ICIS 1998 Proceedings

International Conference on Information Systems (ICIS)

December 1998

The Life Cycle Effects of Software Process Improvement: A Longitudinal Study

Donald Harter
Carnegie Mellon University

Sandra Slaughter
Carnegie Mellon University

Mayuram Krishnan
University of Michigan, Ann Arbor

THE LIFE CYCLE EFFECTS OF SOFTWARE PROCESS IMPROVEMENT: A LONGITUDINAL ANALYSIS

Donald E. Harter
Sandra A. Slaughter
Carnegie Mellon University
U.S.A.

Mayuram S. Krishnan
University of Michigan, Ann Arbor
U.S.A.

Abstract

Rapid innovation, intense competition, and the drive to survive have compelled information technology (IT) firms to seek ways to develop high quality software quickly and productively. The critical issues faced by these firms are the inter-relationships, sometimes viewed as trade-offs, between quality, cycle time, and effort in the software development life cycle. Some believe that higher quality can only be achieved with increased development time and effort. Others argue that higher quality results in less rework, with shorter development cycles and reduced effort. In this study, we investigate the inter-relationships between software process improvement, quality, cycle time, and effort. We perform a comprehensive analysis of the effect of software process improvement and software quality on all activities in the software development life cycle. We find that software process improvement leads to higher quality and that process improvement and quality are associated with reduced cycle time, development effort, and supporting activity effort (e.g., configuration management, quality assurance). We are in the process of examining the effect of process improvement and quality on post-deployment maintenance activities.

Keywords: Software quality, IS development time, IS development effort.

1. RESEARCH OBJECTIVE

The IT industry has seen dramatic, 500% growth worldwide over the past decade (Mowrey 1996). As competition intensifies and customers demand additional capabilities, IT firms must deliver improved software to the market faster (McConnell 1998). However, firms may be reluctant to sacrifice quality or incur higher development costs in order to shorten development times. Investment in process improvement focused on software quality is a potential solution to simultaneously achieving higher quality, shorter cycle times, and reduced costs.

In this study, we develop a framework for assessing the economic value of software process improvement and quality over the software life cycle. We quantify the costs and benefits of improving software processes and quality in different life cycle stages. Specifically, we examine the inter-relationships between software process maturity, quality, development effort, cycle time, supporting activities effort, and maintenance effort (Figure 1). Our objective is to measure the effect of process improvements where process maturity is defined in the terms of the Software Engineering Institute's (SEI) capability maturity model (CMM).

<table>
<tr><th rowspan="3">Life Cycle Phase</th><th colspan="2" rowspan="2">Design</th><th colspan="2" rowspan="2">Development</th><th colspan="3">Testing</th><th rowspan="2">Production</th></tr>
<tr><th>Development</th><th colspan="2">Conformance</th></tr>
<tr><td>System Level</td><td>Program Level</td><td>Code</td><td>Unit Test</td><td>System</td><td>Func-tional</td><td>Stress</td><td>Maintenance</td></tr>
<tr><td>Process Maturity</td><td colspan="7">The Effect of Process Improvement on Software Quality</td><td rowspan="3">The Effects of Process Improvement and Quality on Maintenance Effort</td></tr>
<tr><td>Engineering Activities</td><td colspan="7">The Effect of Process Improvement and Quality on Development Effort and Cycle Time</td></tr>
<tr><td>Supporting Activities</td><td colspan="7">The Effect of Process Improvement and Quality on Supporting Activities Effort</td></tr>
</table>

Figure 1. Overview of Research

2. THEORETICAL FOUNDATIONS

There are different perspectives on the business value of quality even outside of the software context. There are those who believe that it is economical to maximize quality. This is the "quality is free" perspective espoused by Crosby (1979), Feigenbaum (1991), and others. Their key argument is that, as the voluntary costs of defect prevention are increased, the involuntary costs of rework decrease by more than the increase in prevention costs. The net result is lower total costs, and thus quality is "free." However, there are those who believe it is uneconomical to have high levels of quality and assume they must sacrifice quality to achieve other objectives such as reduced development time. Managers in industry have reported to the Software Engineering Institute, "I'd rather have it wrong than have it late. We can always fix it later" (Paulk 1995).

Manufacturing experience relating to return on investment of quality improvements suggests that improvements often result in cost savings that outweigh the investment (Juran and Gryna 1988). Kaplan (1986) discusses the connection between quality and the direct costs of labor and materials. Bohn (1995) reports evidence of process improvement enhancing process yield and product quality. Nandakumar, Datar, and Akella (1993) argue that low quality in manufacturing leads to longer product development cycles.

Humphrey (1995) supports a similar view on software processes. This belief is further supported by surveys and case reports (Herbsleb et al. 1997). Empirical research on software costs has linked software quality to maintenance costs (Banker, Davis, and Slaughter 1998) and to life-cycle costs (Krishnan 1996). While prior research has made many important contributions, comprehensive analysis has been limited, and it provides only a partial understanding of the value of improving software processes and quality.

Our research examines whether the lessons from manufacturing can be extended to software development. Specifically, we investigate the following questions:

1. What effect does process maturity have on software quality?
2. What effect does software quality improvement have on development effort and cycle time?
3. What effect does software quality improvement have on supporting organization costs?
4. How does software quality improvement affect maintenance costs?

3. RESEARCH METHODOLOGY

3.1 Data Collection

To examine our research questions, we collected detailed longitudinal data on a 12 year software development effort. The research site selected for this study is the systems integration division of $1 billion/year information technology firm. The division developed 3.5 million lines of code from 1984 to 1996 as part of a material resource planning system and aggressively pursued process and quality improvements. Data on 30 software products were collected in order to explore the relationships between process, quality, development effort, and cycle time. We collected 72 consecutive months of longitudinal cost data to investigate the effect of process and quality on supporting activities and maintenance effort.

Data for this study indicate that process improvements resulted in a significant reduction in defect rates with quality improving at a diminishing rate. This pattern of incremental process improvements in quality serves as the foundation for measuring the effect of software quality on development effort, cycle time, supporting activity effort, and maintenance expenditures. We first examine the relationship between process maturity and software quality. We continue our analysis by quantifying the benefit of software quality in terms of (1) development effort and cycle time, (2) supporting costs, and (3) maintenance costs.

3.2 Software Quality

We develop and test a model that links software process improvements to software quality. Since process improvements tend to be discrete actions over time, we measure the cumulative impact of process improvements using SEI's CMM. We integrate two models that inter-relate software process maturity, development quality, and conformance quality. Development quality is measured as defects found by the development organization prior to customer testing. Conformance quality is based on defects found in customer testing prior to acceptance. We predict that increased software process maturity will result in improved quality.

The effects of size and other factors in software development are not linear. Researchers of software quality (Newfelder 1993) have observed economies and diseconomies of scale. Thus, we adopt a log-linear specification for our models:

$$\mathit{ln(Development\text{-}quality)} = \alpha_{01} + \alpha_{11} * \mathit{ln(Process\text{-}Maturity)} + \alpha_{21} * \mathit{ln(Product\text{-}Size)} + \alpha_{31} * \mathit{ln(Product\text{-}Complexity)} + \varepsilon_{Q1}$$

$$\mathit{ln(Conformance\text{-}quality)} = \alpha_{02} + \alpha_{12} * \mathit{ln(Process\text{-}Maturity)} + \alpha_{22} * \mathit{ln(Product\text{-}Size)} + \alpha_{32} * \mathit{ln(Requirements\text{-}Ambiguity)} + \alpha_{42} * \mathit{ln(Development\text{-}Quality)} + \varepsilon_{Q2}$$

We find that improved processes significantly increase development quality. The effect of process maturity on conformance quality appears to be mediated through development quality. This suggests that quality cannot be "tested into" the product, but is a deeper characteristic reflecting the success of the design and development processes.

3.3 Development Effort and Cycle Time

We next develop and test a model that links software quality to development effort and cycle time. We integrate three models that inter-relate process maturity, product quality, development cycle time, and development effort. Cycle time and effort are specified as a function of process maturity and product quality, controlling for the size of the product, product complexity, and requirements ambiguity.

Researchers of software costs (Banker, Chang and Kemerer 1994; Banker and Slaughter 1997) have observed economies and diseconomies of scale. Adopting a log-linear specification for our models:

$$\begin{aligned}
ln(\textit{Product-Quality}) &= \beta_{01} + \beta_{11}{*}ln(\textit{Process-Maturity}) + \beta_{21}{*}ln(\textit{Product-Size}) + \\
&\quad \beta_{31}{*}ln(\textit{Product-Complexity}) + \varepsilon_{D1} \\
ln(\textit{Cycle-Time}) &= \beta_{02} + \beta_{12}{*}ln(\textit{Product-Quality}) + \beta_{22}{*}ln(\textit{Process-Maturity}) + \\
&\quad \beta_{32}{*}ln(\textit{Product-Size}) + \beta_{42}{*}ln(\textit{Requirements-Ambiguity}) + \varepsilon_{D2} \\
ln(\textit{Development-Effort}) &= \beta_{03} + \beta_{13}{*}ln(\textit{Product-Quality}) + \beta_{23}{*}ln(\textit{Process-Maturity}) + \\
&\quad \beta_{33}{*}ln(\textit{Product-Size}) + \beta_{43}{*}ln(\textit{Requirements-Ambiguity}) + \varepsilon_{D3}
\end{aligned}$$

We estimated these models using ordinary least squares (OLS) regression. We also estimated a two stage least squares model to correct for any bias in the OLS estimators, seemingly unrelated regression (SURE) parameters using a feasible generalized least squares (FGLS) procedure allowing for correlation of disturbances across equations, and a rank regression as a robustness check. All results were similar in sign, significance, and magnitude to the OLS estimates. Results indicate significant relationships between product quality, cycle time, and development effort. A 1% increase in quality reduces cycle time by 0.45% and effort by 0.61%.

3.4 Supporting Activity Costs

In our third analysis, we assess the effect of process maturity and quality on support costs controlling for software size. Support costs include resources expended by organizations not directly involved in the development of software (Table 1). For example, quality assurance (QA) is not directly involved in software development but is an important support activity.

We form a log-linear multivariate regression to estimate the relationship between software maturity, quality, and support costs:

$$\begin{aligned}
ln(\textit{Support-Costs for DED}) &= \gamma_{01} + \gamma_{11}{*}ln(\textit{Prod-Qual}) + \gamma_{21}{*}ln(\textit{Maturity}) + \gamma_{31}{*}ln(\textit{Prod-Size}) + \varepsilon_{S1} \\
ln(\textit{Support-Costs for Integ}) &= \gamma_{02} + \gamma_{12}{*}ln(\textit{Prod-Qual}) + \gamma_{22}{*}ln(\textit{Maturity}) + \gamma_{32}{*}ln(\textit{Prod-Size}) + \varepsilon_{S2} \\
ln(\textit{Support-Costs for Doc}) &= \gamma_{03} + \gamma_{13}{*}ln(\textit{Prod-Qual}) + \gamma_{23}{*}ln(\textit{Maturity}) + \gamma_{33}{*}ln(\textit{Prod-Size}) + \varepsilon_{S3} \\
ln(\textit{Support-Costs for ADPT}) &= \gamma_{04} + \gamma_{14}{*}ln(\textit{Prod-Qual}) + \gamma_{24}{*}ln(\textit{Maturity}) + \gamma_{34}{*}ln(\textit{Prod-Size}) + \varepsilon_{S4} \\
ln(\textit{Support-Costs for Ops}) &= \gamma_{05} + \gamma_{15}{*}ln(\textit{Prod-Qual}) + \gamma_{25}{*}ln(\textit{Maturity}) + \gamma_{35}{*}ln(\textit{Prod-Size}) + \varepsilon_{S5} \\
ln(\textit{Support-Costs for QA}) &= \gamma_{06} + \gamma_{16}{*}ln(\textit{Prod-Qual}) + \gamma_{26}{*}ln(\textit{Maturity}) + \gamma_{36}{*}ln(\textit{Prod-Size}) + \varepsilon_{S6} \\
ln(\textit{Support-Costs for CM}) &= \gamma_{07} + \gamma_{17}{*}ln(\textit{Prod-Qual}) + \gamma_{27}{*}ln(\textit{Maturity}) + \gamma_{37}{*}ln(\textit{Prod-Size}) + \varepsilon_{S7} \\
ln(\textit{Support-Costs for PC}) &= \gamma_{08} + \gamma_{18}{*}ln(\textit{Prod-Qual}) + \gamma_{28}{*}ln(\textit{Maturity}) + \gamma_{38}{*}ln(\textit{Prod-Size}) + \varepsilon_{S8} \\
ln(\textit{Support-Costs for Mgt}) &= \gamma_{09} + \gamma_{19}{*}ln(\textit{Prod-Qual}) + \gamma_{29}{*}ln(\textit{Maturity}) + \gamma_{39}{*}ln(\textit{Prod-Size}) + \varepsilon_{S9}
\end{aligned}$$

Table 1. Support Activities in Quality Cost Centers

Support Cost Center	Support Activity
Data Element Dictionary (DED)	Track data base characteristics
Integration	Manage product interfaces and system integration
Documentation	System, user, and support documentation
ADPT Support	Hardware/system technical support
Operations	Operator support for development, test and production
Quality Assurance (QA)	Auditing of processes and products
Configuration Management (CM)	Management of baseline documents and software
Program Control (PC)	Schedule and budget tracking
Management	Overall project management

Because the equations have identical explanatory variables, OLS and GLS (generalized least squares) are identical (Greene 1993), and OLS is used to individually estimate each support cost model. We found significant relationships between software quality and seven of the support activities. In these cases, higher quality resulted in reduced costs with substantial savings in management, quality assurance, and computer operations. Management costs experienced a large marginal impact from improvements in quality due to the involvement of senior managers when errors occurred and the high cost of their time. Quality assurance costs were also affected because fewer defects led to less reinspection, reappraisal, and retesting activities. Operations costs are driven by software testing, production, and maintenance support. Quality has a high marginal impact on operations because defects influence allocation of operations staff support for regression testing and maintenance workload. An interesting finding is that the marginal savings in support activities are significantly greater than the savings found in software development.

4. CURRENT STATUS AND PRELIMINARY RESULTS

We will also examine the effect of quality on corrective maintenance effort, performing a time series analysis of the maintenance workload over time as processes and quality improve. Data collection is complete for the analysis of the effect of software quality on maintenance activities.

5. WHAT WE WILL PRESENT AT ICIS

The presentation at ICIS will include discussion of the results from our four analyses. All data collection is complete, and the first two analyses are complete. We will present results from the two-stage quality and development models, the multivariate support model, and the time series maintenance analysis, and we will draw overall conclusions on the life cycle effects of software quality.

6. IMPLICATIONS AND CONTRIBUTIONS

There are a number of interesting observations that emerge from our analysis. We find that quality improved at our research site with each process improvement initiative, but at a decreasing rate. Much of the effect of quality improvement may be realized from the initial improvement efforts.

We found that higher quality is associated with reduced cycle times and development effort. Results indicate that the savings accrue due to reduced rework. A significant finding was that support activity savings outweigh development savings. It appears that supporting activities indirectly benefit from the investment in development improvements. This suggests future research efforts should focus on how process improvement strategies affect support activities, not just development activities, since substantial savings occur external to the software development team.

We are conducting a comprehensive investigation of the economic value of improving software quality throughout all stages of the software life cycle. This analysis lends insight into areas of opportunity for process improvement and benefits of reduced cycle time and development, support, and maintenance costs.

References

Banker, R.; Chang, H.; and Kemerer, C. "Evidence on Economies of Scale in Software Development," *Information and Software Technology* (36:5,) May 1994, pp. 275-282.

Banker, R. D.; Davis, G., and Slaughter, S. "Software Development Practices and Software Maintenance Performance: A Field Study," *Management Science* (44:4), 1998.

Banker, R. D., and Slaughter, S. "A Field Study of Scale Economies in Software Maintenance," *Management Science*, December 1997.

Bohn, R.E. "Noise and Learning in Semiconductor Manufacturing," *Management Science* (41:1), 1995.

Crosby, P. B. *Quality is Free*, McGraw-Hill, New York, 1979.

Feigenbaum, A. V. *Total Quality Control*, 3rd ed., McGraw-Hill, New York, 1991.

Greene, W. H. *Econometric Analysis*, 2nd ed., MacMillan Publishers, New York, 1993.

Herbsleb, J.; Zubrow, D.; Goldenson, D.; Hayes, W.; and Paulk, M. "Software Quality and the Capability Maturity Model," *Communications of the ACM* (40:6), 1997, pp. 30-40.

Humphrey, W. S. *A Discipline for Software Engineering*, Addison-Wesley, Reading, MA, 1995.

Juran, J., and Gryna, F. *Quality Control Handbook*, 4th ed., McGraw-Hill, New York, 1988.

Kaplan, R. "Must CIM be Justified by Faith Alone?" *Harvard Business Review* (64:2), 1986, pp.87-95.

Krishnan, M. S. *Cost and Quality Considerations in Software Product Management*, unpublished Ph.D. Thesis, Carnegie Mellon University, Pittsburgh, PA, 1996.

McConnell, S. *RAPID DEVELOPMENT: Taming Wild Software Schedules*, Microsoft Press, Redmond, WA, 1998.

Mowrey, D. C. *The International Computer Software Industry: A Comparative study of Industrial Evolution and Structure*, Oxford University Press, Oxford, England, 1996.

Nandakumar, P.; Datar, S. M.; and Akella, R. "Models for Measuring and Accounting for Cost of Conformance Quality," *Management Science* (39:1), 1993, pp. 1-16.

Newfelder, A. M. *Ensuring Software Reliability*, Marcel Dekker, Inc., New York, 1993.

Paulk, M. C. *The Capability Maturity Model: Guidelines for Improving the Software Process*, Addison-Wesley, Reading, MA, 1995.