# Preface

## INTRODUCTION

Software Engineering (SE) is a disciplined and engineering approach to software development and management of software projects and complexity. Today, software exists in each and every product, from toys, powered tooth brushes, electric shavers (have 10,000 lines of embedded software code), TV and entertainment systems, mobile computers, portable devices, medical systems, and home appliances, and to large scale software such as aircraft, communication systems, weather forecasts, Grid computing, and much more. Current success in software development practices are based on some of the best practices and good design principles which have been employed over last two decades. On the other hand, Knowledge Engineering (KE) has provided some of the best techniques and tools to support encoding knowledge and experiences successfully. Therefore, it is the main aim of this book to bridge those two successes for the purpose of enhancing software development practices.

Software guidelines have been with us in many forms within Software Engineering community such as knowledge, experiences, domain expertise, laws, software design principles, rules, design heuristics, hypotheses, experimental results, programming rules, best practices, observations, skills, and algorithms, all of which have played major role in software development (guidelines based software engineering). Also, there has been a significant development in new software paradigms such as components, agile aspects, and services. How can we best use those 30 years of best practices in solving software related problems? There has been little evidence of automated approaches to the development of large scale systems and software reuse. The demands of Software Engineering imply a growing need for using Artificial Intelligence and Knowledge Engineering (KE) to support all aspects of the software development process. This book aims to provide insights into the application of knowledge based approaches to the development of software systems, agile software development, software product line, software components, software architecture, automated code generators, automated test case generations, intelligent systems supporting software reuse and productivity, domain-specific software modelling, automated requirements and semantic knowledge capturing techniques, intelligent software intensive devices, and applications. In recent years, there has been significant application of neural networks and genetic programming applied to software development for a variety of applications.

This book also aims to identify how software development best practices captured as software guidelines can be represented to automated software development, decision making, and knowledge management. Therefore, a book of this nature can bring industry and academia together to address the need for the growing applications and supporting knowledge based approaches to software development.

## KNOWLEDGE ENGINEERING FOR SOFTWARE DEVELOPMENT LIFE CYCLES

Software development remains costly and laborious as it depends largely on human talents. For faster productivity and to reduce cost in the current knowledge economy, we need to strive for quality; we need to capture best practices and problem solving techniques in all aspects of software development activities. There have been many successful software development life cycles ranging from a water fall model, spiral model, to win-win model, and many more. All of them have proposed a set of activities to be performed and to be checked for maintaining the desired quality and standards. Therefore, it is quite possible to capture and encode those required activities as knowledge, and hence, save us from repetitive tasks that are costly and time consuming. Also, there have been several techniques on modelling and several programming languages, all which require highly skilled people to develop systems. This learning process can be saved by using a number of automated knowledge based activities supported by those emerging technologies. In addition, most of the existing and legacy systems often need to be transformed into those newer technologies due to business competition and enterprise system integration. These activities can also be automated, thus saving cost and effort and at the same time maintaining expected quality.

## RESEARCH ISSUES

Interplay between Software Engineering and Knowledge Engineering will dominate the current and further research in the areas such as large scale software development, global software project management, knowledge management practices for software services, and automated technologies. Some of the specific areas of research are:

- Best practice software guidelines
- Efficient programming techniques supporting large scale reuse
- Knowledge engineering support for software reuse
- Knowledge engineering support for code generation
- KE support for requirements, architecture, components
- KE support for software process improvement
- KE support for software testing, quality and metrics
- Intelligent software intensive applications
- Knowledge Management (KM) for software product line
- KE for agile and aspects-oriented software development
- KE for web services and SOA
- Requirements engineering and knowledge management
- Automated code generation techniques from requirements and design
- Automated support for software project management and process improvement
- Automated support for reusability, extensibility, and refactoring
- Automated support for reusable requirements, design, and test scripts

## BOOK ORGANISATION

Software development and knowledge engineering process and related technologies are the key to today's globalisation which would not have occurred without integrating these two disciplines. New technologies and concepts are being developed, prompting researchers to find continuously new ways of utilizing both older and new technologies. In such an ever-evolving environment, teachers, researchers and professionals of the discipline need access to the most current information about the concepts, issues, trends and technologies in this emerging field. This book will be most helpful as it provides comprehensive coverage and definitions of the most important issues, concepts, trends and technologies in *Software Engineering and Knowledge Engineering*. This important new publication will be distributed worldwide among academic and professional institutions and will be instrumental in providing researchers, scholars, students and professionals access to the latest knowledge related to information science and technology. Contributions to this important publication will be made by scholars throughout the world with notable research portfolios and expertise. The book chapters are organised into three sections:

- **Section 1. Requirements Engineering and Knowledge Engineering:** This part provides chapters on AI support for Software Engineering in compositional era, Natural Language, Requirements and Knowledge Management.
- **Section 2. Design Patterns, Components, Services and Reuse:** This part consists of chapters on approaches to conceptual modelling for software patterns, web services, and software reuse.
- **Section 3. Testing, Metrics and Process Improvement:** This part consists of chapters on knowledge based approaches to software testing, framework for security improvement, KE supporting software process improvement and models.

## REFERENCES

Orwant, J. (2000). EGGG: Automated programming for game generation. *IBM Systems Journal*, *39*(3&4).

Rich, C., & Waters, C. R. (1991). *Knowledge intensive software engineering tools.* (Mitsubishi Electric Research Laboratories, Technical Report 91-03).

Ramachandran, M. (2009). Knowledge Engineering support for software requirements, architecture, and components . In Vadera, S., & Meziane, F. (Eds.), *AI support for SE*. Hershey, PA: IGI Global Publication.

Ramachandran, M. (2008). *Software components: Guidelines and applications*. New York, NY: Nova Publishers.

Ramachandran, M., & Ganeshan, S. (2009). Commonality analysis: Implications over a successful product line . In Ramachandran, M., & de Carvalho, R. A. (Eds.), *Handbook of software engineering and productivity technologies: Implications for globalisation*. Hershey, PA: IGI Publishers.