Editor: Forrest Shull ■ Fraunhofer Center for Experimental Software Engineering, Maryland ■ fshull@fc-md.umd.edu

Creating Software Process Capability/Maturity Models

Christiane Gresse von Wangenheim, Jean C.R. Hauck, Alessandra Zoucas, Clenio F. Salviano, Fergal McCaffery, and Forrest Shull

A seeming multitude of software process capability/maturity models (SPCMMs) have emerged, and many software engineers have had to worry about compliance with them at one time or another. Although using SPCMMs is a well-established practice, the ways they're used can vary widely. At best, they can pull together vast bodies of knowledge about good software practices—the hard-won expertise of many engineers—into a form that's easier to work with. At worst, they're misused as "processes for process' sake," in which conforming to the model stifles opportunities for innovation and tailoring.

If software engineers had better knowledge about how SPCMMs are developed and the basis of their recommendations, they might be able to interpret and use them to optimize their benefits. We therefore studied these issues in a systematic literature review and follow-on questionnaire.

For this article, we define SPCMMs as models that describe best practices for software life-cycle processes, based on good engineering and process-management principles, and process-attribute sets for capability/maturity design aspects.[1] Examples include the Capability Maturity Model Integration for Development (CMMI-DEV) and the ISO/IEC 15504-5 exemplar Process Assessment Model for the ISO/IEC 15504 Software Process Improvement and Capability Determination (Spice) standard. These models are available for evaluating and comparing process improvements or assessments, based on the assumption that higher process capability or organizational maturity are associated with better performance.

SPCMM Trends

The software process improvement community is still actively developing SPCMMs. Existing models are evolving, creating new versions of generic models. However, tailoring generic models isn't an easy process,[2] so several initiatives are underway to develop domain-specific models. Examples include Spice4Space[3] and AutomotiveSpice (www.automotivespice.com). In addition, harmonization initiatives, such as EnterpriseSpice (www.enterprisespice.com) aim to integrate several existing models within a specific context.

Despite these initiatives, little information is available on how to develop SPCMMs that are theoretically sound, rigorously tested, and widely accepted.[4,5] Most models are based on practices or success factors derived from projects that demonstrated good results within an organization or industry, but they lack a basis in theory.[5] Few of the models have been evaluated in terms of validity, reliability, and generalizability, which is one reason for ambiguous SPCMM results in practice.[6] We wanted to clarify the bases for best practices to help overcome a main criticism of SPCMMs. We also wanted to illuminate whether the process used to define them contributed to the creation of valid, reliable models.

Summarizing the Literature

Our first step was to examine the existing literature to understand how SPCMMs are created.

The study team examined all published peer-reviewed English-language articles on SPCMMs that were available on the Web (via digital librar-

 0740-7459/10/$26.00

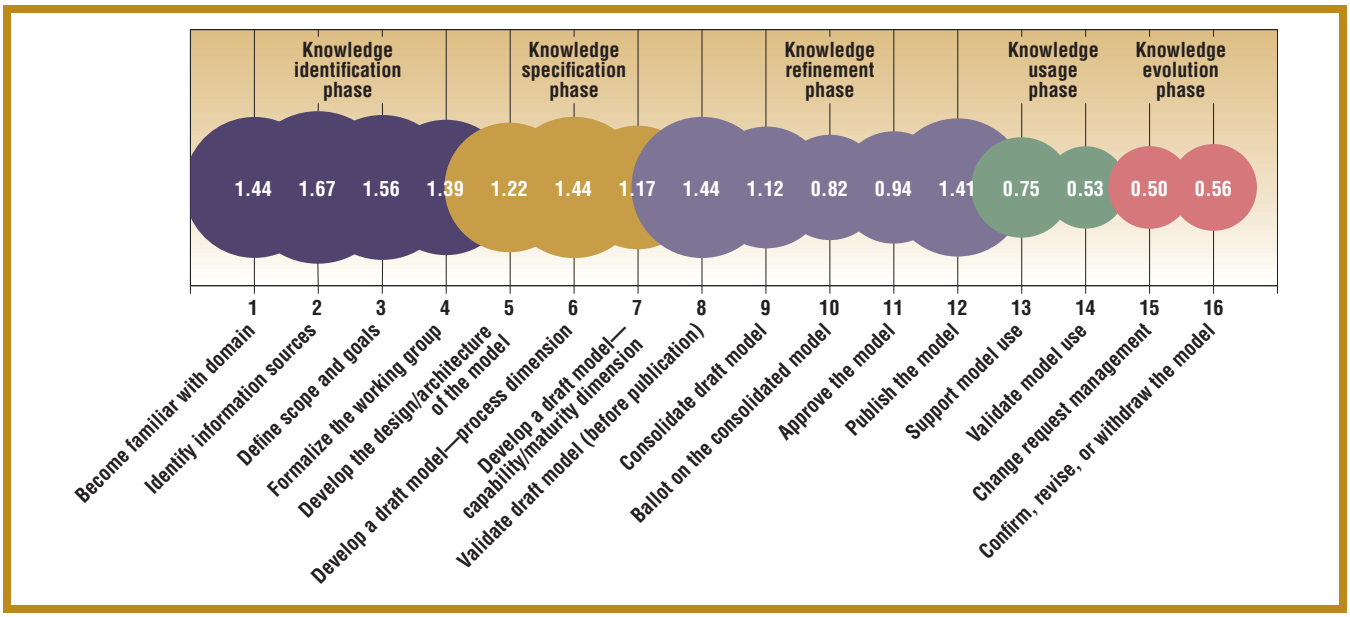

Figure 1. Degree of execution of our reference steps in practice. The bubble sizes show a stronger presence of steps related to knowledge identification, specification, and refinement phases than to knowledge usage and evolution phases.

ies and databases) and published between January 1990 and April 2009. The search returned 1,477 papers, of which the team considered 61 publications relevant to the research focus. (We describe the literature search in more detail in the Web appendix to this article at www.computer.org/software/webextra.)

A review of these publications identified 52 SPCMMs. Besides the evolution of new versions of existing models, the customization of models to specific domains was a clear trend, including customizations for small and medium enterprises, testing and quality assurance, security engineering, extreme programming, and requirements. Model developers have based most of these customizations on one or two existing SPCMMs. The most often-used source models were the CMM (used for 60 percent of the SPCMMs found), the ISO/IEC 15504 standard and its exemplar model (37 percent), and the CMMI framework through its most popular model, CMMI-DEV (21 percent).

The models were developed in diverse ways. Two were defined as standards, following a high-level ISO process. However, in general, we found little information on development processes. Only 21 percent of the papers presented detailed information, 27 percent presented some information, and 52 percent provided no substantial information on the model development.

Contacting SPCMM Developers

Because the literature review yielded little information about how SPCMMs are developed, we conducted a survey in 2009 to obtain more details. We invited the authors of 60 SPCMMs (the 52 found in our literature search plus authors of unpublished SPCMMs that we were aware of) via email and received 18 responses, representing a return rate of 30 percent. The questionnaire included questions to characterize the SPCMM and its developers. (The Web appendix includes a bibliography and other details for the 60 models as well as a copy of the questionnaire.)

We also proposed a reference model for how to develop SPCMMs. We defined five phases and 17 steps to unify those in the ISO stages for developing international standards; the Pro2PI improvement methodology,[7] which is driven by process capability profiles; the framework for process maturity model development proposed by Tonia de Bruin and her colleagues;[6] our own experiences;[8–11] and a general knowledge extraction process.[12] We asked questionnaire respondents to map their process to this reference model.

The responses showed that most SPCMMs were developed in an ad hoc manner. The exceptions were the two models being developed as ISO standards and four models that were developed according to a home-grown methodology: Adept,[13] RequirementCMM,[14] the Software Maintenance Maturity Model (S3M, www.s3m.ca), and iCMM (integrated CMM).[15]

To obtain a more detailed understanding of model development, we analyzed the responses and classified the degree to which each of our reference steps had been executed:

- 2 = executed systematically (using techniques such as literature reviews, surveys, structured interviews, and expert panels);
- 1 = executed but apparently without using a systematic technique; and
- 0 = not executed or not yet executed (the model is still under development).

Figure 1 illustrates the results as a bubble diagram over our reference model's five phases and 16 steps. The bubble size indicates the average degree of execution of each step.

Most models addressed all the steps, at least at an informal level, except in the knowledge-usage and knowledge-evolution phases. The steps in these two places might be missing because not all models have yet seen much use in practice and hence have little feedback to incorporate. Also, we note that models developed as part of a master's or PhD thesis often omit knowledge refinement steps.

For users to have confidence in applying an SPCMM, the knowledge-specification steps are crucial because they capture domain-specific requirements and best practices as a basis for the model. We found the majority of SPCMMs were based on experiences from domain experts and practitioners, using techniques such as brainstorming, focus groups, or interviews. A few models used different techniques, such as GQM (goal, question, metric), to explicitly relate model elements to quality and performance goals. A few others created models based on the abstraction of empirical data, in which the model developers elicited information from the literature or from critical-success-factor interviews and analyzed it by content and frequency to target the model on practical problems.

A critical problem we observed is that the draft model is seldom validated systematically before publication. When the model is validated, it's usually through an expert review with varying degrees of participation. Yet, again with the exception of standards, a defined procedure for achieving consensus among the participants is typically lacking. In rare cases, the developers perform pilot studies to validate the draft model. Studies that report SPCMM effects on the intended quality and performance goals are absent in most cases.

Our research demonstrates the variety of SPCMMs being developed and customized. However, with few exceptions, the work seems to lack methodological support, indicating an area that's still maturing. Among the principal issues is that SPCMM elements aren't explicitly and systematically related to quality and performance goals. We also note a need for methodological support to validate models. This requires a better understanding of the processes used to create the models, which in turn will provide a basis to systematically develop SPCMMs that truly represent the best practices within a specific domain.

Acknowledgments

We thank all participants of the survey for their valuable contributions. CNPq and CAPES, entities of the Brazilian government, supported this work. We also received support in part from Science Foundation Ireland.

References

1. C.F. Salviano and A. Figueiredo, "Unified Basic Concepts for Process Capability Models," *Proc. Int'l Conf. Software Eng. and Knowledge Eng.*, Knowledge Systems Inst., 2008, pp. 173–178.
2. S. Magee and D. Thiele, "Engineering Process Standards: State of the Art and Challenges," *IT Professional*, vol. 6, no. 5, 2004, pp. 38–44.
3. A. Cass et al., "SPICE for SPACE Trials, Risk Analysis, and Process Improvement," *Software Process: Improvement and Practice*, vol. 9, no. 1, 2004, pp. 13–21.
4. T. Mettler, *A Design Science Research Perspective on Maturity Models in Information Systems*, tech. report BE IWI/HNE/03, Universität St. Gallen, 2009; http://eprints.qut.edu.au/cgi/export/25152/DC/quteprints-eprint-25152.txt.
5. S. Matook and M. Indulska, "Improving the Quality of Process Reference Models: A Quality Function Deployment-Based Approach," *Decision Support Systems*, vol. 47, 2009, pp. 60–71.
6. T. Bruin et al., "Understanding the Main Phases of Developing a Maturity Assessment Model," *Proc. 16th Australasian Conf. Information Systems*, Assoc. Information Systems Electronic Library, 2005, http://aisel.aisnet.org/acis2005/109.
7. C.F. Salviano et al., "A Method Framework for Engineering Process Capability Models," *Proc. 16th Conf. European Systems and Software Process Improvement and Innovation*, Publizon, 2009, pp. 6.25–6.36.
8. F. Mc Caffery and G. Coleman, "Developing a Configuration Management Capability Model for the Medical Device Industry," *Int'l J. Information Systems and Change Management*, vol. 2, no. 2, 2007, pp. 139–154.
9. F. McCaffery et al., "Ahaa—Agile, Hybrid Assessment Method for Automotive, Safety Critical SMEs," *Proc. 30th Int'l Conf. Software Engineering*, ACM Press, 2008, pp. 551-560.
10. C. Gresse von Wangenheim et al., "Helping Small Companies Assess Software Processes," *IEEE Software*, vol. 23, no. 1, 2006, pp. 91–98.
11. M.H. Cancian et al., "Discovering Software Process and Product Quality Criteria in Software as a Service," to be published in *Proc. 11th Int'l Conf. Product Focused Software Development and Process Improvement*, Springer, 2010.
12. G. Schreiber et al., *Knowledge Engineering and Management: The CommonKADS Methodology*, MIT Press, 1999.
13. F. McCaffery, P.S. Taylor, and G. Coleman, "Adept: A Unified Assessment Method for Small Software Companies," *IEEE Software*, vol. 24, no. 1, 2007, pp. 24–31.
14. S. Beecham et al., "Defining a Requirements Process Improvement Model," *Software Quality J.*, vol. 13, no. 3, 1997, pp. 247–279.
15. L. Ibrahim and A. Pyster, "A Single Model for Process Improvement," *IT Professional*, vol. 6, no. 3, 2004, pp. 43–49.

IEEE Software (ISSN 0740-7459) is published bimonthly by the IEEE Computer Society. IEEE headquarters: Three Park Ave., 17th Floor, New York, NY 10016-5997. IEEE Computer Society Publications Office: 10662 Los Vaqueros Cir., PO Box 3014, Los Alamitos, CA 90720-1314; +1 714 821 8380; fax +1 714 821 4010. IEEE Computer Society headquarters: 2001 L St., Ste. 700, Washington, DC 20036. Subscription rates: IEEE Computer Society members get the lowest rate of US$52 per year, which includes printed issues plus online access to all issues published since 1984. Go to www.computer.org/subscribe to order and for more information on other subscription prices. Back issues: $20 for members, $163 for nonmembers (plus shipping and handling).

Postmaster: Send undelivered copies and address changes to *IEEE Software*, Membership Processing Dept., IEEE Service Center, 445 Hoes Lane, Piscataway, NJ 08854-4141. Periodicals Postage Paid at New York, NY, and at additional mailing offices. Canadian GST #125634188. Canada Post Publications Mail Agreement Number 40013885. Return undeliverable Canadian addresses to PO Box 122, Niagara Falls, ON L2E 6S8, Canada. Printed in the USA.

Christiane Gresse von Wangenheim is a professor at the Federal University of Santa Catarina (UFSC) and coordinator of the UFSC Software Quality Group at the National Institute for Research and Technology for Digital Convergence. Contact her at gresse@incod.ufsc.br.

Jean Carlo R. Hauck is a visiting researcher at Dundalk Institute of Technology, a member of the Software Quality Group at the National Institute for Research and Technology for Digital Convergence, and a PhD student in knowledge engineering at the Federal University of Santa Catarina. Contact him at jeanhauck@incod.ufsc.br.

Alessandra Zoucas is a professor and master student at the University of the Valley of Itajaí and a professor at the Federal Institute of Santa Catarina. Contact her at alessandra-zoucas@ifsc.edu.br.

Clenio F. Salviano is a researcher in process improvement R&D at CTI (Centro de Tecnologia da Informação) Renato Archer. Contact him at clenio.salviano@cti.gov.br.

Fergal McCaffery is leader of the Regulated Software Research Group at Dundalk Institute of Technology (DkIT) and holds a Science Foundation Ireland Stokes Lecturership in DkIT's Department of Computing and Mathematics. Contact him at fergal.mccaffery@dkit.ie.

Forrest Shull is a senior scientist at the Fraunhofer Center for Experimental Software Engineering, Maryland, and director of its Measurement and Knowledge Management Division. Contact him at fshull@fc-md.umd.edu.