Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2001 Proceedings

Americas Conference on Information Systems (AMCIS)

December 2001

Investigating Information Systems Failures

Xiaoni Zhang
University of North Texas

John Windsor
University of North Texas

Robert Pavur
University of North Texas

INVESTIGATING INFORMATION SYSTEMS FAILURES

Xiaoni Zhang
BCIS Department
University of North Texas
zhangx@unt.edu

John Windsor
BCIS Department
University of North Texas
windsor@unt.edu

Robert Pavur
BCIS Department
University of North Texas
pavur@unt.edu

Abstract

Companies have invested large amounts of money on information systems development. Unfortunately, not all information systems developments are successful. An extensive body of research stream has focused on software reliability studies for the past two decades and many software reliability growth models have been developed for the estimation of software reliability and the number of errors embedded in the software. Comparatively few studies relate software structure to information systems failures. In this study the relationships among variables business requirements change, software complexity, program size and the error rate in each phase of software development life cycle are tested and interpretations of the hypotheses testing are provided.

Keywords: Information systems failure, software complexity

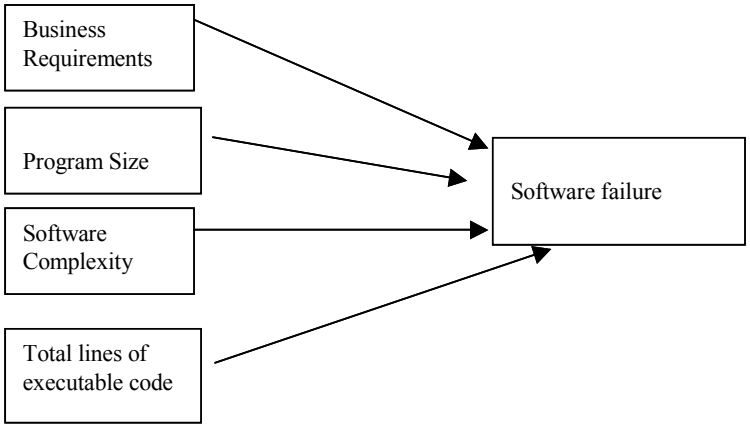
Introduction

Software development includes analysis, design, development and maintenance. Activities carried out and requirements set in each phase determine the final functions of the systems. Errors occurring in any of these phases affect the performance of the systems and affect the costs in the keeping the systems up and running and meeting user's requirements (Huq 2000).

Banker, Davis and Slaughter (2000) developed a two-stage model in which software complexity is a key intermediate variable that links design and development decisions to their downstream effects on software maintenance. Their study suggests that software development practices are related to maintenance performance. Adhikari (1996) stated that software development should put more effort into defining business requirements. As the understanding of the software development process improves, a clearer relationship is found between that software complexity and the location of faults (Mata-Toledo and Gustafson). The literature suggests that software practices in software development life cycle, software complexity, and business requirement are all related to failures (Zhang and Pham 1998).

Previous research has not focused on empirical research investigating a theoretical model which operationalizes variables that cause errors, especially business requirements change. A change in business requirements causes failures. In this study, hypotheses are tested using variables suggested in the literature. These variables include business requirements change, software complexity, and program size. We use a logistic regression model to test the contribution of the variables business requirements change, software complexity, and program size in predicting error rate in the software development life cycle.

Research Framework



The research framework consists of independent variables business requirements, program size, software complexity and dependent variable software failure.

Measurement of the Variables

Table 1. Variables and Measurements

Variables	Measurement
Software Complexity	Is measured by Cyclomatic complexity
Program Size	The total number of lines in the program
Total number of lines in Procedure division	The total number of executable lines in procedure division
Business requirements change	The frequency of the modification that is due to business requirements change
Design failure rate	The frequency of modification that is caused by design stage
Analysis failure rate	The frequency of modification that is caused by analysis stage
Programming failure rate	The frequency of modification that is caused by programming stage

Research Methodology

Longitudinal data analysis was performed in this study by analyzing data collected by professional programmers and auditors in a gas company between 1982-1991. Both qualitative and quantitative data were documented for 100 COBOL programs. These programs include commercial bill, statistical analysis, print programs, report programs, commercial bill cancellation programs, commercial bill zero balance programs, deposit premium programs, list of commercial policies programs, backup statistical system transaction programs, annual warehouse purge of cancelled/expired policies. The smallest program has 101 lines of codes and the largest one has 18488 lines of codes.

To gain insight into the relationship of software failures with attributes about software, 100 COBOL programs are selected for this study from a single company. We are interested in knowing whether the independent variables business requirements change, software complexity and program size are good predictors of error rate in each phase of the software development. Because the independent variables are a mixture of categorical and continuous variables, the multivariate normality assumption will not hold. Logistic regression does not require the strict normality assumptions that standard discriminant analysis assumes. Thus, this statistical procedure is appropriate for analyzing the data associated with software error rate. We present several hypotheses concerning the relationship between independent variables and software failure and use logistic regression to determine if significant relationships existed.

Hypotheses Testing

H_{1a}: There is relationship between design failure rate and program size controlling for business requirements, cyclomatic complexity and total procedure lines.

H_{2a} : There is relationship between design failure rate and business requirements controlling for program size, total procedure lines and cyclomatic complexity.

$H_{3a:}$ There is relationship between design failure rate and cyclomatic complexity controlling for business requirements, program size, and cyclomatic complexity.

$H_{4a:}$ There is relationship between design failure rate and total procedure lines controlling for business requirements, essential complexity, and program size.

H_{5a}: There is relationship between analysis failure rate and business requirements controlling for program size, essential complexity, and cyclomatic complexity.

H_{6a}: There is relationship between analysis failure rate and total procedure lines controlling for program size, business requirements and cyclomatic complexity.

H_{7a}: There is relationship between analysis failure rate and cyclomatic complexity controlling for business requirements, total procedure lines and program size.

H_{8a}: There is relationship between analysis failure rate and program size controlling for business requirements, total procedure lines and cyclomatic complexity.

H_{9a}: There is a relationship between program failure rate and business requirements controlling for program size, total procedure lines, and cyclomatic complexity.

Table 2. Logistic Regression Parameter for Models with Dependent Variable Analysis, Design and Programming Failure Rate

	Analysis	**Design**	**Programming**
Parameter	Pr > ChiSq	Pr > ChiSq	Pr > ChiSq
Intercept	0.0105	0.0062	0.0030
Business requirements	0.0008	<.0001	<.0001
Cyclomatic complexity	0.0410	0.1149	0.6907
Total number of lines	0.0199	0.5095	0.9155
Total lines in procedure	0.0199	0.1619	0.5743

Table 2 presents the significant levels of the four independents variables, business requirements, cyclomatic complexity, total number of line, and total lines in procedure. At analysis stage, all the four independent variables are significant. At design stage only business requirements and cyclomatic complexity are good predicators for failures. At programming stage, only business requirement is significant ($p < 0.0001$).

Conclusions

Practices in different phases of software development life cycle impact the quality, performance and stability of information systems. Software attributes such as software complexity and program size impact the ease of making changes in the program. These attributes also affect software volatility. Our study shows that business requirements have a significant relationship with design failure rate, analysis failure rate and program failure rate. Companies face challenges, regulations and competitions constantly. Opportunities lead to changes in business requirements. Changes in business requirements will inevitably force changes in information systems. Information systems need to be adaptive and flexible to cope with the changes in business

requirements. The findings suggest that management should allocate more resources to analysis and design stages of software development life cycle. Code produced at programming stage is usually decent and not requirement much change. System management needs to monitor environmental change and create incentives for better design.

Software complexity, program size and number of the procedure line play a role in determining analysis failure rate. The more complex a system is, the more likely it will have analysis failure. Typically, the larger the program, the higher the analysis failure rate will be. Extra attention should be given to the large and complex programs during the analysis phase so as to reduce the likelihood of program changes. Developing an adaptive system is the way to have a sustainable system that can live with the changes in business environment.

References

Adhikari, R. "Developers Benefit from a Process Blueprint," *Software Magazine,* (16:3),1996, pp. 59-69.

Banker, R.D., Datar, S.M., Kemerer, C.F., Zweig, D. "Software Complexity and Maintenance Costs," *Communications of the ACM* (36:11), 1993, pp. 81-94

Banker, R. D. Davis, G.B. Slaughter, S. "A Software Development Practices, Software Complexity, and Software Maintenance Performance: A Field Study," *Management Science* (44:4), 1998, pp. 433-450.

Huq, F. "Testing in the Software Development Life-cycle: Now or Later," *International Journal of Project Management* (18:4), 2000, pp. 243-250.

Khoshgoftaar, T. M. Munson, J. C. Lanning, D. L. "Alternative Approaches for the Use of Metrics to Order Programs by Complexity," *Journal of Systems & Software* (24:3), 1994, pp. 211-221.

Mata-Toledo, R. A. Gustafson, D. A. "A Factor Analysis of Software Complexity Measures. *Journal of Systems & Software* (17:3), 1993, pp. 267-273.

Pham, H. *Software Reliability Assessment: Imperfect Debugging and Multiple Failure Types in Software Development*, Technical report, EG&G-RAAM-10737, Idaho National Engineering Laboratory, 1993.

Zhang, X. Pham, H. A software cost model with warranty cost, error removal times and risk costs. *IIE Transactions*. (30:12), 1998, pp. 1135-1142.

Zhang, X. and Pham, H. "An Analysis of Factors Affecting Software Reliability," *Journal of Systems & Software*. 50(1), 2000, pp. 43-56.