Preface

In the past decade, three developments have brought the issues of information systems maintenance and evolution to the fore. First, the creation of large-scale distributed systems, which are dynamically configured and re-configured, and running over global networks. Second, abetted by Internet technologies, the whole business model for many enterprises has been significantly changed. Third, global business competitiveness and complexities have been increased dramatically. Computing technologies, tools, and systems platforms are changing rapidly; so are the business strategies, scope and dimensions. In the past, enterprises could only do business in a particular geographical area. Today, organizations are increasingly becoming location independent and heavily depend on new distributed technologies such as the Internet. Enterprises are finding themselves to be strategically disadvantaged if they do not take advantage of the new technology. The consequence of advances in technologies and the changing business needs have virtually forced the organizations to let their legacy systems evolve more rapidly than they anticipated a few years ago.

In this rapidly evolving world, a corporate information system that interacts with its environment is inevitably required to evolve and to be maintained in order to keep its business competitiveness with others. Information systems (IS) always change and evolve; some more frequently, some are less frequently. System evolution refers to a process of continual change in systems functions and non-functional properties. Evolution of information systems is unavoidable, and organizations need to support and manage systems evolution to take advantage of the new technology and to address the changing business rules. The evolutionary nature of IS products, in fact, leaves us with two options: either maintain the products with their continual changing nature, or accept the degrading performance of the product. In this ever-increasing competitive business environment, the latter one would hardly be an option for most enterprises. A corporate system evolution process often requires managerial and technical skills in order to manage and control system evolution.

Managing systems evolution is generally considered to be complex. This complexity could be further compounded by growing connectivity and integration with other systems such as distributed applications. The rapidly emerging technologies put further pressures on the management task of systems evolution. The ever-dependence of business and enterprises on legacy systems makes the search for effective and viable solutions to the problems of systems evolution and maintenance more urgent. In order to find an effective solution, we need to achieve a deeper and wider understanding of the managerial and technological issues related to the systems evolution.

The importance of IS evolution and maintenance is being increasingly recognized. Global business competitiveness is dependent on the continuous maintenance and evolution of operational legacy systems. In the era of Web-based systems and *e*business, organizations need appropriate maintenance processes and resources that are required to migrate their aging legacy information systems to Web-enabled contemporary systems—for example.

Virtually every organization has to go through some sorts of maintenance and evolution of its systems. The need for a system evolution emerges from various issues such as changes of business rules, emergence of new technology, introducing new functionality, or fixing defects in the systems and so on. Managing system evolution and maintenance involves a great deal of technical, financial and management issues. One of the major concerns of any organization is how to manage the evolution process with minimum cost for maximum benefits. IS maintenance process involves multi-dimensional activities. Very few organizations adopt a separate process for system maintenance because they cannot distinguish the maintenance process from the new development process. The procedures followed in IS evolution and maintenance by practitioners are not well defined. Current practices of system maintenance activities revolve around ad hoc patching that does not follow any defined methodology. A more defined formalism describing various tasks, tools and methods is required to enable a clear understanding of the IS evolution and maintenance activities. The goal of this book is to examine key IS evolution and maintenance issues facing the IS community.

THE CHALLENGES

Any attempt to deal with the management of system evolution and maintenance requires an adequate understanding of the challenges that exist. Such challenges can be classified into different categories:

- The challenge of understanding the existing legacy systems—its architecture, its formation, its use context, and its business values
- The challenge of defining and managing the various tasks of the maintenance process, its resources, and people involved
- The challenge of controlling and containing the impact of changes that are introduced to the legacy system
- The challenge of selecting techniques and methodology to transform the legacy system to a newer form and structure
- The challenge of supporting continuous evolution of large-scale distributed systems that are dynamically self-configured and running over global networks
- The challenge of balancing the business needs and technical requirements of systems evolution
- The challenge of not creating chaos by building overly complex and large systems that we are unable to control the evolution of
- Finally, the challenge of keeping the business goals of the system in an advantageous position in an ever-increasing competitive and complex business environment.

There are no simple solutions to the problems of the topic, and we need a wide spectrum of management and technical approaches to address the challenges.

SEARCHING FOR A SOLUTION

To address the challenges, the approach that we take in this book is to present a broad perspective on systems maintenance and evolution. The focus in this book is, therefore, on the managerial and technical issues related to IS evolution and maintenance to prevent the decay of operational IS products while enhancing their business value and adapting them to a new, rapidly changing environment. The book documents some technical and managerial aspects of systems evolution, providing insights into how a modification to a system could be initiated, managed and technically achieved. The book also addresses how the system evolution has considerable impact on the corporate business process, personnel management, and the technological advantages.

We understand that problem and solution for two legacy systems cannot be the same, but there is an underlying commonality in managing the maintenance tasks from system to system. A general consensus is that a maintenance and evolution process must be manageable and adaptive, the system architecture should be recovered and recreated using defined techniques, the change impact should be controlled and tested, and finally, the business goal should be preserved.

Most books on systems evolution concentrate either on technical aspects or the management issues of the topic. In this book we try to address both issues in a balanced way. The purpose of this book is to disassemble the complex process of information system evolution and maintenance into related managerial and technical tasks so as to aid in the allocation of resources, acquisition of appropriate toolsets, distribution of responsibilities and management of the entire complex process. The book is intended to convey a high-level understanding of managing IS maintenance process and its dimensions. In this book we are particularly interested in exchanging concepts, new ideas, research results, experience and other related issues that could contribute to the academic arena and also benefit the corporate business community.

READERSHIP

The book is primarily aimed at information systems practitioners and researchers. More specifically, the book will suit:

- Researchers who like to keep track of “what is going on” in systems evolution research around the globe
- Practitioners such as systems managers, designers, architects, and planners, who may find the book to be a means of updating their knowledge on the topic
- Graduate students with the goal to acquire the knowledge and skills to manage and understand systems evolution and maintenance process and techniques.

We assume that readers have a basic understanding of general concepts in systems design, programming, and management.

ORGANIZATION OF THE BOOK

Although we have not formally divided the book into parts, each group of related chapters has a different focus on information systems evolution and maintenance. The book is organized into 15 chapters.

Chapter I identifies various issues in the management of information systems evolution and maintenance. The chapter sets the scene for discussions presented by authors in later chapters. In particular, the chapter identifies the global orientation of managing systems evolution and the related issues. Fifteen of the many concerns arising from the interests of the stakeholders of information systems are introduced in this chapter to provide context for the subsequent chapters in this book. This chapter concludes by noting the critical success factor role of leadership in the management of information systems evolution and maintenance.

Chapter II highlights the pervasiveness of change in most developed systems, and the resulting implications on development practice. The chapter emphasizes that the system continuity and evolution require a way of distinguishing between systems that can be assumed to remain stable and those that continuously "live" and evolve in their environment. Developing systems that are expected to evolve calls for a dynamic activity of design that is responsive to changes in the environment. The chapter highlights the implications for the adoption of adaptive design in dynamic environments, thereby justifying the move from discrete and project-based software engineering to a more strategic, continuous and evolving discipline of adaptive design.

Chapter III examines the differences between software maintenance and software development from a service point of view, and the consequences thereof for the maturity of software maintenance organizations. The chapter argues that software maintenance can be seen as providing a service, whereas software development is primarily concerned with the development of products. Differences between products and services affect the way in which customers assess their respective quality. Consequently, customers will judge the quality of software maintenance differently from that of software development. This in turn means that to deliver high quality results in software maintenance, both the functional quality and the technical quality dimensions are important.

Chapter IV presents the process of upfront corrective maintenance at the front-end support level. The chapter is logically divided into two parts. The first part introduces the domain of the upfront corrective maintenance process, and presents its current status within the industry. It lists the problems encountered by the front-support organizations today. The second part provides a glimpse into the Corrective Maintenance Maturity Model (CM^3): a process model specialized in upfront corrective maintenance. The goal of this chapter is to provide detailed insight into the domain of upfront corrective maintenance.

Chapter V reports the development of a novel evolution methodology that integrates the concepts of features, regression testing, and component-based software engineering. The chapter states that regression test cases are untapped resources, full of information about system features. By exercising each feature with its associated test cases using code profilers, code can be located and refactored to create components. These components are then inserted back into the legacy system, ensuring a working system structure. The approach is validated on the evolution of a real-world legacy system to reduce the costs of future maintenance of the system.

Chapter VI provides techniques to re-engineer stand-alone legacy systems into Web-enabled environments. Specifically, it aims for a framework that allows for the identification of reusable business logic entities in large legacy systems in the form of major legacy components, the migration of these procedural components to an object oriented design, the specification of interfaces of these identified components, the automatic generation of CORBA wrappers to enable remote access, and finally, the seamless interoperation with Web services via HTTP based on the SOAP messaging mechanism.

Chapter VII examines the use of patterns for reengineering legacy systems to Web. To address these issues, many different concepts and frameworks need to be well understood, especially legacy system wrapping, connection handling, service abstraction, adaptation techniques, dynamic content generation, and others. In this chapter, we present patterns from different sources that resolve these issues. We integrate them to a pattern language operating in the context of reengineering to the Web, and present pattern variants and examples in this context.

Chapter VIII presents an approach to supporting change impact analysis at the architectural level of software systems based on the architectural slicing and chopping technique. The main feature of the approach is to assess the effect of changes in a software architecture by analyzing its formal architectural specification. The chapter argues and shows that separation of change impact analysis of a system from code level to architectural level is an efficient step to reduce the cost of change impact analysis during software evolution, because this allows a maintenance programmer to assess the effect of changes of the system at the architectural level so that many implementation details of the system need not to be considered.

Chapter IX discusses the requirements that an open architecture for software visualization tools should meet, and details the implementation of such an architecture. The focus in this chapter is on reverse engineering, although the proposed technique could be applicable to forward engineering activities as well. The material presented in this chapter both serves the software architects who are looking for a flexible visualization tool and provides a starting point for the tool designer.

Chapter X addresses the research challenges in the area of software architecture reconstruction and discusses the state-of-the-art solutions to these challenges. The chapter argues that the importance of architecture analysis and reconstruction of a large and complex software system stems from the need to perform continuous maintenance activities, such as migration of a legacy system to a new platform, in which the architectural reconstruction constitutes the major part of these activities. The chapter exposes a systematic approach to the important issues, alternative solutions, and future research in the field of software architecture analysis and reconstruction.

Chapter XI describes the notion of product line models and how to apply them for developing and evolving Web products. A product line captures the common and variable aspects of software systems as key assets under a common architecture. Software companies are increasingly adopting this approach in order to accelerate the development of families of similar software products. The chapter argues that new techniques to engineer Websites are needed in order to reduce the time to market of Web products.

Chapter XII provides insight into the software maintenance process, its features, actors, activities and results, with the emphasis on the importance and complexity of this kind of maintenance. The chapter shows how the software maintenance process is modeled in the form of the queuing network with the applied simulation. Method implementation is described with a case study of telecommunication maintenance. The results of the method show how the process bottlenecks are formally identified and how new proposals for the improved process design are applied.

Chapter XIII proposes a strategy for extracting the requirements for a legacy system evolution from the requirements of the *e*Business evolution. The strategy includes a toolkit composed of a set of decision tables and a measurement framework, both referring to the organization, business processes, and legacy information systems. The decision tables allow the identification of the processes to be evolved, the actions to be performed on them and their activities, and the strategies to be adopted for evolving the embedded software systems. The measurement framework proposed in the chapter aims at achieving a greater understanding of the processes and related problems, taking into account organizational and technological issues.

Chapter XIV discusses some of the common problems in maintenance. The chapter contains a management model for solving these problems, based on theoretical research and practical experience. The central theme of the model is the connection between business and IT in maintenance, and the connected responsibilities of these two. This is achieved by maintaining maintenance objects rather than the system, establishing micro organizations for each maintenance object where business processes are represented. The model is used in more than 50 organizations in Sweden.

Chapter XV concludes the book with a review of a variety of IS quality-oriented approaches based on research findings and dispassionate practitioner accounts, and an indication of their advantages and weaknesses, and the development environments and contexts for which they are most suited. These techniques are available to systems developers to be used individually or in synergistic combinations to confront the prevalent problem of poor system quality and reduce its unfavorable impact on information systems maintenance.

Khaled M. Khan
Yan Zheng, Ph.D.
November 2003