An Empirical Study of the Influence of OCL on Early Learners

Babak Khazaei
School of Computing
Sheffield Hallam University
Furnival Building
Sheffield, S1 1WB, United Kingdom.
B.Khazaei@shu.ac.uk;
Tel. +44 (0)114 225 6899
Fax. +44 (0)114 225 6830

Abstract

This paper reports on an empirical study aiming at investigating the influence of employing the Object Constraint Language (OCL) upon how early learners approach system development. OCL is an object oriented specification notation that is applied to Unified Modeling Language (UML) models. It is recognized as a valuable means of articulating design details beyond what is offered by UML itself.

Twenty six undergraduate second year Software Engineering students and ten postgraduate advanced software engineering students took part in this study as early learners of OCL. They were assigned to a relatively simple specification task. Two main solution types in the form of two models were suggested to the subjects as possible approach to the specification. In stage1 the solutions were informal and based on two models representing different data structures. In stage 2 the two models were represented in OCL. These models and solution types are briefly presented in this paper. The subjects were asked for their preferences for one against the other solution type in the two stages. A major shift in the solution preference between the two stages is observed and reported. More subjects opted for simple solution type rather than the generic solution type in stage 2 after considering the OCL representation.

The statistical results suggest that despite the formal representation increasing the awareness of the characteristics of a given specification problem, the notation appears to be detrimental in the consideration of good quality generic solutions. The different solution types that were used in the study are presented. A comparison to the data for the Z formalism is made and the implications for the choice of simple solutions are discussed.

Keywords: POP-II A. novices, B. formal specification, POP-III .D. specification languages, POP-IV A. simple vs. generic.

1. Introduction

Object Constraint Language (OCL) [2] is an object oriented specification notation that is applied to Unified Modeling Language (UML) models. It is recognized as a valuable means of articulating design details beyond what is offered by UML itself. There are very few empirical studies of use of OCL in the literature. The three notable exceptions are, firstly, L.C. Briand et al. experiment [3] which indicated that once past the initial learning curve of using OCL, the formal notation helps with the defect detection, comprehension and maintenance of a software development project. The second is Offutt et al. [4] who used OCL expressions to develop a tool to generate test cases and gathered experimental data on their tool. Finally Luis Reynoso et al. [5]who established a relationship between object coupling and understanding/modifying OCL expression. This work is closest to Luis Reynoso et al. in that it addresses the characteristics of OCL as a specification notation. However this work also differs as it is concerned specifically with the value and relevance of OCL to early learners. There are many reported studies of usability of notation both in Psychology of Programming Interest Group (PPIG) workshops[9] and Empirical Studies of Programmers (ESP) Conferences [8] for various programming languages and some formal specification language such as Z [14]. The usability of OCL however has yet to be scrutinized in the literature. OCL is as a practical specification notation that is commonly endorsed in software engineering education. At

Sheffield Hallam University we teach OCL as a formal specification language expressing invariants, pre- and post conditions and formal query expressions in relation to UML.

The work reported here builds upon earlier empirical work [1] that was carried out on the influence of formal specification on solution formulations. Our work has a lot in common with the approach such as Britton and Jones [7] as we are investigate how languages for Software Specification support understanding in early learners. In particular we are interested in different solution types to the same specification problem and the issue of subjects' considerations for generic versus simple solution types. In our experiments we provide two main solution types to our subjects and assign them a specification task and ask for their preferences for one against the other. We have used the same problem setting as that of our earlier work in Z formalism [1] but this time we have tailored our solution types to be presented in OCL. Section 2 describes the problem setting. The OCL specification is discussed in section 3. The changes needed from Z to OCL are not superficial and are explained in section 4. Section 5 reports on the experimental study and section 6 provides the discussion and conclusion section.

2. The Problem setting - The Video Horror Browser

The statement of the context of a problem labelled as "Video horror browser" is given as figure 1.

> Here is an extract of a specification for a browser for looking through a library of videos, where the currently selected video is displayed. There can be up to 100 videos in the library. Each video has a unique title and is given a unique horror rating. The higher the rating the more horrific the video. We want to specify an operation to return the position of any particular video in terms of its horror rating and another operation to return the next more horrific video.

Figure 1. Video horror browser

The context of the video horror browser is deliberately specific in order to limit the number of different interpretations and motivations that could possibly shape the problem solving activity. Two alternative sets of data structures are suggested for the Video horror browser. The two alternatives are labelled as model A-Rate and model B-Chart. The two models are provided to our subjects as the experimental material. Typical scenarios for these two models are represented in tabular formats in figures 2 and 3.

Video-to-Title =

Video 1	video 2	video 3	...	video48	Video49	video50
"Frankenstein"	"Psycho"	...	...	...	...	...

Video-to-Horror =

Video 1	video 2	Video 3	...	video48	Video49	video50
78	98	2	...	8	10	4

/*In the above example, each video has a unique title and horror rating*/

Figure 2. Model A- Rate

Video-to-Title =

Video 1	video 2	video 3	...	video48	Video49	video50
"Frankenstein"	"Psycho"	...	...	...	...	...

Horror-to-Video =

1	2	3	...	98	99	100
Video23	video 3	Blank	...	video 2	Blank	blank

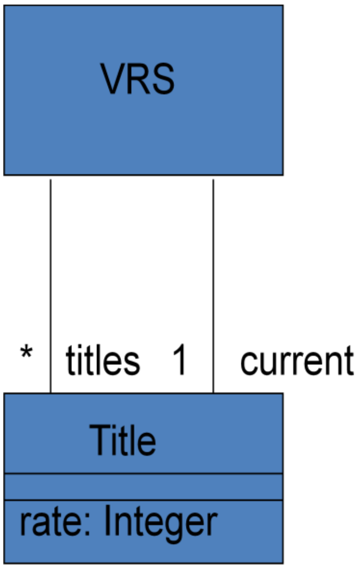
/*In the above example, each rating can hold one video which has a unique title. */

Figure 3. Model B- Chart

It can be argued that Model A-Rate is more logical and more "generic" as it has a common primary key between the two tables. Finding the "position" of any particular video title should not pose any difficulties in either of the models so the tendency to prefer the model A-Rate as more logical and generic can be expected. However, the Model B-Chart is simpler for returning the "next" horrific video operation (i.e. it is located next to its previous one and hence it is a 'simple' table look up). We have provided a fuller explanation and discussion of "generic" versus " simple" considerations in [1]. We will refer to these operations as " position " and "next" respectively in this paper.

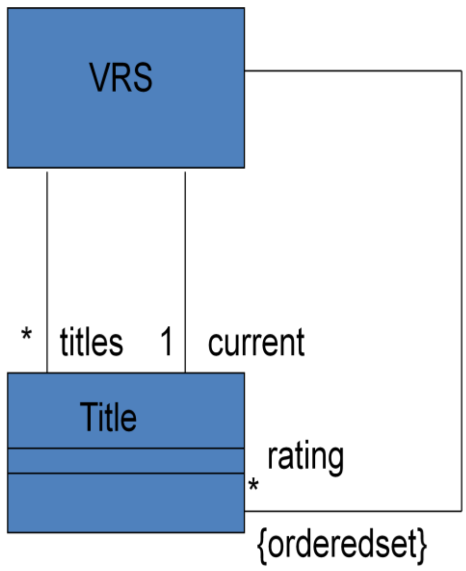
3. The OCL Specification for the Video-Horror Browser

Figures 4 and 5 represent OCL specifications of "next" operation based on the two models. The "position" operation is not very different in the two models and is left out here for brevity. To specify any operation in OCL we need to refer to the UML class diagram and consider the necessary associations and/or attributes before expressing the necessary pre and post conditions. For the rate model A-Rate we need the 'rate' attribute of 'Title' class in our OCL expressions. In contrast for the chart we need to refer to the 'rating' depicted as an ordered set association in order to express the ordering of the videos. The expression for the preconditions for both of the models are comparable in terms of notational complexity and both pre condition express the same meaning that a more horrific title must exists in order for the next operation to be meaningful. The post condition expressions are quite different both in terms of notations (the Model B-Chart model seems to require fewer or simpler notations of OCL) and the meaning (the model A-Rate expresses that the next horrific title should have a rating higher than current video but not higher than any other more horrific title, whilst the Model B-Chart simply states that it should be in the next location in the chart).



context VRS::next(): Title
pre: titles->exists(t| t.rate > current.rate)
post: result = titles->select(t| t.rate > current.rate and
titles->forall(t1| t1.rate>current.rate and t1<>t implies
t1.rate > t.rate)

Figure 4. The model A-Rate and the OCL pre and post expressions for "next" operation



context VRS::next(): Title
pre: rating.indexOf(current) <> rating->size()
post: result = rating.at(rating.indexOf(current)+1)

Figure 5. The model B-Chart and the OCL pre and post expressions for "next" operation

In terms of OCL expression and its meaning it can be argued that the notation is more complex in the model A-Rate. In particular a quick inspection of the post condition for the two models and using cognitive dimension of notation [13] we can see post condition for model A is more verbose and less expressive. A fuller evaluation in terms of cognitive dimension would be useful but it is not part of this particular paper. We investigate these influences in our subjects' preferences in working with one or the other models both in the informal and formal OCL stages.

4. A Comparison to the specification in Z formalism

Figures 6, 7 represent Z specification for the next operation. These were used as the experimental material in our earlier Z study [1].

```
[ID]

VIDEOLIBA ─────────────────────────────
│ lib : ℙ ID
│ current : ID
│ horror_rating : ID ⇸ ℕ
├──────────────────────────────
│ current ∈ lib
│ dom horror_rating = lib
│ ∀ i1,i2 : ID | i1,i2 ∈ lib • i1 ≠ i2 ⇒ horror_rating(i1) ≠ horror_rating(i2)
└──────────────────────────────
```

/* 'lib' is a set of (video) ids and 'current' represents the currently selected id. Every id in the library must have a horror rating and no two distinct ids have the same horror rating.

Figure 6. Model A-Rate in Z

In the two Z models the alternative "direction of mapping" for both models A and B are maintained: Model A is treated as a partial function from videos to a chart number; and model B is treated as a sequence of videos that is formally equivalent to particular function from numbers to videos.

```
[ID]
VIDEOLIBB——————————————————————————
| lib: ℙID
| current:ID
| horror_chart: seq ID
|————————————————————————
| current∈ lib
| ran horror_chart=lib
|  ∀ i,j:ℕ|i,j ∈ dom horror_chart • i ≠ j ⇒ horror_chart(i) ≠ horror_chart(j)
|——————————————————————————————
```

/* 'lib' is a set of (video) ids and 'current' represents the currently selected id. Every id in the library must have a rating in horror-chart and no two entries in horror-chart are for the same (video) id.

Figure 7. Model B-Chart in Z

A close look and comparison of Z solution to that of OCL indicates that there are some similarities and there are fundamental differences in the two formal representations:

The hearts of the solutions for the two Z-models are the same as the two OCL models. This is expected as they both need to be following the informal models as determining the next horror rating for model A and retrieving the video ID of the next horror rating in model B are again the main tasks. Both OCL and Z formalisms are specifications which are based on the predicate logic. However, there are fundamental differences between OCL and Z and as a result it is not just a surface notational issue or a predicate logic issues that are being investigated here. The formulation of class diagrams alongside OCL expressions is to provide an equivalent set of materials to that of the Z. Traditionally Z does not represent the object-oriented approach. Z is relatively detached from data flow and entity relationship diagrams in that it is often written without reference to data flows or entity relationship diagrams. In contrast OCL has to refer to the class diagrams to be of any meaning. We felt that there was a mind set change when we moved from writing our Z specification to that of OCL. We need to consider consistency with a diagram (in this class diagram) as well as object-oriented features. This will have a bearing in our discussion of our results and its comparison to Z. Again using the two cognitive dimensions of role expressiveness and verbosity we can see that Model A-Rate in Z is less expressive in its role and more verbose.

5 Report of The Experimental Study

The experiment conducted explored the approaches taken to specifying the video horror browser as given in section 2 in both informal (data structure) and formal (OCL) stages. This was to be compared with the result from the earlier experiment in Z.

5.1. Stage 1-Informal

5.1.1. The Hypotheses

a) for the operation "position" our subjects would prefer the model A-Rate; b) for the operation "next" our subjects would prefer Model B-Chart; and c) overall the subjects would choose Model B-Chart as it makes specifying both operations easier. The emphasis is on testing the hypothesis c) above.

Testing hypotheses a) and b) were therefore stepping stones before the subjects focused on an overall preferred model.

5.1.2. The Subjects and Setting

Twenty six second year B.Sc. Software Engineering students and ten postgraduate advanced M.Sc. students took part in our experiment. They were all volunteers and the number in each group reflected the size of the respective groups. All these students had successfully completed either two academic years of a computer science or software engineering programme or had completed a computing related B.Sc. degree. The undergraduate students were taking a software engineering module consisting of UML, formal specification in OCL and other relevant topics during an academic semester. The postgraduates had covered similar material and in particular identical OCL sessions with the same lecturer from which both groups had covered 10 hours of OCL and 20 hours of UML as part of their courses. For the purpose of this experiment all the subjects are considered as 'early learners' as they were relatively inexperienced when it came to OCL; other than what had just been taught in the module. Each subject was first provided with:

- a statement of the video horror browser (as in figure 1)
- an example diagram outlining a possible data structure for the model A-rate (as in figure 2)
- an example diagram outlining a possible data structure for the Model B-Chart (as in figure 3)

Each subject was asked to consider the suitability of each model with respect to each operation separately and both operations together (see figure 8).

1. If you were to specify the operation to return the rating for a given title, which model(A or B) would you prefer?
2. If you were to specify the operation to set the currently shown video to the next more horrific video in the library, which model (A or B) would you prefer?
3. If you were to specify both of the operations specified in questions 1 and 2 above, which model (A or B) would you prefer?
4. Please provide some justifications (3/4 sentences) for your preferred Model.

Figure 8. Preference Questions

5.1.3. Preference Results For Stage 1-Informal

The responses to questions 1-3 are summarised in tables 2a,b . Results for question 1 and 2 supported our hypothesis a) i.e. for the position operation more than 90% preferred the model A-Rate ; and for the next operation more than 80% preferred the Model B-Chart. The responses to question 3 show that the majority of subjects more than (60%) opted for the rate model A-Rate as their overall preference. The two subject groups were therefore in good agreement at this stage.

	question 1(position)	Question 2 (next)	question 3 (both)
Model A-Rate	24(99%)	4(2%)	16(62%)
Model B-Chart	2(1%)	22(98%)	10 (38%)

Table 2a. Preferences for the two models at stage 1- OCL undergraduate students

	question 1(position)	Question 2 (next)	question 3 (both)
Model A-Rate	9(90%)	2(20%)	6(60%)
Model B-Chart	1(10%)	8(80%)	4 (40 %)

Table 2b. Preferences for the two models at stage 1- OCL postgraduate students

5.2. Stage 2-OCL

The same 36 subjects were subsequently provided with the same instructions and the same statement of the Video-horror-rating as in stage 1 but were asked to consider the two OCL specifications referred to as "Model A" and "Model B" corresponding closely with the two models as was shown in figures 3 and 4 in section 2.

Table 3a,b provides the classification of answers to questions 1-3 at stage 2. Again expected results for both questions 1 and 2 were obtained supporting our hypothesis that subjects have preferred model B-Chart for retrieving the most horrific video.

	question 1(position)	question 2(next)	question 3 (both)
OCL Model A-Rate	17(65%)	8(31%)	11(42%)
OCL Model B-Chart	9(35%)	18(69%)	15(58%)

Table 3a. Preferences for the two models at stage 2- undergraduates

	question 1(position)	question 2(next)	question 3 (both)
OCL Model A-Rate	9(90%)	3(30%)	4(40%)
OCL Model B-Chart	1(10%)	7(70%)	6(60%)

Table 3b. Preferences for the two models at stage 2- post graduate

However, interestingly, for question 3 our subjects have clearly switched overall preference from model A to model B. One approach commonly used to assess the effect of the treatments is to perform a Mann-Whitney U nonparametric test (Chris 2004 [6]). In comparing the groups' OCL-treatment are significant ($p<0.05$, $U=260$ for undergraduate group; and $p<0.05$, and $U=504$ for postgraduate). The indications are that in stage 2 there are strong influences.

6. Comparison to Z results

A summary of results for the Z experiment are provided in table 4a, b below:

	question 1(position)	Question 2 (next)	question 3 (both)
Model A (rate)	70 (97%)	9(13%)	44(61%)
Model B (chart)	2(3%)	63(87%)	28(39%)

Table 4a. Preferences for the two models at stage 1

	question 1(position)	question 2(next)	question 3 (both)
Z Model A (rate)	45(88%)	5(10%)	10(20%)
Z Model B (chart)	6(12%)	46(90%)	41(80%)

Table 4b. Preferences for the two models at stage 2 in Z

Comparing the results between the two stages the model A has gone down from 61% to 20% and model B has gone up from 39% to 80%. Based on Mann-Whitney U nonparametric test the groups' OCL-treatment are highly significant ($p<0.05$, $U=1218$).

An interesting observation regarding the OCL experiment in contrast to that of the Z experiment was that the participants were finding both models A and B equally as challenging for the 'next' operation. The Follow up interview and closer inspection of justifications made by the subject as part of preference Q4 indicated that the subjects who had switched their choice had found both models challenging to operate with. In question 3 we were getting near to 50-50 where there was 80-20 (table 4c) in Z but for OCL only near 60-40 (table 4a and 4b). Nevertheless, in the OCL experiment the switch of subject preference from A to B was still occurring all the way through from informal to formal stages. In relation to the conclusion of our results it can be argued that the OCL notation has led the user to switch to a ‘Make it Easy’ strategy, however, with new insight these users were now less sure about either way being an easy option to choose. Clearly, the different cognitive tasks for OCL (as compared to Z) of not only writing first predicate logic but also representing appropriate attributes and relations on the class model has had an important effect. It can be argued that there is a bit more flexibility in OCL as the subjects needed to think to either change the class model and/or change the OCL constructs, in our scenario for instance create an ordered set relation rather than any complicated expressions for model B. A possible cognitive dimension evaluation of the two models in Z and OCL could shed some more light on this.

7.Discussion

We have a general interest in how notations that are powerful, and expressively adequate are not necessarily used as such. As suggested in the paper by Loom and Vinter R.[11] , we can see that formal methods are no cure for faulty reasoning amongst subjects and considerations for making a solution generic rather than just simple becomes of lesser priority.

In the results reported here it is important to note that the findings are the product of the *preferences* for models, and thus are not strongly influenced by the ability, or inability, of any subject to work with the specification representation or representation problem. In this sense, the results show an effect upon problem perception, related to formal specification representation, as opposed to the more committing the complex activity of actual working with a representation. Additionally, it is apparent that the representation is of limited benefit for non-expert users and, thus, achieving a suitable level of expertise and capability is inherently complex. We can not therefore make strong claims on comprehension or solution formulation of our early learner subjects of this experiment.

A common claim is that precise formal representations can increase the awareness of the characteristics of a given problem. In published formal the most valued characteristics of specific studies are the early development abstract frameworks and properties, as opposed to immediate solutions [15] and thus the solutions developed are supposed to be more thorough and hence ensure quality. The experimentation present both here and in our previous work shows that this claim cannot always be substantiated. This is because the usability of a notation can have strong influence on the manner in which it is employed. The factor our study substantiates is that the notation and formalisms in themselves doe not ensure the preference for generic solutions and can even draw attention away from more generic considerations.

The same data shows an influence that is not so favorable for notation novices whereby prior to the introduction of OCL, the subjects were satisfied with adopting a solution model that was logically clearer and more generic. However, the introduction of OCL seemed to have disrupted the process. As a result it can be concluded that if we were we to consider the suitability of the overall solution architecture, then the introduction of OCL has apparently discouraged them from adopting what is, on the whole, appropriate solution architecture. Accordingly, the effect on solution quality in this particular case has been negative. The subjects’ preferences were diverted away from software quality and towards formal representational needs, such as, whether to use a partial function or a sequence or an ordered set etc... In the face of the precision demanded by OCL, it appears that the overall problem has been approached in a more naïve manner. In terms of enabling non-expert notation users, the ideal would appear to be focusing on situations for which “Make it Generic” and “Make it Simple” are equally applicable. The results so far concur with the widening body of research in PPIG community that indicates that human factors have a significant role to play in notation use for both programming and software specifications.

Furthermore, through observations of both the experiment and the results there is a general consensus that the subjects experienced a change in mindset when switching from writing Z speciation to OCL.

The reasoning for this is however not inexplicable. The specification language Z is relatively detached from its complementary modelling techniques such as data flow and entity relationship diagrams whereby the user writes a Z specification without needing to directly refer to corresponding data flow or entity relationship diagrams. In contrast, OCL is part and parcel of UML diagrams such that use case diagrams, state diagrams and class diagrams must be referred to in order to understand OCL expressions. OCL, as its name suggests, has an attachment to the names of classes, attributes, methods and relationship of class diagrams, all of which are closely coupled with the artefacts of object-orientation and modern software development. These names are directly used as valid types in OCL expressions, this outlines a key constraint of OCL in that it is somewhat difficult to execute without the respective class diagrams.

In contrast, Z provides no direct cross referencing between particular diagrams and in their absence supports the introduction of user defined types. This, from a user's perspective, makes Z a more detached specification language that enables you to start with a blank sheet of paper for writing your specification. OCL on the other hand requires the user to keep an eye on several UML diagrams and constantly check for cross consistency between different diagrams in the model. As a result, from the user's point of view, this coupling of OCL expressions with UML diagrams creates another layer of difficulty for writing specifications whereby the tailoring class diagrams needs to be carried out alongside writing OCL expressions. Tools such as cognitive dimensions could be helpful in comparing the notational issues of Z and OCL. However, it is not just purely notational issue that we are addressing here as the difference between the two is conceptual as well as notational.

It is common that rich notations lead to very different results depending on unexpected factors, such as the specifics of exercises, the history of the subjects, or the visual detail of a notation for identical semantics. These issues are all worthy of empirical investigations, however we feel the experiments here is highlighting the difficulty of working with a notation that leads to opting for a simple solution.

Acknowledgements

1. Dr. Ali Hamei, Senior Lecturer at University of Brighton for help and advice on teaching of OCL and different solutions for the video browser specification.
2. All the students at Sheffield-Hallam University who took part in the experiment reported here.

A Note of Apology

It has been pointed out by one of the referees for this paper that the use of the word horrific in the scenario used in this study could be disturbing. On reflection perhaps we should have used a less disturbing word and used even a different attribute such as for example a comedy ranking for videos. This did not occur to me and no one had pointed to this amongst my helpers or students. I apologise for this.

References

[1] Khazaei B., Roast C.. The influence of formal representation on solution specification. Requirements Engineering, 2003.

[2] Object Management Group (OMG); Object Constraint Language OMG Available Specification Version 2.0, May 2006.

[3] Briand,L.C., Labiche Y., Dipenta M. & Yan-Bondoc H.D., " An Experimental Investigation of Formality in UML-Based Development", IEEE Trans Software Engineering, Vol31, no10,pp 833-849, October 2005.

[4] Offutt,A.J., Abdurazik, A., Generating Tests From UML Specifications, Proceedings of 2nd int. conference on the Unified Modelling Language (UML'99),Fort Collins, CO, pp416-429,1999.

[5] Reynoso, L., Genero, M., Piattini, M., and Manso, E. 2006. Does object coupling really affect the understanding and modifying of OCL expressions? In Proceedings of the 2006 ACM Symposium

on Applied Computing (Dijon, France, April 23 - 27, 2006). SAC '06. ACM, New York, NY, 1721-1727.

[6] Chris, D. 2004. *Calculating a nonparametric estimate and confidence interval using SAS software* Glaxo Wellcome Inc., Research Triangle Park, NC.

[7] Britton, C. & Jones, S., (1999), "The Untrained Eye: How Languages for Software Specification Support Understanding in Untrained Users", Human-Computer Interaction (14), pages 191-244.

[8] Empirical Studies of Programmers: http://www.interaction-design.org/references/conferences/empirical_studies_of_programmers_-_seventh_workshop.html, last accessed 12/August/2011.

[9] Psychology of Programming Interest Group: http://www.ppig.org/, last accessed 12/August/2011.

[10] Bowen, J. P. & M. Hinchey, (1995),"Ten Commandments of Formal Methods", IEEE Computer, Vol. 8, No 4, pages 56-63.

[11] Loomes M. & Vinter R., (1997), "Formal Methods: No Cure for Faulty Reasoning", In Proceedings of Safety Critical Systems Symposium in Brighton, pages 67-78, Springer-Verlag.

[12] Monk A.F., Curry M.B., & Wright P.C. ,(1994),"Why industry doesn't use the wonderful notations we researchers have given them to reason about their design", In User-centered requirements for software engineering , D. J. Gilmore, R.L. Winder & F. Detienne , pages185-188 , Springer-Verlag.

[13]Green, T. R. G. (1989) Cognitive dimensions of notations. In A. Sutcliffe and L. Macaulay (Eds.) *People and Computers V*. Cambridge University Press. 443-460.

[14] Khazaei B., and Triffitt E., (2002),"A Study of Usability of Z formalism Based on Cognitive Dimensions", 14th Annual Workshop of the Psychology of Programming Interest Group, 18-21 June, Brunel University, London, UK.

[15] Hinchey M .G. and Bowen J.P. "Applications of Formal Methods", Prentice Hall International Series in Computer Science, Prentice Hall, 1996 ISBN 0-13-366949-1.