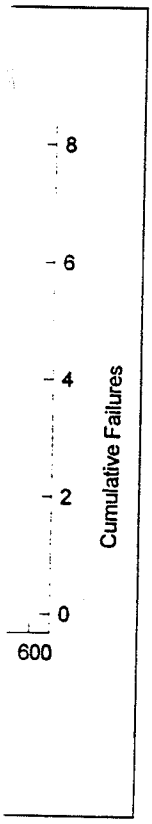

ld be confirmed or

Software Development

l): 46–61.
ration Control Process,

ions on Software Engi-

The Study of Software Maintenance Organizations and Processes

CAROLYN B. SEAMAN
Inst. for Advanced Computer Studies, University of Maryland, College Park, MD

VICTOR R. BASILI
Inst. for Advanced Computer Studies, University of Maryland, College Park, MD

In order to improve the quality of software products, it is necessary to enhance the quality of the software processes used to develop them. This recognition has led to a proliferation of work, both empirical and non-empirical, on software development processes, some of which has dealt specifically with maintenance. One important but often overlooked component of both development and maintenance processes is the organizational context in which they are enacted. Our position is that this is an even more pertinent issue in maintenance than in development because maintainers are not generally the original developers of a system. The software has a history in which a number of people have been involved, and the experience of those people becomes relevant during maintenance. Thus an efficient flow of information is crucial, and this flow can be hindered or facilitated by the organizational relationships between maintainers and other parts of the organization. It is the network of these relationships (reporting relationships, physical proximity, past working relationships, etc.) that constitute organizational structure. Furthermore, a variety of methods must be employed to study organizational issues in maintenance, including both quantitative and qualitative approaches.

The authors have participated in two studies of software maintenance organizations which illustrate two different approaches to studying organizational issues in software maintenance. The first (Seaman, 1996) is a mostly quantitative study in which variables were defined and evaluated, allowing for statistical analysis. The goal of that study was to discover possible relationships between the study variables, which measured organizational structure, communication, and productivity. The second study (Briand, et al., 1995) was a qualitative study which produced a rich descriptive model of the maintenance organization and process. The goal of this study was to highlight anomalies and possible problems in both the organization and process.

In (Seaman, 1996), we examined information flow and organizational structure among a group of software maintainers. The dependent variable in this study was communication effort, defined as the total effort expended to share some type of information. The choice of dependent variable followed directly from the goal of the study. The choice of independent variables was more difficult. They were designed with care, but must be considered exploratory. All the independent variables were measures of the organizational structure, defined as the network of relationships between members of the software development organization. The types of relationships upon which these measures were based are official reporting relationships, physical proximity, and past and present working relationships.

The study took place at IBM Software Solutions Laboratory in Toronto, Canada. The

maintenance system studied was DB2, a commercial database system with several versions for different platforms. During the month of June 1994, data was collected from (mostly design and code) reviews in Toronto. Ten reviews were directly observed. These observations were followed up with interviews with review participants in November 1994, and in April 1995. The data collected was coded into values for communication effort (the dependent variable) and for measures of organizational structure (the independent variables). Simple correlations were then sought between dependent and independent variables, as well as with a number of intervening variables which were also evaluated from the data.

The review process was chosen for study because it is well-defined in the DB2 project, it involves a lot of communication between participants, and much of it is observable. In addition, reviews usually involved the participation of developers who had been part of the history of the code being reviewed. Reviews, in fact, were one of the few parts of the maintenance process in which developers and maintainers interacted.

The study yielded a number of interesting results. Communication, in general, tended to take longer when the communicators were not previously familiar with each others' work, when they worked in physically distant locations, when communication took place during a meeting, when some sort of communication technology was used, and when more people were involved. Also, more effort was required during meetings which included a group of participants from the same part of the organization, plus a few participants from some distant part. This would seem to have some implications for the preferred arrangement of maintainers in relation to developers with relevant information. However, this was an exploratory study, with the aim of generating, not confirming, theory.

This first study strongly suggests that organizational relationships have an effect on maintenance productivity. Thus, organizational factors, and "people" factors in general, are important variables in the empirical study of maintenance processes.

The second maintenance study we will describe (Briand, et al., 1995) was part of the development of a maintenance auditing process (Briand et al., 1994). The auditing process was meant to provide a holistic, qualitative assessment of the strengths, weaknesses, and possible flaws in a currently existing maintenance organization. This assessment would then be used as a basis for a quantitative measurement program for continued improvement. The study took place at NASA/Goddard Space Flight Center's Flight Dynamics Division (FDD). This organization maintains numerous software systems for the control and prediction of satellite orbits, trajectories and attitude, sometimes for very long periods, and new releases are regularly produced.

Data was collected through interviews and from the organization's documents. A model of the FDD's maintenance process and organization was built using a formalism called Actor-Dependency models (Yu and Mylopoulos, 1994). AD models describe an organization as a network of interdependencies among active organizational entities, i.e., actors. A node in such a network represents an organizational actor, and a link indicates a dependency between two actors. Building the AD model of the FDD maintenance organization was the main method of analysis, and it revealed a number of potential flaws in both the process and organization. Among these were an over-reliance on one role (the Task Leader), a lack of quality assurance priorities among maintainers, organizational barriers to eliciting unambiguous requirements, and inaccurate data collection. These were all revealed as

inconsistencies in the dependencies between members of the maintenance organization. For example, one part of the organization (the maintainers) depended on another part (the users) for their requirements, and they also needed these requirements to be unambiguous. However, these two parts of the organization were very far removed from each other and there were no other dependencies between them. Thus, it was difficult to motivate users to spend time and effort on requirements.

These two maintenance studies, one quantitative and one qualitative, provide a good contrast between approaches to the study of organizational issues. Neither approach is better than the other in all cases, but some situations are more amenable to one or the other. In many cases, a combination of the two is the most effective approach.

Maintenance, it would seem, would provide a distinct advantage for empirical researchers because it facilitates the choice of the unit of analysis, which is an important decision that must be made during the planning of any empirical study. In software engineering studies of process, units of analysis range over all levels, from entire projects to enactments of a single subprocess. The larger the unit of analysis, however, the more expensive each data point is in general. Maintenance provides an excellent source of data points based on a variety of units of analysis. This is because most maintenance is arranged in small cycles; it consists of numerous repetitions of a relatively short process. For example, some maintenance processes are based on software changes. There are defined steps for specifying, designing, implementing, integrating, and testing a change, which are then repeated for the next change. At a slightly higher level, some maintenance efforts are organized by system release. The process for planning and executing a release is still fairly short-term, at least in comparison to most development projects. In either case, the researcher interested in studying maintenance processes is presented with a number of data points in a relatively short period of time.

The difficulty comes in determining whether these data points are uniform enough to be compared statistically. Limiting a study to a single maintenance effort eliminates some variation between data points, such as application domain, programming language, and platform. But software changes can vary considerably in nature and difficulty, even in the same system. Releases, too, can vary depending on business goals and conditions.

One approach to handling this variation is to define variables which account for all the variation between data points. These variables can then be factored into the analysis as intervening variables. This approach was used in the study of DB2 maintenance (Seaman, 1996), described briefly above. The unit of analysis chosen for that study was an interaction, defined as a single instance of communication. In order to account for all of the variation in the numerous types of communication observed in that study, many intervening variables were defined, including the communication media used, the skill level of the participants, the size and complexity of information shared, etc. The result was a very rich set of data that was very hard to analyze. The first problem is that, for most types of statistical analysis, the use of more variables requires more data. When there is not enough data corresponding to each level of each variable, then many questions are left open. The second problem is that often the interesting results become lost in the profusion of variables and partitions.

Another approach to dealing with variation among data points is to analyze the data qualitatively. This was the approach taken in (Briand et al., 1995), described above, and in

(Briand et al, 1994), which describes the larger maintenance auditing process. In part of the latter study, a unit of analysis was chosen (a software change), but no attempt was made to define and evaluate variables, or to exhaustively define the variation among software changes. Instead, several changes were chosen, and each was described in detail. The idea was not to provide statistical evidence of far-reaching phenomena, but to find anecdotal evidence of possible problems in the process or organization.

Both quantitative and qualitative methods are appropriate for the study of software maintenance organizations and processes. Quantitative methods are most appropriate when it is easy to define a unit of analysis (which is often the case in maintenance), variation between individual units is minimal or well-understood (which is sometimes the case in maintenance), and the goal of the study is to find extensive and strong evidence to support a hypothesis. Qualitative methods are a good alternative when the above conditions do not hold, particularly in an exploratory study of a problem in which little previous work has been done. Most studies fall somewhere between these two extremes, and there are a number of ways to combine quantitative and qualitative methods (Gilgun, 1992) in such cases.

One way that quantitative and qualitative methods are easily combined is to use (qualitative) information from interviews with study participants to help interpret statistical results based on quantitative data. In essence, this is done frequently in empirical studies in software engineering. Many papers on such studies refer briefly to a "follow-up interview" or "comments from participants". However, in most cases, this opportunity is not fully exploited. If such interviews are planned well, interview techniques from the qualitative research literature (e.g. interview guides, field notes) are used, and the resulting data is more effectively analyzed, then these interviews could yield very useful data. In particular, this qualitative data can be used to help interpret statistical findings, to suggest further analyses of the quantitative data, and to point to future research directions. Even more useful would be interviews conducted throughout the course of a study, not just at the end.

In this position paper we have made a case for the importance of studying the organizational context in which maintenance is carried out. Two different approaches to this type of investigation have been described. Both of the studies which serve as examples of these approaches demonstrate important results which would not have been discovered by studying the maintenance process, or any other aspect of maintenance, by itself. It is only by explicitly including organizational relationships in the study design that important insights are gained, particularly with respect to information flow, which is a crucial issue in maintenance.

References

Briand, L., Basili, V., Kim, Y. M., and Squier, D. R. 1994. A change analysis process to characterize software maintenance projects. *Proceedings of the International Conference on Software Maintenance* Victoria, Canada.

Briand, L., Melo, W., Seaman, C., and Basili, V. 1995. Characterizing and assessing a large-scale software maintenance organization. *Proceedings of the 17th International Conference on Software Engineering* Seattle, WA.

Gilgun, J. F. ed. 1992. *Qualitative Methods in Family Research.* Sage.

POSITION PAPER

Seaman, C. B. 1996.
Science Departmer
Yu, E. S., and Mylo
Proceedings of the

Report fr
Documen

EIRIK TRYGGESE
Department of Com
Trondheim, Norway

I. INTRODU

Several investig
problems of lac
tainers (Lientz a
Dekleva, 1992;

It is my positi
code to provide
bers of the mai
functionality de

A part of my
maintenance pr
system, and ho
the maintenanc
explanation of t

I would not
only show the
several pilot pr
the technologic
seldom possibl
use controlled
experiments sh
research so tha
and generalizat

This paper ou
learned from c
hypotheses test
design. Section

Seaman, C. B. 1996. Communication and organization in software development: An empirical study. Computer Science Department, University of Maryland Technical Report CS-TR-3619.

Yu, E. S., and Mylopoulos, J. 1994. Understanding 'why' in software process modeling, analysis, and design. *Proceedings of the 16th International Conference on Software Engineering* Sorrento, Italy.

Report from an Experiment: Impact of Documentation on Maintenance

EIRIK TRYGGESETH eirik@idt.ntnu.no
Department of Computer and Information Science, Norwegian University of Science and Technology, N-7034 Trondheim, Norway

I. INTRODUCTION

Several investigations on software maintenance problems have given high priority to the problems of lacking or inadequate documentation, and a high turnover rate among maintainers (Lientz and Swanson, 1980; Chapin, 1985; Nosek and Palvia, 1990; Foffani, 1992; Dekleva, 1992; Krogstie, 1994).

It is my position that documentation must be actively maintained concurrently with source code to provide new maintainers with a solid basis for quickly becoming productive members of the maintenance team. I define a software system which have all aspects of its functionality described by thoroughly updated documentation to be in *internal equilibrium*.

A part of my doctoral research is to specify a technological framework for how the maintenance process can be supported in order to ensure internal equilibrium of a software system, and how this valuable asset of a system may be used to ensure that members of the maintenance team obtain a high productivity in their work. I will not delve into a long explanation of this, but rather refer to (Tryggeseth, 1997).

I would not try to "validate" my technological research with case studies which can only show the *usability* of a solution, not its *usefulness*. A full validation would involve several pilot projects over a long time period, and an integrated and robust prototype of the technological solution must be built to be used on those pilots. Such an approach is seldom possible in time constrained research such as a PhD study. Rather I propose to use controlled experiments to provide arguments for the research direction chosen. The experiments should include basic aspects of improvements proposed in the technological research so that the soundness of the proposed solution can be discussed by interpolation and generalization.

This paper outlines the results of an experiment organized for this purpose, and lessons learned from carrying it out. The paper is organized as follows: Section II describes the hypotheses tested in the experiment. Section III gives a short overview of the experiment design. Section IV presents the results of the analysis of the experiment data. In Section V,