An Investigation of Prediction Models for Project Management

Daniel Rodríguez[1], Rachel Harrison[1], Manoranjan Satpathy[1], and Javier Dolado[2]
[1]Dept of Computer Science, The University of Reading, UK
[2]The University of the Basque Country, Spain

{d.rodriguez-garcia, rachel.harrison, m.satpathy}@reading.ac.uk
dolado@si.ehu.es

Abstract

It has been claimed that dynamic prediction models can be used to help project managers make more accurate estimates than static prediction models. However, such a claim needs to be validated so that project managers can use dynamic models with confidence. In this paper, we discuss an experiment we conducted in an academic environment that compared a dynamic model using BBNs with a static model involving the COCOMO and Akiyama models. The results from this experiment in fact validate the above claim. However, we suggest replication of this experiment in order to increase confidence to our results.

Keywords. *Experimentation, Bayesian Networks, Estimates.*

1 Introduction

In this paper, we describe an experiment to study the use of Bayesian Belief Networks (BBN) [10] in project management. Dynamic models such as System Dynamics and BBNs are becoming popular among the Software Engineering research community as they may provide a better solution to some of the problems found in Software Engineering when compared with traditional static models [8]. In principle, dynamic models can help in making good decisions with data that is scarce and incomplete. For example, BBNs provide the following advantages when compared with static models:

- they can deal with uncertainties;
- static models do not take into account the causal relationships that exist between various variables;
- by nature they provide a graphical interface making their use intuitive.

The claim that dynamic models such as BBNs can be used to help project managers to make more accurate predictions needs to be validated. Only then can project managers use dynamic models with confidence. There are three types of validation techniques that are commonly employed in experimental software engineering: surveys, case studies and formal experimentation [11, 9]. Experiments are usually done in a laboratory environment. It is very difficult to perform an experiment correctly [12]. The objective is to manipulate one or more variables and control all other variables at fixed levels. The effect of the manipulation is measured, and statistical analysis is performed over the measured values to validate initial assumptions. Experimentation can also be used to compare the effectiveness of two different methods.

In this study we carried out an experiment in an academic environment to find out if the use of BBNs as a dynamic model is superior to a static model involving COCOMO and the Akiyama equation, for effort and defect estimates respectively. Both models were used in a very simple way to simplify the analysis process.

2 Experiment Background

2.1 BBNs as a Probabilistic Dynamic Model

A Bayesian Belief Network (BBN) [10, 8] is a directed graph in which the nodes represent uncertain variables and the arcs represent the causal relationship between the variables. Each node has a probability table, which stores the conditional probabilities for each possible state of the variable in relation to each combination of its parent state values. For a node without any parent, such a table stores the marginal probabilities for each possible state of that node. If the state of a certain node is known then its probability table is altered to reflect this knowledge. Such knowledge is then propagated to determine the changed probabilities of other nodes. Note that the initial probabilities of the nodes in a BBN are obtained from expert judgement and past project data. In fact, tools are available to help in the generation of BBNs from historical project data [6]. They have been used in various application areas ranging from medical diagnosis to software engineering.

Since the conditional probabilities of the probability tables associated with the nodes of a BBN are determined with respect to past project data and expert judgement, it is expected that they represent the approximate causal relationships with respect to the various quality factors and attributes (identified by the nodes) of the organisation concerned. So, when the knowledge about certain factors (i.e. nodes of the BBN) are known, the probability tables can be used to effectively predict the values associated with other nodes in the network. Note that when an organisation evolves,

Proceedings of the 26 th Annual International Computer Software and Applications Conference (COMPSAC'02)
0730-3157/02 $17.00 © 2002 IEEE

the effect of this evolution can be incorporated into the BBNs by modifying the probability tables accordingly.

BBNs can serve as decision support systems when working with uncertainty. In software engineering, it is almost impossible to predict exact values for quality estimations; in fact, it is usually sufficient to deal with ranges or intervals of parameters. BBNs allow us to represent intervals indicating values to which the parameter must belong. Also their visual support helps in understanding the causal effects.

2.2 The Static Model

We use a combination of the basic COCOMO model [4] and the Akiyama model [1] as our static model. The COCOMO model predicts total effort that would be necessary to generate a source program of certain complexity. The model by Akiyama presents a relationship between the number of defects discovered in a source program and the size of code: *def = 4.86 + 0.018 LinesOfCode*, where *def* is the number of defects introduced (i.e. the sum of the number of defects found during testing and those discovered within 2 months of delivery). In the experiment, we also used function points (FP) to represent functionality [2]. These formulae were used in the simplest possible way.

3 Experiment Set-up

3.1 Goal of the Experiment

We use a goal definition template [3] to state the objectives for our experiment. This template has five sub-headings which we used as shown below:

Object of study: BBN as a dynamic model and equations as a static model

Purpose: To compare the effectiveness of BBN verses the static equations

Quality Focus: estimation capability of both the models

Perspective: from the viewpoint of project manager/researcher

Context: This experiment is conducted in an academic environment with post graduate students and researchers as subjects. This experiment is conducted as a blocked subject-object study [15].

The BBN that we used for our experiment is shown in Figure 1. This BBN represents a defect estimation model. Each node of the graph represents a variable, and an arrow from node $m$to node n represents a causal relationship between variable m to variable n. Each variable can take one of its allowable values; for instance, the variable *functionality* can take one of the 7 permitted values (Figure 2). In the BBN shown in Figure 1, the variable code size depends on the *functionality* and *complexity* of the software. *Defects introduced* (defects in source code) depends on the *code size* and the *design effort*. *Residual defects* (defects that remain after delivery) depends on the defects in the source code and the defects detected during testing. The number of defects detected during testing depends on the amount of testing effort and the number of

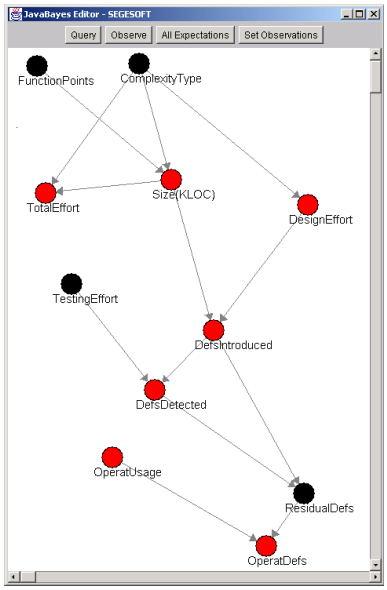

Figure 1. BBN used in the experiment

defects in the source code. Dependencies are characterised by a probability table. For instance, if an organic project with 600 FP has average testing effort and design effort then the BBN infers that the number of defects introduced in the source code will be between 500 and 600. Figure 3 demonstrates such a scenario.

The usage of the tool incorporating this BBN is fairly intuitive. The tool has two modes of operation: *observation mode* which allows insertion of evidence, and a *query mode* where estimations are visualised. Black nodes in the network mean that evidence has been entered, and grey ones mean that no evidence has been entered and we can query their probabilities under the impact of the supplied evidence.

3.2 Hypotheses

The null hypotheses are as follows:

$H_{0,1}$**:** There is no difference between the estimates predicted by the static model verses the same predicted by the BBN.

$H_{0,2}$**:** There is no timing difference between estimations predicted by BBNs with tool support and the same predicted by the static model without

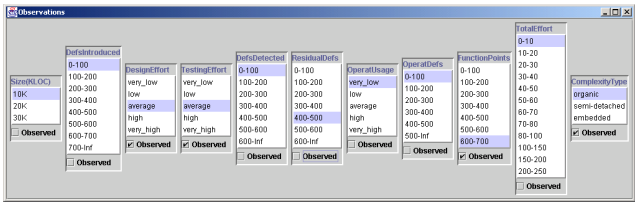

Figure 2. Evidences Window

Proceedings of the 26 th Annual International Computer Software and Applications Conference (COMPSAC'02)
0730-3157/02 $17.00 © 2002 **IEEE**

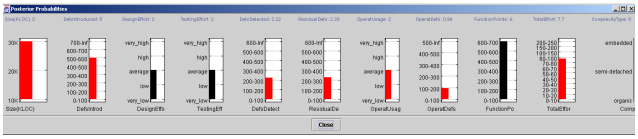

Figure 3. Posterior Probabilities Window

The alternative hypotheses that we expect to prove with this experiment can be defined as follows:

- A BBN makes the context of the problem under study clearer than the static model does.
- BBNs make it easier to answer a broad spectrum of questions relating to software concepts and estimates than static models.

There are some specific management decisions which can be easily calculated by a BBN through backward propagation of probabilities, whereas such questions cannot be answered using a static model, e.g., how resources can be used to produce a product with a given defect density.

3.3 Subjects

The subjects of the experiment were postgraduate students and researchers at the University of Reading. They all had very similar educational qualifications in the field of computer science. Following the terminology of Wohlin et al. [15], we have adopted the approach of *convenience sampling* as regards to the selection of our subjects.

We allocated each subject exactly one treatment, i.e. either the use of BBN or the use of static equations. The experiment was conducted on an individual basis because it was impossible to carry out the experiment with all the subjects simultaneously. In order to minimise threats to the experiment the same documentation was given to all subjects prior to the start and then the subjects were assigned to a pre-determined model. We assigned an equal number of subjects to each category of treatment.

3.4 Experimental Variables

In our experiment, the independent variables were:

- BBN model representing the defects estimation problem
- Static equations represented by the Akiyama and Boehm models

The dependent variables were:

- *interest* (S1) of the subjects in the area under study (we measure the degrees of agreement using a five point Likert scale where 1 is fully agree and 5 fully disagree)
- background *knowledge* (S2) of the subjects in the area under study (1 for each correct answer and 0 for each incorrect one)
- subject-score (S3), the *score* the subjects obtain from their questionnaires (1 for each correct answer and 0 for each incorrect one).

Subj. (BBN)	Score S1	Score S2	Score S3	Min S3
B1	13	3	8	16
B2	11	4	10	21
B3	13	2	8	21
B4	14	1	5	31
B5	12	4	8	22
B6	14	2	7	24
Mean	12.83	2.67	7.67	22.5
Mode	13	4	8	21
STD	1.17	1.21	1.63	4.93
VAR	1.37	1.47	2.67	24.3

Table 1. Scores of the subjects who used BBN

- the *subject-time* (S4), the time that subjects take to complete Section 3 of the questionnaire (in minutes).

In this paper our analysis is mainly based on the *subject-score* and *subject-time* variables. Although further analysis will be performed with the *interest* and *knowledge* variables, we used them mainly to check if the samples in both groups were similar.

3.5 Experiment Procedure

We prepared a questionnaire with four sections. The first section contains questions about the subject's interest in the field of Software Engineering. The questions in Section 2 focus on the subject's knowledge in software testing. Section 3 contains questions which involves the use of a model (static or dynamic) to compute and answer the values of some attributes relating to testing. Section 4 has questions relating to the subject's impression on and interest in the approach. Before starting the experiment, each subject was given 10 minutes of introduction to the experiment. They were then asked to read and make sure they understood the given documentation. They were also allowed to ask questions and clarify doubts before answering the questionnaire. Subjects using the dynamic model were told how to use the tool support which had the Bayesian network encoded in it.

The expected answers to Section 3 were calculated using (i) the static model (equations) and (ii) the dynamic model (BBN). In this way we could guarantee that the expected answers for both the static and dynamic groups were in fact the same.

4 Experiment Results

We wanted to ascertain whether the use of a dynamic model using a BBN improved the estimation of various attribute values, and so we will subject our hypotheses to one-tailed analysis. Data was collected from 12 subjects, half of them used the BBN tool and the other half calculated the equations with spreadsheets or calculators. Tables 1 and 2 show the raw data collected.

Figure 4 shows the box plots of the dependent variables interest and knowledge respectively. It can be seen that the groups had very similar interest and background in the area under concern (they have similar median values).

Proceedings of the 26 th Annual International Computer Software and Applications Conference (COMPSAC'02)
0730-3157/02 $17.00 © 2002 IEEE

Subj. (Static)	Score S1	Score S2	Score S3	Min S3
St1	11	4	10	57
St2	8	3	3	46
St3	11	2	4	40
St4	15	3	8	26
St5	11	1	2	24
St6	13	4	5	39
Mean	11.5	2.83	5.33	38.67
Mode	11	4	#N/A	#N/A
STD	2.35	1.17	3.08	12.39
VAR	5.5	1.37	9.47	153.47

Table 2. Scores of the subjects who used the static model

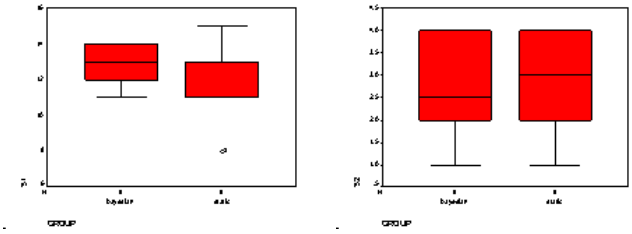

Figure 4. Box plots of the Variables Interest and Knowledge

On the other hand, from the box plots in Figure 5, it can be inferred that on average, subjects in the BBN group scored better that the other group, and furthermore, they also took less time. It can be said that the group using tool support showed improved performance in terms of time compared to the other group. However, we believe that the static group needed much more time in order to decide which equations to apply.

4.1 Dependent Variables: subject-score (S3)

Since we wanted to analyse the hypotheses involving two treatments, we applied the independent t-test to investigate the effect of the independent variables on the dependent variables subject-score

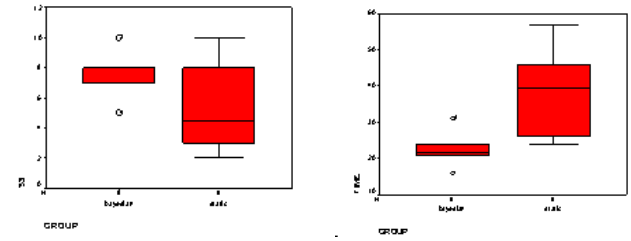

Figure 5. Box plots of the Variables Subject-score and Subject-time

S3	N	Mean	STD	Err Mean
Bayesian	6	7.67	1.63	0.67
static	6	5.33	3.08	1.26

Table 3. Group Statistics for the subject-score variable

TIME	N	Mean	STD	Std. Error Mean
Bayesian	6	22.5	4.93	2.01
static	6	38.67	12.38	5.05

Table 4. Group statistics for the Subject-time variable

and subject-time under both treatments. As the sample is small, we decided to use an alpha value of 0.1 ($\alpha = 0.1$). Therefore the confidence of all decisions to reject or accept the hypotheses $H_{0,1}$ and $H_{0,2}$ is 90%.

Table 3 shows the *subject-score* variable group statistics for Section 3 of the questionnaire for both treatments. It also shows that for the variable subject-score, the mean is 7.67 for the group using BBNs, compared with 5.33 for the group using static equations.

Levene's Test checks to define whether the variances are homogenous [14]. Table 4.2 shows that the value of $p = 0.11 > 0.05$, and therefore, we can apply the t-test assuming that the variances are equal. The rest of the columns in Table 4.2 show the different parameters of the t-test. It is possible to reject $H_{0,1}$ because the probability is below the $\alpha = 0.1$ considered. However, we need to accept this result with caution since both the t-values are relatively close.

It is worth mentioning that one of the questions asked the subject to estimate design effort for a given defect density. Such a question is difficult to answer using a static model since the defect density depends on many variables and the influence of each variable on the outcome can not be known. In our experiment, only two of the subjects using the static model answered the question correctly. However, subjects using BBN modelling could answer the question easily because of the backward propagation of probabilities in the BBN.

4.2 Dependent Variables: subject-time (TIME)

We also performed a one tailed t-test with an alpha value of 0.1 ($\alpha = 0.1$) with the variable subject-time, see Table 4.2. Table 4.2 shows the group statistics of the timing measures for Section 3 of the questionnaire for both the treatments. Observe that the means are significantly different.

Since the p-value of Levene's test is $p = 0.11 > 0.05$, we can assume that the condition for equal variance holds for the t-test. We can clearly reject the null hypothesis $H_{0,2}$ as the p-value is smaller than the proposed $\alpha = 0.1$.

Proceedings of the 26 th Annual International Computer Software and Applications Conference (COMPSAC'02)
0730-3157/02 $17.00 © 2002 IEEE

S3	Levene's Test		t-test Equality Means						
	F	Sig.	t	df	Sig (1-tail)	Mean Diff	StdErr Diff	95% Conf Interval	
Eq var assumed	3.06	0.11	1.64	10	0.067	2.33	1.42	-0.83	5.50
Equal var not assumed			1.64	7.61	0.07	2.33	1.42	-0.97	5.64

Table 5. Independent t-test for Equality of Means (Subject-Score variable)

TIME	Levene's Test		t-test for Equality of Means						
	F	Sig.	t	Df	Sig. (1-tail)	Mean Diff	StdErr Diff	95% Conf Interval	
Equal var assumed	3.09	0.11	-2.97	10	0.007	-16.17	5.44	-28.29	-4.03
Equal var not assumed			-2.97	6.54	0.11	-16.17	5.44	-29.22	-3.11

Table 6. Independent t-test for Equality of Means (Subject-time variable)

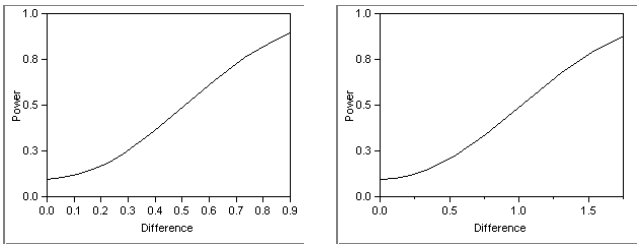

Figure 6. Power Analysis

4.3 Power Analysis and Sample Size

Power analysis is directed towards exploring the different situations that could arise with respect to the effect size, significance level and power level [7]. This exploration, jointly with the post-analysis of the actual results will help us to formulate future similar experiments.

The significance level (α) is the probability of committing a Type I error, i.e., to reject incorrectly the null hypothesis, when it is true. In our case, this will tend to promote, incorrectly, one of the methods (the BBN approach). However, we assume that this will not have an adverse effect on the estimations. On the other side, committing a Type II error, that is, to accept incorrectly the null hypothesis when it is false, would imply that we will be losing the benefits of one of the methods (the BBN method).

In the current setting we did not have previous values of the effect size, so we depicted the combinations allowed, by assuming an error standard deviation of 0.5 (and 1), α of 0.1 and the fixed sample size of 12 subjects. This kind of exploration is the last resort we have when no other similar studies are available.

In Figure 6 (left), we have an exploration of one of the possible situations and the power curve obtained (with the software JMP [13]). In Figure 6 (right), we depict another curve, by assuming another error standard deviation of 1. We observe that a power level above of 0.5 is obtained if the difference detected between the means is above 0.64. If the assumed error standard deviation is 1 we have a power level above 0.5, if the difference detected is above 1.03.

The final differences observed in TIME are above 1. Therefore, given the results shown by Figures 6, we can be reasonably confident about the conclusions of the tests of significance.

5 Threats to the Experiment

A pilot study was carried out with colleagues in the Applied Software Engineering Research Group of the University of Reading. This pilot study helped to improve the questionnaire and the associated documentation.

5.1 Conclusion Validity

Conclusion validity is concerned with the relationship between the treatment and the outcome. One factor affecting the experiment could be the small number of subjects. This is known to have a negative effect on the power of the statistical methods reducing the chance of finding an effect if it exists.

5.2 Internal Validity

Internal validity is concerned with the relationship between the treatment and the outcome; i.e., if the conclusions can be obtained from the causal effect of the independent variable. A crucial step in the experimental design consists of minimising the impact of the threats to the validity; i.e., minimising factors that can affect the dependent variables without the researcher's knowledge.

The volunteers for our experiment were chosen from a group of postgraduate students and researchers in our department. This may explain why no subject took the experiment lightly and also why no subjects dropped out of the experiment. In addition, the subjects were randomly assigned to one of the treatments in order to avoid selection effects.

There could be some maturation effect due to learning and practice as the experiment proceeded. Some of the subjects in both the groups modified some of their initial answers as they became familiar with the tool or with the equations in the static model (some of the subjects wanted to understand the questions better). There is also a risk that the people using the static model needed more training time to master how to apply the formulae correctly.

5.3 Construct Validity

Construct validity is concerned with the degree to which the variables used in the study accurately match the concepts they intend to measure. Our BBN model may not adequately capture the

Proceedings of the 26 th Annual International Computer Software and Applications Conference (COMPSAC'02)
0730-3157/02 $17.00 © 2002 IEEE

advantages of using BBNs in a general software engineering scenario, since our experiment took a simplistic view of the problem and the specific advantages or disadvantages of using BBNs can not be captured by the experiment.

The BBN and the static models were created in such a way that it would be possible to answer the questionnaire using both models. However, it was difficult to create fair scenarios to compare both models. To deal with this threat, the static model was quite simple and the BBN simulated the static model, so the 'correct' answers in the questionnaire were same for both groups.

In order to gauge the background knowledge of the subjects in software engineering in general and on testing in particular, we referred to the *Software Defect Reduction Top 10 List* [5]. After analysing the results of the experiment, we believe that more general questions could have given more accurate results. It is also very difficult to measure interest objectively.

5.4 External Validity

External validity is concerned with the degree to which the results of the research can be generalised. Although the subjects had experience in developing software, most of them had no industrial experience. Therefore, it is difficult to generalise the results. Further studies should be carried out to assess the usefulness of BBNs as used by experienced project managers.

As the experiment was carried out on an individual basis, we believe that no misunderstanding over the questions occurred and all of the subjects took the experiment seriously, but we also think that some of the subjects may have tried to guess some of the answers.

6 Conclusions

We have described an experiment that compared the use of a BBN and a static model involving the basic COCOMO and Akiyama models for estimates. The results show statistically positive results in favour of the group using the BBN.

From the dependent variable, *subject-score*, i.e., the number of correct answers in a section of the questionnaire regarding estimates and cause-effect relationships, we conclude that a probabilistic approach to project management using BBNs can be useful because:

- they can explain some cause-effect relationships of software engineering better than static models;
- the backwards propagation of the probabilities in the BBNs can help in taking managerial decisions that might not be possible using static models;
- they are intuitive an easy to use because of their graphical notation, so it is possible to use them with little knowledge of the area.

From the dependent variable *subject-time*, i.e., time needed to complete a section of the questionnaire regarding estimates and cause-effect relationships, it can be quite trivial to conclude that using specific tool support is better than using calculator and spreadsheets but we also believe that it may reflect that understanding the processes involved in calculating estimates is not easy.

Replication and further empirical studies are necessary to demonstrate the usefulness of BBNs in Software Engineering. In the case of replication, this experiment should be improved with some of the comments discussed in Section 5. However, these preliminary results are encouraging.

Acknowledgements

This work has been supported by The University of Reading, UK. This work is also based on a project management tool developed by the ARGO Group, supported by the CICYT grant 2001-1143-C03-01 (Spain). The authors wish to thank all the participants in the experiment and the members of the Applied Software Engineering group at The University of Reading.

References

[1] F. Akiyama. An example of software system debugging. *Information Processing*, 71:353–379, 1971.

[2] A. J. Albretch. Measuring application development. In *Proc. of INM Applications Development Joint SHARE/GUIDE Symposium*, pages 83–92, Monterey CA, 1979.

[3] V. R. Basili, G. Caldiera, and H. Rombach. The goal question metric paradigm. In *Encyclopedia of Software Engineering*, pages 528–532. John Wiley Sons, 1994.

[4] B. W. Boehm. *Software Engineering Economics*. Prentice-Hall, Englewood Cliffs, N.J. ; London, 1981. Bibliography: p733-749. - Includes index.

[5] B. W. Boehm and V. R. Basili. Software defect reduction top 10 list. *IEEE Computer*, 34(1):135–137, January 2001 2001.

[6] J. Cheng, D. Bell, and W. Liu. Learning belief networks from data: an information theory based approach. In *Sixth ACM International Conference on Information and Knowledge Management*. ACM Press, 1997.

[7] J. Cohen. *Statistical Power Analysis for the Behavioral Sciences. Second Edition*. Academic Press, 1988.

[8] N. E. Fenton and M. Neil. Software metrics: successes, failures and new directions. *Journal of Systems and Software*, 47(2-3):149–157, 1999.

[9] N. E. Fenton and L. S. Pfleeger. *Software metrics : a rigorous and practical approach*. International Thomson Computer Press : PWS Pub, LondonbInternational Thomson Computer Pr., 1997.

[10] F. Jensen. *An Introduction to Bayesian Networks*. UCL Press, 1996.

[11] B. A. Kitchenham. Desmet methodology: Guidelines for evaluation method selection. Technical Report Project Deliverable D2.3.1, The National Computing Centre Ltd, October 1993.

[12] B. A. Kitchenham, S. L. Pfleeger, L. M. Pickard, P. W. Jones, D. C. Hoaglin, E.-K. Emam, and J. Rosenberg. Preliminary guidelines for empirical research in software engineering. In *EASE'01 (Empirical Assessment in Software Engineering)*, Keele University, 2001.

[13] SAS. Jpm4, 2001.

[14] SPSS. *SPSS Base 8.0. Applications Guide*. SPSS, 1998.

[15] C. Wohlin, P. Runeson, M. Host, M. Ohlsson, B. Regnell, and A. Wesslen. *Experimentation in Software Engineering*. Kluwer Acad., 2000.

Proceedings of the 26 th Annual International Computer Software and Applications Conference (COMPSAC'02)
0730-3157/02 $17.00 © 2002 **IEEE**