An Overview of the SWEBOK Guide

Lead Authors: *Hironori Washizaki, Maria-Isabel Sanchez-Segura, Juan Garbajosa, Steve Tockey, Kenneth E Nidiffer, and Annette D. Reilly*

Software is everywhere. Software and systems engineers are leading the way for the digital transformation of social, economic, military, and environmental systems at an increasing rate of change. Advanced software-enabled capabilities allow software and systems engineers to address and anticipate emerging and complex challenges. Most systems of any complexity today are modeled and controlled by software; hence, the term “software intensive systems” is commonly used to describe them. At the end of 2022, the IEEE Computer Society (IEEE-CS) released beta version 4 of the Software Engineering Body of Knowledge (SWEBOK) (IEEE Computer Society 2022). When the final version 4.0 is published, it will replace version 3 which appeared in 2014 (Bourque and Fairley 2014). As have previous versions, SWEBOK 4.0 will help guide both systems engineering and software engineering professionals to long-established software life cycle processes and new insights for dealing with the demands of the digital era. This article uses SWEBOK 4.0 beta to explain how and why the new SWEBOK takes into consideration the needs of digital era software engineers immersed in an industry demanding visionary solutions.

Contents

Introduction

ISO/IEC/IEEE Systems and Software Engineering Vocabulary (SEVOCAB) defines software engineering as "the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software." (ISO/IEC/IEEE n.d.)

It is important to note that systems engineering, and software engineering are distinct disciplines with different but somewhat overlapping core competencies. Due to the inherent characteristics of software intensive systems, these disciplines often approach problem solving from different perspectives. For example, systems engineers apply their problem-solving skills to develop physical, computational, social, and hybrid systems. Depending on the system, those skills could be based on a wide range of disciplines including mathematics, any of the physical and social sciences, law, management, and a myriad of other disciplines. Software engineers tend to apply their problem-solving skills on a narrower set of disciplines, especially discrete mathematics and computer science, to develop computational systems and subsystems. Also, software Is a logical medium. Software components of software-enabled systems are logical constructions expressed in algorithmic form. This contrasts with the physical and social components of systems that are realized through mechanical, electrical, chemical, biological, social, human and other elements. Both disciplines are critical to solving the complex challenges of the future as evidenced by disruptive innovation and increasing complexity in new systems and systems of systems which is causing a convergence among disciplines.

SWEBOK Guide Objectives

The Guide to the Software Engineering Body of Knowledge (SWEBOK), published by the IEEE Computer Society (IEEE CS), represents the current state of generally accepted, consensus-based knowledge

emanating from the interplay between software engineering theory and practice. (Bourke and Fairley 2014, IEEE Computer Society 2022). The objectives include the provision of guidance for learners, researchers, and practitioners to identify and share a mutual understanding of "generally accepted knowledge" in software engineering. This guidance defines the boundary between software engineering and related disciplines, and provides a foundation for certifications and educational curricula.

The Guide should not be confused with the Body of Knowledge itself which exists in published literature. It is similar in this regard to the SEBoK. For example, the Guide addresses knowledge areas (KAs) in terms of "what" activities are being done versus "how" they are accomplished. Bodies Of Knowledge (BOKs) are often a formal way of referring to core competencies, or what should be known to be successful in a practitioner's area of expertise. (Washizaki, et al. 2023) Every profession is based on a body of knowledge, although that knowledge is not always defined in a concise manner. In cases where normality exists, the body of knowledge is "generally recognized" by practitioners and may be codified in a variety of ways for a variety of different uses. But in many cases, a guide to a body of knowledge is formally documented, usually in a form that permits it to be used for such purposes as development and accreditation of academic and training programs, certification of specialists, or professional licensing. (IEEE Computer Society 2022, IEEE Computer Society 2014) The purpose of the Guide is to describe the portion of the Body of Knowledge that is accepted, to organize that portion, and to provide topical access to it. The Guide is also aligned with the following objectives:

1. To promote a consistent view of software engineering worldwide
2. To specify the scope of, and clarify the place 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
4. To provide topical access to the Software Engineering Body of Knowledge
5. To provide a foundation for curriculum development and for individual certification and licensing material.

Overview of the SWEBOK Guide

The Guide represents the current state of accepted, consensus-based knowledge emanating from the interplay between software engineering theory and practice. The origins of the Guide go back to the late 1990s. Much like the software engineering discipline, the Guide has continued to evolve over the last 20 years to reflect the educational, industrial, social, technical, and technological changes in society. Version 4 of the Guide will be released in 2023 to improve its currency, readability, consistency, and usability.

Table 1 provides the high-level table of contents from Guide V4. Note that sections with an "*" are new (were not previously included in Guide 3). The current draft Guide V4 contains 18 knowledge areas (KAs), followed by several appendices. A KA is an identified area of software engineering defined by its knowledge requirements and described in terms of its concepts, component processes, practices, inputs, outputs, tools, and techniques. All KAs have been updated in V4 to reflect changes in software engineering since the publication of Guide V3. The new version reflects modern development practices, techniques, and the advancement of standards. Especially, agile and DevOps have been incorporated into almost all KAs since these models have been widely accepted since Guide V3 was published. Agile models typically have frequent demonstrations of working software to a customer in short iterative cycles with agile practices involving KAs. Furthermore, emerging platforms and technologies, including artificial intelligence (AI), machine learning (ML), and Internet of Things (IoT), have been incorporated into the foundation KAs. Also, there are three new KAs: Software Architecture, Software Operations, and Software Security. It is important to note there has also been more emphasis on Software Security in other relevant KAs such as software requirements. These additions better reflect contemporary software engineering practice, the pressing need to address cybersecurity as early as possible in developing distributed, networked systems, and non-distributed systems for use by the public at large, and the need for specialized skills to be able to work effectively in these areas.

Table 1. SWEBOK v4 Table of Contents (SEBoK Original)

Knowledge Areas
Requirements
Architecture*

Design
Construction
Testing
Operations*
Maintenance
Configuration Management
Engineering Management
Process
Models and Methods
Quality
Security*
Professional Practice
Economics
Computing Foundations
Mathematical Foundations
Engineering Foundations

The 18 KAs in Guide V4 are:

KA1 - Software Requirements: Describes the activities involved in eliciting, analyzing, specifying, validating, and managing software requirements. The importance of requirements documentation for long-term maintenance is added. The "what's" and "how's" of work on software requirements in a project should be determined by the type of constructed software and not by the project life cycle. Also, it includes a deeper, broader coverage of requirements specification techniques, including model-driven requirements specification and acceptance criteria-based requirements specification.

KA2 – Software Architecture: This is a new KA resulting from a realization that architecture is a significant and distinct discipline over and above software design. Software architecture consists of fundamental structures of software elements, relations among them, and properties of both elements and relations. As software tends toward larger, ever more complex systems, architecture concerns move beyond construction to connectivity while assuring new levels of quality for safety, security, and dependability. Software architecture aims to "satisfice" all stakeholders, while the focus of software design is on the transformation of that vision into a feasible implementation.

KA 3 - Software Design: Focuses on the process of transforming requirements into a representation of the software system. Software is pervasive and critical in all aspects of modern life. The success of pervasive software

depends on broadening the background of people with design skills, especially as emerging technologies, faster delivery times, and emphasis on agile, lean, and incremental design impact the development process.

KA4 - Software Construction: Covers the activities involved in translating the software design into executable code. This section has been updated to reflect modern construction techniques and practices: managing dependencies, cross-platform development and migration, feedback loops for construction, visual programming, and low-code/zero-code (i.e., called no code) platforms. Furthermore, major updates were made to some of the existing sections, especially about reuse, life-cycle models, construction languages and environments.

KA5 - Software Testing: Deals with the various approaches and techniques used to assess software correctness and verify its conformance to requirements. This material has been revised to align with the shift left testing movement. This KA provides new content about software testing in recent development processes (like Agile, DevOps, and Test-Driven Development), application domains (like automotive, healthcare, mobile, legal, and IoT), emerging technologies (such as AI, blockchain, cloud) and quality attributes (like security and privacy).

KA6 - Software Engineering Operations: This is a new knowledge area that addresses the evolution of the role of software engineers to include DevOps and infrastructure as a Service (IaaS) activity while eliminating the organizational silos between development, maintenance, and operations. This new KA describes operations fundamentals, planning, delivery and control activities, and techniques.

KA 7 - Software Maintenance (SM): Software maintenance addresses fundamentals, processes, and techniques to provide cost-effective support for software in operation. Activities, such as determining the logistical support needed for the transition, are performed, and applied during the pre-delivery stage, whereas software surveillance, modification, training, and operating or interfacing with a help desk are post-stage activities. Software Maintenance categories and the software process have been updated to align with the 2022 version of ISO/IEC/IEEE 14764. (ISO/IEEE/IEEE 2022) Also, continuous integration, delivery, testing, and deployment have been added as a new topic in response to the growing popularity of

regrouping development, maintenance, and operations tasks to improve software engineering productivity.

KA 8 - Software Configuration Management (SCM): Focuses on four basic functions of SCM: Configuration Identification: identifying all configuration items (CI) of a software system's state; Control changes: coordinating and limiting access to configurations in a software system; Configuration auditing: a series of reviews confirming that changes are being made to conform to a software system's desired state; and Status Accounting: automatically recording information about CIs, relationships, baselines and changes on any CI developed, as well as why they were made, when they were made, and who made them. In SWEBOK Guide V4, SCM has been explained considering all its facets. It stresses that the SCM process plan must be defined first before all the decisions and commitments included in the plan are somehow incorporated into existing tools for execution rather than the other way around, as was the usual practice in the past. Keeping track of configuration items (CI) relationships is essential to visualize the potential impact of a change on a CI over other CIs.

KA9 - Software Engineering Management: Describes the activities of the planning, organizing, estimating, and tracking of software projects. Practices are changing to meet the needs of society caused by increases in the size and complexity of software as well as greater pressure to quickly release software enabling products to market in rapidly changing environments. The results are fundamental management shifts in how work is performed.

KA10 - Software Engineering Process: Describes the activities, methods, and techniques used to define, implement, assess, and improve the software development process. Software engineers face a challenging and evolving, highly technological landscape due to fast-moving technology, societal events and changes, and a trend towards digitalization involving very many different domains. This is a point of no return that has facilitated the ongoing adoption of Agile and DevOps. Nevertheless, now more than ever, the software engineering processes applied to create products will continue to evolve, profiting from advances in the engineering dimension and learning, new and more advanced tools, and the latest knowledge from the other KAs. The KA describes this new consensus, highlighting the interaction with other KAs.

KA 11 - Software Engineering Models and Methods:

Introduces different models, methods, and tools that support the software development process. Models have been updated and accurately classified by type. Aspect-oriented development has been added in the document as well as fundamental model-driven and model-based methods, have also been introduced. Agile methods have been extended a great deal to incorporate modern techniques, such as lean development, as well as large-scale and enterprise agile methods. On this note, DevOps and release engineering have also been introduced to clarify the release aspect of agile methods.

KA12 - Software Quality: Addresses the activities and techniques involved in achieving and evaluating software quality. This chapter was overhauled for alignment with notions of processes/product quality. It now includes new topics: (1) Software Dependability and Integrity Levels, (2) Standards, Models and Certifications, (3) Policies, Processes and Procedures, and (4) Quality Control and Testing.

KA13 – Software Security: This is a new knowledge area that focuses on the broader topics of security, particularly security fundamentals, security management, security tools, and domain-specific security, as well as major security engineering activities (such as security testing) that were briefly described under the Computing Foundations KA in the last SWEBOK edition. The existence of a separate KA emphasizes that security must be a first-class quality attribute in any development since almost any software system is connected to others, resulting in increased security risks.

KA14 - Software Engineering Professional Practice: Covers ethical considerations, professional codes of conduct, and the role of software engineering in society. Additions and revisions address the following areas: (1) Professional practices following generally accepted practices, standards, and guidelines set forth by the applicable professional societies, (2) User interface/user experience (UI/UX) inclusive design, (3) Considerations on diversity and inclusion, and (4) Considerations on agility.

KA15 - Software Engineering Economics: Focuses on the essence of engineering economics, the science of making decisions, and broadening the more traditional, purely financial view of engineering economics. Value does not always derive from money alone; that is, value can also derive from the “unquantifiable” such as, corporate citizenship, employee well-being,

environmental friendliness, customer loyalty, and so on. This KA also focuses on more systematic pre-project decisions where a project is not under development but is being envisioned.

Note: Minor improvements in topic presentation in the next three KA 16-18. Discussions of Artificial Intelligence and Machine Learning were added. Furthermore, the revision includes a deeper coverage of measurement, particularly its implications for programming languages, and a more comprehensive discussion of root cause analysis.

KA16 - Computing Foundations: Covers the fundamental concepts and theories that underlie the discipline of software engineering.

KA 17 - Mathematical Foundations: Introduces mathematical techniques and principles relevant to software engineering, such as logic, probability, and discrete mathematics.

KA - 18 Engineering Foundations: Covers engineering principles and practices applicable to software engineering, including system engineering and project management.

Concluding Comments

A significant component of productivity today is the capabilities enabled by software systems. Software is a maturing science and engineering discipline; therefore, a need exists to continuously provide new guidance materials that reflect areas that are becoming important in modern software engineering. The Software Engineering Body of Knowledge Version V4 is a leading product which will help address this issue. It will serve as a reference for educators, practitioners, and organizations to understand the essential concepts and skills needed to design, develop, and maintain current and future software systems.

SWEBOK V4 represents the next step in the evolution of the software engineering profession and is written under the auspices of the Profession and Education Activities Board of the IEEE Computer Society. An international team of subject matter experts was assembled with the goal of developing the update. Given the purpose of this article, a significant portion of this document includes verbatim or near-verbatim elements of the SWEBOK Guide V4 which is about to go out for public review and that of (Washnizaki, et al. 2023) written by several

members of the international team.

SWEBOK is described inside the SEBoK and there are some clear similarities and differences between the two BOKs. Each BOK is a comprehensive guide that outlines the core knowledge areas, principles, and best practices. SWEBOK concentrates specifically on the software engineering domain, providing a detailed overview of software development processes and methodologies. SEBoK, on the other hand, has a broader scope and addresses the entire systems engineering domain, including software and beyond, considering the interaction and integration of various components in complex systems.

References

Works Cited

Bourque, P. and R.E. Fairley (eds.). 2014. *Guide to the Software Engineering Body of Knowledge (SWEBOK)*. Los Alamitos, CA, USA: IEEE Computer Society. Available at: http://www.swebok.org.

IEEE Computer Society. 2014. Software engineering competency model (SWECOM), Version 1.0. Accessed August 29, 2023. Available at https://www.computer.org/volunteering/boards-and-committees/professional-educational-activities/software-engineering-competency-model.

IEEE Computer Society. 2022. SWEBOK Guide Version 4.0 beta. Accessed August 28, 2023. Available at https://waseda.app.box.com/s/elnhhnezdycn2q2zp4fe0f2t1fvse5rn.

ISO/IEC/IEEE. 2017. ISO/IEC/IEEE Systems and Software Engineering – Vocabulary. Accessed August 28, 2023. Available at https://www.iso.org/standard/71952.htmlhttps://www.iso.org/standard/71952.html.

ISO/IEC/IEEE. 2022. ISO/IEC/IEEE 14764-2022 Software Life Cycle Processes – Maintenance. Accessed August 28, 2023. Available at https://www.iso.org/standard/80710.html.

Washizaki, H., Sanchez-Segura, M-I., Garbajosa, J., Tockey, S., Nidiffer, K.E. 2023. “Envisioning Software Engineer Training Needs in The Digital Era Through the SWEBOK V4 Prism”, Proceedings of the IEEE

International Conference on Software Engineering Education and Training (CSEE&T 2023), August 8-9, 2023. Waseda University, Tokyo Japan.

Primary References

Bourque, P. and R.E. Fairley (eds.). 2014. *Guide to the Software Engineering Body of Knowledge (SWEBOK)*. Los Alamitos, CA, USA: IEEE Computer Society. Available at: http://www.swebok.org.

IEEE Computer Society. 2022. *SWEBOK Guide Version 4.0* beta. Accessed August 28, 2023. Available at https://waseda.app.box.com/s/elnhhnezdycn2q2zp4fe0f2t1fvse5rn.

Additional References

None.

< Previous Article | Parent Article | Next Article >
SEBoK v. 2.10, released 06 May 2024
