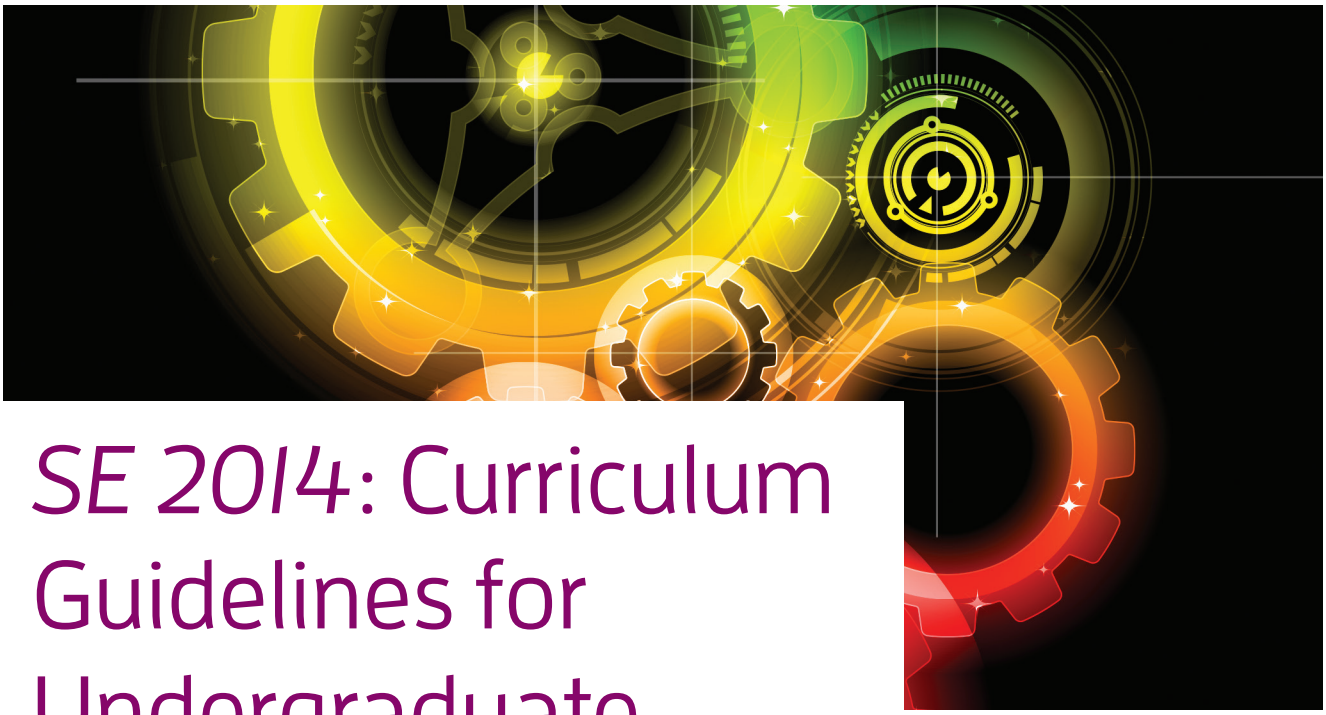


SE 2014: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering

Mark Ardis, Stevens Institute of Technology
David Budgen, Durham University
Gregory W. Hislop, Drexel University
Jeff Offutt, George Mason University
Mark Sebern, Milwaukee School of Engineering
Willem Visser, University of Stellenbosch

Revised curriculum guidelines help university faculty create or update undergraduate software engineering programs.

Over the past decade, undergraduate curriculum guidelines for computer engineering, computer science, information technology, information systems, and software engineering, produced under the aegis of ACM and the IEEE Computer Society, have supported worldwide curriculum development.

ACM and the IEEE Computer Society recently approved a newly revised version, the *Software Engineering 2014 Curriculum Guidelines* (*SE 2014*; http://securriculum.org)—a resource for university faculty seeking to create new undergraduate programs or revise and update existing ones. The guidelines also provide useful information to potential employers about the abilities and knowledge to expect from such programs' graduates.

In this article, we discuss the processes involved and the changes in the newly issued guidelines.

0018-9162/15/$31.00 © 2015 IEEE

EDITOR **CHUCK WALRAD**
Davenport Consulting; cwalrad@daven.com

TASK FORCES

The *SE 2004* guidelines were assembled by an ACM/IEEE Computer Society task force led by Rich LeBlanc and Ann Sobel. However, because software engineering knowledge continues to grow and evolve, the societies created a new task force in 2010 (see "Software Engineering 2014 Curriculum Guidelines Task Forces" sidebar), asking members to consider whether revisions were needed and, if so, how extensive they should be. The group duly sought feedback by running events at major conferences; consulting with individuals; and organizing an online survey, which received 477 completed returns from 42 countries.

In addition to these efforts, the task force looked at activities related to the *Guide to the Software Engineering Body of Knowledge* (*SWEBOK*; www.computer.org/web/swebok), such as the Software Engineering Competency Model (SWECOM; www.computer.org/web/peb/swecom), and related efforts, such as Computer Science Curricula 2013 (CS2013; www.acm.org/education/CS2013-final-report.pdf).

Although survey participants generally agreed on *SE 2004*'s usefulness, their responses indicated a need for a modest revision. Figure 1 summarizes the survey responses to questions about the usefulness of the knowledge units (KUs) forming major elements of the Software Engineering Education Knowledge (SEEK), which summarizes the recommended topics of a curriculum.

A second task force began the revision in late 2012. As with most software evolution projects, the task force underestimated the revision's extent and complexity! The final recommendations emerged in late 2014.

EVOLVING NEEDS

Since the 2004 guidelines' completion, software engineering as a discipline has become more distinct. And, as computing has become more pervasive in daily life, nonprofessionals are more familiar with the term "software engineering." Thus, the revision task force didn't need to explicitly differentiate software engineering from other computing fields and

FROM THE EDITOR

These are some of the IEEE standards that support software engineering practices:

- IEEE Std. 730-2002: IEEE Standard for Software Quality Assurance
- IEEE Std. 828-2012: IEEE Standard for Software Configuration Management
- IEEE Std. 829-2008: IEEE Standard for Software and System Test Documentation
- IEEE Std. 1012-2012: IEEE Standard for System and Software Verification and Validation
- IEEE Std. 1028-2008: IEEE Standard for Software Reviews
- IEEE Std. 1044-2009: IEEE Standard Classification for Software Anomalies
- IEEE Std. 1063-2007: IEEE Standard for Software User Documentation
- IEEE Std. 1062-2002: IEEE Recommended Practice for Software Acquisition
- IEEE Std. 1074-2006: IEEE Standard for Developing a Software Project Life Cycle Process
- IEEE Std. 1228-2010: IEEE Standard for Software Safety Plans
- IEEE Std. 1517-2010: Standard for Information Technology—Software Life Cycle Processes—Reuse Processes
- IEEE Std. 1633-2009: Recommended Practice for Software Reliability
- IEEE Std. 14471-2010: Information Technology—Software Engineering—Guidelines for the Adoption of CASE Tools
- ISO/IEC/IEEE Std. 14764-2006: ISO/IEC/IEEE Standard for Software Engineering—Software Life Cycle Processes—Maintenance (replaces IEEE Std. 1219-1998)
- IEEE Std. 15026-2-2011: System and Software Engineering—System and Software Assurance—Part 2: Assurance Case
- IEEE Std. 15026-1-2011: System and Software Engineering—System and Software Assurance—Part 1: Concepts and Vocabulary
- ISO/IEC/IEEE Std. 15288-2008: System Engineering—System Life Cycle Processes
- IEEE Std. 16326-2009: Standard for Software Engineering—Project Management
- IEEE Std. 16085-2006: Standard for System and Software Life Cycle Processes—Risk Management (previously IEEE Std. 1540-2001)
- IEEE Std. 24765-2010: Systems and Software Engineering—Vocabulary (SEVOCAB)
- ISO/IEC/IEEE Std. 29148-2011: IEEE Standard for Software and System—Life Cycle Processes—Requirements Engineering
- ISO/IEC/IEEE Std. 42010-2011: IEEE Standard for Architectural Descriptions
- IEEE Std. 90003-2008: Software and Systems Engineering—Guidelines for the Application of ISO 9001:2000 to Computer Software

—Chuck Walrad

SOFTWARE ENGINEERING 2014 CURRICULUM GUIDELINES TASK FORCES

Task force I, which determined the need for and extent of the *Software Engineering 2004 Curriculum Guidelines*' revision, consisted of Mark Ardis (chair), Jo Atlee, David Budgen, Gregory Hislop, Renee McCauley, and Mark Sebern.

Task force 2, which produced the revised guidelines, consisted of Mark Ardis (chair), David Budgen, Gregory Hislop, Jeff Offutt, Mark Sebern, and Willem Visser.

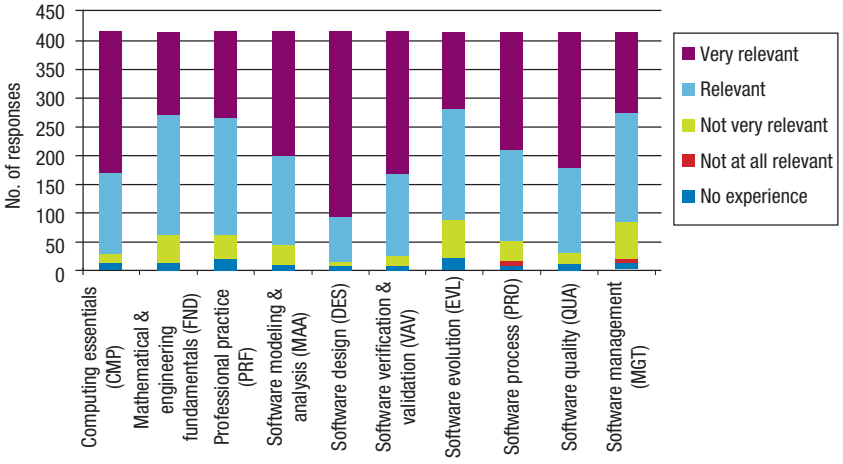
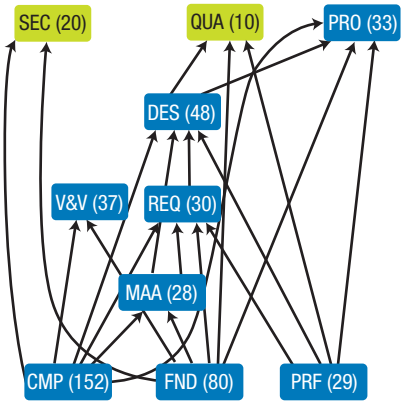


Figure 1. Survey feedback on the usefulness of the Software Engineering 2004 Guidelines' Software Engineering Education Knowledge (SEEK) elements.



Module key:
CMP—Computing essentials
FND—Mathematical & engineering fundamentals
PRF—Professional practice
MAA—Software modeling & analysis
REQ—Requirements analysis & specification
DES—Software design
VAV—Software verification & validation
PRO—Software process
QUA—Software quality
SEC—Security

Figures in brackets indicate a topic's presentation hours (or lecture hours).

Figure 2. SEEK's knowledge unit sizes, interdependencies, and suggested teaching times.

instead focused its efforts on relating the curriculum to a much wider and richer set of disciplines.

In addition, there has been a gradual shift to encompass different development processes, from traditional "waterfall" to lighter-touch agile forms. This in turn required the task force to structure various elements of the guidelines in a process-neutral form.

Finally, owing partly to these factors, the team needed to restructure SEEK and its KUs. Feedback indicated a need for a new cross-cutting security KU (to reflect the view that KUs aren't meant to be templates for modules). The software quality KU was similarly restructured to have a cross-cutting role. Note that SWECOM treats security and software quality as cross-cutting topics, and CS2013 recently added security as a separate topic.

The team restructured other KUs as well, separating requirements analysis and specification from modeling and analysis and merging three smaller process-oriented KUs (software process, software evolution, and software management) into a single software process KU. Figure 2 shows the SEEK elements' interdependencies and each topic's suggested teaching time. Chapter 4 of *SE 2004*, addressing SEEK, further breaks down each KU into knowledge areas (KAs) and indicates an appropriate depth of coverage to adopt in teaching each KA's material.

Individual *SE 2004* chapters provide extensively revised and extended material to assist faculty in different ways, such as preparing a case for (chapter 2) and designing (chapters 5 and 6) a new program, and appendices offer detailed example curriculum modules.

The team ensured that the *SE 2004* guidelines maintained the key philosophy that underpinned *SE 2004*. The guidelines and material are meant to be a configurable knowledge structure about important matters in software

engineering and a flexible template for organizing delivery of this knowledge. Rather than an "off-the-shelf" model, it is usable across different national and institutional structures and can be adapted to fit local needs and other requirements.

Please see http://securriculum.org to obtain a copy of *SE 2004*.

MARK ARDIS is a distinguished service professor at the Stevens Institute of Technology. Contact him at mark.ardis@stevens.edu.

DAVID BUDGEN is a professor at Durham University. Contact him at david.budgen@durham.ac.uk.

GREGORY W. HISLOP is a professor at Drexel University. Contact him at hislop@drexel.edu.

JEFF OFFUTT is a professor at George Mason University. Contact him at offutt@gmu.edu.

MARK SEBERN is a professor emeritus of software engineering at the Milwaukee School of Engineering. Contact him at sebern@msoe.edu.

WILLEM VISSER is a professor at the University of Stellenbosch. Contact him at willem@gmail.com.