AN EMPIRICAL ANALYSIS OF THE EFFECT OF CRITICALITY, COMPLEXITY, AND ORGANISATIONAL INFLUENCE ON SOFTWARE RELIABILITY

Paul L. Bowen, Ph.D., CPA[1], Jon Heales PhD[2] Anne E. Speed, MIS[3]
[1]Department of Commerce
University of Queensland
Brisbane, Queensland 4072
Australia
Phone: +61 7 3365 6584; Fax: +61 7 3365 6788
Internet: bowen@lorien.commerce.uq.edu.au
[2]The University of Queensland
Brisbane, Queensland 4072
Australia
Phone: +61 7 3365 6433; Fax: +61 7 3365 6788
Internet: Heales@commerce.uq.edu.au
[3]The University of Queensland
Brisbane, Queensland 4072
Australia

ABSTRACT

This paper is motivated by a desire to improve software reliability, in particular, the reliability of software that impacts the operational and financial viability of organisations. We examine the effects of Criticality, Complexity, and Organisational Influence on information systems reliability. Questionnaires were used to gather quantitative data for statistical analysis. Post-hoc in-depth interviews were used to help explain results of the statistical analysis. Surprisingly, no associations were observed between Reliability and Criticality or between Reliability and Complexity. A positive relationship was found, however, between system Reliability and Organisational Influence. The interviews indicated that organisations mitigated the potential negative effects of Complexity through additional planning, and achieved more reliable software by assigning more competent project managers. They managed Criticality by assigning more competent project managers to more critical systems. The significant relationship between system Reliability and Organisational Influence indicates that IS managers respond to internal political pressures. This result implies that senior management should take steps to ensure that excessive Organisational Influence does not cause IS managers to misallocate resources. For example, for each major project, the IS steering committee can determine the desired level of reliability, appoint project mangers with the appropriate skill set, and periodically communicate with these project managers about the activities used to achieve each system's desired level of reliability.

The authors are grateful to Fiona Rohde, Ron Weber, the anonymous reviewers, and the editor for their helpful comments.

INTRODUCTION

This paper is motivated by the desire to improve levels of software reliability[1], in particular, software that impacts the operational and financial viability of an organisation. Software errors[2] incur three types of costs: the cost associated with the error itself, the subsequent cost of correcting the error, and the cost associated with the loss of stakeholders' confidence and goodwill[3]. Because of the economic impacts information systems have on organisations, a greater understanding of factors affecting reliability can help minimise costly errors resulting from inadequate levels of software reliability.

Project managers control many factors that affect an information system's quality during development, e.g., they allocate resources, exercise management control, and determine testing regimes. When making decisions about these control factors, the project managers take into account a number of factors. This paper examines the effects of three of these factors on information systems reliability:

---

1 Software reliability is the probability that the software will execute without failure or will perform successfully on demand (Rook, 1990).

2 Shooman (1983) defined software errors as problems in the external operation of a system caused by internal software faults.

3 For example, Westpac Bank blamed insufficient software testing for a fault that caused automatic teller machines to allow customers to overdraw their accounts. This error cost Westpac several million dollars, incurred unexpected software maintenance costs, and resulted in unfavourable publicity (Software Engineering Notes, 1991).

Criticality[4], Complexity[5], and Organisational Influence[6]. The effects are examined at three levels: IS Management level, Project Management level, and Programmer level.

Project managers, as agents of the steering committee and the system owners, should ensure that information systems critical to the organisation's success have high reliability. A positive relationship between Criticality and Reliability indicates that organisations can potentially improve their benefit/cost relationships by allocating more resources to improving the reliability of critical systems. That is, information system costs would increase but would be outweighed by the lower risk of losses or missed opportunities. Second, greater software complexity has long been associated with lower reliability (see, e.g., Halstead 1977, 84-91; Ropponen and Lyytinen, 2000). This paper explores this relationship and increases our understanding of the relationship between Complexity and Reliability. Management can use this understanding to improve information systems reliability in increasingly complex systems environments. Third, in a perfect world, organisational influence should not affect reliability, i.e., subject to complexity constraints, criticality rather than politics should determine the reliability of organisational information systems. A positive relationship between Organisational Influence and Reliability indicates a potential misapplication of the organisation's resources that senior management should address.

Surveys and interviews in three organisations were used to gather evidence about the associations between information systems Criticality, Complexity, Organisational Influence, and Reliability. Questionnaires were used to gather quantitative data for statistical analysis, and post-hoc in-depth interviews were used to supplement and explain the results of the statistical analysis. Surprisingly, no associations were observed between Reliability and Criticality or between Reliability and Complexity. A positive relationship was found, however, between system Reliability and Organisational Influence. Post-hoc interviews were undertaken to try to explain the unexpected statistical results of the survey. For example, the post-hoc interviews revealed that mitigating factors may explain the lack of a statistical association between Complexity and Reliability. For example, the organisations had, by additional planning, achieved more reliable software by assigning more competent project managers. That is, they managed Criticality by assigning more competent project managers to more critical systems. The research found a statistically significant relationship between system Reliability and Organisational Influence that indicates IS managers respond to internal political pressures.

THEORY AND HYPOTHESES

Complexity, Criticality, and Organisational Influence have very different types of relationships with software reliability. Complexity has a direct *technical* association with software reliability, i.e., more complex software is typically less reliable, or equivalently, organisations must expend more computing and information systems resources to achieve a fixed level of reliability for more complex software. The link between criticality and software reliability is primarily *economic*, i.e., economic rationality dictates that organisations should expend more resources on reliability for software that is critical to organisational success. Organisational influence and software reliability exhibit a *behavioural* relationship, i.e., like other members of an organisation, information systems personnel are subject to political pressure and are likely to expend greater efforts on information systems for people or groups they perceive as more powerful or influential.

Reliability

Musa et al. (1987, 15) define software reliability as the probability of failure-free operation of an information system in a specified time frame and environment. They note that their view of reliability represents a user-oriented view of software quality (Musa et al. 1987, 5). Chillarege (1996) takes a similar view when he states that software failures occur when users' expectations are not met or the users are unable to perform useful work with the software. Furthermore, Pitt et al. (1995) and Watson et al. (1998) assert that reliability, defined as the dependable and accurate performance of the promised service or function, is one of five major dimensions of information systems effectiveness.

---

4 Criticality refers to the importance of the software to the organisation. For example, reservation systems are extremely critical to airlines because the airlines cannot conduct business without them.

5 Complexity depends on the size, intricacy, and sophistication of the software.

6 Organisational Influence refers to the power structure within the organisation, i.e., the capacity of an individual or group to modify the conduct of the other individuals or groups.

IS Management Control of Reliability

The Capability Maturity Model (Sallis, Tate et al., 1995; Herbsleb et al. 1997) presents an overall approach that management can use to improve software quality. For example, key practices used to determine an organisation's position on the software process maturity scale include projecting design errors, test errors, and remaining errors; analysing error causes; controlling design changes, requirements changes, and code changes; training developers and review leaders; and maintaining and analysing numerous metrics about software process improvement (Dekleva and Drehmer 1997).

Software reliability is determined by events that occur during all phases of the systems development life cycle.[7] CASE tools can help reduce errors throughout the process (Orlikowski 1993). For example, upper CASE tools facilitate communication, documentation, and co-ordination during the requirements elicitation and definition stages.

Lower CASE tools can help reduce errors translating the requirements specifications into the target programming language. Modern programming techniques, including structured programming (Linger et al. 1979) and program verification (Dyer 1992), reduce initial software faults. Organisations use various forms of testing to detect remaining faults, and to ensure adequate levels of software quality and reliability (Beizer 1984). Increasing software reliability by using rigorous programming techniques or performing extensive testing may increase the costs of developing that software but will reduce the expected costs of software errors (Zhao and Xie 1993; Yang and Chao 1995).

Complexity

Software complexity refers to those characteristics that make the software difficult to create, understand, or change (Curtis et al. 1979). Increasing complexity places progressively greater cognitive demands on people and reduces their performance (Campbell 1988). The most cited software complexity measures are those proposed by McCabe (1976) and Halstead (1977). McCabe's cyclomatic complexity computes the maximum number of linearly independent paths in the program, i.e., it focuses on the number of decision points in the program. Halstead defined a number of complexity measures all of which are related to the number of operators (keywords of the programming language) and operands (variables). Of particular relevance to this study, he demonstrated that his complexity measure for mental discriminations (programmer effort) is positively related to program error rates (Halstead 1977, 84-91).

More recent empirical studies also provide evidence of the negative impacts of complexity. Grady (1993) found that post-release defect density was highly correlated with structural complexity. One reason for this relationship was that significant portions of the code were not tested prior to release because of the difficulty of testing all possible paths through complex code. Card and Glass (1990) found a close correlation between defect density and design complexity.[8] Furthermore, empirical studies at a large commercial bank and at a mass merchandising retailer provide compelling evidence of the negative effects of complexity on maintenance activities (Banker et al. 1993; Banker et al. 1998). Ropponen and Lyytinen (2000) found that Managing Project Complexity was a component of software development risk in both scheduling and timing, and in resource usage and performance.

Thus, software complexity poses a major obstacle to producing and maintaining reliable software. This gives rise to the following hypothesis:

H1: A negative relationship exists between software Reliability and the Complexity of that software.

Criticality

Information systems are critical to the success of most organisations. Indeed, many organisations attempt to use their information systems to create competitive advantages (see, e.g., Mata et al. 1995).

---

[7] The phases of the Systems Development Life Cycle (SDLC) are problem definition, feasibility study, analysis, general systems definition, detailed systems definition, implementation, and evaluation and maintenance. In this view, the implementation stage includes programming and (module) testing, and systems and acceptance testing (Leitch and Davis, 1992).

[8] Design complexity is a function of structural complexity, data complexity, and procedural complexity.

For example, Chrysler Corporation depends on its EDI system to achieve benefits of over $100 per vehicle thereby helping it remain competitive and profitable (Mukhopadhyay et al. 1995). Perhaps the epitome of the criticality of information systems is SABRE, American Airline's reservation system (Copeland and McKenney 1988). Not only the airlines themselves, but also related organisations such as travel agencies, are totally dependent on the continuous, accurate, and reliable functioning of the reservation systems.

Sherer and Paul (1993) developed a model for assessing the expected financial consequences of software errors. Their model is a function of the expected financial consequence of hazards that can result from each possible use, weighted by the probability each hazard will occur when the information system is employed for that use, and adjusted for the extent to which the system is expected to perform that use during the time period of interest. *Ceteris paribus*, the expected net benefits of reliability are greater for systems that are used by more stakeholders and for systems that relate to core functions of the organisation. Hence, wealth maximising organisations would seek to increase the reliability of those software applications that are most critical to their success, i.e., the information systems most closely aligned with the organisation's goals, objectives, and critical success factors. This gives rise to the following hypothesis:

H2: A positive relationship exists between Software Reliability and Criticality of that software to the success of the organisation.

Organisational Influence

Another possible obstacle in producing software with the appropriate level of Reliability is the influence of organisational sub-units. Organisational Influence is the capacity of individuals or groups to obtain the outcome they prefer in particular situations (Salancik and Pfeffer 1977). Persons or groups with organisation influence can induce others to perform tasks they might not otherwise perform (Lansbury and Spillane 1991, 98). Although organisational sub-units should align themselves with the goals of the entire organisation, the possibility exists for organisational sub-units to exert their influence in a manner that is in conflict with the goals of the organisation as a whole (Raghunathan and Raghunathan 1992). For example, organisational units with substantial influence can cause the information systems department to expend resources on projects that do not match the optimal priorities of the organisation as a whole.

Alternatively, organisational sub-units with less influence may not be able to obtain information systems department resources for projects that, if implemented, would make substantial contributions to the objectives of the overall organisation. IS personnel may expend more effort or exercise greater care when developing software for clients with greater Organisational Influence. A positive relationship is likely to exist between software Reliability and the Organisational Influence of the requesting department. This gives rise to the following hypothesis:

H3: A positive relationship exists between software Reliability and the Organisational Influence of the requesting department.

Figure 1 shows the effect on Software Reliability of Complexity, Criticality, and Organisational Influence.

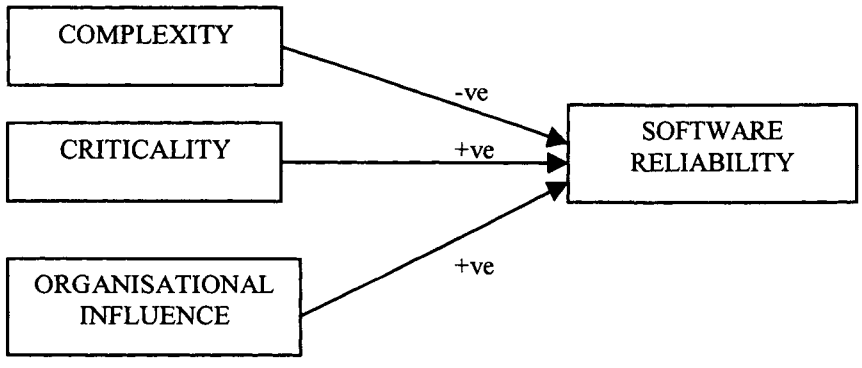

Figure 1. The Effects of Complexity, Criticality and Organisational Influence on Software Reliability.

METHODOLOGY

To obtain information on software reliability, organisations considered likely to develop rather than purchase a substantial portion of their application software were identified. IS managers from three organisations agreed to participate. Details of the organisations, projects, project managers, and programmers are provided in Table 1. The Land Titles Department provides an accurate, secure system for recording property ownership and other interests in freehold and state leasehold land. The system enables protection of the rights and interests of proprietors and the community. It also supplies efficient, readily available services for accessing land registry information. The Insurance Company offers a wide range of insurance options including home and its contents insurance, motor vehicle and compulsory third party coverage, boat coverage (both at sea and ashore), life insurance, income protection and trauma insurance, and commercial and rural business insurance. The Government IT Service Organisation provides a wide range of services and consultancies across the full spectrum of IT activity from designing and executing multimedia web pages to providing fully integrated IT solutions for an entire organisation. It is the largest information technology services provider in the state.

Table 1: Organisations, Projects, and Personnel Surveyed

	Land Titles Department	Insurance Company	Government IT Service Organisation	Totals
IT Managers	1	1	1	3
Projects	5	6	4	15
Project Managers*	5	2	2	9
Programmers	7	8	5	20

*Some Project Managers were in charge of more than one project

Two forms of analysis were used. First, quantitative analyses were performed on questionnaire responses to statistically test the significance of the associations in the model (see Figure 1). The questionnaires are summarised in Appendix A. Second, because of the small sample size and to obtain a richer data set, in-depth interviews were conducted to supplement and explain the quantitative results. These interviews were conducted with the three IS managers and five project managers. The project managers were identified by the IS managers.[9] At least one project manager from each organisation was interviewed. Information was obtained from more than one level to provide different views and perceptions of the effects that Criticality, Complexity, and Organisational Influence may have on Software Reliability (Gough, 1993; Moynihan, 1990; Nath, 1989). Each interview lasted approximately one hour and consisted of open-ended questions eliciting each participant's opinions about:

- the Reliability of the software they develop,
- the Complexity and Criticality of the software,
- their programming techniques and testing procedures, and
- the factors that would affect the programming techniques, testing procedures, and overall reliability of individual software applications.

All participants allowed their interviews to be tape-recorded, agreed to be available for follow-up questions,[10] and requested copies of the results of the study.

After each survey interview, each participant was asked to complete a set of questionnaires (see Appendix A). Each participant read the questionnaires and any questions the participant had were answered. IS managers completed questionnaires about their organisation's programming techniques and testing procedures, and about the ability of each of their project managers. They also completed questionnaires about the Criticality, Complexity, Organisational Influence, and Reliability of several application software projects.

The project managers completed questionnaires about their organisation's programming techniques and testing procedures, and about the ability of each of their programmers. The project managers also

[9] Interviewing project managers identified by the IS managers introduces the possibility of bias and care must be taken when placing reliance on the results.

[10] Two follow-up phone calls were made.

completed questionnaires about the Reliability, Criticality, Complexity, and Organisational Influence of each application software project they managed.

All participants returned completed questionnaires. The three participating organisations returned questionnaires on a total of 15 separate application software projects. The participants rated the Reliability, Criticality, Complexity, and Organisational Influence associated with each software project on a seven-point Likert scale (see Appendix A). The IS managers rated the project managers in terms of their ability to manage software projects on the dimensions of Reliability, Criticality, Complexity, and Organisational Influence of the requesting department.[11] Project managers rated their programmers on the Reliability of their software, and the programmer's ability to handle critical projects, complex projects, and projects for influential members of the organisation. Responses for these ratings were also on seven-point Likert scales. To identify any weaknesses in the questionnaires[12] including any difficulties by the participants, the responses on each questionnaire were compared with the qualitative statements made by each participant during their interview.

Figure 2 shows the summary data for each variable by organisation. Table 2 lists the constructs and variable names. Because of the small sample size, the data were also analysed using non-parametric statistics. Both Pearson (parametric) and Spearman (non-parametric) correlation coefficients are shown in Appendix B. Results from these two methods are essentially identical and are consistent with the regression analyses reported in the results section.

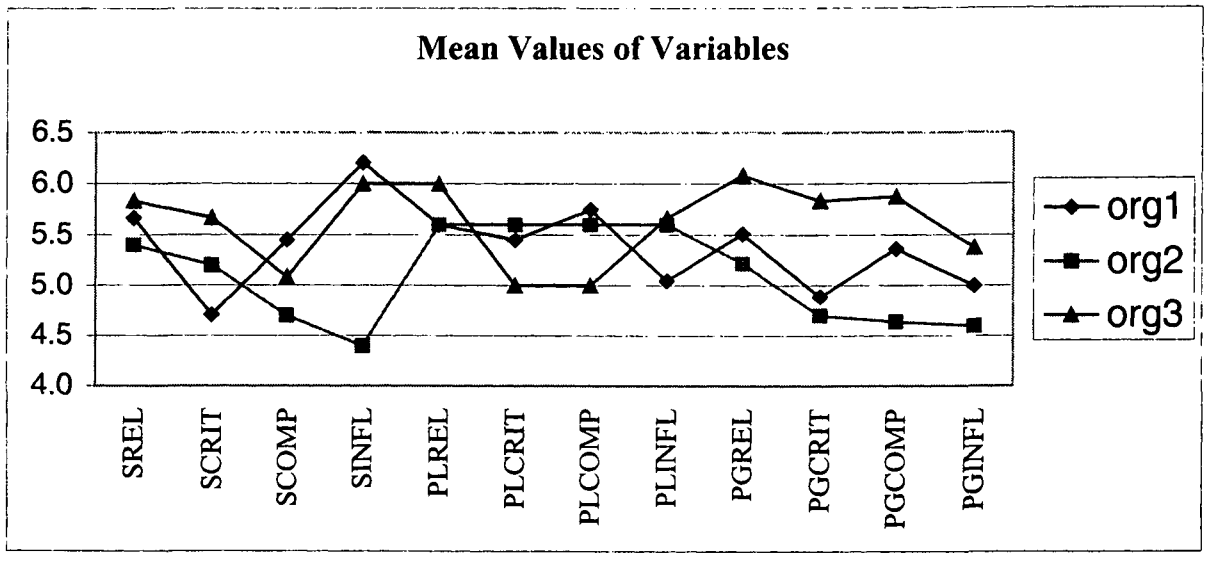

Figure 2. Summary Data of Variables by Organisation.

Table 2: Table of Constructs and Variable Names

Variable	Construct
SREL	System Reliability
SCRIT	System Criticality
SCOMP	System Complexity
SINFL	System Organisational Influence
PLREL	Project Manager Reliability
PLCRIT	Project Manager Criticality
PLCOMP	Project Manager Complexity
PLINFL	Project Manager Organisational Influence
PREL	Programmer Reliability

[11] Quantitative measures have been developed for both software reliability (e.g., Jelinski and Moranda, 1972; Littlewood, 1981; Musa, 1975; Schick and Wolverton, 1973) and complexity (e.g., Halstead, 1977; McCabe, 1976). Unfortunately, none of the organisations contacted had the necessary data to calculate either of these measures.

[12] Prior to their use in this study, the questionnaires were reviewed by three persons each having a minimum of four years of IS experience in diverse industry groups.

PCRIT	Programmer Criticality
PCOMP	Programmer Complexity
PINFL	Programmer Organisational Influence

RESULTS

The relationships of Reliability to software Criticality, Complexity, and Organisational Influence were analysed using OLS regressions. Table 3 contains the results of analysing the model:[13]

$$\text{Reliability} = \beta_0 + \beta_1 * \text{Criticality}(+) + \beta_2 * \text{Complexity}(-) + \beta_3 * \text{Organisational Influence}(+)$$

Criticality

The analysis did not reveal the expected positive relationship between Reliability and Criticality (**H1**). A review of the transcripts of the interviews disclosed that all three organisations used structured programming techniques and rigorous testing methodologies. Although the managers in one of the organisations stated that critical projects would cause them to pay closer attention to detail, managers at all organisations indicated that programming and testing procedures were essentially the same for all projects. The IS managers did, however, make comments indicating that they considered criticality when assigning project managers and other IS personnel to specific projects. That is, the IS managers assigned more capable project managers to critical projects to enhance the reliability of critical software systems. Hence, comments by the IS managers tend to support the hypothesised positive relationship between Reliability and Criticality but the relationship is not supported by the statistical analysis. A number of reasons may explain this apparent conflict. First, the correlation analyses in Appendix B reveal significant positive relationships between Criticality and Complexity for IS managers, project leaders, and programmers. Therefore, the lack of a positive statistically significant relationship between Reliability and Criticality may be because the critical software is more complex. Second, this research may not have used sufficiently accurate measurements of Reliability or Criticality to reveal statistically significant results. Third, using essentially the same programming and testing procedures may mitigate the greater attention to detail and the assignment of more capable personnel.

TABLE 3: Effect of Criticality, Complexity, and Organisational Influence on Software Reliability[14]

Source	DF	Mean Square	F Value	Pr > F	Parameter Estimate
Model	3	0.5682	3.11	0.0708	
Error	11	0.1828			
Criticality	1	0.0006	0.00	0.9551	-0.0068
Complexity	1	0.0157	0.09	0.7752	0.0256
Org. Influ.	1	1.4449	7.91	0.0169	0.3073

$R^2 = 0.46$

Complexity

Quantitative analysis of the questionnaire data also did not reveal the expected negative relationship between Reliability and Complexity (**H2**). Post-hoc interviews revealed that although the organisations

[13] The direction of the expected impact of each independent variable is in parenthesis.

[14] Reported mean squares are based on Type III sum of squares.

followed the same type of programming and testing procedures for all projects, two of the organisations stated that complex projects required more planning. This additional planning included increased attention to analysis and construction of specifications and more careful planning and organisation of programming tasks before actually beginning coding and testing tasks. One manager stated, "With a complex program, you need to map out the program and write a test plan first." Managers also indicated that they considered size and Complexity when assigning personnel to projects. Using personnel assignments to compensate for size and Complexity helped explain the lack of a statistically significant relationship between Complexity and Reliability. Other possible explanations include measurement problems, or the greater attention to planning and organisation.

Organisational Influence

The expected positive relationship between Reliability and Organisational Influence (**H3**) was highly significant (p-value < 0.02, two-tailed test). Project managers at one of the organisations were adamant that Organisational Influence did not change their programming and testing procedures. The IS manager at that organisation, however, does pay more attention to projects for influential clients and frequently mentioned Organisational Influence during the interviews. Managers at the other two organisations stated that they altered the nature or extent of their programming and testing procedures for projects requested by clients with high levels of influence. One of these organisations was explicitly market driven with the IS manager stating that project managers would perform additional testing and other reliability enhancing procedures if the client was willing to pay for it. The effect of Organisational Influence appears to be different in the three organisations.[15]

Post-hoc Analysis of Organisational Influence

The relationship between Reliability and Organisational Influence (**H3**) was supported both statistically and by the case study interviews in all three organisations. The level of influence that clients were able to exert, however, did vary across organisations, i.e., influence was important in all three organisations but was more important in some of the organisations than in others. The interaction of influence with organisation is also significant (p-value = 0.0319) for a regression with Reliability as the dependent variable and Organisational Influence, Organisation, and the interaction of Organisational Influence and Organisation as the independent variables.

Post-hoc Analysis of Project Manager Competency

A recurrent comment from IS managers was that, to increase software reliability, they assigned more competent project managers. The results of an OLS regression of the relationship of software reliability with project managers' perceived abilities to produce reliable software and to deal with Criticality, Complexity, and Organisational Influence support this view (see Table 4). Because IS managers assigned more competent project managers to improve reliability, one might expect project managers to follow a similar strategy with programmers. The results of an OLS regression of the relationship of software reliability with programmers' perceived abilities to produce reliable software and to deal with Criticality, Complexity, and Organisational Influence did not indicate any statistically significant relationships (see Table 5, $F_{(4,10)} = 1.10$, p-value=0.4084). Follow-up conversations with the participants indicated that one organisation tried to assign more competent programmers to obtain more reliable software and in another organisation programmers were assigned from a pool. The third organisation considered coding as a generic process and relied on the project manager and other procedures to obtain the desired reliability levels.

[15] The Pearson correlation coefficient between reliability and influence for the first organisation is 0.4677 but 0.6124 and 0.9956 for the other two.

TABLE 4: Effect of Project Managers' Abilities on Software Reliability

Source	DF	Mean Square	F Value	Pr > F	Parameter Estimate
Model	4	0.6231	5.10	0.0168	
Error	10	1.2226			
Reliability	1	0.5591	4.57	0.0582	0.6258
Criticality	1	0.7422	6.07	0.0335	-1.1407
Complexity	1	0.4216	3.45	0.0930	0.7409
Org. Influ.	1	0.0072	0.06	0.8132	0.0629

$R^2 = 0.67$

TABLE 5: Effect of Programmers' Abilities on Software Reliability

Source	DF	Mean Square	F Value	Pr > F	Parameter Estimate
Model	4	0.2837	1.10	0.4084	
Error	10	0.2580			
Reliability	1	0.1210	0.47	0.5091	0.2685
Criticality	1	0.3744	1.45	0.2561	0.3220
Complexity	1	0.0111	0.04	0.8397	-0.0650
Org. Influ.	1	0.1535	0.59	0.4584	-0.3366

$R^2 = 0.31$

Future Research

The research has focussed on the relationships in the model outlined in Figure 1. The case study data has helped to explain in more depth the relationships in the model. On a broader front, there are a number of organisational issues that could impact on the model, and the results of this research.
Future research can, for example, examine the impact on Organisational Influence and/or Criticality that the following factors may have:

- management attitude and how IT activities are managed;
- the impact of outsourcing, resource allocation, and project management; and
- the type of organisation and its maturity in IT terms (CMM stage); and the degree of centralisation/de-centralisation.

CONCLUSION

This study investigated the relationship between application systems Reliability and its Criticality, Complexity, and Organisational Influence. No statistically significant relationship was found for either Criticality or Complexity. A statistically significant association was found between Organisational Influence and Reliability. Interviews and follow-up conversations indicated that organisations mitigated the potential negative effects of complexity by additional planning and that they achieved more reliable software by assigning more competent project managers.

Because of the lack of a significant relationship between Reliability and Criticality and because reliability is managed at project manager level by the assignment of project managers with varying degrees of capability, senior management (e.g., via a steering committee) need to determine the criticality of application software projects so that they can control reliability through the assignment of project managers. These managers should install and monitor control systems to ensure critical software applications attain appropriate levels of reliability.

The lack of a significant relationship between Reliability and Complexity indicates that IS managers have developed effective strategies for controlling software complexity. Because a primary component of these strategies includes assigning more competent project managers, organisations need to devote significant effort to attracting and developing high quality project managers. Organisations might also examine the possible benefits of assigning more competent programmers to important software projects.

The significant effect between Reliability and Organisational Influence indicates that IS managers respond to internal political pressures. This result implies that senior management should take steps to ensure that excessive organisational influence does not cause IS managers and staff to misallocate resources. Management should ensure that individual software clients align their goals with those of the entire organisation and that IS managers achieve appropriate levels of reliability for software projects requested by clients with differing amounts of organisational influence.

The appointment of project managers by senior IS management indicates their concern for reliability. This concern is not reflected by the appointment of programmers by project managers. Senior IS managers and project managers need to be aware of the need to control software reliability at the programmer level. Where reliability is a concern, steps should be taken at the project management level to assign programmers with the appropriate level of expertise. Furthermore, IS managers, project leaders, and programmers should ensure that the development methodology and all components of the SDLC are appropriate for the level of reliability desired.

REFERENCES

Banker, R.D., S.M. Datar, C.F. Kemerer, and D. Zweig. 1993. "Software Complexity and Maintenance Costs," **Communications of the ACM**, vol. 36, no. 11 (November) 81-94.

Banker, R.D., G.B. Davis, and S.A. Slaughter. 1998. "Software Development Practices, Software Complexity, and Software Maintenance Performance," **Management Science**, vol. 44, no. 4 (April) 433-450.

Basili, V. R. (1979), in Shooman, M. L. (1989) **Software Engineering: Design / Reliability / Management**

Beizer, B. 1984. **Software Testing Techniques**. New York: Van Nostrand Reinhold.

Campbell, D.J., 1988. "Task Complexity: A Review and Analysis," **Academy of Management Review**, vol. 13, no. 1, 40-52.

Card, D. and Glass, R. **Measuring Software Design Quality**. Prentice-Hall, Englewood Cliffs. N.J., 1990. 55.

Chillarege, R., 1996. "What Is Software Failure?" **IEEE Transactions on Reliability**, vol. 45, no. 3 (September) 354-355.

Copeland, D.G., and J.L. McKenney, 1988. "Airline Reservation Systems: Lessons from History," **MIS Quarterly**, vol. 12, no. 3 (September) 353-370.

Curtis, B., S.B. Sheppard, P. Milliman, M.A. Borst, and T. Love. 1979. "Measuring the Psychological Complexity of Software Maintenance Tasks with the Halstead and McCabe Metrics," **IEEE Transactions on Software Engineering**, vol. SE-5, no. 2, 96-104.

Dekleva, S., and D. Drehmer, 1997. "Measuring Software Engineering Evolution: A Rasch Calibration." **Information Systems Research**, vol. 8, no. 1 (March) 95-104.

Dyer, M. 1992. **The Cleanroom Approach to Quality Software Development**. New York: John Wiley & Sons.

Gough, C.A. 1993. **Business and Information Strategy Alignment: A Study of Queensland Government Agencies**. Unpublished Thesis: University of Queensland.

Grady, R. B. 1993. "Practical Results from Measuring Software Quality" **Communications of the ACM**. **36**(11), 62-69.

Halstead, M.H. 1977. **Elements of Software Science**. New York: Elsevier North-Holland.

Herbsleb, J., D. Zubrow, D. Goldenson, W. Hayes, and M. Paulk, 1997. **Software Quality and the Capability Maturity Model**, vol. 40, no. 6 (June) 30-40.

Jelinski, Z., and P.B. Moranda. 1972. "Software Reliability Research." in **Statistical Computer Performance Evaluation**. (W. Freiberger, Editor) New York: Academic: 465-484.

Lansbury, R.D., and R. Spillane. 1991. **Organisational Behavior: The Australian Context**. Melbourne: Longman Cheshire.

Leitch, R.A., and K.R. Davis. 1992. **Accounting Information Systems: Theory and Practice**. 2nd ed. Englewood Cliffs, N.J.: Prentice Hall.

Linger, R.C., H.D. Mills, and B.I. Witt. 1979. **Structured Programming: Theory and Practice**. Reading, Mass.: Addison-Wesley.

Littlewood, B. 1981. "Stochastic Reliability Growth: A Model for Fault Removal in Computer Programs and Hardware Design." **IEEE Transactions on Reliability**. **30**(4): 313-320.

Mata, F.J., W.L. Fuerst, and J.B. Barney, 1995. "Information Technology and Sustained Competitive Advantage: A Resource-Based Analysis," **MIS Quarterly**, vol. 19, no. 4 (December 1995) 487-505.

McCabe, T.J. 1976. "A Complexity Measure." **IEEE Transactions on Software Engineering**. **2**(4): 308-320.

Moynihan, T. 1990. "What Chief Executive and Senior Managers want from their IT Departments." **MIS Quarterly** (March): 15-25.

Mukhopadhyay, T., S. Kekre, and S. Kalathur, 1995. "Business Value of Information Technology: A Study of Electronic Data Interchange," **MIS Quarterly**, vol. 19, no. 2 (June) 137-156.

Musa, J.D. 1975. "A Theory of Software Reliability and its Application." **IEEE Transactions on Software Engineering**. **1**(3): 312-327.

Musa, J.D., A. Iannino, K. Okumoto, 1987. **Software Reliability: Measurement, Prediction, Application**. New York: McGraw-Hill.

Nath, R. 1989. "Aligning MIS with the Business Goals." **Information and Management** (16): 71-79.

Orlikowski, W., 1993, "CASE Tools as Organizational Change: Investigating Incremental and Radical Changes in Systems Development," **MIS Quarterly**, vol. 17, no. 3 (September) 309-340.

Pitt, L.F., R.T. Watson, and C.B. Kavan, 1995. "Service Quality: A Measure of Information Systems Effectiveness." **MIS Quarterly**, vol. 19, no. 2 (June) 173-187.

Raghunathan, B., and T.S. Raghunathan, 1992. "The Relationship Between Organizational Factors and Accounting for Information Systems Costs," **Journal of Information Systems,** vol. 6, no. 2 (Fall) 115-126.

Rook, Paul ed. (1990): **Software Reliability Handbook**. Elsevier Applied Science, London.

Ropponen, J. and K. Lyytinen (2000). "Components of Software Development Risk: How to Address Them? A Project Manager Survey." **IEEE Transactions on Software Engineering**, Vol. 26, No 2: 98-111

Salancik, G.R., and J. Pfeffer. 1977. "Who Gets Power - And How They Hold on to It: A Strategic-Contingency Model of Power." **Organizational Dynamics** (Winter).

Sallis, P., G. Tate, et al. (1995). **Software Engineering: Practice, Management, Improvement,** Addison-Wesley

Sherer, S.A., and J.W. Paul, 1993. "Focusing Audit Testing on High Risk Software Modules: A Methodology and Application," **Journal of Information Systems**, vol. 7, no. 2 (Fall) 65-84.

Schick, G.J., and R.W. Wolverton. 1973. "Assessment of Software Reliability." **Proceeding Operations Research**. Wurzburg-Wien: Physica-Verlag: 395-422.

Shooman, M.L. 1983. **Software Engineering: Design, Reliability, and Management**. New York: McGraw-Hill.

Yang, M. C., and Chao, A. 1995. "Reliability-Estimation & Stopping-Rules for Software Testing, Based on Repeated Appearances of Bugs." **IEEE Transactions on Reliability**. **44**(2): 315-321.

Watson, R.T., L.F. Pitt, and C.B. Kavan, 1998. "Measuring Information Systems Service Quality: Lessons From Two Longitudinal Case Studies," **MIS Quarterly**, vol. 22, no. 1 (March) 61-79.

Zhao and Xie, M. 1993. "Robustness of Optimum Software Release Policies." **IEEE Transactions on Reliability**: 42(2): 218-225.

Appendix A

Questionnaires

The following appendix contains relevant extracts from the questionnaires provided to EDP Managers, Project Managers, and Programmers. Questions were asked of IT Managers, Project Managers, and Programmers, and about the projects with which they were associated. The Questionnaire is titled "The Effect of Programming and Testing Methodologies on Software Reliability"

The following questions were asked of IT Managers (n=3), Project Managers (n=5), and Programmers (n=20):

Question	Scale
How effective do you believe these methodologies are in producing reliable software ?	7 point Likert
How important is the programming process for ensuring software reliability?	7 point Likert
How important is the testing phase for ensuring software reliability?	7 point Likert
List the major departments or groups and circle the number corresponding to their level of organisational influence. (Likert scale for each department)	7 point Likert

The following questions were asked of IT Managers and Project Managers:

How ***reliable*** is the resulting software? (Does it have a low failure rate?)	7 point Likert
Rank their (subordinates) ability to handle projects ***critical*** to the organisation	Rank
Rank their ability to handle large or ***complex*** projects, where complexity is associated with lines of code or function points.	Rank
Rank their ability to handle projects for very ***influential*** members of your organisation	Rank

The following questions relating to a total of 15 projects were asked of IT Managers, Project Managers, and Programmers:

How reliable was the resulting software?	7 point Likert
How critical was the application?	7 point Likert
How complex was the application?	7 point Likert
List the major departments or groups and circle the number corresponding to their level of organisational influence.	7 point Likert

APPENDIX B

Correlation Analyses

The following table shows the Pearson parametric correlation coefficients and p-values for the data variables. Table 2 above shows the legend for the variable names.

Table B1. Pearson Correlation Coefficients

	SREL	SCRIT	SCOMP	SINFL	PLREL	PLCRIT	PLCOMP	PLINFL	PGREL	PGCRIT	PGCOMP
SREL	1.000										
p-value	0.000										
SCRIT	-0.040	1.000									
p-value	0.889	0.000									
SCOMP	0.206	0.505	1.000								
p-value	0.462	0.055	0.000								
SINFL	0.674	-0.098	0.205	1.000							
p-value	0.006	0.729	0.464	0.000							
PLREL	0.630	-0.129	-0.150	0.628	1.000						
p-value	0.012	0.646	0.594	0.012	0.000						
PLCRIT	-0.459	-0.285	-0.091	-0.199	-0.118	1.000					
p-value	0.085	0.303	0.746	0.477	0.676	0.000					
PLCOMP	-0.191	-0.453	-0.046	0.025	-0.053	0.862	1.000				
p-value	0.496	0.090	0.870	0.929	0.850	0.000	0.000				
PLINFL	0.135	-0.009	-0.239	0.065	0.538	0.257	0.089	1.000			
p-value	0.632	0.975	0.392	0.818	0.039	0.356	0.752	0.000			
PGREL	0.447	0.092	0.213	0.308	0.287	-0.670	-0.522	-0.153	1.000		
p-value	0.095	0.745	0.447	0.264	0.300	0.006	0.046	0.586	0.000		
PGCRIT	0.484	0.311	0.310	0.336	0.328	-0.413	-0.316	0.160	0.827	1.000	
p-value	0.068	0.259	0.261	0.221	0.233	0.126	0.251	0.569	0.000	0.000	
PGCOMP	0.382	0.320	0.552	0.364	0.159	-0.504	-0.421	-0.158	0.878	0.869	1.000
p-value	0.160	0.246	0.033	0.182	0.571	0.055	0.119	0.575	0.000	0.000	0.000
PGINFL	0.335	0.263	0.504	0.269	0.141	-0.360	-0.284	-0.012	0.836	0.882	0.893
p-value	0.222	0.344	0.055	0.332	0.617	0.188	0.305	0.965	0.000	0.000	0.000

The following table shows the Spearman (non-parametric) correlation coefficients and p-values for the data variables. Table 2 above shows the legend for the variable names.

Table B2. Spearman Correlation Coefficients

	SREL	SCRIT	SCOMP	SINFL	PLREL	PLCRIT	PLCOMP	PLINFL	PGREL	PGCRIT	PGCOMP
SCRIT	-0.071										
p-value	0.800										
SCOMP	0.261	0.638									
p-value	0.348	0.011									
SINFL	0.741	-0.019	0.271								
p-value	0.002	0.947	0.328								
PLREL	0.578	-0.226	-0.172	0.493							
p-value	0.024	0.418	0.540	0.062							
PLCRIT	-0.414	-0.242	-0.073	-0.328	-0.122						
p-value	0.125	0.385	0.797	0.233	0.664						
PLCOMP	-0.148	-0.354	-0.032	-0.049	-0.081	0.855					
p-value	0.599	0.196	0.911	0.863	0.773	0.000					
PLINFL	0.117	0.049	-0.121	-0.033	0.495	0.308	0.141				
p-value	0.679	0.863	0.668	0.907	0.061	0.264	0.617				
PGREL	0.736	-0.090	0.140	0.596	0.526	-0.641	-0.432	-0.169			
p-value	0.002	0.751	0.620	0.019	0.044	0.010	0.107	0.548			
PGCRIT	0.675	0.018	0.156	0.376	0.578	-0.364	-0.279	0.317	0.821		
p-value	0.006	0.949	0.579	0.167	0.024	0.182	0.313	0.250	0.000		
PGCOMP	0.707	0.067	0.505	0.500	0.341	-0.412	-0.340	-0.066	0.790	0.743	
p-value	0.003	0.811	0.055	0.058	0.213	0.127	0.215	0.816	0.001	0.002	
PGINFL	0.538	0.025	0.455	0.266	0.264	-0.216	-0.161	0.049	0.702	0.774	0.818
p-value	0.039	0.929	0.088	0.338	0.341	0.439	0.566	0.862	0.004	0.001	0.000