Measuring the Comprehensibility of a UML-B Model and a B Model

Rozilawati Razali, and Paul W. Garratt

***Abstract*—**Software maintenance, which involves making enhancements, modifications and corrections to existing software systems, consumes more than half of developer time. Specification comprehensibility plays an important role in software maintenance as it permits the understanding of the system properties more easily and quickly. The use of formal notation such as B increases a specification's precision and consistency. However, the notation is regarded as being difficult to comprehend. Semi-formal notation such as the Unified Modelling Language (UML) is perceived as more accessible but it lacks formality. Perhaps by combining both notations could produce a specification that is not only accurate and consistent but also accessible to users. This paper presents an experiment conducted on a model that integrates the use of both UML and B notations, namely UML-B, versus a B model alone. The objective of the experiment was to evaluate the comprehensibility of a UML-B model compared to a traditional B model. The measurement used in the experiment focused on the efficiency in performing the comprehension tasks. The experiment employed a cross-over design and was conducted on forty-one subjects, including undergraduate and masters students. The results show that the notation used in the UML-B model is more comprehensible than the B model.

***Keywords*—**Model comprehensibility, formal and semi-formal notation, empirical assessment.

I. Introduction

SOFTWARE understandability is a characteristic of software quality, which means ease of understanding of software systems [1]. In Boehm's quality model for instance, understandability is considered an important aspect of software maintenance. Software maintenance in general accounts for the largest cost in the software lifecycle [2], where software understandability or comprehensibility plays a crucial and costly role in the software maintenance process [3].

Software specification is the fundamental software artefact as it captures what a system should do. It is the primary point of reference for engineers who need to deal with a system including the maintainers. Maintainers scrutinise specifications to understand not only the localised properties of a system that need to be changed but also the context within which the changes should take place. These tasks are not straightforward particularly when the specifications tend to be unreliable and when the notations used are not familiar to them or difficult to interpret. Maintenance in essence costs engineers time, effort and money. This requires that the maintenance process be as efficient as possible. Specification comprehensibility is necessary for an efficient maintenance process as it allows maintainers to understand the system properties quickly prior to the modification task.

The notation used in a specification plays a vital role in specification comprehensibility. The comprehension process and the subsequent tasks related to it will be affected if the people involved have to struggle in deciphering the notation used rather than concentrating on the specification's contents. It has been known that the use of formal notation in a specification increases its precision, which in turn enables greater consistency and correctness to be obtained, e.g. [4],[5]. Nevertheless, it has been a concern that the notation could also cause comprehension difficulties, e.g. [6]-[8]. The notation is seen as being so difficult to comprehend that highly trained engineers are required to employ them. In fact, it has been recommended that a formal method specialist should support the engineers in order to ensure that the notation is interpreted and employed correctly [9]. Moreover, the notation is seen as more usable for programmers rather than for the engineers who need to specify the system requirements [10]. It is believed that the widespread adoption of formal methods in industry will only happen when the formal notation is more accessible to a wide range of users.

The accessibility of visualisation techniques such as the use of graphical notation in software specification has been recognised for some time [11]-[13]. The graphical notation is considered as easy to grasp quickly as in some sense it is analogous to the world that it represents [14],[15]. On the other hand, the graphical notation is not as expressive as the textual notation since some aspects of system characteristics could not be specified completely using only diagrams [16]. Besides, the underlying factors that contribute to the superiority of graphical representation are not well understood [17]. Perhaps the combination of the graphical notation and the textual notation together in a specification could be a strategy to establish synergy between both notations.

Formal notation normally appears as textual whereas semi-formal notation mainly as graphical. The graphical representation in semi-formal notation is easy to understand and supports refinement activities. On the other hand, formal notation supports consistent and precise representation of system requirements. By integrating formal and semi-formal notations, practitioners could indeed benefit from both graphical and textual notations. One of the ideas towards this integration is to combine the formal notation used in a formal method, namely the B method [18], and the semi-formal notation used in the Unified Modelling Language (UML) [19].

Manuscript received August 31, 2006.

R. Razali is with the Dependable Systems and Software Engineering Group, School of Electronics and Computer Science, University of Southampton, UK (e-mail: rr04r@ecs.soton.ac.uk).

P. W. Garratt is with the Dependable Systems and Software Engineering Group, School of Electronics and Computer Science, University of Southampton, UK (e-mail: pwg@ecs.soton.ac.uk).

A method called the UML-B [20],[21] is one of the products of this integration. The rationale behind this integration is that the B method has strong industrial supporting tools such as Atelier-B [22], B-Toolkit [23] and Click'n'Prove [24], and the UML has become the de facto standard for system development [25].

This paper presents an experiment conducted on the UML-B method. The experiment aimed to evaluate the notation used in the UML-B method in order to know whether it could improve the specification or model comprehensibility. The evaluation was based on the comparison made between the notation used in the UML-B method and the notation used in the B method. The main objective of the experiment was to investigate whether the notation used in the UML-B method is more comprehensible than the notation used in the B method. The measurement used in the evaluation focused on the efficiency in performing the comprehension task, that is, accuracy over time.

II. Objectives

The main objective of this experiment was to evaluate the comprehensibility of a UML-B model compared to a traditional B model. The treatments of the experiment were therefore the UML-B model and the B model. A UML-B model comprises the subsets of semi-formal and formal notations used in the UML and the B method respectively. On the other hand, a B model comprises the formal notation used in the B method.

The experiment was conducted to confirm a theory that suggests the notation used in the UML-B method has a particular effect on the practitioners, making it better in some way than the notation used in the B method. In general, the experiment attempted to answer the following broad research questions:

Is a UML-model easier to understand than a B model for practitioners with limited hours of training?

The null hypothesis stated for this experiment was:

H_0 The UML-B model is as comprehensible as the B model

to be rejected in favour of the alternative hypothesis:

H_1 The UML-B model is more comprehensible than the B model

A one-sided alternative hypothesis was employed because the UML-B method can only be considered as worthwhile if it could overcome the current barriers against formal notation such as used in the B method. In other words, the UML-B model should be better than the B model. After all, this is the theory that the experiment aimed to confirm or refute by providing some empirical evidence.

III. Design

The experiment employed a related within-subject design where each of the subjects was trained and assigned a task on both models. Since there were two treatments to be tested in this experiment, the subjects were allocated randomly into two groups. To reduce subjects' variability across groups, they were firstly blocked based on their ability and previous knowledge on the object-oriented technology and formal methods, prior to the group allocation.

The experiment was designed in such a way that at one point of time, one group was assigned a task on the UML-B model while the other was assigned the same task on an equivalent B model. The reverse was then carried out later. This means there were two sessions in the experiment, which were run consecutively within one hour and forty minutes. The design which is called cross-over trial [26] was employed in order to eliminate any task direction bias and subsequently any ability effect.

IV. Variables

This experiment identified the notations used in the models as its independent variable. Since the experiment aimed to examine the effect of the notations on the efficiency in understanding the models, the identified dependent variables were:

Score: This variable is the mark obtained. Each question was given a specific allocation of marks. Since the questions were open-ended, the marking was based on specific keywords expected from the answers. Marks were awarded for the presence of these keywords and zero for otherwise. The questions had been carefully constructed so that the marks could be easily decided. One person did the marking so that there was consistency throughout the process.

Time Taken: This variable is the time taken to answer each question in minutes, excluding the time to read and understand the question.

The score was chosen as the measure of comprehension because it is believed that the subjects could only answer a question correctly only if they understand the object being evaluated. Although there is a possibility that the subjects could still give a correct answer based on wild guess or hunch, the chances of this are low. This is because the questions were constructed in such a way that the subjects could only answer the questions by reading and understanding the models. The time taken was decided to be the other measure because it is the most frequent measure of software engineers' effort in any software development, particularly maintenance [27]. One method is better than the other if it allows software engineers to do their tasks correctly in least possible time.

The quality aspect measured in this experiment was efficiency in understanding the models. This means a model is considered as more comprehensible than the other if it allows the subjects to answer the questions accurately in a minimum period of time. Therefore, the score and the time taken measures were used to determine another important measure namely rate of scoring. The rate of scoring was obtained by dividing the score by the time taken.

V. Subjects

There were forty-one students that participated in the experiment. This included twenty-seven third-year Undergraduate students and fourteen Master students of

Computer Science and Software Engineering courses at the University of Southampton, United Kingdom. They were students from various continents including Europe, Asia and Africa. The international students, who came from outside the United Kingdom constituted half of the subjects and the proportion of women to men was 1:4.

The subjects were students who registered for a course on the B method. The subjects were taught formally on the B method for about nine hours and on the UML-B method for one hour. All subjects had gone through courses on the object-oriented technology and formal methods at some points of their studies. The subjects therefore were familiar with all the methods used in this experiment but were not very experienced.

The subjects were aware that the experiment was intended for research purposes. Nevertheless, the subjects were motivated to participate because the level of understanding tested in the experiment was considered to be necessary for them to do their coursework and prepare for the examination.

VI. Materials and Procedure

Since the experiment had two treatments to be examined in each of the two sessions, four models that represented two separate case studies were developed. In the first session, one group was given a UML-B model and the other was given the equivalent B model on the first case and vice-versa on the second case in the subsequent session, as shown in the Table I below. Two separate case studies were needed in order to eliminate any learning effect caused by the problem domain. However, the models for the second session were made closely equivalent in terms of size and complexity to the first session so that the effect to be tested remained the same, that is, the treatment effect.

TABLE I
GROUP AND TASK ALLOCATION

	Group A	Group B
1st session Case 1	Tasks on UML-B model	Equivalent tasks on B model
2nd session Case 2	Tasks on B model	Equivalent tasks on UML-B model

Several indicators were employed in order to gauge the comprehension level of the subjects on the models. This included the interpretation of the symbols used in the models, the tracing of input and output, the mapping between models and problem domains, and finally, the modification of the models by adding new features.

There were five questions for each model. One question was constructed for each of the indicators above except for the mapping between models and problem domains. This particular indicator had two questions because it is critical for ensuring accuracy and completeness; a quality that is expected from any specification [28],[29]. The questionnaires on both models were similar except for the question on the symbols used in the models. This cannot be avoided as each model has its own unique symbols that are important for subjects to interpret in order to comprehend the models.

The importance of performing a pilot study before the execution of an experiment cannot be over emphasised. Performing a pilot study can mean the difference between a success and a failure of an empirical assessment [30],[31]. For this experiment, a pilot study with five participants had been conducted to validate and verify the accuracy of the materials prepared for the experiment. These included the clarity of the instruction, the validity and complexity of the questions and the practicality of the tasks required relative to the allocated time.

During the experiment, subjects were given a specific model and its questionnaire in each session. The instruction sheet was given at the beginning of the first session. The materials for the first session were collected after thirty-five minutes had passed and the materials for the second session were distributed right after. During this time, the subjects had a short wash out or break before starting the second case study. The materials for the second session were accompanied with an additional set of questions, where the subjects were asked about the models comprehensibility subjectively. An additional five minutes were allocated for this purpose. After the forty minutes had passed, the materials were collected whether or not the subjects had completed answering all the questions.

The subjects were not allowed to talk to each other and leave the room at any time until the experiment ended. The subjects were separated from each other as if in an examination session. During the tasks however, the subjects were allowed to refer to textbooks or notes. The subjects were also instructed to inform the researcher if they had any trouble in understanding the questions.

VII. Results and Analysis

The rate of scoring was the measure of interest in this experiment as it determines the model comprehensibility in terms of accuracy over time. The scale used for the rate of scoring was marks per minute (marks/min). This means a model with a higher rate of scoring is better than otherwise since it indicates a higher accuracy with least time taken to understand the model.

There were two types of comprehension measurement and analysis; *overall comprehension task* and *comprehension for modification task*. The measurement for overall comprehension task was obtained by consolidating the total score and the total time taken for all five questions. Meanwhile, the measurement for the modification task was obtained by considering the score and the time taken for the question on model modification only.

The Table II below illustrates the measures of centre and spread for the overall comprehension task distribution. Column *Min* shows the minimum values, column *1st Q* shows the first quartile values, column *Mean* shows the average values, column *Median* shows the middle values, column *3rd Q* shows the third quartile values, column *Max* shows the maximum values, column *Std Dev* shows the degree of variation, and column *N* gives the number of collected data. Rows *C1:U* and *C1:B* present the rate of scoring of UML-B model and B model respectively for the first case. Rows *C2:U* and *C2:B* present the rate of scoring of the respective models for the second case. The last two rows present the grouped rate of scoring based on the models used, regardless of the case.

TABLE II
RATE OF SCORING DISTRIBUTION FOR OVERALL COMPREHENSION TASK

	Min	1st Q	Mean	Median	3rd Q	Max	Std Dev	N
C1:U	0.13	0.59	0.74	0.70	1.00	1.33	0.33	21
C1:B	0.17	0.41	0.60	0.63	0.78	1.12	0.26	20
C2:U	0.28	0.68	0.76	0.75	0.86	1.14	0.19	20
C2:B	0.43	0.53	0.73	0.71	0.91	1.18	0.23	21
U	0.13	0.63	0.75	0.74	0.90	1.33	0.27	41
B	0.17	0.48	0.66	0.67	0.87	1.18	0.25	41

The Table III below illustrates the measures of centre and spread for the modification task distribution. The description for the columns and rows are similar to Table II above. For the modification task, the data considered for the analysis were slightly less. This is due to data cleaning, which was conducted in order to ensure the validity of the analysis. In particular, the analysis excluded the subjects who did not attempt the modification task at all, which numbers are stated in the brackets under the *N* column. On the other hand, the subjects who had attempted the modification task for some time but failed to get any score were included in the analysis, which numbers are stated in the brackets under the *Min* column. The data essentially resulted in zero values for the rate of scoring. The implication of this data is that the subjects had struggled to understand the model or perhaps had misunderstood the model. Either possibility indicates that there was a problem on the model comprehensibility. This is the reason why they were included in the analysis.

TABLE III
RATE OF SCORING DISTRIBUTION FOR MODIFICATION TASK

	Min	1st Q	Mean	Median	3rd Q	Max	Std Dev	N
C1:U	0.00 (2)	1.00	1.20	1.21	1.69	2.00	0.62	18 (3)
C1:B	0.00 (2)	0.41	0.80	0.58	1.13	2.00	0.64	16 (4)
C2:U	0.33 (0)	0.46	0.72	0.63	0.77	1.60	0.37	19 (1)
C2:B	0.00 (1)	0.32	0.59	0.50	0.89	1.20	0.36	21 (0)
U	0.00	0.55	0.98	0.95	1.37	2.00	0.56	37
B	0.00	0.39	0.69	0.54	0.91	2.00	0.51	37

From the descriptive statistics shown above, it can be seen that the rate of scoring on the UML-B models is higher than the B models. The differences of mean and median values between both models are particularly apparent for the modification task. These differences may be a reflection of true differences in the population from which the samples were taken. On the other hand, it is possible that the differences may be due to errors inherent in random sampling or sampling errors. In order to assume that the differences obtained from the samples to be true differences in the population and not due to sampling errors, the standard statistical inference needs to be applied.

This experiment employed a robust statistical method called bootstrap methods and permutation tests for the statistical inference [32]. The strength of these methods is that they do not rely on characteristics of the distribution and do not require large samples but are capable of generating results that are more accurate than those from the traditional methods [33]. The bootstrap methods were used in this experiment to calculate the standard errors and the confidence intervals [34], whereas the permutation tests were used to test the significance level of the observed effects. The analysis was done using the S-PLUS® 7.0 for Windows-Enterprise Developer [35] software.

The experiment employed a cross-over design and thus had to consider the period effect [26]. Period effect concerns the chances of detecting effects due to the period or session when the treatment is applied rather than the treatment itself. The analysis for the period effect was performed by obtaining the period differences between the two periods for both sequence groups. Later, two-sample test was performed on the period differences using the bootstrap methods. Whereas differences between period differences in the same sequence group can be regarded as being random, differences between any two period differences in different sequence groups would also reflect treatment differences. Therefore, comparing the means of the period differences for the two sequence groups would allow the treatment effect to be examined [36]. On the other hand, the mean period difference for each sequence group is an estimate of the difference between two treatments and also between two periods. This means that eliminating the period differences from the two sequence groups will give an estimate of twice the difference between two treatments [26]. Thus, the differences in means and the standard errors obtained from the test had to be adjusted by dividing by two. The subsequent statistical analysis was based on these adjusted values. The true treatment effect that considers the period effect at 95% confidence interval for the respective comprehension tasks are shown below. Indeed, they are the estimated differences between the expected rate of scoring under the UML-B model and that under the B model at 95% confidence interval.

Overall Comprehension Task:
$0.01 <= t <= 0.16$ (to the nearest 2 decimal places)

Comprehension for Modification Task:
$0.03 <= t <= 0.50$ (to the nearest 2 decimal places)

To test the significance of the results, the p-values (P) were assessed against the significance criterion (α=0.05). The p-value for the overall comprehension task is 0.012 (one-sided) in favour of the UML-B model. This means that the difference in the treatment effect between the UML-B model and the B model is statistically significant ($P<0.05$). This concludes that the UML-B is more comprehensible than the B model in terms of the efficiency in overall understanding.

The same testing was applied on the modification task's data. The p-value for the modification task is 0.011 (one-sided) in favour of the UML-B model. Similarly, the difference in the treatment effect between the UML-B model and the B model for the modification task is statistically significant ($P<0.05$). Therefore, this concludes that the UML-B is more comprehensible than the B model in terms of the efficiency in understanding a model for modification task purposes.

Another finding has also been found, which seems to suggest that even with very limited training on the UML-B method, one could still understand the model well. During the experiment, the subjects were asked to state whether or not they had attended the one-hour lecture on the UML-B method. There were eight subjects who did not attend the lecture and thus depended on the available references or own knowledge to answer the questions. The rate of scoring for these eight subjects is shown in the Table IV below. From the data, it can be seen that seven out of eight subjects performed better on the UML-B model. Therefore, despite the fact that these subjects had no training on the UML-B method, the quantitative measures suggest that they still performed better on the UML-B model. Because of the size of this sample is too small, the statistical significance testing could not be done however, as it would be unreliable.

TABLE IV
RATE OF SCORING FOR SUBJECTS WHO WERE ABSENT DURING LECTURE

Subject	UML-B model	B model
A08	0.63	0.61
A12	0.63	0.53
A13	0.64	0.73
A16	0.50	0.44
A18	0.66	0.48
B01	0.87	0.42
B11	0.57	0.48
B20	0.77	0.71

The UML-B model was commented by the subjects as being easy to visualise and understand the scenario more quickly, easy to understand the relationships between operations, easy to develop especially on computers, easy for novices and more logical to developers. Nevertheless, the model was said to be useful only with good tool support. The UML-B model was also commented as being quite 'messy' since the information was scattered around the class and statechart diagrams. On the other hand, the B model was commented as being more formal, less ambiguous and easy to read since the information was kept together as a flow of information. However, the B model was claimed as being harder to develop, lacking visualisation, lengthy and too much text.

VIII. CONCLUSION AND FUTURE WORK

This paper describes an empirical assessment or experiment conducted on the UML-B method and the B method. In particular, the experiment involved an assessment on the notations used in the respective UML-B and B models. The objective was to investigate whether the notation used in the UML-B model is more comprehensible than the notation used in the B model. The model comprehensibility was measured based on the subjects' efficiency in understanding the notations used in the models and performing the required tasks.

The empirical data have revealed that the notation used in the UML-B model expedites the subjects' comprehension task with accuracy, even with limited hours of training. Compared to the notation used in the B model, the notation in the UML-B model allows the subjects to grasp the required information more quickly and use it to perform the subsequent tasks correctly. This finding is particularly appealing as it shows that introducing some graphical features of semi-formal notation into the formal notation significantly improves the formal notation's accessibility. Besides allowing the formal notation to be more understandable, the features seem to make the daunting mathematical notation interesting and approachable. The UML-B method is thus seen as a possible solution that has been proven empirically to be able to overcome practitioners' mathematical barriers towards formal notation.

It has been pointed out that the hallmark of good experimentation is the accumulation of data and insights over time [37]-[40]. Therefore, one possible way of improvement is through replication, where the experiment will be repeated on different samples of the population with different problem domains or perhaps the same problem domains but larger in size. The experiment could also be repeated using different research design and different aspects of comprehension measurement. This will help in determining how much confidence can be placed in the results of the experiment and also strengthening its external validity. It also enables the characteristics of the method and the effects that it has on its environment to be better understood. The knowledge could then be shared with the practitioners so that they could think of the best possible way to effectively employ the method. The researchers meanwhile could formulate new ideas on how to improve the method further so that it could penetrate the industry's acceptance.

ACKNOWLEDGMENT

R. Razali thanks Dr. C. F. Snook and Dr. M. R. Poppleton for their invaluable advice and support during the planning and execution of the experiment. Dr. C. F. Snook has provided technical advice on the UML-B method and the empirical assessment, and Dr. M. R. Poppleton has provided advice and mechanism for the experiment execution. The author also thanks the postgraduate students of DSSE for their participation in the pilot study and the students of COMP3011 (Sem II/ 2005-06) for their participation in the experiment.

REFERENCES

[1] B. W. Boehm, J. R. Brown, and J. R. Kaspar, "Characteristics of Software Quality", *TRW Series of Software Technology*, 1978.

[2] I. Sommerville, *Software Engineering.* 6th Edition, Addision-Wesley, 2001, ISBN: 020139815.

[3] T. M. Pigoski, *Practical Software Maintenance: Best Practices for Managing your Software Investment.* Wiley Computer Publishing, 1996, ISBN: 0471170011.

[4] D. Craigen, S. Gerhart, and T. Ralston, *Applications of Formal Methods*, In M. Hinchey, J. Bowen, Eds., Prentice-Hall, Englewoodcliffs, NJ, 1995, ISBN:0133669491

[5] M. G. Hinchey, "Confessions of a Formal Methodist", In P. Lindsay, Ed., *Conferences in Research and Practice in Information Technology*, Vol.15, Australian Computer Society, 2002, pp. 17-20.

[6] K. Finney, and A. Fedorec, "An Empirical Study of Specification Readability Teaching and Learning Formal Methods", In: N. Dean, M. Hinchey, Eds., *Teaching and Learning Formal Methods*, Academic Press, New York, 1996, ISBN: 0123490405.

[7] K. Finney, "Mathematical Notation in Formal Specification: Too difficult for the Masses?", *IEEE Transactions on Software Engineering*, Vol.22, No.2, pp. 158-159, 1996.

[8] D. Carew, C. Exton, and J. Buckley, "An Empirical Investigation of the Comprehensibility of Requirements Specifications", *International Symposium on Empirical Software Engineering*, 2005, pp. 256-265.
[9] J. P. Bowen, and M. G. Hinchey, "Ten Commandments of Formal Methods... Ten Years Later", *IEEE Computer*, Vol.39, No.1, 2006, pp. 40-48.
[10] A. van Lamsweerde, "Formal Specification: A Roadmap", In *The Future of Software Engineering Track (ICSE'00)*, ACM Press, Los Angeles CA, 2000, pp. 147-159.
[11] I. Vessey, and R. Weber, "Structured Tools and Conditional Logic: An Empirical Investigation", *Communications of the ACM*, Vol.29, No.1, 1986, pp. 48-57.
[12] D. A. Scanlan, "Structured Flowcharts Outperform Pseudocode: An Experiment Comparison", *IEEE Software*, Vol.6, No.5, 1989, pp. 28-36.
[13] N. Cunniff, and R. P. Taylor, "Graphical vs Textual Representation: An Empirical Study of Novices' Program Comprehension", In *Empirical Studies of Programmers: 2nd Workshop*, 1987, pp. 114-131.
[14] M. Bauer, and P. Johnson-Laird, "How Diagrams Can Improve Reasoning", *Psychological Science*, Vol.4, 1993, pp. 372-378.
[15] K. Stenning, and J. Oberlander, "A Cognitive Theory of Graphical and Linguistic Reasoning: Logic and Implementation", *Cognitive Science*, Vol.19, 1995, pp. 97-140.
[16] M. Petre, "Why looking isn't always seeing: Readership skills and graphical programming", *Communications of the ACM*, Vol.38, 1995, pp.33-44.
[17] M. Scaife, and Y. Rogers, "External Cognition: How do Graphical Representations Work?", *International Journal of Human-Computer Studies*, Vol.45, 1996, pp. 185-213.
[18] J. R. Abrial, *The B-Method - Assigning Programs to Meanings*, Cambridge University Press, 1996, ISBN: 0521496195.
[19] Object Management Group (2006). *Introduction to OMG's Unified Modeling Language (UML)*. [Online]. Available: http://www.omg.org/gettingstarted/what_is_uml.htm
[20] C. Snook, I. Oliver, and M. Butler, "The UML-B Profile for Formal Systems Modelling in UML", In J. Mermet, , Ed., *UML-B Specification for Proven Embedded Systems Design*, Springer, 2004, ch. 5, ISBN: 1402028660.
[21] C. Snook, and M. Butler, "UML-B: Formal Modelling and Design Aided by UML", *ACM Transactions on Software Engineering and Methodology*, Vol.15, No.1, 2006, pp.92-122.
[22] ClearSy, *AtelierB User Manual V3.6*, ClearSy System Engineering, 2003, Aix-en-Provence, France.
[23] B-Core (UK) Limited, Oxon, UK (1999). *B-Toolkit*, *On-line manual*, [Online]. Available: http://www.b-core.com/ONLINEDOC/Contents.html
[24] J. R. Abrial, and D. Cansell (2003) *Click 'n' Prove – Interactive Proofs within Set Theory B*. [Online]. Available: http://www.b4free.com/ and http://www.loria.fr/cansell/cnp.html
[25] T. Pender, *UML Bible*, Wiley, 2003, ISBN: 0764526049.
[26] S. Senn, *Cross-over Trials in Clinical Research (Statistics in Practice)*, John Wiley & Sons, 2002, ISBN: 0471496537.
[27] J. Foster, "Program Lifetime: A Vital Statistic for Maintenance", In Proceedings of the *IEEE Conference on Software Maintenance*, 1991, pp. 98-103.
[28] A. Davis, S. Overmyer, K. Jordan, J. Caruso, F. Dandashi, A. Dinh, G. Kincaid, G. Ledeboer, P. Reynolds, A. Sitaram, A. Ta, and M. Theofanos, "Identifying and Measuring Quality in a Software Requirements Specification", In Proceedings of the *1st. International Software Metrics Symposium*, *IEEE*, 1993, pp. 141-152.
[29] M. Piattini, M. Genero, G. Poels, and J. Nelson, "Towards a Framework for Conceptual Modelling Quality", In M. Genero, M. Piattini, and C. Calero, Eds., *Metrics for Software Conceptual Models*, London : Imperial College Press, 2005, ISBN: 1860944973.
[30] S. L. Pfleeger, "Experimental Design and Analysis in Software Engineering: Part 1-5", *ACM SIGSOFT Software Engineering Notes*, Vol.20, No.1-5, 1995.
[31] B. A. Kitchenham, and S. L. Pfleeger, "Principles of Survey Research: Part 1-6", *ACM SIGSOFT Software Engineering Notes*, Vol.27, No.1-6, 2002.
[32] B. Efron, and R. Tibshirani, *An Introduction to the Bootstrap*, Chapman and Hall, New York, London, 1993.
[33] D. S. Moore, and G. P. McCabe, *Introduction to the Practice of Statistics*, 5th Edition, W. H. Freeman, New York, 2006, ISBN: 071676282.
[34] B. Efron, and R. Tibshirani, "The Bootstrap Method for standard errors, confidence intervals and other measures of statistical accuracy", *Statistical Science*, Vol.1, No. 1, 1986, pp 1-35.
[35] Insightful Corporation (2006). Available: http://www.insightful.com/products/splus/default.asp
[36] B. Jones, and M. G. Kenward, *Design and Analysis of Cross-over Trials*, 2nd Edition, Chapman and Hall, London, 2003, ISBN: 0412606402.
[37] V. R. Basili, R. W. Selby, and D. H. Huthchens, "Experimentation in Software Engineering", *IEEE Transactions on Software Engineering*, Vol.12, No.7, 1986, pp. 733-743.
[38] V. R. Basili, F. Shull, and F. Lanubile, "Building Knowledge through Families of Experiments", *IEEE Transactions on Software Engineering*, Vol.25, No.4, 1999, pp. 456-473.
[39] R. Jeffery, and L. Scott, "Has Twenty-five Years of Empirical Software Engineering Made a Difference?", In Proceedings of the *9th Asia-Pacific Software Engineering Conference*, 2002, pp. 539-546.
[40] W. F. Tichy, "Should Computer Scientists Experiment More?", *IEEE Computer*, Vol.31, No.5, 1998, pp. 32-40.

