SWEBOK:软件工程知识体

SWEBOK:Software Engineering Body of Knowledge

白 征

(深圳职业技术学院计算机系 深圳518055)

Abstract The paper introduces the guide to Software Engineering Body of Knowledge,a project developed jointly by IEEE computer society and ACM The background,objectives and viewpoints of the project are presented ,with an emphasis on the core of the SWEBOK.

Keywords Software engineering,SWEBOK

1.引言

尽管在现代社会中计算机软件处处可见,并在世界范围内已有不少的软件专业人员从事软件开发工作,软件工程却并没有被认为是一个合法的工程学科专业。自1993年以来,IEEE 计算机学会和 ACM 学会开始积极促进软件工程成为独立专业,并在1994年成立了一个专门小组负责研究这一问题。根据项目的规模和进展情况,软件工程势必成为等同于计算机工程、电气工程和建筑工程类的专业学科。1999年1月软件工程协调委员会(SWECC:Software Engineering Coordinating Committee)正式成立,替代了原来的专门小组接手了这一工作。为了促成软件工程成为独立的计算机技术学科专业,IEEE 计算机学会和 ACM 学会联合开展了三个项目来达到目的。1998年由 SWECC 发起的软件工程知识体(SWEBOK:Software Engineering Body of Knowledge)研究项目便是其中之一。另外两个研究项目分别是"软件工程专业课程计划"和"软件工程道德法律和专业实践"。

软件工程知识体指南(Guide to SWEBOK)分三阶段完成,这三个阶段分别称为"草人(straw man)"、"石人(stone man)和"铁人(iron man)"阶段。"草人"版 SWEBOK 指南于1998年9月完成,它形成了软件工程知识体的组成框架;"石人"版 SWEBOK 指南于2000年2月完成,是试用版,也是本文的主要参照;"铁人"阶段则将耗时三年,预定2003年完成,它将包括更深层次的分析和更广泛的评审过程,并包括试用所获的经验。

2. SWEBOK 项目的目标

SWECC 开展 SWEBOK 项目的目的是为了建立一组标准和规范,作为软件工程专业实践中进行产业决策、专业认证和制定教育课程计划的基础,该目标包括五点:①促进国际上对软件工程的一致理解,为达此目的,SWECC 聘请了42个国家的近500名专家承担了该项目的评审工作;②确定软件工程相对于其它学科,如计算机科学、工程管理、计算机工程和数学等的地位和范围;③描述软件工程学科内容和特征;④提供获取软件工程知识体相关信息的目录;⑤提供软件工程课程计划的开发、个人专业证书和专业许可的依据。

由于软件工程知识体是在30多年软件工程实践过程中发展而形成的产物,是现实存在的东西,因此 SWEBOK 项目的目的不在于描述该知识体本身,而在

方法可能不是最佳的,但从例2可以看出,该求法具有相当的一般性。底集定义中的条件2似乎可以去掉,但这会破坏 $\mathcal{O}_S$ 的唯一确定性,因而加大定理证明的难度,本方法适合于特定领域弱终止性证明的自动化。可以使用动态项重写计算[2]进行形式自动证明,其实现细节将在今后的论文中进行论述。

参 考 文 献

1 Dershowitz N. Termination of rewriting. J. Symb. Comput.,1987,3:69~115

2 Feng S,et al. Confluence property of simple frames in dynamic term rewriting calculus. IEICE Trans. Inf. & Syst.,1997,E80-D(6):625~645

3 冯速.项重写系统的等价性的归纳证明.计算机科学,2000,27(8),5~7

4 Huet G Confluent reductions:abstract properties and applications to term rewriting systems J. ACM,1980,27:797~821

5 Toyama Y. How to prove equivalence of term rewriting systems without induction. Theoret.Comput. Sci.,1991,90:369~390

于达到对能够刻画软件工程学科本质特征的核心知识子集的一致认识，提供对该知识体的指南，它对各个企业组织对软件工程达成共识，满足他们在教育与培训方面的需求、进行工作分类和发展性能评估策略等方面有所帮助，并为软件工程师的专业实践，软件工程的专业认证，大学课程计划的制定与评估、学生学习提供指南。

3. SWEBOK 指南的特点

1)软件工程应当以计算机科学为基础，正如电气工程以物理科学为基础一样，但软件工程与计算机科学相比有显著不同的侧重点。科学家做的事情是延伸自然定律，而工程师做的事情是在一些限制条件下，运用自然定律来建造有用的产品。因此，该指南强调的是构造有用的软件产品。

2)该指南并没有包括许多对构成软件工程知识可能很重要的信息技术，如特定的编程语言、关系数据库和网络等，因为特定技术的变化太快，容易过时。工程师必须掌握一些基础知识，这些基础知识能够支撑他在适当的环境和适当的时间选择适当的技术。

3)该指南包括的软件工程知识对软件工程师来说是必备的但未必是足够的知识。进行实践的软件工程师还需要知道很多关于计算机科学、工程管理和系统工程等方面的知识，这些知识不属于该指南描述的知识体范围，但该指南提供其它学科相关信息的参考资料的目录。

4)该指南强调工程实践，从而导致与标准文献具有密切关系。标准文件规定了在特定场合下工程师应当做什么，而不是仅提供有用的信息。两个主要的软件工程标准化团体(IEEE 软件工程标准委员会和 ISO/IEC JTC1/SC7)派代表参与了该项目。

5)该指南遵循两个原则："透明"和"共识"。"透明"是指该指南的开发过程本身是用文档描述的，可公开发表，并可宣传，使项目的重要决策和状态对所有参与人员是可见的。"共识"是指实现软件工程学科专业合法化的实际方法只有通过广泛的参与并使各方重要人士和团体取得一致意见。

6)该指南描述被广泛接受的软件工程核心知识，"被广泛接受的"是指那些由多数组织机构推荐的，实际使用的知识。

7)该指南确认了与软件工程密切相关的7个学科，SWEBOK 的内容可以参照这些相关领域知识。这7个学科包括：认知科学和人类因素、计算机工程，计算机科学，管理和管理科学，数学，工程管理和系统工程等。

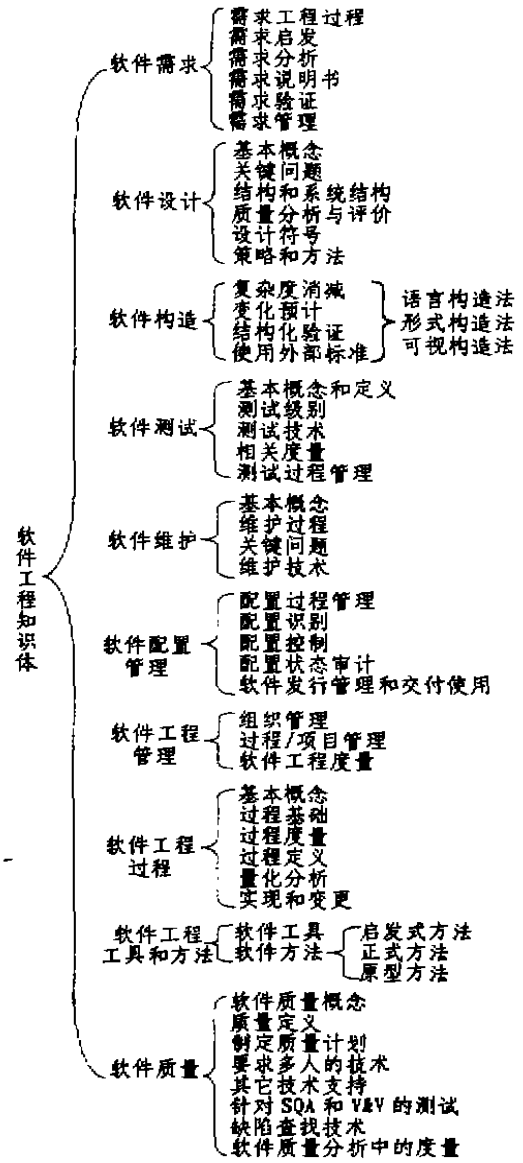

图1　软件工程知识体的知识域分解

4. SWEBOK 的组成

SWEBOK 指南对软件工程知识体进行了分层描述，将软件工程知识分解成"知识域"和"知识域组成部分"，并将该知识体组织成多级层次结构(见图1)，以此确定软件工程学科的内容和边界。在 SWEBOK 指南中，软件工程知识体被分解成10个知识域：软件需求；软件设计；软件构造；软件测试；软件维护；软件配置管理；软件工程管理；软件工程过程；软件工程工具和方法；软件质量。

4.1　软件需求知识域

需求被定义为解决现实问题所必须展示的特性。它包含六个知识子域：

1)需求工程过程：与整个软件工程过程吻合，描述过程模型、过程参与人、过程支持和管理、过程质量改进。

2)需求启发(requirement elicitation)：描述从何处获取需求及需求工程师收集需求的方法，包括需求来源与启发技术。

3)需求分析：描述分析需求的过程，如发现并解决需求之间的冲突，发现系统边界和系统必须怎样与环境相互作用，详细了解系统需求等。需求分析可再分解成：需求分类、建立概念模型、系统结构设计、需求分配和需求谈判等知识子域。

4)软件需求说明书(SRS)：描述需求文档的结构、质量和标准，包括系统需求定义文档和软件需求说明书两类。

5)需求验证：目的是在提交需求分析结果之前找出问题，保证需求文档定义了正确的(用户所期望的)系统，该子域描述审查需求文档的过程，并再分解成四个子域：需求评审的执行、原型制作、模型验证和验收测试。

6)需求管理：是一个跨越整个软件生命周期的活动，从本质来说是关于需求的维护和需求的变更管理的知识，目的是保证需求说明准确地反映了待开发的或已经开发的软件，它包括变更管理、需求属性和需求跟踪等三个子域。

4.2 软件设计知识域

按照IEEE的定义，软件设计是定义一个系统或子系统的结构、组成部分、接口和其他特征的过程，也是该过程的结果。如果把软件设计看成过程，则它是一种活动，该活动的任务是分析软件需求，并生成对系统内部结构和组成的说明作为软件构造的依据；如果把软件设计看成结果，则它是一个说明文档，该文档描述系统分解和组成的方式、各组成部件之间的接口，并对各组成部件进行足够详细的描述，以便于其构造。

软件设计知识域包括六个知识子域：

1)软件设计基本概念：理解软件设计的作用和范围的基础，包括软件设计的一般概念、软件设计的内容、设计过程和可采用的技术。

2)软件设计关键问题：包括并发性、分布性、事件控制和处理、错误和异常处理、交互式系统和持续性等问题。

3)软件设计结构和系统结构：按特定的构造结构和观点看，系统结构的风格、设计模式、以及程序及其构架的最终划分和组合。

4)软件设计质量分析和评价：包括软件设计的质量属性、质量分析和评估工具(包含子域：软件设计评审、静态分析、仿真和原型制作)和度量(包含子域：面向功能/结构的设计度量和面向对象的设计度量)。

5)软件设计符号：包括结构描述(静态观)和行为描述(动态观)。

6)软件设计策略和方法：包括一般设计策略、面向功能的方法、面向对象的方法、面向数据结构的方法、其他方法，如形式方法和变换方法。

4.3 软件构造知识域

软件构造是软件工程的基本活动，其任务是通过编码、验证和单元测试构造出有意义的、可工作的软件产品。分解软件构造知识子域时，最重要的一点是认识对软件构造影响最强的四项原则，即：

·复杂度消减：包括软件构造期间用于减小复杂度的三个主要技术：复杂度消除、复杂度自动化消减和复杂度局部化。

·变化预估：对软件在生存期间发生各种变化的预测。在构造软件时预测变化的三个主要技术是：普遍化、实验法和局部化。

·结构化验证：以结构化方式建立软件，这种方式能够容易地在单元测试和后继的测试活动期间检出错误和遗漏。

·使用外部标准：专用语言建立的软件在被长期使用的过程中会遇到很多问题，如难以理解进而难以维护等，因此，应当采用符合外部标准的构造语言，如一般编程语言所用的标准。否则须提供足够详细的"语法"说明，使该构造语言过后能被其他人所理解。

分解软件构造知识子域的另一点是认识软件构造的三种方式和方法，即：语言方法、形式方法和可视方法。

4.4 软件测试知识域

软件测试是用有限的测试用例集合对照预期指定的行为对程序实际的行为进行动态证明的过程，测试用例通常是从无限执行域中适当挑选的。该知识域包含下述五个子域：

1)基本概念：包括测试术语、测试理论基础以及测试与其他活动的关系。

2)测试级别：包括测试目标(单元、集成、系统)和测试目的(接受测试、安装测试、回退测试、恢复测试、等等)。

3)测试技术：包括测试用例选取标准，测试技术(白盒、黑盒技术等等)本身，以及如何选用适当的技术。

4)测试相关的度量：包括对被测试的程序的评价和对所进行的测试的评价。

5)测试过程管理：包括测试管理关注问题和测试

活动。

4.5 软件维护知识域

一旦软件交付使用，软件生命期的维护阶段即开始。维护活动的任务包括发现软件运行中的错误、响应运行环境变化和用户新的要求。软件维护知识子域包括：

1）基本概念：包括软件维护的定义、主要活动和问题。

2）维护过程：描述基于IEEE1219和ISO/IEC14764标准的维护过程。

3）关键问题：包括技术、管理、费用、预算和度量等问题。

4）维护技术：包括程序理解、再工程、逆向工程和效果分析等。

4.6 软件配置管理知识域

软件配置管理是在明确的时间点上确定系统配置的方法，其目的是在整个系统生命期中系统地控制配置的变化并维护配置的完整性和可跟踪性。软件配置管理包括六个知识子域：

1）配置过程管理：关于软件配置的组织环境、限制和指南、计划和计划制定、监督等。

2）软件配置识别：确定要控制的配置项、为配置项及其版本建立标识方案；建立在采集和管理所控制的配置项中要用的工具和技术。包括"要控制和配置项标识"和"软件库"两个知识子域。

3）软件配置控制：对软件生命期中的变化进行管理，包括软件变化的请求、评估和批准、软件变化的实现。软件变化的偏离和放弃等三个知识子域。

4）软件配置状态审计：包括软件配置状态信息和软件配置状态报告。

5）软件配置审计：包括软件功能配置审计、软件物理配置审计、在进行中的软件基线审计。

6）软件发行管理和交付使用：包括软件建立和软件发行管理。

4.7 软件工程管理知识域

包括三个知识子域：

1）组织管理：再分解为政策管理、个人管理、沟通管理、协调管理、采购管理等。

2）过程/项目管理：包括项目启动和范围定义、制定计划、建立规定、项目评审和评价、项目结束等。

3）软件工程度量：包括软件度量的一般原理，如：度量程序的目标、度量的选择、软件度量及其发展、数据收集和软件计量模型。

4.8 软件工程过程知识域

关于软件工程过程本身的定义、实施、度量、管理、变更和改进、分解为六个知识子域：

1）基本概念：目的和术语。

2）过程基础建设：软件工程过程小组和经验工厂。

3）过程度量：评价过程的方法和范例。

4）过程定义：包括过程定义类型、生命期框架模型、软件生命期过程模型、过程定义符号、过程定义方法和自动化。

5）量化分析：包括过程定义评审和原因分析。

6）过程实施和变更：包括过程实施和变更的范例、指南和效果评价等。

4.9 软件工程工具和方法知识域

包括软件开发环境和开发方法。软件开发环境是辅助软件开发过程的计算机工具有机集合，这里共分解成11个知识子域、包括：需求分析、设计、构造、测试、维护、过程、质量、配置、管理、基础建设和其他活动所需的支撑工具；开发方法是指对软件开发活动的组织方法，目的是让这些活动有组织地进行从而使成功的可能性最大，包括启发式方法、形式方法、原形方法和其他方法等4个知识子域。

4.10 软件质量知识域

软件质量是贯穿于整个软件工程活动的关注点，它包括四个知识子域：

1）质量概念：包括质量值的度量，ISO9126的质量描述、特殊类型的系统和质量要求。

2）软件质量保证（SQA）和证明与确认（V&V）的目的和计划制定。

3）SQA和V&V包含的活动，用于SQA和V&V的动态技术和静态技术。

4）SQA和V&V采用的度量方法。

结束语 SWEBOK指南对软件工程学科的特征范围提供了一致而有效的描述，并提供了查找支持该学科知识体的参考文献的线索，它不仅反映了三十多年来软件工程发展的状态和成果，而且指出了该学科发展的方向。此外，该指南强调软件工程基础知识和实践性，强调实施方法和标准，对我国软件产业和软件工程教育的发展均具有很大的参考价值。

参考文献

1 Guide to the Software Engineering Body of Knowledge. IEEE-Stoneman (Trial Version, Version 0.9), February 2001.202

2 Parnas D L. Software engineering programs are not computer science programs. IEEE software, 1999, 11/12: 19～30

3 Bourque P, et al. The Guide to the Software Engineering Body of knowledge. IEEE Software, 1999, 11/12: 35～44

4 Hilburn T B, et al, Guidance for development of software engineering education programs USA. The Journal of System and Software, 1999, 49(2/3): 163～169