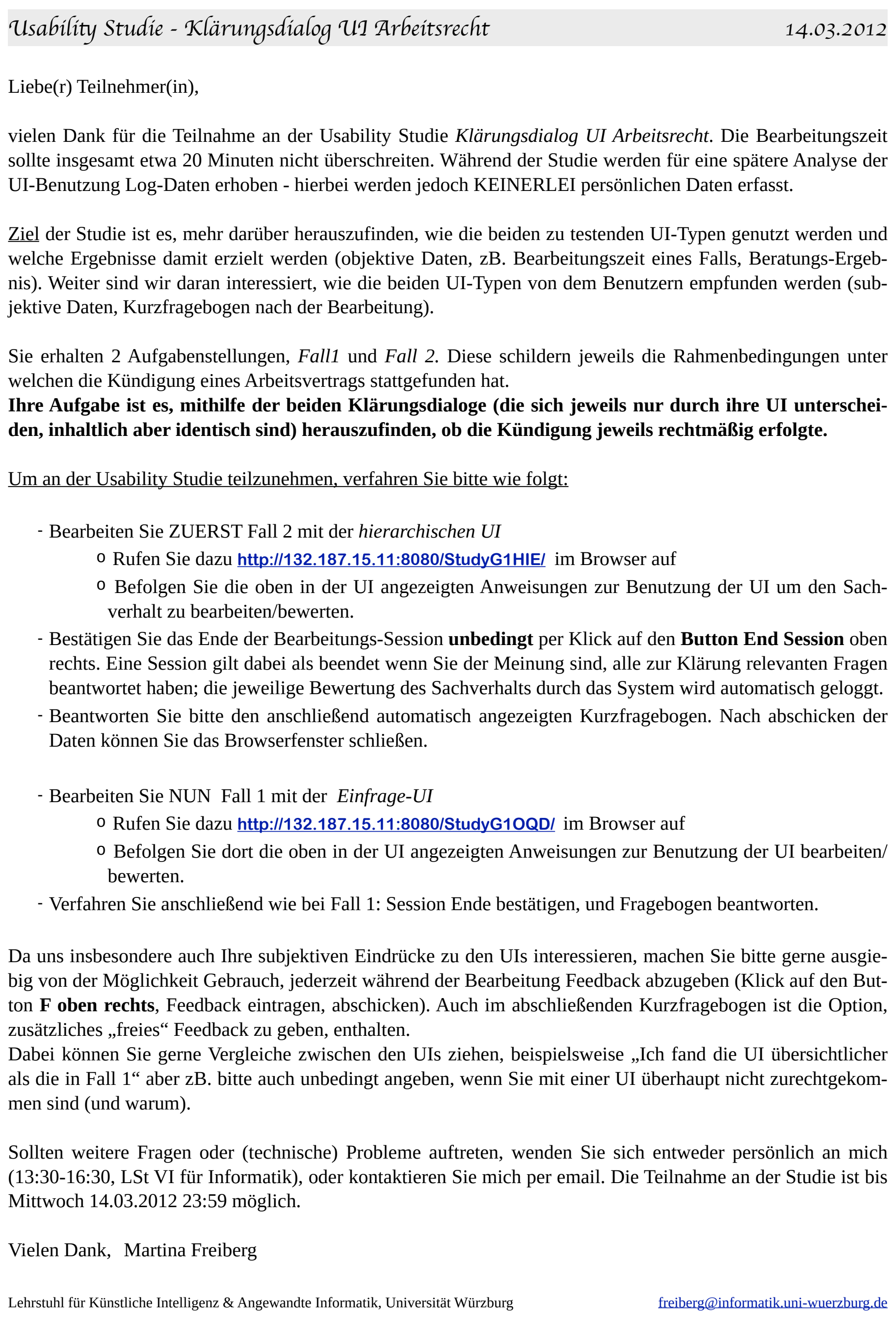
C.5.2 Concluding Questionnaire, Question Catalogue

Bitte bewerten Sie die getesteten Dialoge nach verschiedenen Einzelkriterien mit Schulnoten:
1 (sehr gut) bis 6 (sehr schlecht)

Bitte berücksichtigen Sie bei der Bewertung die Komplexität der jeweils zugrunde liegenden Wissensbasis

Bitte tragen Sie in jedes Kästchen eine Schulnote ein!	**Fall 1: ITree (Arbeits-recht Kün-digung)**	**Fall 2: Entschei-dungsbaum Statistik**	**Fall 3: Interview (Statistik)**	**Fall 4: Fragebogen Boxlayout (Pflanzen)**	**Fall 5: Fragebogen Daily Layout (Pflanzen)**
1. Wie bewerten Sie die Nützlichkeit des Beratungssystems insgesamt?					
2. Bitte geben Sie die Lösung des Falles ein.					
3. Wie sicher sind Sie sich, dass Ihr Ergebnis korrekt ist? 1=sehr sicher, 6=kein Vertrauen in das Ergebnis					
4. Wie bewerten Sie die inhaltliche Qualität der Wissensbasis? (Verständlichkeit und Strukturierung)					
5. Haben Sie das Gefühl, dass das System Ihnen zusätzliches Wissen über die Thematik vermitteln konnte?					
6. Wie bewerten Sie die Einfachheit der Bedienung?					
7. Wie bewerten Sie die Effizienz des Systems?					
8. Bitte geben Sie die Größe des Bildschirms an, mit dem Sie den Fall bearbeitet haben.					
9. Wie bewerten Sie die optische Gestaltung der Eingabeoberfläche?					
10. Weitere Kommentare (ggf. Rückseite): Dies können sowohl positive wie negative Beobachtungen sein. Bitte geben Sie jeweils den betroffenen Dialogtyp zum jeweiligen Kommentar an.					

Table C.1: KBS UI Pattern Assessment: User Study 05/2014, Questionnaire.



Usability Studie - Klärungsdialog UI Arbeitsrecht 14.03.2012

Liebe(r) Teilnehmer(in),

vielen Dank für die Teilnahme an der Usability Studie *Klärungsdialog UI Arbeitsrecht*. Die Bearbeitungszeit sollte insgesamt etwa 20 Minuten nicht überschreiten. Während der Studie werden für eine spätere Analyse der UI-Benutzung Log-Daten erhoben - hierbei werden jedoch KEINERLEI persönlichen Daten erfasst.

Ziel der Studie ist es, mehr darüber herauszufinden, wie die beiden zu testenden UI-Typen genutzt werden und welche Ergebnisse damit erzielt werden (objektive Daten, zB. Bearbeitungszeit eines Falls, Beratungs-Ergebnis). Weiter sind wir daran interessiert, wie die beiden UI-Typen von dem Benutzern empfunden werden (subjektive Daten, Kurzfragebogen nach der Bearbeitung).

Sie erhalten 2 Aufgabenstellungen, *Fall1* und *Fall 2*. Diese schildern jeweils die Rahmenbedingungen unter welchen die Kündigung eines Arbeitsvertrags stattgefunden hat.
Ihre Aufgabe ist es, mithilfe der beiden Klärungsdialoge (die sich jeweils nur durch ihre UI unterscheiden, inhaltlich aber identisch sind) herauszufinden, ob die Kündigung jeweils rechtmäßig erfolgte.

Um an der Usability Studie teilzunehmen, verfahren Sie bitte wie folgt:

- Bearbeiten Sie ZUERST Fall 2 mit der *hierarchischen UI*
  - Rufen Sie dazu **http://132.187.15.11:8080/StudyG1HIE/** im Browser auf
  - Befolgen Sie die oben in der UI angezeigten Anweisungen zur Benutzung der UI um den Sachverhalt zu bearbeiten/bewerten.
- Bestätigen Sie das Ende der Bearbeitungs-Session **unbedingt** per Klick auf den **Button End Session** oben rechts. Eine Session gilt dabei als beendet wenn Sie der Meinung sind, alle zur Klärung relevanten Fragen beantwortet haben; die jeweilige Bewertung des Sachverhalts durch das System wird automatisch geloggt.
- Beantworten Sie bitte den anschließend automatisch angezeigten Kurzfragebogen. Nach abschicken der Daten können Sie das Browserfenster schließen.

- Bearbeiten Sie NUN Fall 1 mit der *Einfrage-UI*
  - Rufen Sie dazu **http://132.187.15.11:8080/StudyG1OQD/** im Browser auf
  - Befolgen Sie dort die oben in der UI angezeigten Anweisungen zur Benutzung der UI bearbeiten/bewerten.
- Verfahren Sie anschließend wie bei Fall 1: Session Ende bestätigen, und Fragebogen beantworten.

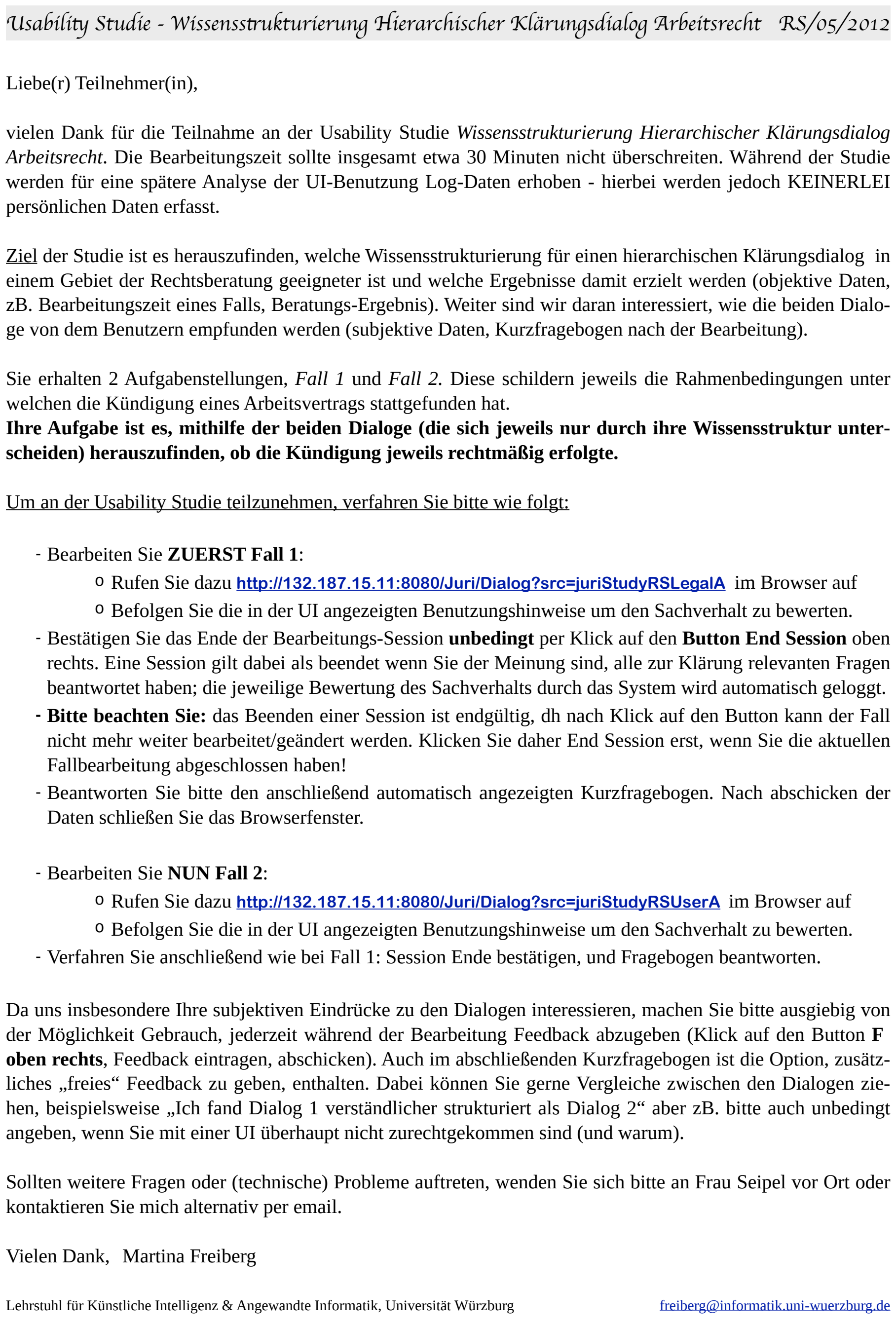
Da uns insbesondere auch Ihre subjektiven Eindrücke zu den UIs interessieren, machen Sie bitte gerne ausgiebig von der Möglichkeit Gebrauch, jederzeit während der Bearbeitung Feedback abzugeben (Klick auf den Button **F oben rechts**, Feedback eintragen, abschicken). Auch im abschließenden Kurzfragebogen ist die Option, zusätzliches „freies“ Feedback zu geben, enthalten.
Dabei können Sie gerne Vergleiche zwischen den UIs ziehen, beispielsweise „Ich fand die UI übersichtlicher als die in Fall 1“ aber zB. bitte auch unbedingt angeben, wenn Sie mit einer UI überhaupt nicht zurechtgekommen sind (und warum).

Sollten weitere Fragen oder (technische) Probleme auftreten, wenden Sie sich entweder persönlich an mich (13:30-16:30, LSt VI für Informatik), oder kontaktieren Sie mich per email. Die Teilnahme an der Studie ist bis Mittwoch 14.03.2012 23:59 möglich.

Vielen Dank, Martina Freiberg

Lehrstuhl für Künstliche Intelligenz & Angewandte Informatik, Universität Würzburg freiberg@informatik.uni-wuerzburg.de

Figure C.1: Exemplary instruction sheet, used in ITree Study I, March 2012.



Usability Studie - Wissensstrukturierung Hierarchischer Klärungsdialog Arbeitsrecht RS/05/2012

Liebe(r) Teilnehmer(in),

vielen Dank für die Teilnahme an der Usability Studie *Wissensstrukturierung Hierarchischer Klärungsdialog Arbeitsrecht*. Die Bearbeitungszeit sollte insgesamt etwa 30 Minuten nicht überschreiten. Während der Studie werden für eine spätere Analyse der UI-Benutzung Log-Daten erhoben - hierbei werden jedoch KEINERLEI persönlichen Daten erfasst.

Ziel der Studie ist es herauszufinden, welche Wissensstrukturierung für einen hierarchischen Klärungsdialog in einem Gebiet der Rechtsberatung geeigneter ist und welche Ergebnisse damit erzielt werden (objektive Daten, zB. Bearbeitungszeit eines Falls, Beratungs-Ergebnis). Weiter sind wir daran interessiert, wie die beiden Dialoge von dem Benutzern empfunden werden (subjektive Daten, Kurzfragebogen nach der Bearbeitung).

Sie erhalten 2 Aufgabenstellungen, *Fall 1* und *Fall 2.* Diese schildern jeweils die Rahmenbedingungen unter welchen die Kündigung eines Arbeitsvertrags stattgefunden hat.
Ihre Aufgabe ist es, mithilfe der beiden Dialoge (die sich jeweils nur durch ihre Wissensstruktur unterscheiden) herauszufinden, ob die Kündigung jeweils rechtmäßig erfolgte.

Um an der Usability Studie teilzunehmen, verfahren Sie bitte wie folgt:

- Bearbeiten Sie **ZUERST Fall 1**:
  - Rufen Sie dazu **http://132.187.15.11:8080/Juri/Dialog?src=juriStudyRSLegalA** im Browser auf
  - Befolgen Sie die in der UI angezeigten Benutzungshinweise um den Sachverhalt zu bewerten.
- Bestätigen Sie das Ende der Bearbeitungs-Session **unbedingt** per Klick auf den **Button End Session** oben rechts. Eine Session gilt dabei als beendet wenn Sie der Meinung sind, alle zur Klärung relevanten Fragen beantwortet haben; die jeweilige Bewertung des Sachverhalts durch das System wird automatisch geloggt.
- **Bitte beachten Sie:** das Beenden einer Session ist endgültig, dh nach Klick auf den Button kann der Fall nicht mehr weiter bearbeitet/geändert werden. Klicken Sie daher End Session erst, wenn Sie die aktuellen Fallbearbeitung abgeschlossen haben!
- Beantworten Sie bitte den anschließend automatisch angezeigten Kurzfragebogen. Nach abschicken der Daten schließen Sie das Browserfenster.

- Bearbeiten Sie **NUN Fall 2**:
  - Rufen Sie dazu **http://132.187.15.11:8080/Juri/Dialog?src=juriStudyRSUserA** im Browser auf
  - Befolgen Sie die in der UI angezeigten Benutzungshinweise um den Sachverhalt zu bewerten.
- Verfahren Sie anschließend wie bei Fall 1: Session Ende bestätigen, und Fragebogen beantworten.

Da uns insbesondere Ihre subjektiven Eindrücke zu den Dialogen interessieren, machen Sie bitte ausgiebig von der Möglichkeit Gebrauch, jederzeit während der Bearbeitung Feedback abzugeben (Klick auf den Button **F oben rechts**, Feedback eintragen, abschicken). Auch im abschließenden Kurzfragebogen ist die Option, zusätzliches „freies" Feedback zu geben, enthalten. Dabei können Sie gerne Vergleiche zwischen den Dialogen ziehen, beispielsweise „Ich fand Dialog 1 verständlicher strukturiert als Dialog 2" aber zB. bitte auch unbedingt angeben, wenn Sie mit einer UI überhaupt nicht zurechtgekommen sind (und warum).

Sollten weitere Fragen oder (technische) Probleme auftreten, wenden Sie sich bitte an Frau Seipel vor Ort oder kontaktieren Sie mich alternativ per email.

Vielen Dank, Martina Freiberg

Lehrstuhl für Künstliche Intelligenz & Angewandte Informatik, Universität Würzburg freiberg@informatik.uni-wuerzburg.de

Figure C.2: Exemplary instruction sheet, used in ITree Study II, May 2012.

Appendix D
Usability Standards

D.1 Norm [ISO 9241–110, 2006]

1 **Task Adequacy [Aufgabenangemessenheit]:** Ein interaktives System ist aufgabenangemessen, wenn es den Benutzer unterstützt, seine Arbeitsaufgabe zu erledigen, d. h., wenn Funktionalität und Dialog auf den charakteristischen Eigenschaften der Arbeitsaufgabe basieren, anstatt auf der zur Aufgabenerledigung eingesetzten Technologie.

2 **Self-Descriptiveness [Selbstbeschreibungsfähigkeit]:** Ein Dialog ist in dem Maße selbstbeschreibungsfähig, in dem für den Benutzer zu jeder Zeit offensichtlich ist, in welchem Dialog, an welcher Stelle im Dialog er sich befindet, welche Handlungen unternommen werden können und wie diese ausgeführt werden können.

3 **Controllability [Steuerbarkeit]:** Ein Dialog ist steuerbar, wenn der Benutzer in der Lage ist, den Dialogablauf zu starten sowie seine Richtung und Geschwindigkeit zu beeinflussen, bis das Ziel erreicht ist.

4 **Expectation Conformity [Erwartungskonformität]:** Ein Dialog ist erwartungskonform, wenn er den aus dem Nutzungskontext heraus vorhersehbaren Benutzerbelangen sowie allgemein anerkannten Konventionen entspricht.

5 **Error Tolerance [Fehlertoleranz]:** Ein Dialog ist fehlertolerant, wenn das beabsichtigte Arbeitsergebnis trotz erkennbar fehlerhafter Eingaben entweder mit keinem oder mit minimalem Korrekturaufwand seitens des Benutzers erreicht werden kann. Fehlertoleranz wird mit den Mitteln erreicht: Fehlererkennung und -vermeidung (Schadensbegrenzung), Fehlerkorrektur, Fehlermanagement (um aufgetretene Fehler zu behandeln).

6 **Individual Adaptability [Individualisierbarkeit]:** Ein Dialog ist individualisierbar wenn Benutzer die Mensch-System-Interaktion und die Darstellung von Informationen ändern können, um diese an ihre individuellen Fähigkeiten und Bedürfnisse anzupassen.

7 **Learnability [Lernförderlichkeit]:** Ein Dialog ist lernförderlich, wenn er den Benutzer beim Erlernen der Nutzung des interaktiven Systems unterstützt und anleitet.

D.2 The Ten Heuristics of [Nielsen, 1994]

1 **Visibility of the system status:** The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

2 **Match between the system and the real world:** The system should speak the user's language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

3 **User control and freedom:** Users often choose system functions by mistake and will need a clearly marked emergency exit to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

4 **Consistency and standards:** Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

5 **Error prevention:** Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

6 **Recognition rather than recall:** Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

7 **Flexibility and efficiency of use:** Accelerators, unseen by the novice user, may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

8 **Aesthetic and minimalist design:** Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

9 **Help users recognize, diagnose, and recover from errors:** Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

10 **Help and documentation:** Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Appendix E
Additional Materials

The e-book version of this thesis, as well as all additional materials like compiled- and source code version of the tool ProKEt and the entire comprehensive evaluation documents, are provided in a freely accessible online repository of the university of Würzburg:

http://nbn-resolving.de/urn:nbn:de:bvb:20-opus-106072

Usage of the proket.war (quick usage, less adaptability) Copy the proket.war to the webapps directory of your local tomcat. In the course of this work, tomcat version 7 was used. Start the tomcat server. Call the corresponding local URL, e.g. `http://localhost:8085/proket/`. Please not that the particular URL may vary, depending on your local installation—e.g., regarding the ports used by tomcat. For using the separate KBS UI .war files, one for each dialog type assessed in the expert evaluation and user study, follow the above steps and replace *proket* by the name of the respective .war file.

Setup of the ProKEt development project/sources (comprehensive setup, all possibilities for adaption) The sources enclosed on the DVD constitute a ready to use, functional maven programming project. This is setup in NetBeans IDE (7.1 at the time of this writing) in a straightforward manner: Netbeans → File → Open Project, then choose the proketSources folder from the DVD. Once the import/opening is completed, the project is ready to use by the menu Run → Clean and Build Project.

Please note, that the project requires various other dependencies from the university and d3web repositories. Therefore, it is required to operate in online mode (do NOT set the –offline tag for maven compilation). This can be configured in Netbeans by the menu Netbeans → Settings → Miscellaneous → Maven (Tab) → Global Execution Options. Once successfully built, the project can be used by executing Run → Run Project. Therefore, a locally working, functional Tomcat Server needs to be configured for the use with Netbeans. This can be configured via Tools → Servers → Right-click on Servers, enter Tomcat Server Data according to your local installation setup.

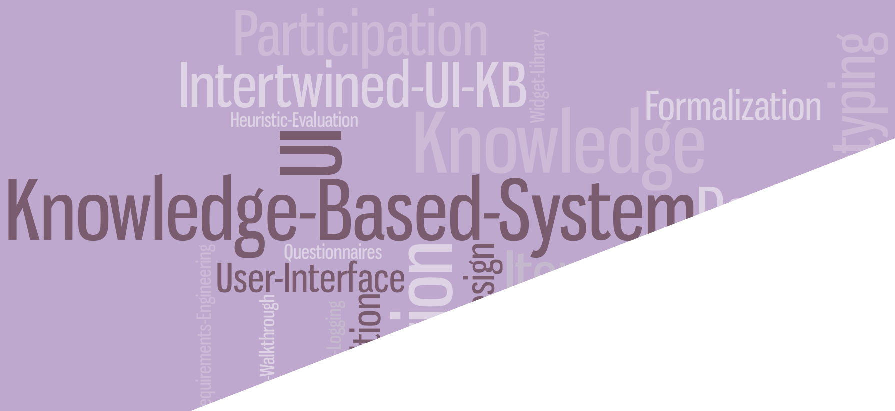
Please note, that at the time of this writing, ProKEt is a highly active, ongoing, development project, still under continuous refinement and extension. Thus, the ProKEt sources on this DVD mirror the development state at the submission date, i.e., June 2014. The latest sources can be retrieved from the university svn—therefore, please contact the department staff.

As a starting point, the ControlCenter can be used for accessing all KBS types currently implemented (or still under construction). The following demo systems, available under *d3web Dialogs*, are safe (i.e., functional) starting points, they comply to the *(06/2014) implementation state*:

- Box Questionnaire: StudyQuestionaryPlantsExpertsJusti, QuestionaryDebug_Statistics
- Daily Questionnaire: Study Daily Kneipe, DailyDebugStatistik
- Strict Interview: StudyInterviewStatistik, DefaultInterview
- ITree: StudyITreeKuend, StudyITreeUnfall

Hierarchical Interview so far is only available as (interactive) prototype, thus called from the ControlCenter via: *PurePrototypeDialogs* → StudyDecTreeStatistic.

Additionally, in the \proket\dialogs folder on the DVD, five compiled .war files are enclosed that contain the reference implementations of those five KBS UI patterns as used in the expert evaluation and comparative user study (*implementation state 04/2014*). Usage: see above, usage of the proket.war (just replace *proket* with the respective name of the .war).



Knowledge-based diagnosis or documentation systems are successfully applied in research projects and in industrial settings today. Often, their correct operation is critical—e.g., considering medical diagnosis. Thus, an intuitively usable UI that supports potential users whilst minimizing the chances of faulty operation is a key requirement. Still, current research mostly focusses on knowledge formalization and reasoning whereas UI and usability aspects remain neglected. This book deals particularly with those latter aspects by proposing amore encompassing development methodology. Based thereupon, a foundational collection of UI-centered patterns is specified. For practical support, further a tailored tool that supports the development paradigm and the patterns in a hands-on manner is provided.

ISBN 978-3-95826-012-2
9 783958 260122

Würzburg University Press