Software Engineering Features - Models, Methods, Tools, Standards, and Metrics

Lead Author: *Tom Hilburn*

In recent decades, software has become ubiquitous. Almost all modern engineered systems include significant software subsystems; this includes systems in the transportation, finance, education, healthcare, legal, military, and business sectors. Along with the increase in software utility, capability, cost, and size there has been a corresponding growth in methods, models, tools, metrics and standards, which support software engineering.

Chapter 10 of the SWEBOK discusses modeling principles and types, and the methods and tools that are used to develop, analyze, implement, and verify the models. The other SWEBOK chapters on the software development phases (e.g., Software Design) discuss methods and tools specific to the phase. Table 1 identifies software engineering features for different life-cycle phases. The table is not meant to be complete; it simply provides examples. In Part 2 of the SEBoK there is a discussion of models and the following is one of the definitions offered: “an abstraction of a system, aimed at understanding, communicating, explaining, or designing aspects of interest of that system” (Dori 2002).

For the purposes of Table 1 the definition of a model is extended to some aspect of the software system or its development. As an example, “Project Plan” is listed as a model in the Software Management area. The idea is that the Project Plan provides a model of how the project

is going to be carried out: the project team organization, the process to be used, the work to be done, the project schedule, and the resources needed.

Table 1: SWE Features (SEBoK Original)

Life-Cycle Activity	Models	Methods & Tools	Standards
Software Management	• Life-Cycle Process Model • Work Breakdown Structure • Constructive Cost Model (COCOMO) • Project Plan • Configuration Management (CM) Plan • Risk Management Plan	• Effort, Schedule and Cost Estimation • Risk Analysis • Data Collection • Project Tracking • CM Management • Iterative/Incremental Development • Agile Development	• [IEEE 828] • [IEEE 1058] • [IEEE 1540] • [IEEE 12207]
Software Requirements	• Functional Model • User Class Model • Data Flow Diagram • Object Model • Formal Model • User Stories	• Requirements Elicitation • Prototyping • Structural Analysis • Data-Oriented Analysis • Object-Oriented Analysis • Object Modeling Language (OML) • Formal Methods • Requirements Specification • Requirements Inspection	• [IEEE 830] • [IEEE 1012] • [IEEE 12207]

Software Design	• Architectural Model • Structure Diagram • Object Diagram • Class Specification • Data Model	• Structured Design • Object-Oriented Design • OML • Modular Design • Integrated Development Environment (IDE) • Database Management System (DBMS) • Design Review • Refinement	• [IEEE 1012] • [IEEE 1016] • [IEEE 12207] • [IEEE 42010]
Software Construction	• Detail Design Document • Pseudocode • Flow Chart • Program Code • Unit Test Plan • Integration Test Plan	• Detailed Design • Functional Programming • Object-Oriented Programming • IDE • DBMS • Black Box/White Box Testing • Basic Path Testing • Unit Testing • Code Review • Proof of Correctness • Software Reuse • Integration • Integration Testing	• [IEEE 1008] • [IEEE 1012] • [IEEE 1016] • [IEEE 12207]
Software Testing	• System Test Plan • Reliability Model • Software Maintenance Process	• Usability Testing • System Testing • Acceptance Testing • Regression Testing • Reliability Testing • Non-Functional Software Testing	• [IEEE 829] • [IEEE 1012] • [IEEE 12207]
Software Maintenance	• Software Maintenance Process	• Automated Testing Tools • Maintenance Change • Impact Analysis • Inventory Analysis • Restructuring • Reverse Engineering • Re-engineering	• [IEEE 1219] • [IEEE 12207] • [IEEE 14764]

Software Metric

A software metric is a quantitative measure of the degree a software system, component, or process possesses a given attribute. Because of the abstract nature of software and special problems with software

schedule, cost, and quality, data collection and the derived metrics are an essential part of software engineering. This is evidenced by the repeated reference to measurement and metrics in the SWEBOK. Table 2 describes software metrics that are collected and used in different areas of software development. As in Table 1 the list is not meant to be complete, but to illustrate the type and range of measures used in practice.

> **Table 2: Software Metrics** * (SEBoK Original)

Category	Metrics
Management Metrics	• Size: Lines of Code (LOC*), Thousand Lines of Code (KLOC) • Size: Function points, Feature Points • Individual Effort: Hours • Task Completion Time: Hours, Days, Weeks • Project Effort: Person-Hours • Project Duration: Months • Schedule: Earned Value • Risk Projection: Risk Description, Risk Likelihood, Risk Impact
Software Quality Metrics	• Defect Density - Defects/KLOC (e.g., for system test) • Defect Removal Rate – Defects Removed/Hour (for review and test) • Test Coverage • Failure Rate
Software Requirements Metrics	• Change requests (received, open, and closed) • Change request frequency • Effort required to implement a requirement change • Status of requirements traceability • User stories in the backlog
Software Design Metrics	• Cyclomatic Complexity • Weighted Methods per Class • Cohesion - Lack of Cohesion of Methods • Coupling - Coupling Between Object Classes • Inheritance - Depth of Inheritance Tree, Number of Children

Software Maintenance and Operation	• Mean Time Between Changes (MTBC) • Mean Time to Change (MTTC) • System Reliability • System Availability • Total Hours of Downtime

**Note: Even though the LOC metric is widely used, using it comes with some problems and concerns: different languages, styles, and standards can lead to different LOC counts for the same functionality; there are a variety of ways to define and count LOC– source LOC, logical LOC, with or without comment lines, etc.; and automatic code generation has reduced the effort required to produce LOC.*

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.

Dori, D. 2003. "Conceptual modeling and system architecting." *Communications of the ACM*, 46(10), pp. 62-65.

[IEEE 828] IEEE Computer Society, IEEE Standard for *Computer Configuration Management in Systems and Software Engineering*, IEEE Std 828- 2012, 20012.

[IEEE 829] IEEE Computer Society, IEEE Standard for *Software and System Test Documentation*, IEEE Std 829-2008, 2008.

[IEEE 830] IEEE Computer Society, IEEE Standard for *Recommended Practice for Software Requirements Specifications*, IEEE Std 830-1998, 1998.

[IEEE 1008] IEEE Computer Society, IEEE Standard for *Software Unit Testing,* IEEE Std 1008-1987, 1987.

[IEEE 1012] IEEE Computer Society, IEEE Standard for *System and Software Verification and Validation*, IEEE Std 1012-2002, 2012.

[IEEE 1016] IEEE Computer Society, IEEE Standard for *Recommended Practice for Software Design*

Descriptions, IEEE Std 1016-2002, 2002.

[IEEE 1058] IEEE Computer Society, IEEE Standard for *Software Project Plans*, IEEE Std 1058-1998, 1998.

[IEEE 1219] IEEE Computer Society, IEEE Standard for *Software Maintenance*, IEEE Std 1219-1998, 1998.

[IEEE 1540] IEEE Computer Society, IEEE Standard for *Risk Management*, IEEE Std 1540-2001, 2001.

[IEEE 12207] IEEE Computer Society, IEEE Standard for *Systems and Software Engineering —Software Life Cycle Processes*, IEEE Std 12207-2008, 2008.

[IEEE 14764] IEEE Computer Society, IEEE Standard for *Software Engineering - Software Life Cycle Processes - Maintenance*. IEEE Std 14764-2006, 2006.

[IEEE 42010] IEEE Computer Society, IEEE Standard for *Systems and Software Engineering — Architecture Description*, IEEE Std 42010-2011, 2011.

Primary References

None.

Additional References

Chidamber, S.R., C.F. Kemerer. 1994. "A Metrics Suite for Object Oriented Design", *IEEE Transactions on Software Engineering*. Vol. 20, No. 6. June 1994.

Kan, Stephen H. 2003. *Metrics and Models in Software Quality Engineering*, 2nd edition. Reading, Massachusetts, USA: Addison-Wesley.

Li, M. and Smidts, C. 2003. "A ranking of software engineering measures based on expert opinion." *IEEE Transactions on Software Engineering*. September 2003.

McConnell, Steve. 2009. *Code Complete*, 2nd Ed. Microsoft Press.

Moore, James. 1997. *Software Engineering Standards: A User's Road Map*. Hoboken, NJ, USA: Wiley-IEEE Computer Society Press.

Sommerville, I. 2010. *Software Engineering*. 9th Ed. Boston, MA, USA: Addison Wesley.

< Previous Article | Parent Article | Next Article >
SEBoK v. 2.7, released 31 October 2022