Preface

After years of experience with building systems, the information systems (IS) community is still challenged by systems delivery, that is, planning the implementation project, determining system features and requirements, sourcing and deploying the software, and managing its evolution. High-quality systems are still elusive. Yet organizations invest heftily in IS to support their business operations and realize corporate priorities, either in search of competitive advantage or as a competitive necessity. One of the major paradoxes of our era is the disparity between the many innovations that have been enabled by information technology (IT) and the failure of the IS community, comprising developers, managers, and users, to exploit these advances to consistently produce high-quality IS that provide value to organizations. This phenomenon, which is highlighted by Brynjolfssen (1993), Gibbs (1994), and others, has been dubbed "the software crisis."

The IS Quality Landscape

The effects of the software crisis are demonstrated in the number of projects that are abandoned before completion (The Standish Group, 2003), deployed with poor quality, consuming inordinate maintenance resources (Banker et al., 1998), or remain unused after implementation (Markus & Keil, 1994). The IS community has rightly focused on how to reverse the trend of low-quality systems, especially as IS become more central to the accomplishment of organizational mission. The quality drive in IS delivery is reminiscent of the intensified focus on quality in the manufacturing and service areas, and the IS discipline has built on many of the concepts articulated by Deming and others.

However, the systems delivery process has at least two key complicating factors that are absent from most other manufacturing processes. First, IS are not confined to single operations, but typically address a network of interdependent business processes and interfaces to other technical systems (Liu, 2000), and are embedded within the social systems of the organization (Robey et al., 2001). Second, while the erection of physical structures takes increasing physical shape as the construction progresses, software artifacts remain "invisible" throughout development (Brooks, 1987). Progress is therefore much harder to monitor, and systems delivery more difficult to manage.

It is little wonder that the pursuit of quality-enhancing approaches has taken many turns and has focused on a variety of factors. IS quality has been defined in various ways and encompasses several interpretations depending on the perspective of the particular stakeholder. To address the many perspectives, the IS community has variously focused on the contribution of people, delivery processes, and development philosophies (methods) to the attainment of defined quality attributes of intermediary and final software products.

User involvement in systems delivery facilitates the capture of the knowledge of appropriate domain experts, leads to greater commitment to system decisions and outcomes, and reduces the probability of usage failures (Barki & Hartwick, 1994). The IS community has also embraced the underlying logic of continuous process improvement programs, namely that process quality largely determines product quality (Deming, 1986). This has been substantiated in IS research (Harter et al., 1998, Khalifa & Verner, 2000; Ravichandran & Rai, 2000). Yet scholars have lamented that software process interventions are not well employed in IS organizations to improve quality (Fichman & Kemerer, 1997). Similarly, several systems delivery methods (design principles for driving specific development techniques), such as rapid application development, object-oriented development, and agile methods (Duggan, 2004), all purport to improve IS quality.

Despite the fact that the IS discipline is over a half a century old and numerous articles have been written about software quality, there is, as yet, very little convergence of opinions on approaches for attaining quality. In several cases, the same method attracts simultaneous claims of efficacy and ineffectiveness in different quarters. There is not even a commonly accepted definition for IS quality. This obviously affects agreement on important questions such as what are its determinants, what are the mechanisms by which quality is incorporated into the IS delivery process, and under what conditions are particular techniques likely to be successful.

What Does This Book Offer?

This book presents an international focus on IS quality that represents the efforts of authors from several countries. Despite this diversity and the presentation of the uncoordinated research findings and objective, conceptual analyses of multiple dimensions of IS quality by several authors, the result is integrative. It reflects the common position that improving objective knowledge of potential quality-enhancing methods is far more likely to assist the production of high-quality software than wanton experimentation with each new "gadget." Hence, the editors and chapter authors share a common motivation to reduce the uncertainty and ambivalence about the efficacy of particular methods, the contexts and conditions under which they are effective, and what synergies might obtain from combined approaches.

The book therefore provides thoughtful analyses and insights to explain some of the contradictions and apparent paradoxes of the many IS quality perspectives. It offers prescriptions, grounded in research findings, syntheses of relevant, up-to-date literature, and verifiable leading practices to assist the assimilation, measurement, and management of IS quality practices in order to increase the odds of producing higher quality systems. In addition to both descriptive contributions (to elucidate and clarify quality concepts) and prescriptive solutions (to propose quality-enhancing approaches), there are also normative features to rationalize perceptive gaps and balance conflicting views.

The Intended Audience

The book is intended to serve the diverse interests of several sectors of the IS community. It will be supportive of teaching, research, and application efforts in the increasingly important area of information systems delivery quality. It will both provide answers to enhance our state of knowledge and generate questions to motivate further research. In teaching, it may be used to provide supplementary material to support courses in systems analysis and design, software engineering, and information systems management.

This book will be useful for researchers in several ways. First, it provides an up-to-date literature review of this subject area. Second, it clarifies terminology commonly used in IS quality literature and generates some points of convergence; as noted by Barki et al. (1993), the absence of a common vocabulary and inconsistent use of terms in the IS literature severely constrains cumulative research. Third, most chapters suggest a variety of research questions and propose directions for future research for both seasoned researchers and doctoral students who intend to focus on IS process improvement and quality issues.

It is also valuable for IS practitioners by providing objective analyses of quality-enhancing methods and practices in context. Practitioners typically sift through a maze of claims and hype about the effectiveness of several techniques from their proponents and need useful information to evaluate these claims. This book helps by clarifying the muddy waters, explicating the quality requisites of a successful information systems, providing information for effective decision-making about appropriate process strategies and their fit to particular IS delivery settings, and assessing the strengths, weaknesses, and limits of applicability of proposed approaches.

Organization of This Book

The 15 chapters of this book contain a mixture of conceptual and research papers, and are organized into five sections, each with a distinct focus:

- Section 1 introduces IS quality concepts, definitions, perspectives, and management techniques and issues.
- Section 2 discusses quality issues in the early and formative stages of an information system;
- Section 3 is devoted to the contribution of process-centricity to quality IS products.
- Section 4 describes and evaluates metamethods and frameworks for evaluating software process improvement (SPI) programs and process structuring methodologies.
- Section 5 analyzes quality issues in less researched applications and institutions.

Section I: Introduction and Overview of Quality Concepts and Dimensions

The three chapters of Section I provide a general introduction to and overview of the critical definitions, concepts, perspectives, and nuances of IS quality and its determinants and measures. It lays the appropriate foundation and sets the tone for further examination of the concepts, issues, and controversies presented in later sections.

Chapter I provides a solid opening in its introductory overview, which covers the breadth of IS quality considerations to prepare readers for the more de-

tailed analyses in subsequent chapters. It provides an overview of the threats to IS delivery and the quality of the final product and highlights the many perspectives and heterogeneity of IS stakeholders. Consequently, many approaches (sometimes uncoordinated) have been adopted to improve the quality, usage, and perceived value of IS within organizations. The authors undertake an extensive review of the IS quality literature to develop a general conceptual model of the elements — people, process, and practices — that contribute to software quality. They posit that the objective quality attributes of a system do not necessarily offer system success; the perceptions of users, however formed, are also influential. Beyond setting the tone for the topics that follow, Chapter I also makes its own contributions and provides some insights on IS quality issues that are not covered elsewhere in the book.

Chapter II provides an overview of the concepts and management issues that surround the measurement and incorporation of quality features into an IS development project. It provides a detailed review of the literature in this field in general, and of software quality, in particular. It also presents an overview of what to expect in the upcoming international standards on software quality requirements, which transcend the life cycle activities of IT processes. The chapter provides validation for a sub-theme of the book, namely that the concept of software quality cannot be defined abstractly, but that any adequate definition must take into account the application context of the software.

Chapter III examines several definitions of quality, categorizes the differing views of the concept, and compares different models and frameworks for software quality evaluation. The author emphasizes the notion that a particular view can be useful in varying contexts, and that a particular context may be better or worse served by varying views. The chapter examines both historical and current literature to provide a focused and balanced discussion of recent research on the Software Evaluation Framework (SEF). SEF gives the rationale for the choice of characteristics used in software quality evaluation, supplies the underpinning explanation for the multiple views of quality, and describes the areas of motivation behind software quality evaluation. The framework which has its theoretical foundations in value-chain models found in the disciplines of cognitive psychology and consumer research, introduces the use of cognitive structures to describe the many definitions of quality.

Section II: Quality in the Early Stages of IS Delivery

Section II contains three chapters which examine various dimensions of IS quality, particularly in the early stages of the information systems life cycle. System failures resulting from poor deliverables in the early stages of develop-

ment are pervasive (Dodd & Carr, 1994). It is generally acknowledged that quality, in the early stages of IS delivery, sets the tone for subsequent stages and determines the extent of design errors and rework (Byrd et al., 1992; Kang & Christel, 1992). It also tends to ripple through to the final product.

Chapter IV examines the problem of requirements defects, which remain a significant issue in the development of all software intensive systems including information systems, and reduces the quality of the product. The author asserts that progress with this fundamental problem is possible once we recognize that individual functional requirements represent fragments of behavior, while a design *satisfying* a set of functional requirements represents integrated behavior. The chapter offers a solution to the problem that accommodates this perspective by using behavior trees, formal representation for individual functional requirements, to construct a design *out of* its requirements. Behavior trees of individual functional requirements may be composed, one at a time, to create an integrated design behavior tree (DBT) to detect different classes of defects at various stages of the development process. This has significant quality-enhancing implications.

Chapter V builds on the theme that improper specification of systems requirements has thwarted many splendid efforts to deliver high-quality information systems and links this problem largely to poor communication among systems developers and users at this stage of systems development. It focuses on the devastating impact of user-developer miscommunication and the inadequacy of some of the models available for communicating specifications to users to obtain the degree of participation and high-quality feedback necessary for effective validation of systems requirements. The chapter presents an overview of both longstanding and newer requirements specification models and representational schemes for communicating specifications and evaluates their capability to advance user participation in this process and incorporate stated quality attributes. It also reports on preliminary evaluations of animated system engineering (ASE), the author's preferred (newer) technique, which adds animation and the capability to model dynamic and concurrent activities. The evaluation of ASE indicates that it has the potential to improve the effectiveness of requirements specification.

Chapter VI introduces the concept of scenarios (rich picture stories) to promote collaboration in the design of new information systems and provides insights into the opportunities available for the application of valid user scenarios. Systems designers need contextual information about their users in order to design and provide information systems that will function effectively and efficiently within those contexts. Storytelling or scenarios allow developers and users to create that all important "meeting place" that permits collaborative efforts in the design of new systems and richer information flows to raise the quality of the design. The chapter also explores the question of validation, which

is one of the primary issues inhibiting the wider use of storytelling or scenarios in information systems development. The chapter demonstrates how the structured use of user-driven scenarios can assist practitioners in improving the quality of information systems design and encourages researchers to explore this interesting approach.

Section III: Process Contribution to IS Quality

Credible evidence that sound systems development processes contribute to the delivery of better information systems (Harter et al., 1998; Kalifa & Verner, 2000; Ravichandran & Rai, 2000) provides the justification for organizations to invest in process improvement programs. Such programs assess the capability of an organization's IS processes and based on the results, define goals and plans to institutionalize documented standards and practices to guide the execution of repetitive development activities in order to reduce variability from one development project to the next. This section contains four chapters that address process contributions to IS quality.

Chapter VII explores the reasons for the persistent anxiety about low-quality IS, on the one hand, and the demonstrated relationship between systematic IS development processes and system success on the other. It presents a broad overview of IS process improvement concepts, applications, and contributions to IS delivery quality. The chapter synthesizes core facts from real world experiences, practitioner reports and anecdotes from current practice, and insights gleaned from scholarly research to provide a general exposition of the perspectives and issues surrounding this increasingly important and interesting topic. The chapter provides the foundation and background for the deeper analyses of various aspects of process-centered approaches in succeeding chapters of this section of the book. In addition, it makes a meaningful contribution to the ongoing debate about the relevance and potential of process-centricity in the delivery of high-quality IS by assessing the contexts in which quality-enhancing software process improvements can realistically thrive.

In Chapter VIII, the authors report on the results of a set of research projects that investigated the SPI movement against the background that the search for new ideas and innovations to improve software development productivity and enhance software system quality continues to be a key focus of industrial and academic research. They sought answers to the apparent disconnect between the extensive focus on SPI programs based on the premise that system development outcomes are largely determined by the capabilities of the software development process, and the slow diffusion and utilization of SPI initiatives in the software engineering community. They found that (1) software developers'

perceived control over the use of a SPI impacts its diffusion success and (2) that software developers' perceptions of enhanced software quality and increased individual productivity achieved through the use of SPI impact the successful diffusion of the SPI. Results of these research efforts support the compilation of a clear set of management guidelines for the effective use of SPI initiatives in software development organizations.

Chapter IX introduces, defines, and elaborates on agile development methods and how quality information systems are created through the values and practices of people using agile approaches. The chapter covers key differences among agile methods, the SDLC, and other development methodologies and offers suggestions for improving quality in IS through agile methods. Following this objective analysis, the authors recommend adopting the principles of agile methods, propose several IS quality improvement solutions that could result, and raise future research issues and directions that could provide further insights into the contribution of agile methods to information systems quality. They also call for more education about the value of agile approaches and the expansion of their application to include more people, use in a variety of organizational cultures, and renaming agile methods to signify the value system inherent in the approach. The chapter provides excellent background coverage for Chapter X.

Chapter X presents an empirical assessment of the quality of the process of building software systems with agile development methods, which were designed to help with the development of higher quality information systems under given conditions. The research assessed eXtreme Programming (XP), one of the several agile development approaches. It compares XP with a traditional (design-driven) software construction process by observing and measuring the work of several student groups using different approaches to produce software for commercial companies during a semester. The data collected were analyzed following the Bayesian approach. The results indicate that that XP could cope with small to medium sized projects and delivered measurable improvement in the quality of the system as judged by business clients.

Section IV: Managing Risks of SPI Projects and Methodologies

Section IV focuses on process contributions to IS quality and the several vehicles for accommodating SPI programs and institutionalizing streamlined software processes through system development methodologies (SDM). However, there are risks in the adoption and use of these methodologies; their success is highly dependent on how well these processes fit the organizational culture and

can be interwoven into its social systems (Curtis et al., 1995; Perry et al., 1994). The two chapters in this section address risk mitigation approaches to adopting SPI programs and meta-methodologies for evaluating the effectiveness of methodologies respectively.

Chapter XI shows how action research can help practitioners develop IT risk management approaches that are tailored to their organization and the specific issues they face. Based on literature and practical experience, the authors present a generic method for developing risk management approaches for use in real-world software innovation projects. The method is based on action research into an organization's specific risk management context and needs. The chapter illustrates the method by presenting the results of the authors' experiences in developing the tailored approach to risk management in SPI projects at a large Danish bank.

Chapter XII discusses the evaluation of information systems development methodologies, which are considered cornerstones for building quality into an information system. If methodologies are indeed pivotal to system quality, then the effective evaluation of methodologies becomes crucial. This evaluation is usually achieved through the use of evaluation frameworks and metamodels, both of which are considered meta-methodologies. The chapter provides a comprehensive overview of how to construct efficient and cost-effective meta-methodologies and identify their quality attributes in a scientific and reliable manner. It reviews representative meta-methodologies and summarizes their quality features, strengths and limitations, and compares the functional and formal quality properties that traditional meta-methodologies and method evaluation paradigms offer in addressing properties such as computability and implementability, testing, dynamic semantics capture, and people's involvement.

Section V: IS Quality Issues in Under-Researched Areas

The quality and SPI foci have been dominated by issues in a corporate context, where software is sourced through traditional means. The three sections in this chapter widen the perspective to include quality concerns in areas that are not as well discussed and researched. The first examines quality issues with open source software (OSS) development, which has grown from the preoccupation of hackers to viability as a software development alternative (Henderson, 2000). The others examine peculiar problems of quality in non-corporate organizations — general issues in government organizations and ERP implementation difficulties in a University environment.

Chapter XIII examines quality features of Open Source Software (OSS) processes against the background that these processes are not well researched.

The chapter describes the principles and processes used to generate OSS and their differences with processes used for proprietary software, the responsibilities of producer and consumer communities, and the protocols that govern their actions. The author then uses Bass et al.'s (2000) quality model that assesses software by attributes of the system such as performance, security, modifiability, reliability, and usability to explore the challenges these attributes pose to the open source development process and how the open source community measures up to these challenges.

Chapter XIV analyzes the peculiar challenges public agencies face in creating and sustaining IS quality. Unlike private sector firms, governments exist to meet mandated service requirements with limited ability to create revenue streams to fund IS programs that are essential to managing organizational bureaucracies and serving the public interest. The author provides a historical perspective of IS quality (with selected examples of both low- and high-quality systems) in the public sector, where high-profile failures have masked the relative successes of the majority of systems that work adequately within a complicated network of several interrelated governmental entities. The chapter denotes that the expectation of IS quality varies, but operational success over the long run may be considered the most highly valued feature of quality information systems in that sector. However, demands for productivity and innovation, increased responsiveness, and more sophisticated leadership may shape new quality profiles for the future.

Chapter XV presents a case study of the implementation of an ERP system in a large Australian university, viewed through the lens of Eriksson and Törn's (1991) Software Library Evolution (SOLE) quality model. The study explores the relationship between ERP systems capability and the quality requirements of individual users and highlights the problems encountered by organizations such as universities, where implementation failures of such mammoth systems could be relatively more disastrous than in corporations. The literature suggests other differences in the deployment of these systems compared to stand-alone systems, particularly in terms of the nature and timing of user involvement. Given these and other peculiar difficulties of non-traditional adoption of ERP systems, the authors refer to the lessons of this case to prescribe useful practices for such implementations and suggest some solid directions for similar research in the future.

References

Banker, R.D., Davis, G.B., & Slaughter, S.A. (1998). Software development practices, software complexity, and software maintenance performance: A field study. *Management Science, 44*(4), 433-450.

Barki, H., & Hartwick, J. (1994). Measuring user participation, user involvement, and user attitude. *MIS Quarterly, 18*(1), 59-82.

Barki, H., Rivard, S., & Talbot, J. (1993). A keyword classification scheme for IS research literature: An update. *MIS Quarterly, 17*(2), 209-226.

Brooks, F.P., Jr. (1987). No silver bullet: Essence and accidents of software engineering. *IEEE Computer, 20*(4), 10-19.

Brynjolfssen, E. (1993). The productivity paradox of information technology. *Communications of the ACM, 36*(12), 67-77.

Byrd, T.A., Cossick, K.L., & Zmud, R.W. (1992). A synthesis of research on requirements analysis and knowledge acquisition techniques. *MIS Quarterly, 16*(3), 117-138.

Curtis, B., Hefley, W.E., & Miller, S. (1995). *Overview of the people capability maturity model* (Tech. Rep. No. CMU/SEI-95-MM-01). Carnegie Mellon University, Software Enginering Institute.

Dodd, J.L., & Carr, H.H. (1994). Systems development led by end-users. *Journal of Systems Management, 45*(8), 34-40.

Duggan, E.W. (2004). Silver pellets for improving software quality. *Information Resources Management Journal, 17*(2), 1-21.

Eriksson, I., & Törn, A. (1991). A model of IS quality. *Software Engineering Journal, 6*(4), 152-158.

Fichman, R., & Kemerer, C. (1997). The assimilation of software process innovations: An organizational learning perspective. *Management Science, 43*(10), 1345-1363.

Gibbs, W.W. (1994). Software's chronic crisis. *Scientific American, 271*(3), 86-95.

Harter, D.E., Slaughter, S.A., & Krishnan, M.S. (1998). The life cycle effects of software quality: A longitudinal analysis. *Proceedings of the International Conference on Information Systems* (pp. 346-351).

Henderson, L. (2000). Requirements elicitation in open-source programs. Hill Air Force Base Software Technology Support Center (STSC): *CrossTalk, 13*(7).

Kang, K.C., & Christel, M.G. (1992). *Issues in requirements elicitation* (Tech. Rep. No. SEI-92-TR-012). Pittsburgh: Carnegie Mellon University.

Khalifa, M., & Verner, J.M. (2000). Drivers for software development method usage. *IEEE Transactions on Engineering Management, 47*(3), 360-369.

Liu, K. (2000). *Semiotics in information systems engineering*. Cambridge, UK: Cambridge University Press.

Markus, M.L., & Keil, M. (1994). If we build it they will come: Designing information systems that users want to use. *Sloan Management Review, 35*(4), 11-25.

Perry, D.E., Staudenmayer, N.A., & Votta, L.G. (1994). People, organizations, and process improvement. *IEEE Software, 11*, 36-45.

Ravichandran, T., & Rai, A. (2000). Quality management in systems development: An organizational system perspective. *MIS Quarterly, 24*(3), 381-415.

Robey, D., Welke, R., & Turk, D. (2001). Traditional, iterative, and component-based development: A social analysis of software development paradigms. *Information Technology and Management, 2*(1), 53-70.

Standish Group, The. (2003). *Chaos – the state of the software industry.* Retrieved December 1, 2005, from http://www.standishgroup.com

Evan W. Duggan

Han Reichgelt