Evaluating the Usability of Interactive Verification Systems

Bernhard Beckert[1] and Sarah Grebing[2]

[1] Karlsruhe Institute of Technology (KIT)
beckert@kit.edu
[2] University of Koblenz-Landau
sarahgrebing@uni-koblenz.de

Abstract. Usability is an important criterion for measuring and comparing the quality of software systems. It is particularly important for interactive verification systems, which heavily rely on user support to find proofs and that require various complex user interactions.
In this paper, we present a questionnaire for evaluating interactive verification systems based on Green and Petre's Cognitive Dimensions. In a first case study, we have used this questionnaire to evaluate our own tool, the KeY System. The lessons learned from this evaluation relate (1) to the usability of the KeY System and interactive verification systems in general and also (2) gave us insights on how to perform usability evaluations for interactive verification systems.

1 Introduction

Overview For the acceptance, success, and widespread use of software its usability plays a central role. It is an important criterion for measuring and comparing the quality of software systems. And usability is particularly important for interactive verification systems, which heavily rely on user support to find proofs and that require various complex user interactions. However, measuring usability has so far not been the focus of developers of verification systems – instead the community generally concentrates on measuring performance of systems without evaluating to what degree usability effects a system's efficiency.

In general, there are a variety of methods for integrating usability in the development process. This includes analysis methods such as task and user analysis at the planning stage, testing methods where users are monitored while using a system (or a prototype), but also evaluation methods such as questionnaires and interviews [1, 2].

In this paper, we discuss and present a questionnaire for evaluating interactive verification systems based on Green and Petre's Cognitive Dimensions [3]. In a first case study, we have used this questionnaire to evaluate our own tool, the KeY System. The lessons learned from this evaluation relate (1) to the usability of the KeY System and interactive verification systems in general and also (2) to insights on how to perform usability evaluations for interactive verification systems, where such an evaluation can form a basis for improving usability.

Though usability should be taken into account from the beginning of the software life cycle, in this work we concentrate on evaluating the usability of an already existing tool.

Related Work As said above, there is not a lot of work reported on evaluating usability of verification systems or deduction systems in general. A noteworthy exception is Kadoda et al.'s list of desirable features for educational theorem provers [4], which resulted from an evaluation of proof systems based on a questionnaire using Green and Petre's Cognitive Dimensions' framework.

Griffioen and Huisman [5] present a comparison of PVS and Isabelle from a user's perspective. They propose to have something similar to consumer reports for reasoning tools, which can help users choosing the right tool for their application. In this work the two proof tools Isabelle and PVS are compared with respect to their logic, specification language, proof commands, strategies, and the availability of decision procedures, as well as system architecture, proof manager, and user interface. In addition, the user manuals and support tool is taken into account. At the end, the authors give a list of criteria on which the tools have been tested. The lessons we have learned from our evaluation mainly coincide with the aspects given by the authors.

Aitken and Melham analyze errors in interactive proof attempts [6]. They propose the error taxonomy by Zapf et al. as a usability metric. For this the authors have done two user studies with the interactive theorem provers Isabelle and HOL. Experienced users of both systems had to solve a task and the user's interactions were recorded. User errors where then categorized into three types: logical errors, interaction errors, and syntax errors. The authors draw conclusions about the usability of interactive theorem provers. Based on their evaluation, the authors provide some practical advice for the design of proof environments and the user interface of interactive theorem provers.

There are also attempts to improve the user interface of theorem provers, which plays a central role for usability – besides other factors such as the system's response to the user, the design of program logics and specification languages, system documentation etc. For example, Bertot and Théry [7] propose ways to build a user-friendly interface for theorem provers, which include concepts such as "proof-by-pointing", "script management" and "textual explanation of proofs". The user interfaces of particular deduction systems have been evaluated in a number of papers, for example, that of Isabelle [8], that of PVS [9], and that of KIV [10].

There are various competitions (such as CASC, SMT-COMP and SAT), where the performance of automated deduction systems is compared. In recent years, attempts to compare interactive verification systems have changed from comparing just the effectiveness (counting the number of problems that can be solved in arbitrary time) to the comparison of effectiveness and efficiency

(counting how many problems can be solved in limited time). Competitions of that form where held at VSTTE 2010[3] [11], VSTTE 2012[4], and FoVeOOS 2011[5].

Structure of this Paper This paper is structured as follows: Section 2 gives an overview of usability and its evaluation in general. Section 3 briefly describes the KeY System, which we used for our evaluation case study. In Section 4, we present our questionnaire and its design; and in Section 5 we discuss the recruitment of participants for the evaluation. Section 6 contains the results of the evaluation and the lessons learned on the usability of KeY, the usability of verification systems in general, and on the design of the questionnaire. Finally, in Section 7 we draw conclusions and discuss future work.

2 Usability of Software

Software usability is mostly investigated as part of research in the area of human-computer interaction. However, besides a well-designed user interface, other aspects of a system – e.g., good documentation – play an important role for usability.

According to ISO 9241 Part 11 [12], usability is defined as the "extent to which a product can be used by specified users to achieve specified goals with (i) effectiveness, (ii) efficiency, and (iii) satisfaction in a specified context of use." The standard also defines the three terms effectiveness, efficiency, and satisfaction:

Effectiveness: Accuracy and completeness with which users achieve specified goals.

Efficiency: Resources expended in relation to the accuracy and completeness with which users achieve goals

Satisfaction: Freedom from discomfort and positive attitudes towards the use of the product.

Usability is also often defined via five attributes (1) learnability, (2) efficiency, (3) user retention over time, (4) error rate, and (5) satisfaction. Depending on the system, these attributes differ in relevance and may also directly affect each other. For example, high efficiency often leads to reduced learnability (as, for example, key shortcuts need to be learned) [2].

There are standardized questionnaires and evaluation methods for usability that have been widely accepted. One such method is the Software Usability Measurement Inventory (SUMI)[6]. The SUMI questions are statements with which an interviewed user can "agree" or "disagree" (there is also an "undecided" option).

[3] `http://www.macs.hw.ac.uk/vstte10/Competition.html`
[4] `https://sites.google.com/site/vstte2012/compet`
[5] `http://foveoos2011.cost-ic0701.org/verification-competition`
[6] `http://sumi.ucc.ie`

Besides an introductory text about how to answer the statements, the SUMI questionnaire consists of a main part of 50 statements, which have the three possible answers already mentioned, and a smaller part addressing the interviewee's experience level and the importance of the software.

As SUMI is an established method, with a concise questionnaire, it is a low-effort method both for the evaluator and the interviewee. However, there is also a major disadvantage, which is that the feedback consists only of numbers. Therefore no information on how to improve usability in a particular area is gained. The result of evaluating a system with the SUMI method is a "score" reflecting how well the system performed in each of five dimensions (corresponding to the five attributes mentioned above), and in what dimension the system needs improvement.

Another important method in the area of software usability are the cognitive dimensions of notations, first described by Green and Petre [3] and modified into the Cognitive Dimensions framework proposed by Green and Blackwell. It provides a "practical usability tool for everyday analysts and designers" [13]. Rather than being an analytic method, the cognitive dimensions provide a vocabulary to discuss aspects that are cognitively relevant. The concept should of cognitive dimensions allows designers to evaluate their system, to a certain extent, by themselves, without the help of experts [13, 14].

Table 1 (taken from Green's tutorial on cognitive dimensions [13] with modifications from [3]) briefly summarizes the 14 Cognitive Dimensions of Information Artefacts.

3 Evaluation Target: The KeY System

The target for our usability-evaluation case study is the KeY Program Verification System [15, 16] (co-developed by the authors' research group at the Karlsruhe Institute of Technology, Germany and groups at TU Darmstadt, Germany and at Chalmers University, Gothenburg, Sweden).

The target language for verification in the KeY system is Java Card 2.2.1. Java 1.4 programs that respect the limitations of Java Card (no floats, no concurrency, no dynamic class loading) can be verified as well. Specifications are written using the Java Modeling Language (JML).

The program logic of KeY, called Java Card DL, is axiomatised in a *sequent calculus*. Those calculus rules that axiomatise program formulas define a symbolic execution engine for Java Card and so directly reflect the operational semantics. The calculus is written in a small domain-specific language called the *taclet* language [15] that was designed for concise description of rules. Taclets specify not merely the logical content of a rule, but also the context and pragmatics of its application. They can be efficiently compiled not only into the rule engine, but also into the automation heuristics and into the GUI. Depending on the configuration, the axiomatisation of Java Card in the KeY prover uses 1000–1300 taclets.

Table 1. Definition of the cognitive dimensions by Green and Petre [13, 3]

Cognitive Dimension	*Description*
Visibility and Juxtaposability	Visibility: ability to view components easily, respectively is every part of the code simultaneously visible (assuming a large enough display) Juxtaposability: ability to place/view any two components side by side
Error-proneness	Does the design of the notation induce 'careless mistakes'?
Abstraction	An abstraction is a class of entities or a grouping of elements to be treated as one entity, either to lower the viscosity or to make the notation more like the user's conceptual structure
Hidden dependencies	A hidden dependency is a relationship between two components such that one of them is dependent on the other, but that the dependency is not fully visible. In particular, the one-way pointer where A points to B but B does not contain a back-pointer to A
Premature commitment	Constraints on the order of doing things force the user to make a decision before the proper information is available
Secondary notation	Extra information carried by other means than the official syntax
Viscosity	Resistance to change; the cost of making small changes
Closeness of mapping	Closeness of representation to domain
Consistency	Similar semantics are expressed in similar syntactic forms, respectively when some of the language has been learnt, how much of the rest can be inferred?
Diffuseness	Verbosity of language, respectively how many symbols or graphic entities are required to express a meaning?
Hard mental operations	High demand on cognitive resources
Progressive evaluation	Work-to-date can be checked at any time
Provisionality	Degree of commitment to actions or marks
Role expressiveness	The purpose of a component (or an action or a symbol) is readily inferred

The KeY system is not merely a verification condition generator (VCG), but a theorem prover for program logic that combines a variety of automated reasoning techniques with interactive theorem proving. It employs a free-variable sequent calculus for first-order dynamic logic for Java.

While striving for a high degree of automation, the KeY prover features a user interface for presentation of proof states and rule application, aiming at a seamless integration of automated and interactive proving.

The KeY System's user interface consists of a window divided into two parts. On the left side of the window the proof tree is displayed showing the applied rules and case distinctions.

On the right side of the window shows the sequent currently in focus. KeY supports navigation of the proof tree and the application of rules through clicking on (sub-)formulas, and it offers comfortable interaction mechanisms such as drag-and-drop for quantifier instantiations. More on the user interface of the KeY System can be found in [17].

4 Constructing the Questionnaire

Concept of the Questionnaire The main idea behind the design of our questionnaire is to cover the cognitive dimensions [18, 19]. Our idea was to investigate – in general – whether these dimensions can serve as a method for evaluating interactive verification systems and – specifically – in which way the KeY System should be changed to improve its usability.

Table 1 shows the cognitive dimensions [13], and Table 2 shows examples of the corresponding questions from our questionnaire. As the table shows, our questions cover almost all cognitive dimensions except the dimension of *hidden dependencies*.

Note that some dimensions are covered by more than one question. These questions often differ in how specific they are for our chosen target. For example, a question for the dimension of *visibility and juxtaposability* that applies in general to interactive theorem provers is:

> How clear is the arrangement of the formulas in the open goal? Is it possible to determine where a formula results from?.

A question that also relates to *visibility and juxtaposability* but is more specific to the KeY System is:

> To handle formulas to be proven, the KeY System transforms it with normal form rules (e.g., arithmetic rules). Which normal form rules (the ones KeY applies during the automatic verification process) are most annoying or confusing when the automatic process stops and you have to continue interactively?

Specific questions may lead to more specific answers, which may be more useful for improving the system. Some questions we asked even mention suspected problems with usability and ask for a confirmation. On the other hand, such specific questions are often leading and presuppose certain answers. They also make it harder to compare different systems w.r.t. their usability. For that reason our questionnaire includes both general and specific questions. Some questions are half-way between general and specific, such as

> Your automatic proof stops with 1000 closed and 1 open goal. What are the first steps you do?

Table 2. Example questions from the questionnaire (the full version can be found at `http://userpages.uni-koblenz.de/~sarahgrebing/questionnaireForm.pdf`).

Cognitive Dimension	*Questions*
Visibility and Juxtaposability	How clear is the arrangement of the formulas in the open goal? Is it possible to determine where a formula results from?
Error-proneness	Do some kind of mistakes during the interactive verification process seem particularly common or easy to make?
Abstraction	Would you like to have user-defined abstract datatypes?
Premature commitment	Your automatic proof stops with 1000 closed and 1 open goal. What are the first steps you do?
Secondary notation	In JML it is possible to use comments for notes or explanations of the annotation. Would it be useful to have such a feature for proof nodes/subsequents/proof branches in KeY?
Viscosity	If you need to make a change to the previous work (proof, program, or annotation), how easy is it to make the change? Why?
Closeness of mapping	Does the JML notation or the dynamic logic notation allow you to express your intuition why a program is correct with respect to its annotation? Are there cases where the notation is not sufficient?
Consistency	Where there are different parts of the proof/open goal that have a similar meaning, is the similarity clear from the way they appear? Please give examples.
Diffuseness	Does the JML notation or dynamic logic notation (a) let you say what you want reasonably briefly, or is it (b) long-winded? Why?
Hard mental operations	Verifying programs using KeY, what proportion of the time (approximately) are you spending on: quantifier instantiation, finding the right invariant, ...
Progressive evaluation	How easy is it to stop in the middle of creating a proof and check your work so far? Can you find out how much progress you have made?
Provisionality	Other proof systems allow to sketch the proof at a more abstract/higher level (like Isabelle/HOL's tactics). Do you think it could be useful in KeY to sketch proofs if you have an idea how the proof might look like, without giving detailed interactive guidance? If yes, do you have an idea what such a functionality might look like?
Role expressiveness	Would you like to have labels at formulas that indicate the application of which rule the formula resulted from?

Besides instantiating the cognitive dimensions framework, we also included some questions taken from or inspired by the SUMI questionnaire. In addition, there are some questions aimed at gaining information about the interviewees, e.g., their experience level.

The structure of the questionnaire and examples for the questions are described in more detail in the following subsections.

Structure of the Questionnaire Our questionnaire[7] contains 48 questions in total, of which 44 are open questions. It has the following overall structure: after an introductory text describing the evaluation and its goals, we start with some general questions about the interviewee's experience with the KeY System. Next, we ask questions about performing proof tasks and then questions about the proof presentation. The next parts of the questionnaire cover the notation, in our case the JML notation, and the error messages provided by the system. In the last part of the questionnaire, we ask questions about usability in general, about experiences with other proof systems, and about some auxiliary information such as a contact address.

Questions Covering the Cognitive Dimensions Our questions related to cognitive dimensions are mostly instances of the questions in the "cognitive dimensions questionnaire optimized for users" [18, 19]. For example

> When *looking at an open goal*, is it easy to tell what each *sub-sequent* is for in the overall scheme? Why?

is an instance of

> When *reading the notation*, is it easy to tell what each *part* is for in the overall scheme? Why?

where we have instantiated the "notation" with the "open goal" and "part of notation" with "sub-sequent". Note, that such instantiations cannot be done uniformly for all questions. For example, the term "notation" may have to be instantiated with "open goal", "JML notation" or "proof presentation" in different contexts.

According to Kadoda's checklist [4], there are additional dimensions for verification systems, such as assistance (proof plan, next step) and meaningful error messages, which we covered as well. The dimension "assistance" we covered with the question "When the automatic proof process stops, is there enough information given on the screen to continue interactively or do you have to search (e.g., scroll on the screen or click onto the proof tree / search in the proof tree) for the appropriate information?".

SUMI Questions As already recorded, we included a few questions inspired by SUMI besides questions covering the cognitive dimensions. In particular, some of the more general questions resulted from turning the SUMI question into an open question. For example, the SUMI question (question number 2)

[7] There is an online and an offline version of the questionnaire. The offline version can be downloaded at `http://userpages.uni-koblenz.de/~sarahgrebing/questionnaireForm.pdf`.

I would recommend this software to my colleagues.

was turned into the open question

I would recommend the KeY System to people who ...

SUMI also has a few open questions such as "What do you think is the best aspect of this software, and why?" or "What do you think needs most improvement, and why?". The first question we divided into two questions: "List the three most positive or most helpful aspects of the KeY System for the interactive verification process" and "List the three most negative or annoying aspects of KeY concerning the interactive verification process".

Questions Regarding the Interviewee's Experience Level Different user groups with different experience levels have different needs w.r.t. a system's usability. It is therefore important to get information on the interviewee's experience level and how it relates to their answers.

The questions concerning the interviewee's experience with the KeY System included different ratings of the experience level. The interviewees had to choose one of the experience levels (a) little experience, (b) average, (c) above average, and (d) expert (these levels are defined in the questionnaire in more detail). They had to name the largest and most complex project they had verified using the KeY System. They also had to state since when they have been using the KeY System.

5 Recruiting the Participants

To recruit the participants, we asked 25 KeY users either personally or in personalised emails to participate in our evaluation. We got responses from 17 participants. Their experience levels were almost evenly distributed between "average", "above average" and "expert". We had one participant with "little experience". Many (but not all) participants had some relationship with the KeY project. The time they had been using KeY ranged from the early beginnings of the KeY System's development (around 1999) to a few months during a university course.

For a usability evaluation, this is a small sample of interviewees. But this sample gave us some interesting and useful insights both into the usability of KeY and the design of usability questionnaires (as explained in the following sections), and we feel that this is a sufficient sample size for a first evaluation.

For many interactive verification systems, which are developed in academia, recruiting a larger number of participants is difficult. Even more so, if one wants to recruit participants that know different systems and are able to compare them, or if one wants to only recruit participants that have no relation to the institution were the system is being developed (e.g., are enrolled as students) and are thus more likely to give unbiased responses.

For Kadoda's paper [20] about the differences between designers and users of theorem proving assistants, a questionnaire was sent to 27 interviewees, i.e., a

sample size similar to ours. The case study identified the gap between different system views of the designers on the one hand and the users on the other hand. It also highlighted that the cognitive dimensions have an effect on these differences. However, due to the small sample size, Kadoda found it hard to identify particular areas where system designers have to take special care in order to build a usable system.

6 Lessons Learned

6.1 Lessons Learned About Features that are Important for Usability

Many questions we asked in the questionnaire are posed in a way specific to the KeY System. Nevertheless, many of the answers we got and the lessons learned from the survey apply just as well to interactive verification systems in general. In the following these lessons are discussed in more detail.

Proof Presentation: A Major Point for Improvement Almost all participants agreed that the presentation of (partial) proofs – and the formulas and sequences of which proofs consist – is central to the usability of KeY. It is a time-consuming task to inspect the sequences and to reconstruct the structure of the proof by tracing the origin of each subsequence and formula. Thus, a (further) improvement of KeY in this area, which relates to the cognitive dimension of *visibility and juxtaposability*, should take high priority when usability is to be increased.

Documentation: Not Even the Specialists Know Everything Another important area where the usability of KeY could be improved is documentation. About 50% of the participants mentioned a lack of documentation in at least one of their answers. However, it is not one particular part or feature of the system that seems to be in particular need of better documentation, but different participants with different experience voice a wish for different areas where the documentation should be improved: the proof-search strategy settings, the proof rules, the annotation language, and various other system features.

KeY is a rather long-running project and various documentation exists, including a book [15]. This may be the reason why only very few participants asked for documentation in general or for a manual to be written. But the answers show that good documentation is essential. Even highly experienced users and members of the development team asked for certain aspects of the documentation to be improved, which shows that even specialists cannot be expected to know everything about a tool without referring to documentation.

Proof and Change Management: For a Better Overview of What to Prove A good proof and change management contributes to usability as well. This relates to the cognitive dimensions of *viscosity* and *hidden dependencies*. We asked the question of how easy it is to make a change to previous work (proof, program, or annotation). In the KeY System, changing the program to be verified or its

annotation is a simple task. However, if the proofs contain interactive steps, it is time consuming work to redo the proofs which are affected by changes. There is some functionality in KeY for replaying proofs automatically [21, 22], but responses to our questionnaire show that users would like to have more support in this area.

Additional Annotation Mechanisms: A Possibly Helpful Mechanism for Inexperienced Users In the questionnaire we asked about a possible extension of KeY that allows to the addition of comments to nodes in proof trees:

> In JML it is possible to use comments for notes or explanations of the annotation. Would it be useful to have such a feature for proof nodes/subsequents/proof branches in KeY?

This relates to *secondary notation* in the cognitive dimensions. The range of answers to this question shows that users have mixed feelings. The positive answers emphasised that such a feature may be particularly helpful for inexperienced users. Some participants also suggested that proof annotations may be added automatically by the system. That, however, would go beyond the dimension of *secondary notation* and also relate to better *visibility*.

6.2 Lessons Learned About How Users Define Usability of Interactive Verification Systems

We asked the participants what usability of an interactive verification system means for them. The answers we got were manifold, but they mostly supported our assumption that the cognitive dimensions framework is a good model and provides the right concepts for evaluating usability of interactive verification systems.

Areas that were mentioned frequently as being of particular importance for usability are related to proof presentation and the cognitive dimension of *visibility and juxtaposability*. Some typical answers are: "understandable/easy to read presentation of (partial) proofs/open goal, if the verification has to be continued interactively" and "easy location of proof branches".

Another important area is proof guidance as the following answers suggest: "good feedback and guidance when proof attempt fails" and "suggestions on how to proceed".

Documentation of the tool and its rules as well as easy tool usage (with less theoretical background) have been mentioned as features that contribute to the usability for interactive verification systems. The dimension of *viscosity* was also mentioned (e.g., "proof management, what remains to do").

Finally, there were answers relating to the performance of the system and to the user interface in general (e.g., "easy to use interface" and "interaction via mouse and keyboard").

All in all, the answers covered our expectations for features that contribute to the cognitive dimensions and, thus the usability of interactive verification systems:

– proof presentation,
– documentation,
– change management,
– proof guidance,
– feedback mechanism,
– quality of the user interface,
– good performance of the automatic proof search.

6.3 Lessons Learned About How to Evaluate Usability of Interactive Verification Systems

Lessons on How to Pose Questions

Open vs. Closed Questions One important lesson we have learned is that it is not a good idea to have too many open questions as they take more time to answer. The response rate goes down with too many open questions and the answers tend to get shorter and less useful. Also, answers to open questions are harder to evaluate. One should therefore carefully balance open and closed questions and use closed questions where possible.

Clarity of Questions is Important The cognitive dimensions as vocabulary for the usability of a system are a good starting point. But – not surprisingly – one has to carefully think about how to instantiate the concepts. For example, the question

> Where there are different parts of the proof/open goal that have a similar meaning, is the similarity clear from the way they appear? Please give examples.

is an instance of a question from the cognitive dimensions questionnaire. But with this question we gained almost no information because participants did not know how to interpret "similar meaning". The problem seems to be that in the world of formal methods, "meaning" is identified with "semantics". And to a formal-methods person, it is clear that similar semantics may or may not be indicated by appearance (syntax). On the other hand, "meaning" in the above question could also be interpreted as "purpose". So participants got confused.

Another example for a question that was often misunderstood is

> If the performance of the automatic proof process would be improved, would that lead to a better productivity? Or wouldn't it help, because most time is spent elsewhere during the proof process?

More than half of the users interpreted the "performance" as referring to effectiveness, i.e., which problems can be solved at all, and not as referring to efficiency (computation time), which was our intended meaning.

Rating Participants' Experience Level To interpret participant's answers, it is very important to know if there are different classes of users and to what class a participant belongs. A typical example for such classes are users with different experience levels.

We asked the participants to (subjectively) rate their own experience level and we asked for objective measures, namely the size of the biggest project they did with KeY and for how long they have been using KeY.

It turns out that users are not good at judging their experience level. Some who had just used the system in a lab course for two months rated their level to be "average", while some who had been working frequently with the system for years rated themselves as "average" as well. Thus, asking for a self-rating only makes sense if the various levels are well-defined (such as "you are an experienced user if . . . ").

Improving Usability vs. Measuring Usability One major drawback of the cognitive dimensions is the fact that there is no measure or score for usability with which one can compare different systems. On the other hand, a SUMI evaluation provides a score but does not provide detailed feedback on how to improve usability. So, in follow-up evaluations, we plan to use a better combination of both kinds of questions to get both a score and some feedback on how to improve the system.

Designing Questionnaires for Evaluating Other Systems? It is rather difficult to design a questionnaire that can be used to evaluate arbitrary verification systems. In particular, questions that relate to the cognitive dimensions depend on the system and on the information the systems designers want to gain.

The authors of the "cognitive dimensions questionnaire for users" [18] propose to let interviewees instantiate each question for themselves and let them choose the issues to which they would like to draw attention. This strategy can work, but it requires motivated interviewees that are willing to think hard about what their answers are.

7 Conclusions and Future Work

Our evaluation provided some important insights and lessons both on the usability of the KeY System and interactive verification systems in general and on how to perform usability evaluations for interactive verification systems. Cognitive dimensions have turned out to be a useful basis. And our questionnaire worked very well in giving us the results and feedback we planned for this first evaluation of the KeY System's usability.

As a result, some new features are now being implemented in KeY to improve its usability, in particular a better traceability of formulas and sequents in (partial) proofs. We also investigate how to change the automatic application of simplification rules to improve KeY w.r.t. the cognitive dimensions of *diffuseness* and *hard mental operations*.

Based on our experience, we will improve our questionnaire such that participants can rate the system w.r.t. different dimensions and, thus, provide a measure of usability. In particular, we plan to use that questionnaire on a larger and uniform group of participants, namely students who have used KeY in a lab course.

Acknowledgements

We would like to thank all participants of this evaluation for spending their time to answer the questionnaire. Their help towards improving the usability of the KeY System is greatly appreciated.

References

1. Ferré, X., Juzgado, N.J., Windl, H., Constantine, L.L.: Usability basics for software developers. IEEE Software **18**(1) (2001) 22–29
2. Heinsen, S., ed.: Usability praktisch umsetzen: Handbuch für Software, Web, Mobile Devices und andere interaktive Produkte. Hanser, München (2003)
3. Green, T.R., Petre, M.: Usability analysis of visual programming environments: A 'cognitive dimensions' framework. Journal of Visual Languages and Computing **7** (1996) 131–174
4. Kadoda, G., Stone, R., Diaper, D.: Desirable features of educational theorem provers: A Cognitive Dimensions viewpoint. In: Proceedings of the 11th Annual Workshop of the Psychology of Programming Interest Group. (1996)
5. Griffioen, W.O.D., Huisman, M.: A comparison of PVS and Isabelle/HOL. In Grundy, J., Newey, M.C., eds.: Theorem Proving in Higher Order Logics, 11th International Conference, TPHOLs'98, Canberra, Australia, September 27 - October 1, 1998, Proceedings. LNCS 1479, Springer (1998) 123–142
6. Aitken, J.S., Melham, T.F.: An analysis of errors in interactive proof attempts. Interacting with Computers **12**(6) (2000) 565–586
7. Bertot, Y., Théry, L.: A generic approach to building user interfaces for theorem provers. Journal of Symbolic Computation **25**(2) (1998) 161–194
8. Gast, H.: Engineering the prover interface. In: Proceedings of the User Interfaces for Theorem Provers Workshop (UITP 2010). (2010)
9. Owre, S.: A brief overview of the PVS user interface. In: 8th International Workshop User Interfaces for Theorem Provers (UITP'08), Montreal, Canada (August 2008) Available at `http://www.ags.uni-sb.de/~omega/workshops/UITP08/UITP08-proceedings.pdf`.
10. Haneberg, D., Bäumler, S., Balser, M., Grandy, H., Ortmeier, F., Reif, W., Schellhorn, G., Schmitt, J., Stenzel, K.: The user interface of the KIV verification system: A system description. In: Proceedings of the User Interfaces for Theorem Provers Workshop (UITP 2005). (2005)
11. Klebanov, V., Müller, P., Shankar, N., Leavens, G.T., Wüstholz, V., Alkassar, E., Arthan, R., Bronish, D., Chapman, R., Cohen, E., Hillebrand, M.A., Jacobs, B., Leino, K.R.M., Monahan, R., Piessens, F., Polikarpova, N., Ridge, T., Smans, J., Tobies, S., Tuerk, T., Ulbrich, M., Weiß, B.: The 1st verified software competition: Experience report. In Butler, M., Schulte, W., eds.: FM 2011: Formal Methods - 17th International Symposium on Formal Methods, Limerick, Ireland, June 20-24, 2011. Proceedings. LNCSS 6664, Springer (2011) 154–168

12. ISO: ISO 9241-11:1998 Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability. (1998)
13. Green, T.R., Blackwell, A.: Cognitive dimensions of information artefacts: A tutorial. Technical report, BCS HCI Conference (1998) Available at `http://www.cl.cam.ac.uk/~afb21/CognitiveDimensions/CDtutorial.pdf`.
14. Blackwell, A., Green, T.R.: Notational systems – the cognitive dimensions of notations framework. In Carroll, J.M., ed.: HCI Models, Theories, and Frameworks: Toward a Multidisciplinary Science. Interactive Technologies. Morgan Kaufmann, San Francisco, CA, USA (2003) 103–134
15. Beckert, B., Hähnle, R., Schmitt, P.H., eds.: Verification of Object-Oriented Software: The KeY Approach. LNCS 4334. Springer-Verlag (2007)
16. Ahrendt, W., Beckert, B., Hähnle, R., Rümmer, P., Schmitt, P.H.: Verifying object-oriented programs with KeY: A tutorial. In de Boer, F., Bonsangue, M., Graf, S., de Roever, W., eds.: Revised Lectures, 5th International Symposium on Formal Methods for Components and Objects (FMCO 2006), Amsterdam, The Netherlands. LNCS 4709, Springer (2007)
17. Giese, M.: Taclets and the KeY prover. In Aspinall, D., Lüth, C., eds.: Proceedings, User Interfaces for Theorem Provers Workshop, UITP 2003. Volume 103-C of Electronic Notes in Theoretical Computer Science., Elsevier (2004) 67–79
18. Blackwell, A.F., Green, T.R.: A cognitive dimensions questionnaire optimized for users. In: Proceedings of the Twelfth Annual Meeting of the Psychology of Programming Interest Group (PPIG-12). Volume 1-2. (2000) 137–153
19. Blackwell, A., Green, T.R.: A cognitive dimensions questionnaire. Version 5.1.1. At `http://www.cl.cam.ac.uk/~afb21/CognitiveDimensions/CDquestionnaire.pdf` (February 2007)
20. Kadoda, G.: A Cognitive Dimensions view of the differences between designers and users of theorem proving assistants. In: Proceedings of the 12th Annual Workshop on Psychology of Programming (PPIG 12). (2000) Available at `http://ppig.org/papers/12th-kadoda.pdf`.
21. Beckert, B., Klebanov, V.: Proof reuse for deductive program verification. In Cuellar, J., Liu, Z., eds.: Proceedings, Software Engineering and Formal Methods (SEFM), Beijing, China, IEEE Press (2004)
22. Klebanov, V.: Proof reuse. In [15] (2007)