

# SWEBOK

Guide to the

# Software Engineering Body of Knowledge

2004 Version

Executive Editors

Alain Abran, École de technologie supérieure  
James W. Moore, The MITRE Corp.

Editors

Pierre Bourque, École de technologie supérieure  
Robert Dupuis, Université du Québec à Montréal



Project managed by:



**Guide to the Software Engineering  
Body of Knowledge**

**2004 Version**

**SWEBOK<sup>®</sup>**

**A project of the IEEE Computer Society  
Professional Practices Committee**



# Guide to the Software Engineering Body of Knowledge

2004 Version

# SWEBOK<sup>®</sup>

Executive Editors

*Alain Abran*, École de technologie supérieure  
*James W. Moore*, The MITRE Corp.

Editors

*Pierre Bourque*, École de technologie supérieure  
*Robert Dupuis*, Université du Québec à Montréal

Project Champion

*Leonard L. Tripp*, Chair, Professional Practices Committee,  
IEEE Computer Society (2001-2003)



Los Alamitos, California

Washington • Brussels • Tokyo

---

Copyright © 2004 by The Institute of Electrical and Electronics Engineers, Inc. All rights reserved.

*Copyright and Reprint Permissions:* This document may be copied, in whole or in part, in any form or by any means, as is, or with alterations, provided that (1) alterations are clearly marked as alterations and (2) this copyright notice is included unmodified in any copy. Any other use or distribution of this document is prohibited without the prior express permission of IEEE.

You use this document on the condition that you indemnify and hold harmless IEEE from any and all liability or damages to yourself or your hardware or software, or third parties, including attorneys' fees, court costs, and other related costs and expenses, arising out of your use of this document irrespective of the cause of said liability.

IEEE MAKES THIS DOCUMENT AVAILABLE ON AN "AS IS" BASIS AND MAKES NO WARRANTY, EXPRESS OR IMPLIED, AS TO THE ACCURACY, CAPABILITY, EFFICIENCY MERCHANTABILITY, OR FUNCTIONING OF THIS DOCUMENT. IN NO EVENT WILL IEEE BE LIABLE FOR ANY GENERAL, CONSEQUENTIAL, INDIRECT, INCIDENTAL, EXEMPLARY, OR SPECIAL DAMAGES, EVEN IF IEEE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

IEEE Computer Society Order Number C2330

ISBN 0-7695-2330-7

Library of Congress Number 2005921729


*Additional copies may be ordered from:*

IEEE Computer Society  
Customer Service Center  
10662 Los Vaqueros Circle  
P.O. Box 3014  
Los Alamitos, CA 90720-1314  
Tel: + 1-714-821-8380  
Fax: + 1-714-821-4641  
E-mail: [cs.books@computer.org](mailto:cs.books@computer.org)

IEEE Service Center  
445 Hoes Lane  
P.O. Box 1331  
Piscataway, NJ 08855-1331  
Tel: + 1-732-981-0060  
Fax: + 1-732-981-9667  
[http://shop.ieee.org/store/  
customer-service@ieee.org](http://shop.ieee.org/store/customer-service@ieee.org)

IEEE Computer Society  
Asia/Pacific Office  
Watanabe Bldg., 1-4-2  
Minami-Aoyama  
Minato-ku, Tokyo 107-0062  
JAPAN  
Tel: + 81-3-3408-3118  
Fax: + 81-3-3408-3553  
[tokyo.ofc@computer.org](mailto:tokyo.ofc@computer.org)

Publisher: Angela Burgess  
Group Managing Editor, CS Press: Deborah Plummer  
Advertising/Promotions: Tom Fink  
Production Editor: Bob Werner  
Printed in the United States of America

  
IEEE  
COMPUTER  
SOCIETY

 **IEEE**

# TABLE OF CONTENTS

FOREWORD.....	vii
PREFACE.....	xvii
CHAPTER 1 INTRODUCTION TO THE GUIDE.....	1-1
CHAPTER 2 SOFTWARE REQUIREMENTS.....	2-1
CHAPTER 3 SOFTWARE DESIGN.....	3-1
CHAPTER 4 SOFTWARE CONSTRUCTION.....	4-1
CHAPTER 5 SOFTWARE TESTING.....	5-1
CHAPTER 6 SOFTWARE MAINTENANCE.....	6-1
CHAPTER 7 SOFTWARE CONFIGURATION MANAGEMENT.....	7-1
CHAPTER 8 SOFTWARE ENGINEERING MANAGEMENT.....	8-1
CHAPTER 9 SOFTWARE ENGINEERING PROCESS.....	9-1
CHAPTER 10 SOFTWARE ENGINEERING TOOLS AND METHODS.....	10-1
CHAPTER 11 SOFTWARE QUALITY.....	11-1
CHAPTER 12 RELATED DISCIPLINES OF SOFTWARE ENGINEERING.....	12-1
APPENDIX A KNOWLEDGE AREA DESCRIPTION SPECIFICATIONS FOR THE IRONMAN VERSION OF THE GUIDE TO THE SOFTWARE ENGINEERING BODY OF KNOWLEDGE.....	A-1
APPENDIX B EVOLUTION OF THE GUIDE TO THE SOFTWARE ENGINEERING BODY OF KNOWLEDGE.....	B-1
APPENDIX C ALLOCATION OF IEEE AND ISO SOFTWARE ENGINEERING STANDARDS TO SWEBOK KNOWLEDGE AREAS.....	C-1
APPENDIX D CLASSIFICATION OF TOPICS ACCORDING TO BLOOM’S TAXONOMY.....	D-1



# FOREWORD

In this Guide, the IEEE Computer Society establishes for the first time a baseline for the body of knowledge for the field of software engineering, and the work partially fulfills the Society's responsibility to promote the advancement of both theory and practice in this field. In so doing, the Society has been guided by the experience of disciplines with longer histories but was not bound either by their problems or their solutions.

It should be noted that the Guide does not purport to define the body of knowledge but rather to serve as a compendium and guide to the body of knowledge that has been developing and evolving over the past four decades. Furthermore, this body of knowledge is not static. The *Guide* must, necessarily, develop and evolve as software engineering matures. It nevertheless constitutes a valuable element of the software engineering infrastructure.

In 1958, John Tukey, the world-renowned statistician, coined the term *software*. The term *software engineering* was used in the title of a NATO conference held in Germany in 1968. The IEEE Computer Society first published its *Transactions on Software Engineering* in 1972. The committee established within the IEEE Computer Society for developing software engineering standards was founded in 1976.

The first holistic view of software engineering to emerge from the IEEE Computer Society resulted from an effort led by Fletcher Buckley to develop IEEE standard 730 for software quality assurance, which was completed in 1979. The purpose of IEEE Std 730 was to provide uniform, minimum acceptable requirements for preparation and content of software quality assurance plans. This standard was influential in completing the developing standards in the following topics: configuration management, software testing, software requirements, software design, and software verification and validation.

During the period 1981-1985, the IEEE Computer Society held a series of workshops concerning the application of software engineering standards. These workshops involved practitioners sharing their experiences with existing standards. The workshops also held sessions on planning for future standards, including one involving measures and metrics for software engineering products and processes. The planning also resulted in IEEE Std 1002, Taxonomy of Software Engineering Standards (1986), which provided a new, holistic view of software engineering. The standard describes the form and content of a software engineering standards taxonomy. It explains the various types of software engineering standards, their functional and external relationships, and the role of various functions participating in the software life cycle.

In 1990, planning for an international standard with an overall view was begun. The planning focused on reconciling the software process views from IEEE Std 1074 and the revised US DoD standard 2167A. The revision was eventually published as DoD Std 498. The international standard was completed in 1995 with designation, ISO/IEC 12207, and given the title of Standard for Software Life Cycle Processes. Std ISO/IEC 12207 provided a major point of departure for the body of knowledge captured in this book.

It was the IEEE Computer Society Board of Governors' approval of the motion put forward in May 1993 by Fletcher Buckley which resulted in the writing of this book. The Association for Computing Machinery (ACM) Council approved a related motion in August 1993. The two motions led to a joint committee under the leadership of Mario Barbacci and Stuart Zweben who served as cochairs. The mission statement of the joint committee was "To establish the appropriate sets(s) of criteria and norms for professional practice of software engineering upon which industrial decisions, professional certification, and educational curricula can be based." The steering committee organized task forces in the following areas:

1. Define Required Body of Knowledge and Recommended Practices.
2. Define Ethics and Professional Standards.
3. Define Educational Curricula for undergraduate, graduate, and continuing education.

This book supplies the first component: required body of knowledge and recommend practices.

The code of ethics and professional practice for software engineering was completed in 1998 and approved by both the ACM Council and the IEEE Computer Society Board of Governors. It has been adopted by numerous corporations and other organizations and is included in several recent textbooks.

The educational curriculum for undergraduates is being completed by a joint effort of the IEEE Computer Society and the ACM and is expected to be completed in 2004.

Every profession is based on a body of knowledge and recommended practices, although they are not always defined in a precise manner. In many cases, these are formally documented, usually in a form that permits them to be used for such purposes as accreditation of academic programs, development of education and training programs, certification of specialists, or professional licensing. Generally, a professional society or related body maintains custody of such a formal definition. In cases where no such formality exists, the body of knowledge and recommended practices are “generally recognized” by practitioners and may be codified in a variety of ways for different uses.

It is hoped that readers will find this book useful in guiding them toward the knowledge and resources they need in their lifelong career development as software engineering professionals.

The book is dedicated to Fletcher Buckley in recognition of his commitment to promoting software engineering as a professional discipline and his excellence as a software engineering practitioner in radar applications.

**Leonard L. Tripp, IEEE Fellow 2003**

**Chair, Professional Practices Committee, IEEE Computer Society (2001-2003)**

**Chair, Joint IEEE Computer Society and ACM Steering Committee**

**for the Establishment of Software Engineering as a Profession (1998-1999)**

**Chair, Software Engineering Standards Committee, IEEE Computer Society (1992-1998)**

# ASSOCIATE EDITORS

The following persons served as Associate Editors for either the Trial version published in 2001 or for the 2004 version.

## Software Requirements

Peter Sawyer and Gerald Kotonya, Computing Department, Lancaster University, UK, {p.sawyer}{g.kotonya}@lancaster.ac.uk

## Software Design

Guy Tremblay, Département d'informatique, UQAM, Canada, [tremblay.guy@uqam.ca](mailto:tremblay.guy@uqam.ca)

## Software Construction

Steve McConnell, Construx Software, USA, [Steve.McConnell@construx.com](mailto:Steve.McConnell@construx.com)

Terry Bollinger, the MITRE Corporation, USA, [terry@mitre.org](mailto:terry@mitre.org)

Philippe Gabrini, Département d'informatique, UQAM, Canada, [gabrini.philippe@uqam.ca](mailto:gabrini.philippe@uqam.ca)

Louis Martin, Département d'informatique, UQAM, Canada, [martin.louis@uqam.ca](mailto:martin.louis@uqam.ca)

## Software Testing

Antonia Bertolino and Eda Marchetti, ISTI-CNR, Italy, {antonia.bertolino}{eda.marchetti}@isti.cnr.it

## Software Maintenance

Thomas M. Pigoski, Techsoft Inc., USA, [tmpigoski@techsoft.com](mailto:tmpigoski@techsoft.com)

Alain April, École de technologie supérieure, Canada, [aapril@ele.etsmtl.ca](mailto:aapril@ele.etsmtl.ca)

## Software Configuration Management

John A. Scott, Lawrence Livermore National Laboratory, USA, [scott7@llnl.gov](mailto:scott7@llnl.gov)

David Nisse, USA, [nissed@worldnet.att.net](mailto:nissed@worldnet.att.net)

## Software Engineering Management

Dennis Frailey, Raytheon Company, USA, [DJFrailey@Raytheon.com](mailto:DJFrailey@Raytheon.com)

Stephen G. MacDonell, Auckland University of technology, New Zealand, [smacdne@aut.ac.nz](mailto:smacdne@aut.ac.nz)

Andrew R. Gray, University of Otago, New Zealand

## Software Engineering Process

Khaled El Emam, served while at the Canadian National Research Council, Canada, [khaled.el-emam@nrc-cnrc.gc.ca](mailto:khaled.el-emam@nrc-cnrc.gc.ca)

## Software Engineering Tools and Methods

David Carrington, School of Information Technology and Electrical Engineering, The University of Queensland, Australia, [davec@itee.uq.edu.au](mailto:davec@itee.uq.edu.au)

## Software Quality

Alain April, École de technologie supérieure, Canada, [aapril@ele.etsmtl.ca](mailto:aapril@ele.etsmtl.ca)

Dolores Wallace, retired from the National Institute of Standards and Technology, USA,

[Dolores.Wallace@nist.gov](mailto:Dolores.Wallace@nist.gov)

Larry Reeker, NIST, USA, [Larry.Reeker@nist.gov](mailto:Larry.Reeker@nist.gov)

## References Editor

Marc Bouisset, Département d'informatique, UQAM, [Bouisset.Marc@uqam.ca](mailto:Bouisset.Marc@uqam.ca)

# INDUSTRIAL ADVISORY BOARD

At the time of the publication, the following people formed the Industrial Advisory Board:

Mario R. Barbacci, Software Engineering Institute, representing the IEEE Computer Society

*Carl Chang, representing Computing Curricula 2001*

François Coallier, École de technologie supérieure, speaking as ISO/IEC JTC 1 / SC7 Chairman

Charles Howell, The MITRE Corporation

Anatol Kark, National Research Council of Canada

Philippe Kruchten, University of British Columbia, served as representative of Rational Software

Laure Le Bars, SAP Labs (Canada)

Steve McConnell, Construx Software

Dan Nash, Raytheon Company

Fred Otto, Canadian Council of Professional Engineers (CCPE)

Richard Metz, The Boeing Company

Larry Reeker, National Institute of Standards and Technology, Department of Commerce, USA

The following persons served along with the IAB in the Executive Change Control Board for the 2004 edition:

Donald Bagert, Rose-Hulman Institute of Technology, representing the IEEE Computer Society Professional Practices Committee

Ann Sobel, Miami University, representing the Computing Curricula Software Engineering Steering Committee

# **PANEL OF EXPERTS**

The following persons served on the panel of experts for the preparation of the Trial version of the Guide:

Steve McConnell, Construx Software

Roger Pressman, R.S. Pressman and Associates

Ian Sommerville, Lancaster University, UK

# REVIEW TEAM

The following people participated in the review process of this Guide for the Trial version and/or for the 2004 version.

Abbas, Rasha, Australia  
Abran, Alain, Canada  
Accioly, Carlos, Brazil  
Ackerman, Frank, USA  
Akiyama, Yoshihiro, Japan  
Al-Abdullah, Mohammad, USA  
Alarcon, Miren Idoia, Spain  
Alawy, Ahmed, USA  
Alleman, Glen, USA  
Allen, Bob, Canada  
Allen, David, USA  
Amorosa, Francesco, Italy  
Amyot, Daniel, Canada  
Andrade, Daniel, Brazil  
April, Alain, Canada  
Arroyo-Figueror, Javier, USA  
Ashford, Sonny, USA  
Atsushi, Sawada, Japan  
Backitis Jr., Frank, USA  
Bagert, Donald, USA  
Baker, Jr., David, USA  
Baker, Theodore, USA  
Baldwin, Mark, USA  
Bales, David, UK  
Bamberger, Judy, USA  
Banerjee, Bakul, USA  
Barber, Scott, USA  
Barker, Harry, UK  
Barnes, Julie, USA  
Barney, David, Australia  
Barros, Rafael, Colombia  
Bastarache, Louis, Canada  
Bayer, Steven, USA  
Beaulac, Adeline, Canada  
Beck, William, USA  
Beckman, Kathleen, USA  
Below, Doreen, USA  
Benediktsson, Oddur, Iceland  
Ben-Menachem, Mordechai, Israel  
Bergeron, Alain, Canada  
Berler, Alexander, Greece  
Bernet, Martin, USA  
Bernstein, Larry, USA  
Bertram, Martin, Germany  
Bialik, Tracy, USA  
Bielikova, Maria, Slovakia

Bierwolf, Robert, The Netherlands  
Bisbal, Jesus, Ireland  
Boivin, Michel, Canada  
Bolton, Michael, Canada  
Bomitali, Evelino, Italy  
Bonderer, Reto, Switzerland  
Bonk, Francis, USA  
Booch, Grady, USA  
Booker, Glenn, USA  
Börstler, Jürgen, Sweden  
Borzovs, Juris, Latvia  
Botting, Richard, USA  
Bourque, Pierre, Canada  
Bowen, Thomas, USA  
Boyd, Milt, USA  
Boyer, Ken, USA  
Brashear, Phil, USA  
Briggs, Steve, USA  
Bright, Daniela, USA  
Brosseau, Jim, Canada  
Brotbeck, George, USA  
Brown, Normand, Canada  
Bruhn, Anna, USA  
Brune, Kevin, USA  
Bryant, Jeanne, USA  
Buglione, Luigi, Italy  
Bullock, James, USA  
Burns, Robert, USA  
Burnstein, Ilene, USA  
Byrne, Edward, USA  
Calizaya, Percy, Peru  
Carreon, Juan, USA  
Carroll, Sue, USA  
Carruthers, Kate, Australia  
Caruso, Richard, USA  
Carvalho, Paul, Canada  
Case, Pam, USA  
Cavanaugh, John, USA  
Celia, John A., USA  
Chalupa Sampaio, Alberto Antonio, Portugal  
Chan, F.T., Hong Kong  
Chan, Keith, Hong Kong  
Chandra, A.K., India  
Chang, Wen-Kui, Taiwan  
Chapin, Ned, USA

Charette, Robert, USA  
Chevrier, Marielle, Canada  
Chi, Donald, USA  
Chiew, Vincent, Canada  
Chilenski, John, USA  
Chow, Keith, Italy  
Ciciliani, Ricardo, Argentina  
Clark, Glenda, USA  
Cleavenger, Darrell, USA  
Cloos, Romain, Luxembourg  
Coallier, François, Canada  
Coblentz, Brenda, USA  
Cohen, Phil, Australia  
Collard, Ross, New Zealand  
Collignon, Stephane, Australia  
Connors, Kathy Jo, USA  
Cooper, Daniel, USA  
Councill, Bill, USA  
Cox, Margery, USA  
Cunin, Pierre-Yves, France  
DaLuz, Joseph, USA  
Dampier, David, USA  
Daneva, Maya, Canada  
Daneva, Maya, Canada  
Daughtry, Taz, USA  
Davis, Ruth, USA  
De Cesare, Sergio, UK  
Dekleva, Sasa, USA  
Del Castillo, Federico, Peru  
Del Dago, Gustavo, Argentina  
DeWeese, Perry, USA  
Di Nunno, Donn, USA  
Diaz-Herrera, Jorge, USA  
Dieste, Oscar, Spain  
Dion, Francis, Canada  
Dixon, Wes, USA  
Dolado, Javier, Spain  
Donaldson, John, UK  
Dorantes, Marco, Mexico  
Dorofee, Audrey, USA  
Douglass, Keith, Canada  
Du, Weichang, Canada  
Duben, Anthony, USA  
Dudash, Edward, USA  
Duncan, Scott, USA  
Duong, Vinh, Canada  
Durham, George, USA

Dutil, Daniel, Canada  
 Dutton, Jeffrey, USA  
 Ebert, Christof, France  
 Edge, Gary, USA  
 Edwards, Helen Maria, UK  
 El-Kadi, Amr, Egypt  
 Endres, David, USA  
 Engelmann, Franz, Switzerland  
 Escue, Marilyn, USA  
 Espinoza, Marco, Peru  
 Fay, Istvan, Hungary  
 Fayad, Mohamed, USA  
 Fendrich, John, USA  
 Ferguson, Robert, USA  
 Fernandez, Eduardo, USA  
 Fernandez-Sanchez, Jose Luis, Spain  
 Filgueiras, Lucia, Brazil  
 Finkelstein, Anthony, UK  
 Flinchbaugh, Scott, USA  
 Forrey, Arden, USA  
 Fortenberry, Kirby, USA  
 Foster, Henrietta, USA  
 Fowler, Martin, USA  
 Fowler, John Jr., USA  
 Fox, Christopher, USA  
 Frankl, Phyllis, USA  
 Freibergs, Imants, Latvia  
 Frezza, Stephen, USA  
 Fruehauf, Karol, Switzerland  
 Fuggetta, Alphonso, Italy  
 Fujii, Roger, USA  
 FUSCHI, David Luigi, Italy  
 Fuschi, David Luigi, Italy  
 Gabrini, Philippe, Canada  
 Gagnon, Eric, Canada  
 Ganor, Eitan, Israel  
 Garbajosa, Juan, Spain  
 Garceau, Benoît, Canada  
 Garcia-Palencia, Omar, Colombia  
 Garner, Barry, USA  
 Gelperin, David, USA  
 Gersting, Judith, Hawaii  
 Giesler, Gregg, USA  
 Gil, Indalecio, Spain  
 Gilchrist, Thomas, USA  
 Giurescu, Nicolae, Canada  
 Glass, Robert, USA  
 Glynn, Garth, UK  
 Goers, Ron, USA  
 Gogates, Gregory, USA  
 Goldsmith, Robin, USA  
 Goodbrand, Alan, Canada  
 Gorski, Janusz, Poland  
 Graybill, Mark, USA  
 Gresse von Wangenheim, Christiane, Brazil  
 Grigonis, George, USA  
 Gupta, Arun, USA  
 Gustafson, David, USA  
 Gutcher, Frank, USA  
 Haas, Bob, USA  
 Hagar, Jon, USA  
 Hagstrom, Erick, USA  
 Hailey, Victoria, Canada  
 Hall, Duncan, New Zealand  
 Haller, John, USA  
 Halstead-Nussloch, Richard, USA  
 Hamm, Linda, USA  
 Hankewitz, Lutz, Germany  
 Harker, Rob, USA  
 Hart, Hal, USA  
 Hart, Ronald, USA  
 Hartner, Clinton, USA  
 Hayeck, Elie, USA  
 He, Zhonglin, UK  
 Hedger, Dick, USA  
 Hefner, Rick, USA  
 Heinrich, Mark, USA  
 Heinze, Sherry, Canada  
 Hensel, Alan, USA  
 Herrmann, Debra, USA  
 Hesse, Wolfgang, Germany  
 Hilburn, Thomas, USA  
 Hill, Michael, USA  
 Ho, Vinh, Canada  
 Hodgen, Bruce, Australia  
 Hodges, Brett, Canada  
 Hoffman, Douglas, Canada  
 Hoffman, Michael, USA  
 Hoganson, Tammy, USA  
 Hollocker, Chuck, USA  
 Horch, John, USA  
 Howard, Adrian, UK  
 Huang, Hui Min, USA  
 Hung, Chih-Cheng, USA  
 Hung, Peter, USA  
 Hunt, Theresa, USA  
 Hunter, John, USA  
 Hvannberg, Ebba Thora, Iceland  
 Hybertson, Duane, USA  
 Ikiz, Seckin, Turkey  
 Iyengar, Dwaraka, USA  
 Jackelen, George, USA  
 Jaeger, Dawn, USA  
 Jahnke, Jens, Canada  
 James, Jean, USA  
 Jino, Mario, Brazil  
 Johnson, Vandy, USA  
 Jones, Griffin, USA  
 Jones, James E., USA  
 Jones, Alan, UK  
 Jones, James, USA  
 Jones, Larry, Canada  
 Jones, Paul, USA  
 Ju, Dehua, China  
 Juan-Martinez, Manuel-Fernando, Spain  
 Juhasz, Zoltan, Hungary  
 Juristo, Natalia, Spain  
 Kaiser, Michael, Switzerland  
 Kambic, George, USA  
 Kamthan, Pankaj, Canada  
 Kaner, Cem, USA  
 Kark, Anatol, Canada  
 Kasser, Joe, USA  
 Kasser, Joseph, Australia  
 Katz, Alf, Australia  
 Kececi, Nihal, Canada  
 Kell, Penelope, USA  
 Kelly, Diane, Canada  
 Kelly, Frank, USA  
 Kenett, Ron, Israel  
 Kenney, Mary L., USA  
 Kerievsky, Joshua, USA  
 Kerr, John, USA  
 Kierzyk, Robert, USA  
 Kinsner, W., Canada  
 Kirkpatrick, Harry, USA  
 Kittiel, Linda, USA  
 Klappholz, David, USA  
 Klein, Joshua, Israel  
 Knight, Claire, UK  
 Knoke, Peter, USA  
 Ko, Roy, Hong Kong  
 Kolewe, Ralph, Canada  
 Komal, Surinder Singh, Canada  
 Kovalovsky, Stefan, Austria  
 Krauth, Péter, Hungary  
 Krishnan, Nirmala, USA  
 Kromholz, Alfred, Canada  
 Kruchten, Philippe, Canada  
 Kuehner, Nathanael, Canada  
 Kwok, Shui Hung, Canada  
 Lacroix, Dominique, Canada  
 LaMotte, Stephen W., USA  
 Land, Susan, USA  
 Lange, Douglas, USA  
 Laporte, Claude, Canada  
 Lawlis, Patricia, USA  
 Le, Thach, USA  
 Leavitt, Randal, Canada  
 LeBel, Réjean, Canada  
 Leciston, David, USA  
 Lee, Chanyoung, USA  
 Lehman, Meir (Manny), UK

Leigh, William, USA  
 Lembo, Jim, USA  
 Lenss, John, USA  
 Leonard, Eugene, USA  
 Lethbridge, Timothy, Canada  
 Leung, Hareton, Hong Kong  
 Lever, Ronald, The Netherlands  
 Levesque, Ghislain, Canada  
 Ley, Earl, USA  
 Linders, Ben, Netherlands  
 Linscomb, Dennis, USA  
 Little, Joyce Currie, USA  
 Logan, Jim, USA  
 Long, Carol, UK  
 Lounis, Hakim, Canada  
 Low, Graham, Australia  
 Lutz, Michael, USA  
 Lynch, Gary, USA  
 Machado, Cristina, Brazil  
 MacKay, Stephen, Canada  
 MacKenzie, Garth, USA  
 MacNeil, Paul, USA  
 Magel, Kenneth, USA  
 Mains, Harold, USA  
 Malak, Renee, USA  
 Maldonado, José Carlos, Brazil  
 Marcos, Esperanza, Spain  
 Marinescu, Radu, Romania  
 Marm, Waldo, Peru  
 Marusca, Ioan, Canada  
 Matlen, Duane, USA  
 Matsumoto, Yoshihiro, Japan  
 McBride, Tom, Australia  
 McCarthy, Glenn, USA  
 McChesney, Ian, UK  
 McCormick, Thomas, Canada  
 McCown, Christian, USA  
 McDonald, Jim, USA  
 McGrath Carroll, Sue, USA  
 McHutchison, Diane, USA  
 McKinnell, Brian, Canada  
 McMichael, Robert, USA  
 McMillan, William, USA  
 McQuaid, Patricia, USA  
 Mead, Nancy, USA  
 Meeuse, Jaap, The Netherlands  
 Meier, Michael, USA  
 Meisenzahl, Christopher, USA  
 Melhart, Bonnie, USA  
 Mengel, Susan, USA  
 Meredith, Denis, USA  
 Meyerhoff, Dirk, Germany  
 Mili, Hafedh, Canada  
 Miller, Chris, Netherlands  
 Miller, Keith, USA  
 Miller, Mark, USA  
 Miranda, Eduardo, Canada  
 Mistrik, Ivan, Germany  
 Mitasiunas, Antanas, Lithuania  
 Modell, Howard, USA  
 Modell, Staiger, USA  
 Modesitt, Kenneth, USA  
 Moland, Kathryn, USA  
 Moldavsky, Symon, Ukraine  
 Montequín, Vicente R., Spain  
 Moreno, Ana Maria, Spain  
 Mosiuoa, Tseliso, Lesotho  
 Moudry, James, USA  
 Msheik, Hamdan, Canada  
 Mularz, Diane, USA  
 Mullens, David, USA  
 Müllerburg, Monika, Germany  
 Murali, Nagarajan, Australia  
 Murphy, Mike, USA  
 Napier, John, USA  
 Narasimhadevara, Sudha, Canada  
 Narawane, Ranjana, India  
 Narayanan, Ramanathan, India  
 Navarro Ramirez, Daniel, Mexico  
 Navas Plano, Francisco, Spain  
 Navrat, Pavol, Slovakia  
 Neumann, Dolly, USA  
 Nguyen-Kim, Hong, Canada  
 Nikandros, George, Australia  
 Nishiyama, Tetsuto, Japan  
 Nunn, David, USA  
 O'Donoghue, David, Ireland  
 Oliver, David John, Australia  
 Olson, Keith, USA  
 Oskarsson, Östen, Sweden  
 Ostrom, Donald, USA  
 Oudshoorn, Michael, Australia  
 Owen, Cherry, USA  
 Pai, Hsueh-Ieng, Canada  
 Parrish, Lee, USA  
 Parsons, Samuel, USA  
 Patel, Dilip, UK  
 Paulk, Mark, USA  
 Pavelka, Jan, Czech Republic  
 Pavlov, Vladimir, Ukraine  
 Pawlyszyn, Blanche, USA  
 Pecceu, Didier, France  
 Perisic, Branko, Yugoslavia  
 Perry, Dale, USA  
 Peters, Dennis, Canada  
 Petersen, Erik, Australia  
 Pfahl, Dietmar, Germany  
 Pfeiffer, Martin, Germany  
 Phillips, Dwayne, USA  
 Phipps, Robert, USA  
 Phister, Paul, USA  
 Phister, Jr., Paul, USA  
 Piattini, Mario, Spain  
 Piersall, Jeff, USA  
 Pillai, S.K., India  
 Pinder, Alan, UK  
 Pinheiro, Francisco A., Brazil  
 Plekhanova, Valentina, UK  
 Poon, Peter, USA  
 Poppendieck, Mary, USA  
 Powell, Mace, USA  
 Predenkoski, Mary, USA  
 Prescott, Allen, USA  
 Pressman, Roger, USA  
 Price, Art, USA  
 Price, Margaretha, USA  
 Pullum, Laura, USA  
 Purser, Keith, USA  
 Purssey, John, Australia  
 Pustaver, John, USA  
 Quinn, Anne, USA  
 Radnell, David, Australia  
 Rae, Andrew, UK  
 Rafea, Ahmed, Egypt  
 Ramsden, Patrick, Australia  
 Rao, N. Vyaghrewara, India  
 Rawsthorne, Dan, USA  
 Reader, Katherine, USA  
 Reddy, Vijay, USA  
 Redwine, Samuel, USA  
 Reed, Karl, Australia  
 Reedy, Ann, USA  
 Reeker, Larry, USA  
 Rethard, Tom, USA  
 Reussner, Ralf, Germany  
 Rios, Joaquin, Spain  
 Risbec, Philippe, France  
 Roach, Steve, USA  
 Robillard, Pierre, Canada  
 Rocha, Zalkind, Brazil  
 Rodeiro Iglesias, Javier, Spain  
 Rodriguez-Dapena, Patricia, Spain  
 Rogoway, Paul, Israel  
 Rontondi, Guido, Italy  
 Roose, Philippe, France  
 Rosca, Daniela, USA  
 Rosenberg, Linda, USA  
 Rourke, Michael, Australia  
 Rout, Terry, Australia  
 Rufer, Russ, USA  
 Ruiz, Francisco, Spain  
 Ruocco, Anthony, USA  
 Rutherford, Rebecca, USA  
 Ryan, Michael, Ireland  
 Salustri, Filippo, Canada

Salustri, Filippo, Canada  
Salwin, Arthur, USA  
Sanden, Bo, USA  
Sandmayr, Helmut, Switzerland  
Santana Filho, Ozeas Vieira,  
Brazil  
Sato, Tomonobu, Japan  
satyadas, antony, USA  
Satyadas, Antony, USA  
Schaaf, Robert, USA  
Scheper, Charlotte, USA  
Schiffel, Jeffrey, USA  
Schlicht, Bill, USA  
Schrott, William, USA  
Schwarm, Stephen, USA  
Schweppe, Edmund, USA  
Sebern, Mark, USA  
Seffah, Ahmed, Canada  
Selby, Nancy, USA  
Selph, William, USA  
Sen, Dhruva, USA  
Senechal, Raymond, USA  
Sepulveda, Christian, USA  
Setlur, Atul, USA  
Sharp, David, USA  
Shepard, Terry, Canada  
Shepherd, Alan, Germany  
Shillato, Rrobert W, USA  
Shintani, Katsutoshi, Japan  
Silva, Andres, Spain  
Silva, Andres, Spain  
Singer, Carl, USA  
Sinnott, Paul, UK  
Sintzoff, André, France  
Sitte, Renate, Australia  
Sky, Richard, USA  
Smilie, Kevin, USA  
Smith, David, USA  
Sophatsathit, Peraphon, Thailand  
Sorensen, Reed, USA

Soundarajan, Neelam, USA  
Sousa Santos, Frederico,  
Portugal  
Spillers, Mark, USA  
Spinellis, Diomidis, Greece  
Splaine, Steve, USA  
Springer, Donald, USA  
Staiger, John, USA  
Starai, Thomas, USA  
Steurs, Stefan, Belgium  
St-Pierre, Denis, Canada  
Stroulia, Eleni, Canada  
Subramanian, K.S., India  
Sundaram, Sai, UK  
Swanek, James, USA  
Swearingen, Sandra, USA  
Szymkowiak, Paul, Canada  
Tamai, Tetsuo, Japan  
Tasker, Dan, New Zealand  
Taylor, Stanford, USA  
Terekhov, Andrey A., Russian  
Federation  
Terski, Matt, USA  
Thayer, Richard, USA  
Thomas, Michael, USA  
Thompson, A. Allan, Australia  
Thompson, John Barrie, UK  
Titus, Jason, USA  
Tockey, Steve, USA  
Tovar, Edmundo, Spain  
Towhidnejad, Massood, USA  
Trellue, Patricia, USA  
Trèves, Nicolas, France  
Troy, Elliot, USA  
Tsui, Frank, USA  
Tsuneo, Furuyama, Japan  
Tuohy, Kenney, USA  
Tuohy, Marsha P., USA  
Turczyn, Stephen, USA  
Upchurch, Richard, USA

Urbanowicz, Theodore, USA  
Van Duine, Dan, USA  
Van Ekris, Jaap, Netherlands  
Van Oosterhout, Bram, Australia  
Vander Plaats, Jim, USA  
Vegas, Sira, Spain  
Verner, June, USA  
Villas-Boas, André, Brazil  
Vollman, Thomas, USA  
Walker, Richard, Australia  
Walsh, Bucky, USA  
Wang, Yingxu, Sweden  
Wear, Larry, USA  
Weigel, richard, USA  
Weinstock, Charles, USA  
Wenyin, Liu, China  
Werner, Linda, USA  
Wheeler, David, USA  
White, Nathan, USA  
White, Stephanie, USA  
Whitmire, Scott, USA  
Wijbrans, Klaas, The  
Netherlands  
Wijbrans-Roodbergen, Margot,  
The Netherlands  
Wilkie, Frederick, UK  
Wille, Cornelius, Germany  
Wilson, Charles, USA  
Wilson, Leon, USA  
Wilson, Russell, USA  
Woechan, Kenneth, USA  
Woit, Denise, Canada  
Yadin, Aharon, Israel  
Yih, Swu, Taiwan  
Young, Michal, USA  
Yrivarren, Jorge, Peru  
Znotka, Juergen, Germany  
Zuser, Wolfgang, Austria  
Zvegintzov, Nicholas, USA  
Zweben, Stu, USA

**The following motion was unanimously adopted by the Industrial Advisory Board  
on 6 February 2004.**

*The Industrial Advisory Board finds that the Software Engineering Body of Knowledge project initiated in 1998 has been successfully completed; and endorses the 2004 Version of the Guide to the SWEBOK and commends it to the IEEE Computer Society Board of Governors for their approval.*

**The following motion was adopted by the IEEE Computer Society Board of  
Governors in February 2004.**

*MOVED, that the Board of Governors of the IEEE Computer Society approves the 2004 Edition of the Guide to the Software Engineering Body of Knowledge and authorizes the Chair of the Professional Practices Committee to proceed with printing.*

# PREFACE

Software engineering is an emerging discipline and there are unmistakable trends indicating an increasing level of maturity:

- ◆ Several universities throughout the world offer undergraduate degrees in software engineering. For example, such degrees are offered at the University of New South Wales (Australia), McMaster University (Canada), the Rochester Institute of Technology (US), the University of Sheffield (UK), and other universities.
- ◆ In the US, the Engineering Accreditation Commission of the Accreditation Board for Engineering and Technology (ABET) is responsible for the accreditation of undergraduate software engineering programs.
- ◆ The Canadian Information Processing Society has published criteria to accredit software engineering undergraduate university programs.
- ◆ The Software Engineering Institute's Capability Maturity Model for Software (SW CMM) and the new Capability Maturity Model Integration (CMMI) are used to assess organizational capability for software engineering. The famous ISO 9000 quality management standards have been applied to software engineering by the new ISO/IEC 90003.
- ◆ The Texas Board of Professional Engineers has begun to license professional software engineers.
- ◆ The Association of Professional Engineers and Geoscientists of British Columbia (APEGBC) has begun registering software professional engineers, and the Professional Engineers of Ontario (PEO) has also announced requirements for licensing.
- ◆ The Association for Computing Machinery (ACM) and the Computer Society of the Institute of Electrical and Electronics Engineers (IEEE) have jointly developed and adopted a Code of Ethics and Professional Practice for software engineering professionals.<sup>1</sup>
- ◆ The IEEE Computer Society offers the Certified Software Development Professional certification for software engineering. The Institute for Certification of Computing Professionals (ICCP)

has long offered a certification for computing professionals.

All of these efforts are based upon the presumption that there is a Body of Knowledge that should be mastered by practicing software engineers. The Body of Knowledge exists in the literature that has accumulated over the past thirty years. This book provides a Guide to that Body of Knowledge.

## PURPOSE

The purpose of the Guide to the Software Engineering Body of Knowledge is to provide a consensually validated characterization of the bounds of the software engineering discipline and to provide a topical access to the Body of Knowledge supporting that discipline. The Body of Knowledge is subdivided into ten software engineering Knowledge Areas (KA) plus an additional chapter providing an overview of the KAs of strongly related disciplines. The descriptions of the KAs are designed to discriminate among the various important concepts, permitting readers to find their way quickly to subjects of interest. Upon finding a subject, readers are referred to key papers or book chapters selected because they succinctly present the knowledge.

In browsing the Guide, readers will note that the content is markedly different from computer science. Just as electrical engineering is based upon the science of physics, software engineering should be based, among other things, upon computer science. In these two cases, though, the emphasis is necessarily different. Scientists extend our knowledge of the laws of nature while engineers apply those laws of nature to build useful artifacts, under a number of constraints. Therefore, the emphasis of the Guide is placed on the construction of useful software artifacts.

Readers will also notice that many important aspects of information technology that may constitute important software engineering knowledge are not covered in the Guide, including specific programming languages, relational databases, and networks. This is a consequence of an engineering-based approach. In all fields—not only computing—the designers of engineering curricula have realized that specific technologies are replaced much more rapidly than the engineering work force. An engineer must be equipped with the essential knowledge that supports the selection of the appropriate technology at the appropriate time in the appropriate circumstance. For

---

<sup>1</sup> The ACM/CS Software Engineering Code of Ethics and Professional Practice can be found at <http://www.computer.org/certification/ethics.htm>.

example, software might be built in Fortran using functional decomposition or in C++ using object-oriented techniques. The techniques for software configuring instances of those systems would be quite different. But, the principles and objectives of configuration management remain the same. The Guide therefore does not focus on the rapidly changing technologies, although their general principles are described in relevant KAs.

These exclusions demonstrate that this Guide is necessarily incomplete. The Guide covers software engineering knowledge that is necessary but not sufficient for a software engineer. Practicing software engineers will need to know many things about computer science, project management, and systems engineering—to name a few—that fall outside the Body of Knowledge characterized by this Guide. However, stating that this information should be known by software engineers is not the same as stating that this knowledge falls within the bounds of the software engineering discipline. Instead, it should be stated that software engineers need to know some things taken from other disciplines—and that is the approach adopted in this Guide. So, this Guide characterizes the Body of Knowledge falling within the scope of software engineering and provides references to relevant information from other disciplines. A chapter of the Guide provides a taxonomical overview of the related disciplines derived from authoritative sources.

The emphasis on engineering practice leads the Guide toward a strong relationship with the normative literature. Most of the computer science, information technology, and software engineering literature provides information useful to software engineers, but a relatively small portion is normative. A normative document prescribes what an engineer should do in a specified situation rather than providing information that might be helpful. The normative literature is validated by consensus formed among practitioners and is concentrated in standards and related documents. From the beginning, the SWEBOK project was conceived as having a strong relationship to the normative literature of software engineering. The two major standards bodies for software engineering (IEEE Computer Society Software Engineering Standards Committee and ISO/IEC JTC1/SC7) are represented in the project. Ultimately, we hope that software engineering practice standards will contain principles directly traceable to the Guide.

#### INTENDED AUDIENCE

The Guide is oriented toward a variety of audiences, all over the world. It aims to serve public and private

organizations in need of a consistent view of software engineering for defining education and training requirements, classifying jobs, developing performance evaluation policies, or specifying software development tasks. It also addresses practicing, or managing, software engineers and the officials responsible for making public policy regarding licensing and professional guidelines. In addition, professional societies and educators defining the certification rules, accreditation policies for university curricula, and guidelines for professional practice will benefit from SWEBOK, as well as the students learning the software engineering profession and educators and trainers engaged in defining curricula and course content.

#### EVOLUTION OF THE GUIDE

From 1993 to 2000, the IEEE Computer Society and the Association for Computing Machinery (ACM) cooperated in promoting the professionalization of software engineering through their joint Software Engineering Coordinating Committee (SWECC). The Code of Ethics was completed under stewardship of the SWECC primarily through volunteer efforts. The SWEBOK project was initiated by the SWECC in 1998.

The SWEBOK project's scope, the variety of communities involved, and the need for broad participation suggested a need for full-time rather than volunteer management. For this purpose, the IEEE Computer Society contracted the Software Engineering Management Research Laboratory at the Université du Québec à Montréal (UQAM) to manage the effort. In recent years, UQAM has been joined by the École de technologie supérieure, Montréal, Québec.

The project plan comprised three successive phases: Strawman, Stoneman, and Ironman. An early prototype, Strawman, demonstrated how the project might be organized. The publication of the widely circulated Trial Version of the Guide in 2001 marked the end of the Stoneman phase of the project and initiated a period of trial usage. The current Guide marks the end of the Ironman period by providing a Guide that has achieved consensus through broad review and trial application.

The project team developed two important principles for guiding the project: *transparency* and *consensus*. By transparency, we mean that the development process is itself documented, published, and publicized so that important decisions and status are visible to all concerned parties. By consensus, we mean that the only practical method for legitimizing a statement of this kind is through broad participation and agreement by all significant sectors of the relevant community.

Literally hundreds of contributors, reviewers, and trial users have played a part in producing the current document.

Like any software project, the SWEBOK project has many stakeholders—some of which are formally represented. An Industrial Advisory Board, composed of representatives from industry (Boeing, Construx Software, the MITRE Corporation, Rational Software, Raytheon Systems, and SAP Labs-Canada), research agencies (National Institute of Standards and Technology, National Research Council of Canada), the Canadian Council of Professional Engineers, and the IEEE Computer Society, has provided financial support for the project. The IAB's generous support permits us to make the products of the SWEBOK project publicly available without any charge (see <http://www.swebok.org>). IAB membership is supplemented with the chairs of ISO/IEC JTC1/SC7 and the related Computing Curricula 2001 initiative. The IAB reviews and approves the project plans, oversees consensus building and review processes, promotes the project, and lends credibility to the effort. In general, it ensures the relevance of the effort to real-world needs.

The Trial Version of the Guide was the product of extensive review and comment. In three public review cycles, a total of roughly 500 reviewers from 42 countries provided roughly 9,000 comments, all of which are available at [www.swebok.org](http://www.swebok.org). To produce the current version, we released the Trial Version for extensive trial usage. Trial application in specialized studies resulted in 17 papers describing good aspects of the Guide, as well as aspects needing improvement. A Web-based survey captured additional experience: 573 individuals from 55 countries registered for the survey; 124 reviewers from 21 countries actually provided comments—1,020 of them. Additional suggestions for improvement resulted from liaison with related organizations and efforts: IEEE-CS/ACM Computing Curricula Software Engineering; the IEEE CS Certified Software Development Professional project; ISO/IEC JTC1/SC7 (software and systems engineering standards); the IEEE Software Engineering Standards Committee; the American Society for Quality, Software Division; and an engineering professional society, the Canadian Council of Professional Engineers.

#### CHANGES SINCE THE TRIAL VERSION

The overall goal of the current revision was to improve the readability, consistency, and usability of the Guide. This implied a general rewrite of the entire text to make the style consistent throughout the document. In several cases, the topical breakdown of the KA was

rearranged to make it more usable, but we were careful to include the same information that was approved by the earlier consensus process. We updated the reference list so that it would be easier to obtain the references.

Trial usage resulted in the recommendation that three KAs should be rewritten. Practitioners remarked that the Construction KA was difficult to apply in a practical context. The Management KA was perceived as being too close to general management and not sufficiently specific to software engineering concerns. The Quality KA was viewed as an uncomfortable mix of process quality and product quality; it was revised to emphasize the latter.

Finally, some KAs were revised to remove material duplicating that of other KAs.

#### LIMITATIONS

Even though the Guide has gone through an elaborate development and review process, the following limitations of this process must be recognized and stated:

- ♦ Software engineering continues to be infused with new technology and new practices. Acceptance of new techniques grows and older techniques are discarded. The topics listed as “generally accepted” in this Guide are carefully selected at this time. Inevitably, though, the selection will need to evolve.
- ♦ The amount of literature that has been published on software engineering is considerable and the reference material included in this Guide should not be seen as a definitive selection but rather as a reasonable selection. Obviously, there are other excellent authors and excellent references than those included in the Guide. In the case of the Guide, references were selected because they are written in English, readily available, recent, and easily readable, and—taken as a group—they provide coverage of the topics within the KA.
- ♦ Important and highly relevant reference material written in languages other than English have been omitted from the selected reference material.

Additionally, one must consider that

- ♦ Software engineering is an emerging discipline. This is especially true if you compare it to certain more established engineering disciplines. This means notably that the boundaries between the KAs of software engineering and between software engineering and its related disciplines remain a matter for continued evolution.

The contents of this Guide must therefore be viewed as an “informed and reasonable” characterization of the software engineering Body of Knowledge and as baseline for future evolution. Additionally, please note that the Guide is not attempting nor does it claim to

replace or amend in any way laws, rules, and procedures that have been defined by official public policy makers around the world regarding the practice and definition of engineering and software engineering in particular.

*Alain Abran*  
École de technologie supérieure

***Executive Editors of the  
Guide to the Software  
Engineering Body of  
Knowledge***

*James W. Moore*  
The MITRE Corporation

*Pierre Bourque*  
École de Technologie Supérieure

***Editors of the Guide to  
the Software Engineering  
Body of Knowledge***

*Robert Dupuis*  
Université du Québec à Montréal

*Leonard Tripp*  
1999 President  
IEEE Computer Society

***Chair of the Professional  
Practices Committee,  
IEEE Computer Society  
(2001-2003)***

*December 2004*

The SWEBOK project Web site is <http://www.swebok.org/>

#### **ACKNOWLEDGMENTS**

The SWEBOK editorial team gratefully acknowledges the support provided by the members of the Industrial Advisory Board. Funding for this project has been provided by the ACM, Boeing, the Canadian Council of Professional Engineers, Construx Software, the IEEE Computer Society, the MITRE Corporation, the National Institute of Standards and Technology, the National Research Council of Canada, Rational Software, Raytheon Company, and SAP Labs (Canada). The team is thankful for the counsel provided by the Panel of Experts. The team also appreciates the important work performed by the Associate Editors. We would also like to express our gratitude for initial work on the Knowledge Area Descriptions completed by Imants Freibergs, Stephen Frezza, Andrew Gray, Vinh T. Ho, Michael Lutz, Larry Reeker, Guy Tremblay, Chris Verhoef, and Sybille Wolff. The editorial team must also acknowledge the indispensable contribution of the hundreds of reviewers.

Serge Oigny, Suzanne Paquette, Keith Paton, Dave Rayford, Normand Séguin, Paul Sinnett, Denis St-Pierre, Dale Strok, Pascale Tardif, Louise Thibaudeau, Dolores Wallace, Évariste Valery Bevo Wandji, and Michal Young.

Finally, there are surely other people who have contributed to this Guide, either directly or indirectly, whose names we have inadvertently omitted. To those people, we offer our tacit appreciation and apologize for having omitted explicit recognition here.

The editorial team also wishes to thank the following people who contributed to the project in various ways: Mark Ardis, Yussef Belkebir, Michel Boivin, Julie Bonneau, Simon Bouchard, François Cossette, Vinh Duong, Gilles Gauthier, Michèle Hébert, Paula Hawthorn, Richard W. Heiman, Julie Hudon, Idrissa Konkobo, Rene Köppel, Lucette Lapointe, Claude Laporte, Luis Molinié, Hamdan Msheik, Iphigénie N'Diyae,



# CHAPTER 1

## INTRODUCTION TO THE GUIDE

In spite of the millions of software professionals worldwide and the ubiquitous presence of software in our society, software engineering has only recently reached the status of a legitimate engineering discipline and a recognized profession.

Achieving consensus by the profession on a core body of knowledge is a key milestone in all disciplines and had been identified by the IEEE Computer Society as crucial for the evolution of software engineering towards professional status. This Guide, written under the auspices of the Professional Practices Committee, is part of a multi-year project designed to reach such a consensus.

### WHAT IS SOFTWARE ENGINEERING?

The IEEE Computer Society defines software engineering as

“(1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.

(2) The study of approaches as in (1).”<sup>1</sup>

### WHAT IS A RECOGNIZED PROFESSION?

For software engineering to be fully known as a legitimate engineering discipline and a recognized profession, consensus on a core body of knowledge is imperative. This fact is well illustrated by Starr when he defines what can be considered a legitimate discipline and a recognized profession. In his Pulitzer Prize-winning book on the history of the medical profession in the USA, he states,

“The legitimization of professional authority involves three distinctive claims: first, that the knowledge and competence of the professional have been validated by a community of his or her peers; second, that this consensually validated knowledge rests on rational, scientific grounds; and third, that the professional’s judgment and advice are oriented toward a set of substantive values, such as health. These aspects of legitimacy correspond to the kinds of attributes—collegial, cognitive, and moral—usually embodied in the term “profession.”<sup>2</sup>

### WHAT ARE THE CHARACTERISTICS OF A PROFESSION?

Gary Ford and Norman Gibbs studied several recognized professions, including medicine, law, engineering, and

accounting.<sup>3</sup> They concluded that an engineering profession is characterized by several components:

- ♦ An initial *professional education* in a curriculum validated by society through *accreditation*
- ♦ Registration of fitness to practice via voluntary *certification* or mandatory *licensing*
- ♦ Specialized *skill development* and *continuing professional education*
- ♦ Communal support via a *professional society*
- ♦ A commitment to norms of conduct often prescribed in a *code of ethics*

This Guide contributes to the first three of these components. Articulating a Body of Knowledge is an essential step toward developing a profession because it represents a broad consensus regarding what a software engineering professional should know. Without such a consensus, no licensing examination can be validated, no curriculum can prepare an individual for an examination, and no criteria can be formulated for accrediting a curriculum. The development of consensus is also a prerequisite to the adoption of coherent skills development and continuing professional education programs in organizations.

### WHAT ARE THE OBJECTIVES OF THE SWEBOK PROJECT?

The Guide should not be confused with the Body of Knowledge itself, which already exists in the published literature. The purpose of the Guide is to describe what portion of the Body of Knowledge is generally accepted, to organize that portion, and to provide a topical access to it. Additional information on the meaning given to “generally accepted” can be found below and in Appendix A.

The Guide to the Software Engineering Body of Knowledge (SWEBOK) was established with the following five objectives:

1. To promote a consistent view of software engineering worldwide
2. To clarify the place—and set the boundary—of software engineering with respect to other disciplines such as computer science, project management, computer engineering, and mathematics
3. To characterize the contents of the software engineering discipline

---

<sup>1</sup> “IEEE Standard Glossary of Software Engineering Terminology,” IEEE std 610.12-1990, 1990.

<sup>2</sup> P. Starr, *The Social Transformation of American Medicine*, Basic Books, 1982, p. 15.

---

<sup>3</sup> G. Ford and N.E. Gibbs, *A Mature Profession of Software Engineering*, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pa., tech. report CMU/SEI-96-TR-004, Jan. 1996.

4. To provide a topical access to the Software Engineering Body of Knowledge
5. To provide a foundation for curriculum development and for individual certification and licensing material

The first of these objectives, a consistent worldwide view of software engineering, was supported by a development process which engaged approximately 500 reviewers from 42 countries in the Stoneman phase (1998–2001) leading to the Trial version, and over 120 reviewers from 21 countries in the Ironman phase (2003) leading to the 2004 version. More information regarding the development process can be found in the Preface and on the Web site ([www.swebok.org](http://www.swebok.org)). Professional and learned societies and public agencies involved in software engineering were officially contacted, made aware of this project, and invited to participate in the review process. Associate editors were recruited from North America, the Pacific Rim, and Europe. Presentations on the project were made at various international venues and more are scheduled for the upcoming year.

The second of the objectives, the desire to set a boundary for software engineering, motivates the fundamental organization of the Guide. The material that is recognized as being within this discipline is organized into the first ten Knowledge Areas (KAs) listed in Table 1. Each of these KAs is treated as a chapter in this Guide.

**Table 1** The SWEBOK Knowledge Areas (KAs)

Software requirements
Software design
Software construction
Software testing
Software maintenance
Software configuration management
Software engineering management
Software engineering process
Software engineering tools and methods
Software quality

In establishing a boundary, it is also important to identify what disciplines share that boundary, and often a common intersection, with software engineering. To this end, the Guide also recognizes eight related disciplines, listed in Table 2 (see Chapter 12, “Related Disciplines of Software Engineering”). Software engineers should, of course, have knowledge of material from these fields (and the KA descriptions may make reference to them). It is not, however, an objective of the SWEBOK Guide to characterize the knowledge of the related disciplines, but rather what knowledge is viewed as specific to software engineering.

**Table 2** Related disciplines

♦ Computer engineering	♦ Project management
♦ Computer science	♦ Quality management
♦ Management	♦ Software ergonomics
♦ Mathematics	♦ Systems engineering

### HIERARCHICAL ORGANIZATION

The organization of the KA descriptions or chapters supports the third of the project’s objectives—a characterization of the contents of software engineering. The detailed specifications provided by the project’s editorial team to the associate editors regarding the contents of the KA descriptions can be found in Appendix A.

The Guide uses a hierarchical organization to decompose each KA into a set of topics with recognizable labels. A two- or three-level breakdown provides a reasonable way to find topics of interest. The Guide treats the selected topics in a manner compatible with major schools of thought and with breakdowns generally found in industry and in software engineering literature and standards. The breakdowns of topics do not presume particular application domains, business uses, management philosophies, development methods, and so forth. The extent of each topic’s description is only that needed to understand the generally accepted nature of the topics and for the reader to successfully find reference material. After all, the Body of Knowledge is found in the reference material themselves, not in the Guide.

### REFERENCE MATERIAL AND MATRIX

To provide a topical access to the knowledge—the fourth of the project’s objectives—the Guide identifies reference material for each KA, including book chapters, refereed papers, or other recognized sources of authoritative information. Each KA description also includes a matrix relating the reference material to the listed topics. The total volume of cited literature is intended to be suitable for mastery through the completion of an undergraduate education plus four years of experience.

In this edition of the Guide, all KAs were allocated around 500 pages of reference material, and this was the specification the associate editors were invited to apply. It may be argued that some KAs, such as software design for instance, deserve more pages of reference material than others. Such modulation may be applied in future editions of the Guide.

It should be noted that the Guide does not attempt to be comprehensive in its citations. Much material that is both suitable and excellent is not referenced. Material was selected in part because—taken as a collection—it provides coverage of the topics described.

### DEPTH OF TREATMENT

From the outset, the question arose as to the depth of treatment

the Guide should provide. The project team adopted an approach which supports the fifth of the project’s objectives—providing a foundation for curriculum development, certification, and licensing. The editorial team applied the criterion of *generally accepted* knowledge, to be distinguished from advanced and research knowledge (on the grounds of maturity) and from specialized knowledge (on the grounds of generality of application). The definition comes from the Project Management Institute: “The generally accepted knowledge applies to most projects most of the time, and widespread consensus validates its value and effectiveness.”<sup>4</sup>

Specialized Practices used only for certain types of software	Generally Accepted Established traditional practices recommended by many organizations
	Advanced and Research Innovative practices tested and used only by some organizations and concepts still being developed and tested in research organizations

**Figure 1** Categories of knowledge

However, the term “generally accepted” does not imply that the designated knowledge should be uniformly applied to all software engineering endeavors—each project’s needs determine that—but it does imply that competent, capable software engineers should be equipped with this knowledge for potential application. More precisely, generally accepted knowledge should be included in the study material for the software engineering licensing examination that graduates would take after gaining four years of work experience. Although this criterion is specific to the US style of education and does not necessarily apply to other countries, we deem it useful. However, the two definitions of generally accepted knowledge should be seen as complementary.

**LIMITATIONS RELATED TO THE BOOK FORMAT**

The book format for which this edition was conceived has its limitations. The nature of the contents would be better served using a hypertext structure, where a topic would be linked to topics other than the ones immediately preceding and following it in a list.

Some boundaries between KAs, subareas, and so on are also sometimes relatively arbitrary. These boundaries are not to be

<sup>4</sup> A Guide to the Project Management Body of Knowledge, 2000 ed., Project Management Institute, www.pmi.org.

given too much importance. As much as possible, pointers and links have been given in the text where relevant and useful.

Links between KAs are not of the input-output type. The KAs are meant to be views on the knowledge one should possess in software engineering with respect to the KA in question. The decomposition of the discipline within KAs and the order in which the KAs are presented are not to be assimilated with any particular method or model. The methods are described in the appropriate KA in the Guide, and the Guide itself is not one of them.

**THE KNOWLEDGE AREAS**

Figure 1 maps out the eleven chapters and the important topics incorporated within them. The first five KAs are presented in traditional waterfall life-cycle sequence. However, this does not imply that the Guide adopts or encourages the waterfall model, or any other model. The subsequent KAs are presented in alphabetical order, and those of the related disciplines are presented in the last chapter. This is identical to the sequence in which they are presented in this Guide.

**STRUCTURE OF THE KA DESCRIPTIONS**

The KA descriptions are structured as follows.

In the introduction, a brief definition of the KA and an overview of its scope and of its relationship with other KAs are presented.

The breakdown of topics constitutes the core of each KA description, describing the decomposition of the KA into subareas, topics, and sub-topics. For each topic or sub-topic, a short description is given, along with one or more references.

The reference material was chosen because it is considered to constitute the best presentation of the knowledge relative to the topic, taking into account the limitations imposed on the choice of references (see above). A matrix links the topics to the reference material.

The last part of the KA description is the list of recommended references. Appendix A of each KA includes suggestions for further reading for those users who wish to learn more about the KA topics. Appendix B presents the list of standards most relevant to the KA. Note that citations enclosed in square brackets “[ ]” in the text identify recommended references, while those enclosed in parentheses “( )” identify the usual references used to write or justify the text. The former are to be found in the corresponding section of the KA and the latter in Appendix A of the KA.

Brief summaries of the KA descriptions and appendices are given next.

**SOFTWARE REQUIREMENTS KA (SEE FIGURE 2, COLUMN A)**

A requirement is defined as a property that must be exhibited in order to solve some real-world problem.

The first knowledge subarea is *Software Requirements Fundamentals*. It includes definitions of software requirements

themselves, but also of the major types of requirements: product vs. process, functional vs. nonfunctional, emergent properties. The subarea also describes the importance of quantifiable requirements and distinguishes between systems and software requirements.

The second knowledge subarea is the *Requirements Process*, which introduces the process itself, orienting the remaining five subareas and showing how requirements engineering dovetails with the other software engineering processes. It describes process models, process actors, process support and management, and process quality and improvement.

The third subarea is *Requirements Elicitation*, which is concerned with where software requirements come from and how the software engineer can collect them. It includes requirement sources and elicitation techniques.

The fourth subarea, *Requirements Analysis*, is concerned with the process of analyzing requirements to

- ◆ Detect and resolve conflicts between requirements
- ◆ Discover the bounds of the software and how it must interact with its environment
- ◆ Elaborate system requirements to software requirements

Requirements analysis includes requirements classification, conceptual modeling, architectural design and requirements allocation, and requirements negotiation.

The fifth subarea is *Requirements Specification*. Requirements specification typically refers to the production of a document, or its electronic equivalent, that can be systematically reviewed, evaluated, and approved. For complex systems, particularly those involving substantial non-software components, as many as three different types of documents are produced: system definition, system requirements specification, and software requirements specification. The subarea describes all three documents and the underlying activities.

The sixth subarea is *Requirements Validation*, the aim of which is to pick up any problems before resources are committed to addressing the requirements. Requirements validation is concerned with the process of examining the requirements documents to ensure that they are defining the right system (that is, the system that the user expects). It is subdivided into descriptions of the conduct of requirements reviews, prototyping, and model validation and acceptance tests.

The seventh and last subarea is *Practical Considerations*. It describes topics which need to be understood in practice. The first topic is the iterative nature of the requirements process. The next three topics are fundamentally about change management and the maintenance of requirements in a state which accurately mirrors the software to be built, or that has already been built. It includes change management, requirements attributes, and requirements tracing. The final topic is requirements measurement.

## SOFTWARE DESIGN KA (SEE FIGURE 2, COLUMN B)

According to the IEEE definition [IEEE 610.12-90], design is both “the process of defining the architecture, components, interfaces, and other characteristics of a system or component” and “the result of [that] process.” The KA is divided into six subareas.

The first subarea presents *Software Design Fundamentals*, which form an underlying basis to the understanding of the role and scope of software design. These are general software concepts, the context of software design, the software design process, and the enabling techniques for software design.

The second subarea groups together the *Key Issues in Software Design*. They include concurrency, control and handling of events, distribution of components, error and exception handling and fault tolerance, interaction and presentation, and data persistence.

The third subarea is *Software Structure and Architecture*, the topics of which are architectural structures and viewpoints, architectural styles, design patterns, and, finally, families of programs and frameworks.

The fourth subarea describes *software Design Quality Analysis and Evaluation*. While there is a entire KA devoted to software quality, this subarea presents the topics specifically related to software design. These aspects are quality attributes, quality analysis, and evaluation techniques and measures.

The fifth subarea is *Software Design Notations*, which are divided into structural and behavioral descriptions.

The last subarea describes *Software Design Strategies and Methods*. First, general strategies are described, followed by function-oriented design methods, object-oriented design methods, data-structure-centered design, component-based design, and others.

## SOFTWARE CONSTRUCTION KA (SEE FIGURE 2, COLUMN C)

Software construction refers to the detailed creation of working, meaningful software through a combination of coding, verification, unit testing, integration testing, and debugging. The KA includes three subareas.

The first subarea is *Software Construction Fundamentals*. The first three topics are basic principles of construction: minimizing complexity, anticipating change, and constructing for verification. The last topic discusses standards for construction.

The second subarea describes *Managing Construction*. The topics are construction models, construction planning, and construction measurement.

The third subarea covers *Practical Considerations*. The topics are construction design, construction languages, coding, construction testing, reuse, construction quality, and integration.

## SOFTWARE TESTING (SEE FIGURE 2, COLUMN D)

Software Testing consists of the dynamic verification of the

behavior of a program on a finite set of test cases, suitably selected from the usually infinite executions domain, against the expected behavior. It includes five subareas.

It begins with a description of *Software Testing Fundamentals*. First, the testing-related terminology is presented, then key issues of testing are described, and finally the relationship of testing to other activities is covered.

The second subarea is *Test Levels*. These are divided between the targets of the tests and the objectives of the tests.

The third subarea is *Test Techniques*. The first category includes the tests based on the tester's intuition and experience. A second group comprises specification-based techniques, followed by code-based techniques, fault-based techniques, usage-based techniques, and techniques relative to the nature of the application. A discussion of how to select and combine the appropriate techniques is also presented.

The fourth subarea covers *Test-Related Measures*. The measures are grouped into those related to the evaluation of the program under test and the evaluation of the tests performed.

The last subarea describes the *Test Process* and includes practical considerations and the test activities.

#### **SOFTWARE MAINTENANCE (SEE FIGURE 2, COLUMN E)**

Once in operation, anomalies are uncovered, operating environments change, and new user requirements surface. The maintenance phase of the life cycle commences upon delivery, but maintenance activities occur much earlier. The Software Maintenance KA is divided into four subareas.

The first one presents *Software Maintenance Fundamentals*: definitions and terminology, the nature of maintenance, the need for maintenance, the majority of maintenance costs, the evolution of software, and the categories of maintenance.

The second subarea groups together the *Key Issues in Software Maintenance*. These are the technical issues, the management issues, maintenance cost estimation, and software maintenance measurement.

The third subarea describes the *Maintenance Process*. The topics here are the maintenance processes and maintenance activities.

*Techniques for Maintenance* constitute the fourth subarea. These include program comprehension, re-engineering, and reverse engineering.

#### **SOFTWARE CONFIGURATION MANAGEMENT (SEE FIGURE 3, COLUMN F)**

Software Configuration Management (SCM) is the discipline of identifying the configuration of software at distinct points in time for the purpose of systematically controlling changes to the configuration and of maintaining the integrity and traceability of the configuration throughout the system life cycle. This KA includes six subareas.

The first subarea is *Management of the SCM Process*. It

covers the topics of the organizational context for SCM, constraints and guidance for SCM, planning for SCM, the SCM plan itself, and surveillance of SCM.

The second subarea is *Software Configuration Identification*, which identifies items to be controlled, establishes identification schemes for the items and their versions, and establishes the tools and techniques to be used in acquiring and managing controlled items. The first topics in this subarea are identification of the items to be controlled and the software library.

The third subarea is *Software Configuration Control*, which is the management of changes during the software life cycle. The topics are: first, requesting, evaluating, and approving software changes; second, implementing software changes; and third, deviations and waivers.

The fourth subarea is *Software Configuration Status Accounting*. Its topics are software configuration status information and software configuration status reporting.

The fifth subarea is *Software Configuration Auditing*. It consists of software functional configuration auditing, software physical configuration auditing, and in-process audits of a software baseline.

The last subarea is *Software Release Management and Delivery*, covering software building and software release management.

#### **SOFTWARE ENGINEERING MANAGEMENT (SEE FIGURE 3, COLUMN G)**

The Software Engineering Management KA addresses the management and measurement of software engineering. While measurement is an important aspect of all KAs, it is here that the topic of measurement programs is presented. There are six subareas for software engineering management. The first five cover software project management and the sixth describes software measurement programs.

The first subarea is *Initiation and Scope Definition*, which comprises determination and negotiation of requirements, feasibility analysis, and process for the review and revision of requirements.

The second subarea is *Software Project Planning* and includes process planning, determining deliverables, effort, schedule and cost estimation, resource allocation, risk management, quality management, and plan management.

The third subarea is *Software Project Enactment*. The topics here are implementation of plans, supplier contract management, implementation of measurement process, monitor process, control process, and reporting.

The fourth subarea is *Review and Evaluation*, which includes the topics of determining satisfaction of requirements and reviewing and evaluating performance.

The fifth subarea describes *Closure*: determining closure and closure activities.

Finally, the sixth subarea describes *Software Engineering*

*Measurement*, more specifically, measurement programs. Product and process measures are described in the Software Engineering Process KA. Many of the other KAs also describe measures specific to their KA. The topics of this subarea include establishing and sustaining measurement commitment, planning the measurement process, performing the measurement process, and evaluating measurement.

**SOFTWARE ENGINEERING PROCESS (SEE FIGURE 3, COLUMN H)**

The Software Engineering Process KA is concerned with the definition, implementation, assessment, measurement, management, change, and improvement of the software engineering process itself. It is divided into four subareas.

The first subarea presents *Process Implementation and Change*. The topics here are process infrastructure, the software process management cycle, models for process implementation and change, and practical considerations.

The second subarea deals with *Process Definition*. It includes the topics of software life cycle models, software life cycle processes, notations for process definitions, process adaptation, and automation.

The third subarea is *Process Assessment*. The topics here include process assessment models and process assessment methods.

The fourth subarea describes *Process and Product Measurements*. The software engineering process covers general product measurement, as well as process measurement in general. Measurements specific to KAs are described in the relevant KA. The topics are process measurement, software product measurement, quality of measurement results, software information models, and process measurement techniques.

**SOFTWARE ENGINEERING TOOLS AND METHODS (SEE FIGURE 3, COLUMN I)**

The Software Engineering Tools and Methods KA includes both software engineering tools and software engineering methods.

The *Software Engineering Tools* subarea uses the same structure as the Guide itself, with one topic for each of the other nine software engineering KAs. An additional topic is provided: miscellaneous tools issues, such as tool integration techniques, which are potentially applicable to all classes of tools.

The *Software Engineering Methods* subarea is divided into four subsections: heuristic methods dealing with informal approaches, formal methods dealing with mathematically based approaches, and prototyping methods dealing with software development approaches based on various forms of prototyping.

**SOFTWARE QUALITY (SEE FIGURE 3, COLUMN J)**

The Software Quality KA deals with software quality considerations which transcend the software life cycle processes. Since software quality is a ubiquitous concern in software engineering, it is also considered in many of the other KAs, and the reader will notice pointers to those KAs throughout this KA. The description of this KA covers three subareas.

The first subarea describes the *Software Quality Fundamentals* such as software engineering culture and ethics, the value and costs of quality, models and quality characteristics, and quality improvement.

The second subarea covers *Software Quality Management Processes*. The topics here are software quality assurance, verification and validation, and reviews and audits.

The third and final subarea describes *Practical Considerations* related to software quality. The topics are software quality requirements, defect characterization, software quality management techniques, and software quality measurement.

**RELATED DISCIPLINES OF SOFTWARE ENGINEERING (SEE FIGURE 3, COLUMN K)**

The last chapter is entitled *Related Disciplines of Software Engineering*. In order to circumscribe software engineering, it is necessary to identify the disciplines with which software engineering shares a common boundary. This chapter identifies, in alphabetical order, these related disciplines. For each related discipline, and using a consensus-based recognized source as found, are identified:

- ◆ an informative definition (when feasible);
- ◆ a list of KAs.

The related disciplines are:

◆ Computer engineering	◆ Project management
◆ Computer science	◆ Quality management
◆ Management	◆ Software ergonomics
◆ Mathematics	◆ Systems engineering

**APPENDICES**

**APPENDIX A. KA DESCRIPTION SPECIFICATIONS**

The appendix describes the specifications provided by the editorial team to the associate editors for the content, recommended references, format, and style of the KA descriptions.

#### **APPENDIX B. EVOLUTION OF THE GUIDE**

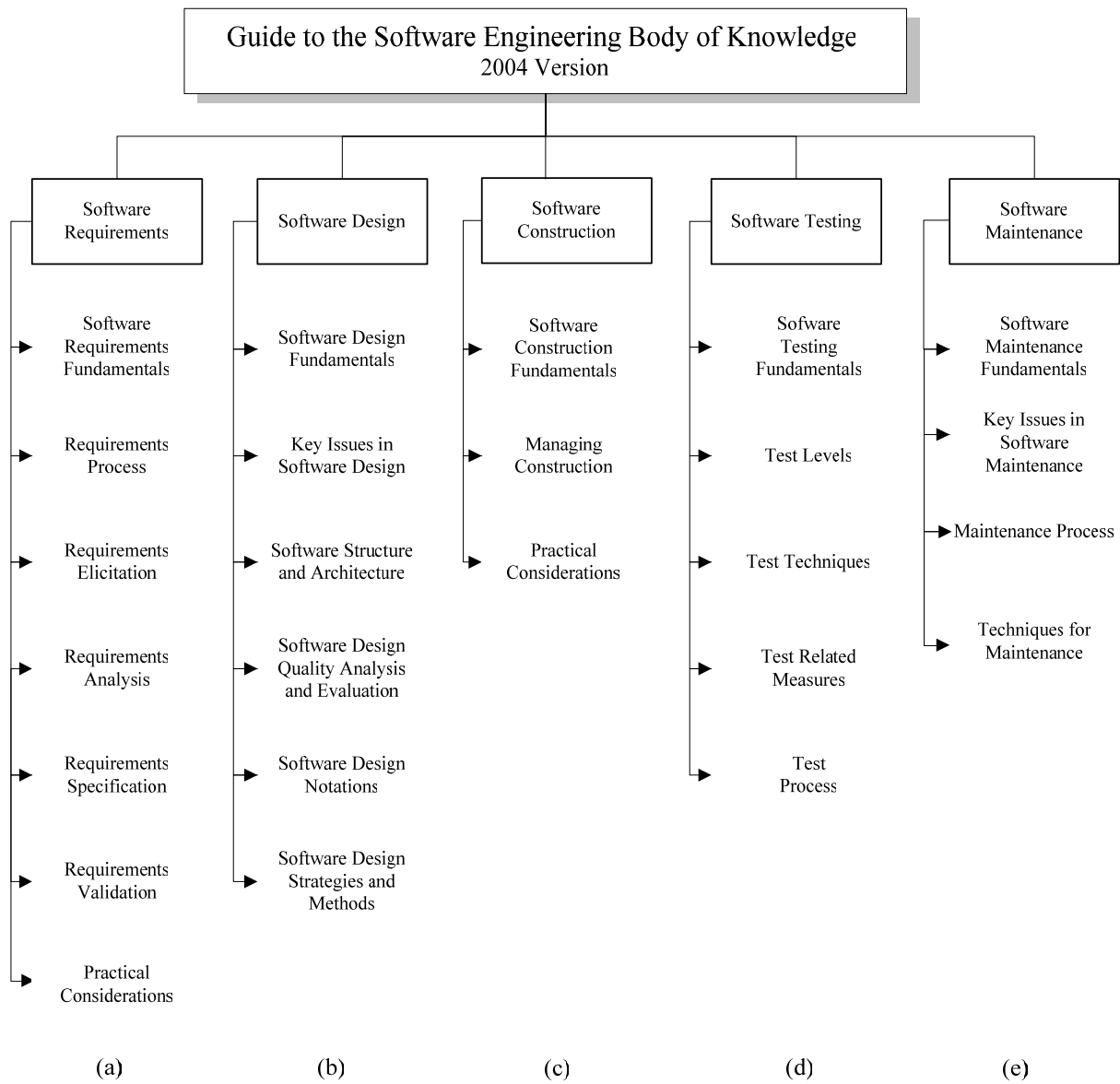
The second appendix describes the project's proposal for the evolution of the Guide. The 2004 Guide is simply the current edition of a guide which will continue evolving to meet the needs of the software engineering community. Planning for evolution is not yet complete, but a tentative outline of the process is provided in this appendix. As of this writing, this process has been endorsed by the project's Industrial Advisory Board and briefed to the Board of Governors of the IEEE Computer Society but is not yet either funded or implemented.

#### **APPENDIX C. ALLOCATION OF STANDARDS TO KAs**

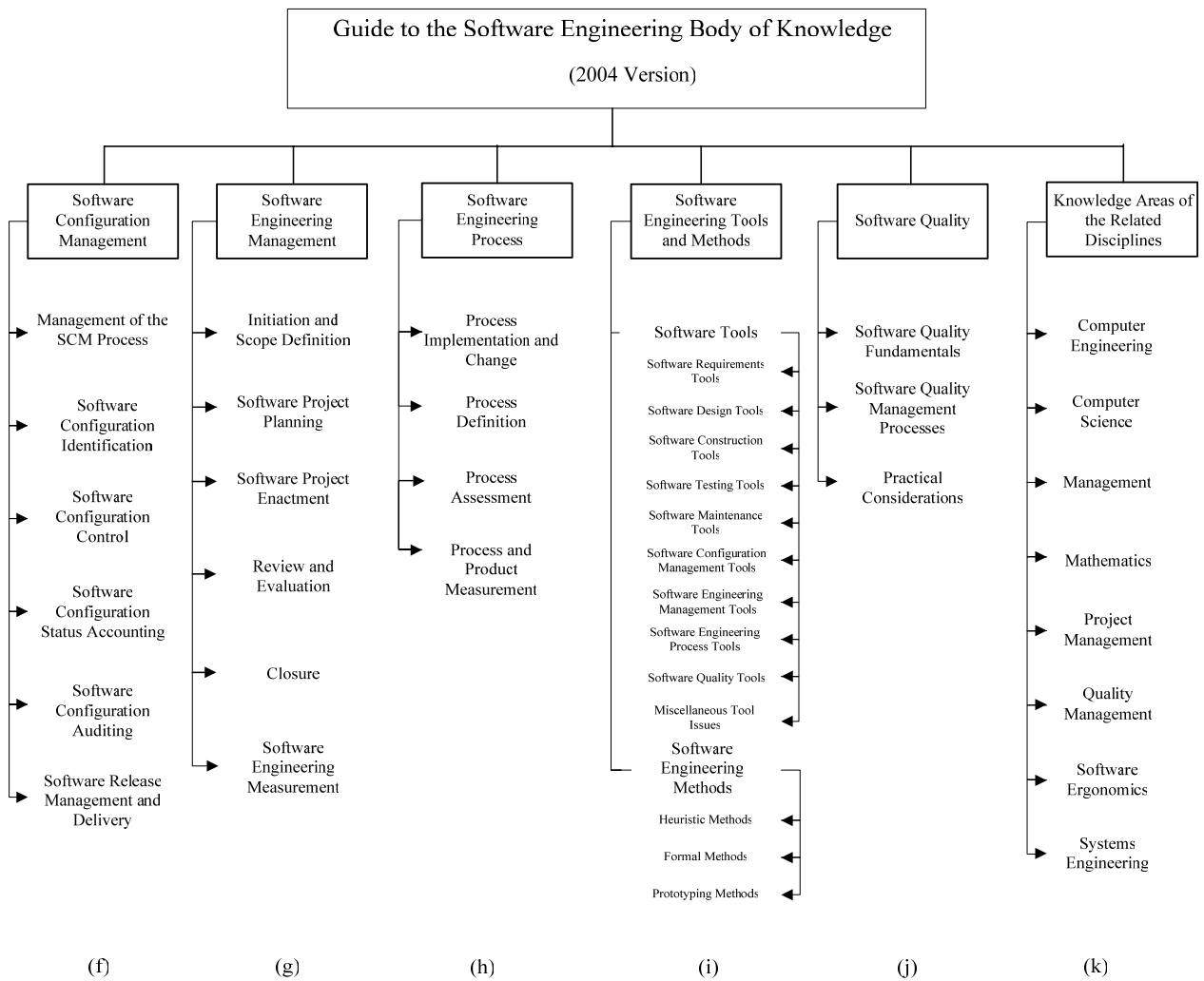
The third appendix is an annotated table of the most relevant standards, mostly from the IEEE and the ISO, allocated to the KAs of the SWEBOK Guide.

#### **APPENDIX D. BLOOM RATINGS**

As an aid, notably to curriculum developers (and other users), in support of the project's fifth objective, the fourth appendix rates each topic with one of a set of pedagogical categories commonly attributed to Benjamin Bloom. The concept is that educational objectives can be classified into six categories representing increasing depth: knowledge, comprehension, application, analysis, synthesis, and evaluation. Results of this exercise for all KAs can be found in Appendix D. This Appendix must not, however, be viewed as a definitive classification, but much more as a starting point.



**Figure 2** First five KAs



**Figure 3** Last six KAs