Formal Specification Comprehension

The Art of Reading and Writing Z

Andreas Bollin
Alpen-Adria Universität Klagenfurt
Universitätsstrasse 65-67
Klagenfurt, Austria
Andreas.Bollin@uni-klu.ac.at

Dominik Rauner-Reithmayer
Carinthia University of Applied Sciences
Primoschgasse 10
Klagenfurt, Austria
D.Rauner-Reithmayer@fh-kaernten.at

ABSTRACT

Formal Methods have been developed to provide systematic and rigorous techniques for software development. They found their place in document-driven development processes as well as in the agile world. However, reading, understanding and working with a formal specification still turns out to be a real challenge.

This paper tries to identify the underlying cause and argues that comprehensibility of a specification is one of the key factors. It presents some first findings of an extensive study investigating the readability of Z specifications and comes up with a set of recommendations in writing formal specifications so that the syntactic gap between the mathematics and the natural language requirements description can be bridged more easily.

Categories and Subject Descriptors

D.2.4 [**Software Engineering**]: Software/Program Verification—*Formal Methods*; K.3.2 [**Computer and Education Science Education**]: Computer Science Education

General Terms

Measurement, Experimentation, Human Factors

Keywords

Formal Specifications, Quality, Comprehensibility

1. INTRODUCTION

The emergence of software engineering as a new term in the discipline of Computer Science / Computing / Informatics created significant challenges for educators and trainers, at Universities as well as in industry. Embracing this new concept required a transition from a discipline of lone wolves and artistic heroes to a discipline of engineers focusing on product development in a planned process under constrained time, budget, and other resources but leading to a predictable quality.

One of the issues involved is the use of formal methods. We now know how to use formal notations to write specifications, use refinement calculus to gradually transform a specification into a correct implementation, and use Hoare or Dijkstra's logics to prove programs correct with the same degree of the rigor that we apply to mathematical theorems. But, none of these techniques is easy to use by ordinary practitioners to deal with real software projects, and working with formal methods still turns out to be a real challenge.

Formal specifications are not documents that are written once and usually they are not just written at the beginning of the software development process [3]. Time is needed to create a first, useful version and then a lot of effort (and revisions) to develop a specification that is close to the concepts the customers (and/or developers) have in mind. As further argued in the work of Bollin [4, p.10], comprehensibility is a key issue (among other quality attributes) when working on and with a formal specification. Already in 2002, van der Poll and Kotzé [15] note that "a specification constructed according to established design criteria could be more comprehensible to users, easier to enhance and thereby facilitate long-term maintenance [...]".

When we want to foster the integration between the formal methods and the software engineering communities, then a closer look at this quality attribute seems to be necessary. The assumption is that, by raising comprehensibility, one is also very likely raising acceptability. This, in turn, would also ease using and teaching formal methods.

With this in mind, the overall objective of the study presented in this paper is to find out whether the style in which we are writing down our specifications has an influence on the comprehensibility (in terms of correct understanding and time needed for a task) of the specification. For this, sample specifications following diverse guidelines from literature have been given to students and colleagues in the form of short tests and qualitative and quantitative data has been collected. Astonishingly, a first assessment of the result shows that not all recommendations to be found in literature or textbooks do hold, whereas others are strongly confirmed.

The paper is structured as follows: Sec. 2 introduces related work, Sec. 3 discusses the motivation and the key questions of this paper in all the details. Sec. 4 presents the setting and the results of the study, and Sec. 5 summarizes and discusses the findings. Finally, Sec. 6 concludes the work with a short recapitulation of the findings.

FormaliSE'14, June 3, 2014, Hyderabad, India
Copyright 2014 ACM 978-1-4503-2853-1/14/06...$15.00
http://dx.doi.org/10.1145/2593489.2593491

2. RELATED WORK

When dealing with the quality of specifications (from the perspective of forward engineering), the absence of defects is relevant. But seen from the perspective of reverse engineering or maintenance, other attributes count. The factors of readability and comprehensibility are important, too. Unfortunately, papers investigating comprehensibility or correctness of formal specifications are rare, and they are quite often not really based on empirical investigations.

In 1990, Gravell [9] proposed a number of principles for constructing a specification written in Z [14], stemming from personal observations. He focused mainly on making specifications more readable and came up with a number of preferable specification fragments and a lot of counter examples (which will be used later on in our study).

In 1997, Finney, Rennolls and Fedorce [8] then addressed the issue of comprehensibility of specifications again. In an empirical study they looked at the influence of comments, naming and structure of Z specifications, and they found incidences for an influence of comments on the scores obtained by their test groups. Already in 1997, they write that "it would be desirable to have available indices of comprehensibility" for formal specifications [8, p.11].

In another study, Vinter, Loomes and Kornbrot [16] demonstrate that even specifications contain errors (be it requirement errors or be it mathematical constructs that are used in a wrong or in an incomprehensible way). They show that the number of defects to be found in formal specifications strongly correlates with the number of implications used in predicates of a Z-specification. Moreover, it also correlates with the perceived complexity of the specification itself.

Finally, and motivated by the "Established Strategy" of Barden et al. [1], van der Poll and Kotzé [15] examine Z specifications as well as some design heuristics from software engineering, and propose ten design heuristics (on a schema level) to be used when writing formal specifications.

Apart from notational (and structural) concerns, there is also the rise of lightweight formal methods [11, 13] and, with it, there are environments and languages supporting the automated transformation and the graphical analysis of models (like the Overture Initiative [12] or the Alloy analyzer [10]). Nevertheless, even though large portions of specification text can be generated automatically, readability stays an issue, and it is important when the specification is to be kept up to date during the software development life-cycle or when it has to be corrected due to problems identified during animation.

3. COMPREHENDING SPECIFICATIONS

One of the key ideas of formal specifications is abstracting away from particulars of the implementation and writing down the details of the system in a very compact way. However, this density of expressing thoughts combined with size complexity and mathematical idioms becomes detrimental for comprehending specifications in later phases.

The complexity of size (which is closely related to problem inherent complexity) can be dealt with by decomposition and structuring techniques as provided by the different specification languages and as suggested by textbooks (e.g. [6, p.65] for Z), but for understanding the huge amount of text (and a lot of dependencies between different terms in the specification text) the help of tools [2] is needed.

Moving away from the structure level, we do not have so much support from specification languages. Some papers, as mentioned in Sec. 2, come up with examples or guidelines we can learn from, and either directly or indirectly they suggest how to write down our formal specification text. But, we do not know yet if the different variants of writing down things are just a matter of style or if they have an impact on errors, the comprehension time or the ease in comprehending. So, a first question arises: What is a comprehensible specification and does it differ from a "good" specification?

Gravel [9, p.12] tries to answer the question about a "good" specification by giving guidelines for supporting the comprehensibility of the terms used in the specification. We think that his set of guidelines is not enough to guarantee a "good" formal specification. Thus, we propose the following working definition:

Definition 1. *A* **good formal specification** *is a syntactically and semantically correct specification which enables a lossless mapping between all the concepts in/behind the specification and the mental model of the specified system. The mapping process should not be perceived as exhausting and it should be completed within reasonable time.*

This definition extends the common perception of what a good formal specification should be: complete, consistent, and adequate. And, needless to say, good specifications are subsets of comprehensible specifications. But, it has to be noted that comprehensibility is not just a "nice to have" add-on, it is an *essential* requirement for deciding about the semantic correctness of the specification itself.

In the following, let us assume that the specifications we are looking at are using the correct types and are at least syntactically and semantically correct (in the sense that they reflect the system thoroughly and at the most suitable level of abstraction). Then only one aspect is missing, namely the "quality" of the mapping process itself, and, therefore, the different aspects of comprehensibility.

4. THE STUDY

In this section we now try to address the issues raised above, and we will not only investigate the sense of style in reading and writing formal specification, but also consider which style (of writing) is less error prone when trying to comprehend some pieces of specification text later on.

The study, conducted during the winter term 2013, looked at more aspects than described here in this paper. But, due to space limitation, we now focus on the following two key questions to learn more about what a good specification according to Def. 1 has to look like:

KQ1 Do common guidelines support the correct understanding of a formal specification?

KQ2 Do common guidelines support an easier and faster understanding of a formal specification?

The first key question addresses the issue of a lossless mapping of the concepts in mind and the concepts described in the specification text. Here, the essential aspects are the correct use (and understanding) of mathematical idioms and symbols and logical operations. In our study we do make use of the Z specification language, and therefore we took a closer look at the following aspects:

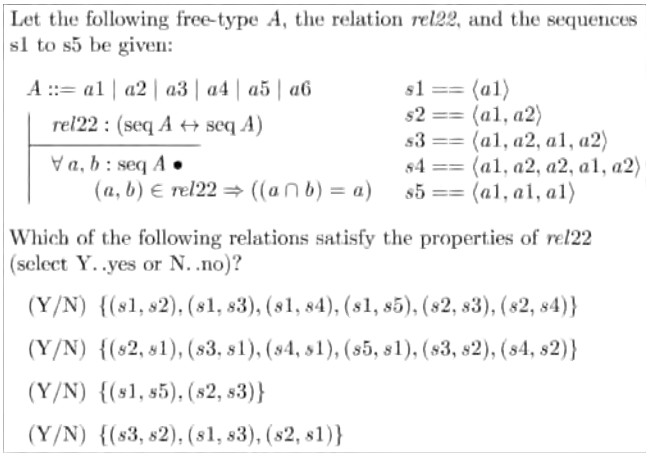


Figure 1: Question for checking skills in reading implications (translated from the German version).

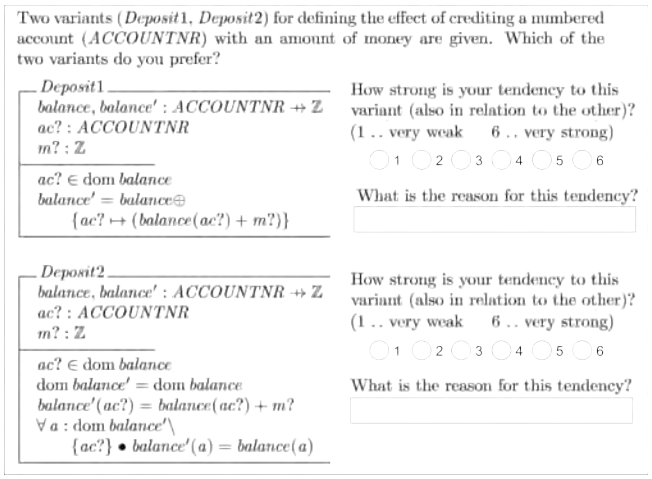


Figure 2: Question 16, according to [9, p.9] for checking preferences in specification style.

A1 Understandability of mathematical idioms (symbols in Z). Here, we focus on the relational override and the use of functions.

A2 Correct perception of the logical implication (following the observations of Vinter, Loomes and Kornbrot [16]). Here, we focus on "natural order" [9, p.4], logic equivalence and its use in orders that are not natural.

A3 Correct interpretation of incomplete operations[1].

The second key question (related to easier and faster understanding) is answered by looking at how developers perceive the comprehensibility of different specifications on a quantitative and qualitative scale. The basic assumption is that specification terms that are preferred are also those terms that will be used and more easily understood. We now take a closer look at the following aspects:

A4 Correctness of (a subset of) the recommendations of Gravell [9, p.12]. We look at the following guidelines:

- A41 Prefer clarity to brevity.
- A42 Choose the state so as to minimize the invariant.
- A43 Choose the state to simplify the description of the operations.
- A44 Give an implication its natural order, or avoid implications entirely.
- A45 Give names to important concepts.
- A46 Where the mathematical idiom is commonly understood, use it.

4.1 Study Setting

Our research questions address two different aspects:

1. Aspects, with the objective of identifying whether someone understood a specification correctly.
2. Aspects, with the objective of identifying the preferred "style" of a specification term.

[1]In Z, states can get partially undefined by using the Δ convention. More information can be found in the textbook of Diller [6].

Therefore, the study was split up into two on-line questionnaires implemented within the Moodle e-Learning platform at the Alpen-Adria Universität Klagenfurt. After a pre-test with experienced colleagues (and some minor corrections), the study started mid December 2013. The tests were conducted in a pleasant atmosphere with no time pressure for solving the tasks.

The first questionnaire contains 14 questions in multiple choice select form using a small specification and input data as the stem and a set of possible results as the select options. We have chosen this form to be able to figure out if a specification was understood and to test for the skills of the experimental subjects (see an example in Fig. 1).

In the second questionnaire (containing 24 tasks) we make use of examples introduced by [9]. In order to minimize the effort needed to understand the problem domain, every task was constructed according to the following layout:

1. Description of the example in natural language.
2. Specification of the example in Z.
3. A question for the participants to decide if the specification represents the described situation in a correct manner.

The layout of the Moodle-questionnaires allowed for breaks between the different tasks, but not for breaks during reading and answering the questions. For every task we also asked for the preferences (including some justifications) according to the different styles of specifications given. Fig. 2 shows an example with a six level Likert scale [7, p.29] for every specification.

The questionnaires were offered to two tutors and 30 students attending the course "Specification and Verification", which is an optional course in the Bachelors and Masters curriculum of Applied Informatics at the Alpen-Adria Universität Klagenfurt. The course (worth 6 European credit

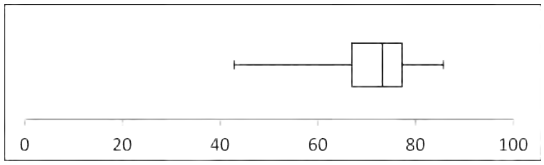


Figure 3: Achieved scores for questionnaire 1 (100 points maximum, n=25).

points) itself aims at writing and reading Z specifications, including first steps in refinement and proof. Since the survey took place in the last third of the course, we assumed that the students are – at least to a certain degree – familiar with the Z notation. Due to intermediate tests during the course, we also know that they are able to write their own Z specifications. The skills of the students were quite high. As Fig. 3 shows, the overall performance (which was also measured in the first questionnaire) is located in the upper range of the Box and Whisker chart, indicating that the skills of the experimental subjects are quite high. The Anderson-Darling test for normal distribution also shows normal distribution of the data.

About 83% (25) of the registered students took part in the survey. Among them, 24% (6) students have been in the Master program and 76% (19) students have been in the Bachelors program. The students did not have previous knowledge in Specification and Verification, but all of them passed previous courses on programming and Software Engineering, and they had at least 28 European credit points on Maths and Theoretical Computer Science.

4.2 Results

Due to limitations of space, this paper only presents some of the questions of the questionnaire. The full tests of the study are to be found on the web site of the ViZ project [5].

4.2.1 Correct Understanding

The first part of the study focused on the correct understanding of a specification. It took a closer look at the following three aspects.

- Aspect A1 (Understandability of mathematical idioms). In this part of the study 89% of the answers were correct.
- Aspect A2 (Correct perception of the logical implication). Here, 79% of the answers were correct. In this part of the study we tested three variants of an implication (single implication, equivalent logical form using negation[2], and an implication contained in another implication) exposing that a single implication and its logically written equivalent are understood in 83% and 82.5% of the cases respectively. Only the case of an implication contained in another implication caused difficulties (only 66% of the answers were correct).
- Aspect A3 (Correct interpretation of incomplete operations). Here, unlike the other tasks, only 63% of the answers are correct. One example (the specification of an Add operation schema using relational override) lead to problems as it was designed in such a way that the pre-condition did not cover all the situations.

[2]Negation means to resolve the implication by the following rewrite rule: $A \Rightarrow B \Leftrightarrow \neg A \vee B$

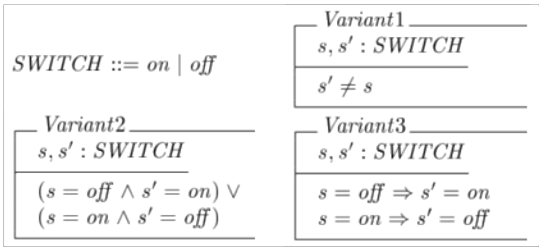


Figure 4: Examples for checking aspect A41.

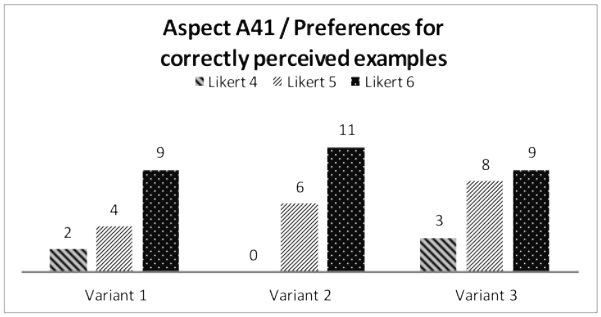


Figure 5: Preferences of those participants for aspect A41 when the specification was assumed to be correct. The bars show the absolute amount of the votes on the Likert scale.

This part of the study shows that the experimental subjects do know Z sufficiently well. Therefore, an interpretation of the results in the next part of the study makes sense.

4.2.2 Developers' Preferences

The second part of the study focused on the preferences a developer has when working with formal specifications. For this, 6 additional aspects were checked:

- Aspect A41 (Prefer clarity to brevity). This part of the study presented three variants (see Fig. 4) of the description of a switch. Fig. 5 shows that the majority tends to either variant 2 or 3 (which are the longer ones). Gravell argues that variant 2 is the least popular, which, in this study, cannot be reproduced.
- Aspect A42 (Choose the state so as to minimize the invariant). Here, we tested two versions of a state describing the collection of an item store, one with an

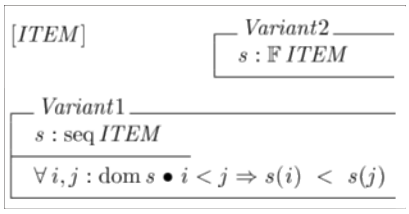


Figure 6: Examples for checking aspect A42.

$[CUSTID, BOOKID, DATE, NAME]$
$[ADDRESS, TITLE, AUTHOR]$

Variant1

$BookDB : BOOKID \pfun (AUTHOR \times TITLE)$
$CustDB : CUSTID \pfun (NAME \times ADDRESS)$
$LoanDB : BOOKID \pfun (CUSTID \times DATE)$

$\dom LoanDB \subseteq \dom BookDB$
$\forall c : CUSTID$
$\quad \mid (\exists b : BOOKID;\ d : DATE \bullet b \mapsto (c, d) \in LoanDB) \bullet$
$\quad c \in \dom CustDB$

Variant2

$author : BOOKID \pfun AUTHOR$
$title : BOOKID \pfun TITLE$
$name : CUSTID \pfun NAME$
$address : CUSTID \pfun ADDRESS$
$borrower : BOOKID \pfun CUSTID$
$due : BOOKID \pfun DATE$

$\dom borrower = \dom due \subseteq \dom author = \dom title$
$\ran borrower \subseteq \dom name = \dom address$

Figure 7: Examples for checking aspect A45.

explicit invariant, and one with an implicit one (see Fig. 6). Our participants preferred (as can be seen in Fig. 8) the first variant. However, in this case the qualitative assessment shows that the majority of the participants were not sure about the exact meaning of the finite set symbol ($\mathbb{F}$) used in this example.

- Aspect A43 (Choose the state to simplify the description of the operations). In this example the participants can choose between two semantically equivalent state descriptions, the first state (variant 1) needs an additional predicate to access elements of the state space, whereas in the second state the predicate is made explicit in the declaration part of the state. As can be seen in Fig. 8, the vast majority prefers the second variant - confirming this guideline of Gravell.

- Aspect A44 (Give an implication its natural order, or avoid implications entirely). Here, an implication is presented in three different variants:

  $Variant\ 1: \quad \forall i, j : \dom s \bullet i < j \Rightarrow s(i) < s(j)$

  $Variant\ 2: \quad \forall i, j : \dom s \bullet s(i) \geq s(j) \Rightarrow i \geq j$

  $Variant\ 3: \quad \forall i, j : \dom s \mid i < j \bullet s(i) < s(j)$

  Most of the participants preferred variant 1 which is the natural one, confirming another guideline of Gravell. However, Fig. 9 also shows that the participants would not avoid the implication entirely.

- Aspect A45 (Give names to important concepts). In this example (see Fig.7) there are two variants of the specification of a library system. The difference is in making some concepts of the specification explicit in the declaration part. As can be seen in Fig. 10, our participants do not have a clear tendency. The majority (total number of votes) tends to variant 1 (where concepts are not explicitly mentioned), whereas the strongest votes are those for variant 2.

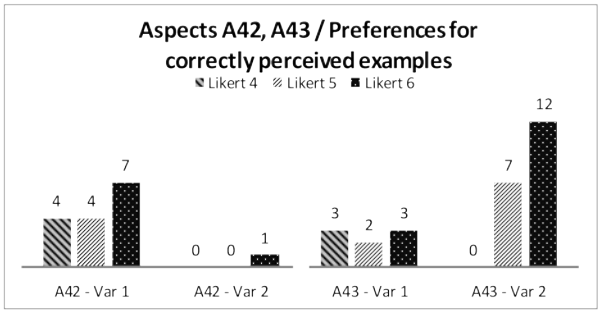


Figure 8: Preferences of those participants for aspects A42 and A43 when the specification was assumed to be correct. The bars show the absolute amount of the votes on the Likert scale.

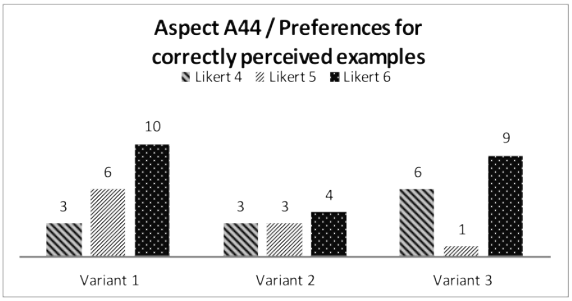


Figure 9: Preferences of those participants for aspect A44 when the specification was assumed to be correct. The bars show the absolute amount of the votes on the Likert scale.

- Aspect A46 (Where the mathematical idiom is commonly understood, use it). In variant 1 the relational override is used in an operation, whereas in variant 2 the override operator is made explicit. As can be seen in Fig. 10, the vast majority votes for version 1, confirming this guideline.

The idea behind this part of the study was, as a first step, to check for the validity of existing guidelines. In three cases (aspects A42, A43 and A46) the results of the study verify the guidelines, but in the other cases clear tendencies are not visible in the data.

4.2.3 Time and Skills

In our study we also tested for the time needed to complete the task of comprehending a specification and answering a question. We did this for two different settings: in one setting we kept the specification the same and varied the question according to different input data, in the other setting we kept the problem description the same, but varied the style of the specification. In both tests it turned out that the time for fulfilling the tasks decreased drastically. Fig. 11 shows the times for some of the tasks of questionnaire two. For example, one can see that the time needed for question A41 (we tested the guideline "prefer clarity to brevity") dropped from 34 seconds, over 18.5 seconds down to 14.5 seconds in the median, which is a speedup close to two from the first to the second variant of the specification. Such a speedup holds for all the other examples.

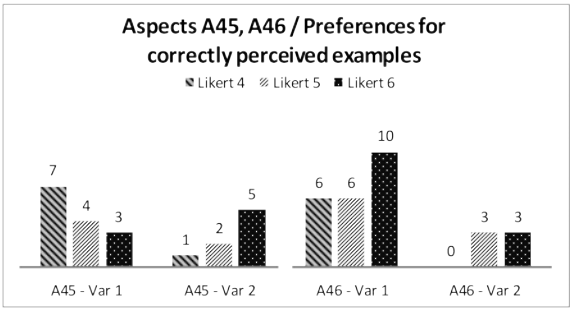
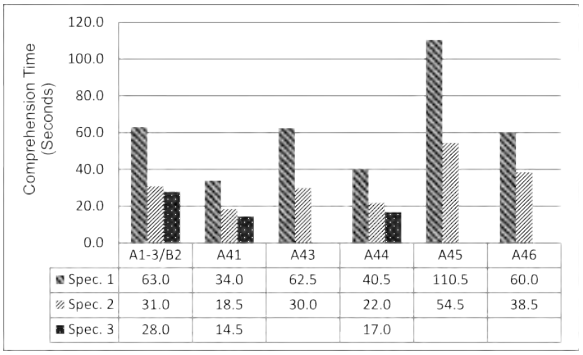


Figure 10: Preferences of those participants for aspects A45 and A46 when the specification was assumed to be correct. The bars show the absolute amount of the votes on the Likert scale.



	A1-3/B2	A41	A43	A44	A45	A46
Spec. 1	63.0	34.0	62.5	40.5	110.5	60.0
Spec. 2	31.0	18.5	30.0	22.0	54.5	38.5
Spec. 3	28.0	14.5		17.0		

Figure 11: Time (median) needed for completing some of the tasks of the questionnaires.

As mentioned above, it turned out that the skills of the participants are quite high. The overall results would have been even higher, when aspect A3 (incomplete description of operations) would not have been part of the test. We therefore looked for correlations between the time needed to fulfill a task and the correctness of the answers. The Anderson-Darling test shows normal distribution of both, time and points. We made use of three different statistical tests for looking for any correlations (Pearson, Spearman and Kendall), but were not able to find any statistically significant correlation between the data sets. E.g., the Kendall test tells us that there is, if at all, just a very weak positive correlation ($\rho = 0.228,\ p = 0.129$) between the time needed for the test and the results. Still ongoing research shows that the reason is the size of the specification, as with slightly larger specifications a correlation, as expected, can be confirmed.

5. DISCUSSION

The first part of the study (A1, A2 and A3) shows that specifications can be understood with some modest training (approx. 2 months in a University's Bachelor curriculum) in reading/writing in a formal specification language. But, the results of our study suggest that the findings of Vinter, Loomes and Kornbrot [16] do not necessarily hold. The semantics of implications is understood widely, and only nested logical implications lead to misunderstandings. We were surprised to see that the logical equivalent to an implication ($A \Rightarrow B \Leftrightarrow \neg\ A \vee B$) is also commonly understood.

Another interesting issue is related to aspect A3 which is a notational feature of Z (the Δ notation) and which allows for incomplete state transition descriptions. We think that here our participants were tempted to treat specifications like programs. There, different from Z, it holds that an identifier stays unchanged when it is not mentioned in a statement.

So, concerning key question one (which covers the first part of Def. 1), our study suggests that the set of existing guidelines does support the understanding of specifications. This statement is confirmed in questionnaire two, as those examples (in comparable cases) have been preferred which are making use of the guidelines. However, we have to admit that already due to the modest training in the classroom also the examples not adhering to the guidelines were understood correctly.

Not part of the guidelines, but an important finding of the study, is the following advice to developers of specifications: *when giving a specification of an operation, always make it total*. It raised comprehensibility a lot.

We think that we can trust the personal preferences as provided in the second questionnaire, as the skills of the participants are quite high. We cross-checked the results of questionnaire one with the final results in the lecture "Specification and Verification", and it turned out that there is a positive correlation ($\rho = 0.69,\ p = 0.001$).

Concerning aspect A41, our results differ from the guideline which tells us to prefer selecting the clearer predicate. In our opinion, a predicate is not comprehensible due to brevity or clarity. Sometimes these two objectives might even be diametrically opposed. Some clear solutions can be very long, but for the reader even more comprehensible. Maybe, longer solutions are sometimes even "more natural".

Taking a closer look at aspect A42, we have to admit that our findings are not necessarily valid. The reason is that the participants did not feel well with the use of the $\mathbb{F}$ symbol. So, we do not know if their tendency of avoiding the second version is due to the symbol or due to real preferences.

Aspects A43, A44 and A46 have been confirmed and do not need more discussion. However, aspect A45 needs a closer look as there we were not able to identify a strict tendency. In the qualitative part of the study we asked the participants for a justification of their choice. Taking a closer look at it, it turns out that there are two different types of participants: those preferring compactness without too much redundancy (e.g. it contained statements like "It is more compact", "This solution is less intimidating!"), and those who prefer names for all concepts (containing statements like "It's so easy to follow all the relationships!", "Set operations are easier to understand than quantifiers!").

So, concerning key questions two (which covers also the second part of Def. 1), we have to be more careful. As we were not able to find correlations between the time needed for solving tasks and the correctness of the answers, key question two can only be answered partially. From ongoing research we think that we are able to address the issue of an easier understanding of a specification, but with the data set at hand we are not able to talk about the issue of time.

According the ease of understanding, we need to consider both questionnaires again. We have chosen the layout of the study in such a way that those aspects (in questionnaire

two) related to preferences were checked in respect to their correct understandability in questionnaire one. As in the first questionnaire the vast majority of the participants understood nearly all variants of the specifications correctly, and as they preferred in most cases those according to the guidelines in questionnaire two, we dare to infer that almost all guidelines support easier understanding. However, there are some exceptions: brevity and clarity are two different aspects and they are not related to each other. Additionally, implications are understood quite well, so that their usage should not be avoided.

What remains is the question of how valid the results of the study are. Though the participants of the study are not working in the formal methods area. The test of their skills show that their choice of preferences is not random, they understood the concepts behind the different variants correctly and were (apart from one symbol) not distracted by an unusual style of the written text. The tested skills also go hand in hand with the results and grades in the classroom, so that we are quite certain that the answers are not incidental. On the other hand, up to now we are not able to say something about the time (effort) needed when trying to comprehend specification variants.

6. SUMMARY

The motivation behind this work was to find out whether the style we are writing down our specifications has an influence on the comprehensibility. For this, we took a look at common guidelines and checked them with two online surveys. Our overall objective is to come up with recommendations to foster the usage of formal methods in the software engineering community and to provide guidelines for improving teaching of this topic.

The study confirmed by large that common guidelines do support comprehensibility, but we also found out that not all of them are valid (at least in the setting that we used). The following guidelines seem to hold:

- Choose the state to simplify the description of the operations.
- Where the mathematical idiom is commonly understood, use it.
- Give an implication its natural order. However, avoiding implications entirely (as suggested) is not necessary.

The next three guidelines could not be confirmed totally:

- Prefer clarity to brevity.
- Choose the state so as to minimize the invariant.
- Give names to important concepts.

Additionally, by reflecting the results, we found another guideline:

- When giving a specification of an operation, always make it total.

Overall, this study is just a first step in a series of necessary investigations. As we still think that comprehension time and complexity are related, more tests, eventually with different study layouts, and complementary guidelines will have to follow.

7. REFERENCES

[1] R. Barden, S. Stepney, and D. Cooper. *Z in Practice*. Prentice-Hall, Inc., NJ, USA, 1995.

[2] A. Bollin. Concept Location in Formal Specifications. *Journal of Software Maintenance and Evolution – Research and Practice*, 20(2):77–105, 2008.

[3] A. Bollin. Is there evolution before birth? Deterioration effects of formal Z specifications. In *Proceedings of the 13th international conference on Formal methods and software engineering*, ICFEM'11, pages 66–81, Berlin, Heidelberg, 2011. Springer-Verlag.

[4] A. Bollin. Do You Speak Z? Formal Methods under the Perspective of a Cross-Cultural Adaptation Problem. In S. Gnesi and N. Plat, editors, *1st FME Workshop on Formal Methods in Software Engineering (FormaliSE)*, pages 8–14. IEEE, 2013.

[5] A. Bollin. ViZ – Visualize, Measure, and Comprehend. http://viz.uni-klu.ac.at, Page last visited: Jan. 2014.

[6] A. Diller. *Z - An Introduction to Formal Methods*. John Wiley and Sons, 1999.

[7] N. E. Fenton and S. L. Pfleeger. *Software Metrics: A Rigorous and Practical Approach*. PWS Publishing Co., Boston, MA, USA, 2nd edition, 1998.

[8] K. Finney, K. Rennolls, and A. Fedorec. Measuring the comprehensibility of z specifications. *J. Syst. Softw.*, 42(1):3–15, July 1998.

[9] A. M. Gravell. What is a Good Formal Specification? In *Proceedings of the Fifth Annual Z User Meeting on Z User Workshop*, pages 137–150, London, UK, 1991. Springer-Verlag.

[10] D. Jackson. *Software Abstractions - Logic, Language, and Analysis*. The MIT Press, Cambridge, Massachusetts, 2006.

[11] C. B. Jones, D. Jackson, and J. Wing. Formal Methods Light. *Computer*, 29(4):20–22, Apr. 1996.

[12] P. G. Larsen, N. Battle, M. Ferreira, J. Fitzgerald, K. Lausdahl, and M. Verhoef. The overture initiative integrating tools for vdm. *SIGSOFT Softw. Eng. Notes*, 35(1):1–6, Jan. 2010.

[13] P. G. Larsen, J. Fitzgerald, and S. Riddle. Learning by doing: Practical courses in lightweight formal methods using vdm+. Cs-tr-992, University of Newcastle upon Tyne, 2006.

[14] J. Spivey. *The Z Notation*. C.A.R. Hoare Series. Prentice Hall, 1989.

[15] J. A. van der Poll and P. Kotzé. What Design Heuristics May Enhance the Utility of a Formal Specification? In *Proceedings of the 2002 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on Enablement Through Technology*, SAICSIT '02, pages 179–194, 2002.

[16] R. Vinter, M. Loomes, and D. Kornbrot. Applying Software Metrics to Formal Specifications: A Cognitive Approach. In *5th International Symposium on Software Metrics*, pages 216–223, Bethesda, Maryland, 1998. IEEE Computer Society.