面向服务的软件开发方法评价分析框架研究

苏红军 尤振华 王国华

(宁夏科技发展战略和信息研究所 银川 750001)

摘 要 面向服务的软件解决方案因其"重用"和"互操作"的核心概念,已经成为实施企业级系统的指导标准。然而,由于面向服务的软件开发的方法学评价仍然缺乏完善的定义,因此在项目实施过程中很难评价和比较各种情况下的相应机制。使用一组定性和定量特征进行评价的分析框架可以从结构、过程、产品3个方面较好地对软件项目进行评价。通过一个实例展示了该框架的灵活性、弹性以及全面的特点。

关键词 面向服务,软件开发,定量,定性,分析框架,评价

中图法分类号 TP31 **文献标识码** A

Study on Evaluation and Analysis Framework for Service-oriented Software Development Method

SU Hong-jun YOU Zhen-hua WANG Guo-hua

(Ningxia Institute of Science and Technology for Development and Information, Yinchuan 750001, China)

Abstract Service-oriented software solution becomes the guiding criterion for the implementation of enterprise-level systems because of its core concept of "reuse" and "interoperability". However, it still lacks a well-defined way for the methodological evaluation of service-oriented software development, so it is difficult to evaluate and compare corresponding mechanisms of different situations in the implementation of projects. Analysis framework using a group of qualitative and quantitative features for evaluation can achieve perfect evaluation of software projects from three aspects of structure, process and product. An example shows the flexible, elastic and comprehensive characteristics of the framework.

Keywords Service-oriented, Software development, Quantify, Qualitative, Analysis framework, Evaluation

1 引言

面向服务的计算(Service-Oriented Computing, SOC)和面向服务的体系结构(Service-Oriented Architecture, SOA)是企业级分布式系统的开发标准。SOC的核心思想是发现和组合,业务流程在运行时找到最合适的服务,并相互协调,以便满足特定业务的需求。SOA是构造分布式计算的应用程序的方法,它将应用程序功能作为服务发送给最终用户或者其他服务。其基本原则包括可重复使用、模组性、可组合型、构件化以及交互操作性等。开发基于这两种标准的系统通常都是庞大而且复杂的工作。随着用户需求日益增加,系统的开发难度变得越来越大。因此,软件开发人员引入一种开发方法的概念,开发方法可以带来明确的、清晰的过程,可以指导软件工程的开发周期。具体讲就是将一个明确的成熟的方法应用在软件开发过程中,用来提高工作效率和改善产品质量。同时,开发方法为构建软件系统提供特定的范式及规则的支持。

目前,在面向服务解决方案中的不同业务领域已经有一些开发方法被提出或者应用,针对这些开发方法现在也有一些评价方法,但是还没有完全满足面向服务方案的特性,有一定的局限性。

因此,本文提出一个采用特征分析和度量评价为基础的,可以客观评价软件开发方法的框架。该框架规定了一套理想的特征,用定性检查(如成熟度和易用性)来体现各种方法的特点,用一组定量指标来评价内部属性(如任务和产品之间的关系)。为了支持更有条理和针对性的评价过程,该框架对功能指标和质量模型从"结构"、"过程"、"产品"3个不同角度进行方法评价,具有一定的通用性特征,包括质量的特征,以及常用软件开发方法。同时,根据这些特点可以设定相适应的特定模式,这些将在本文SOC的案例中体现。

2 背景

2.1 开发方法元模型

本文所演示的软件开发方法元模型(见图1)将为推导一致的、形式化的评价机制提供理论基础和依据。该元模型结合核心概念和定义通过成熟的方法提取出一个集合过程类的通用模型。它包含了开发方法的重要元素,以及它们之间的关系和适用的限制。该元模型使用UML2描述,并具有一个面向对象的方法。"软件与系统过程工程元模型2.0(SPEM)"规定了一个全面的过程工程元模型,定义了一组通

苏红军(1977—),男,助理研究员,主要研究方向为模型驱动构架、软件工程、软件构架、数据库设计,E-mail:nxshj@hotmail.com;尤振华(1978—),男,工程师,主要研究方向为信息安全、软件工程与自动化;王国华(1984—),男,助理研究员,主要研究方向为软件工程、人机交互。

用的 UML2 衍型来建模和制定软件开发方法及其组件。本文元模型仅包含简单且直观的主进程类，用来配合本文所提出的评价分析框架。

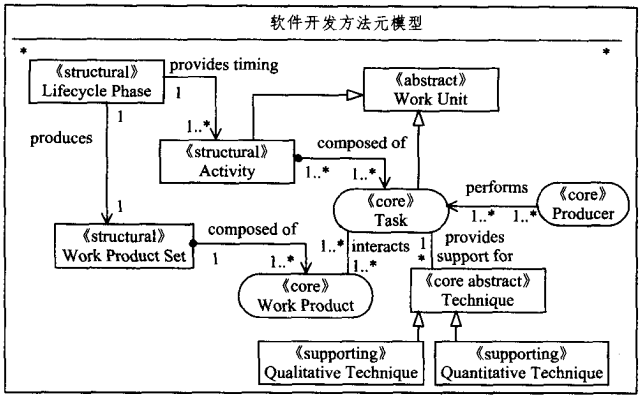

图 1　软件开发方法元模型

进程类被分为 4 种类型(见图 1)：1)核心类代表开发过程中的重要组成；2)结构类为相关核心类提供逻辑结构，通过结构的逻辑对应关系，支持针对性的方法评价；3)抽象类是一种为进程类提供服务的概念类；4)支持类为核心类提供帮助。

2.2　评价方法现状

评价方法的过程大致分内部和外部两类。内部评价被应用在方法本身，以确定其有效性或符合某种公认的标准。外部评价建立压力或者重要影响，在实际应用中用来量化一个特定方法的影响、产品的质量、开发的效率。目前，主要有 3 种评价方法作为内部和外部评价的评价过程：特征分析、本体评价、度量评价。

特征分析是一种广泛引用的定性评价方法，其特点是提取重要特征组成清单，并通过特定方法对清单特征的比较来实现评价。其优点是高效快速，并可以灵活运用；缺点是这种评价具有一定的主观性，且评价特征的确定基于特定原因，使其难以完整和一致地进行评价。除此之外，实际评价过程通常是在一个非正式的方式下进行的，从而很大程度上取决于评价方法特征对标准清单的主观评分。

本体评价就是根据各种评价指标，应用各种科学评价方法，将本体与特定的应用需求及黄金标准进行比较，对影响本体质量的各个因素进行全面、综合测评，得出评价结论，提出修改方案以改善本体结构或功能的过程。本体评价的主要优点是提供了强大的理论基础和正式评价的语义比较特征分析。然而，本体描述语言多种多样，缺乏一致性，另外其复杂的语言规则会给建模和评价带来巨大困难。

基于度量评价可以为内部和外部提供一个正式的和客观的方法评价机制。以度量为基础的评价方法的主要优点是支持正式的、客观的、自动化的方法学评价。缺点在于目前可用的内部度量标准有限，并且新的度量推导和验证可用是一个复杂的过程；其次，外部度量数值只能通过观察生产方法在实践中的行为，以及评价产品质量来计算。

表 1 总结了特征分析、本体评价和基于度量评价的特点：1)表示了涵盖评价框架的具体方法；2)阐述了一个特定评价方法的内部和外部评价指标的整体适用性。3)表示了评价工作的相关性、客观性和评价程序。本文提出的评价框架下的特征分析和度量评价方法可以涵盖内部和外部的评价指标，如表 1 所列。

表 1　评价方法概要

评价方法	评价指标					评价过程
	内部(设计)	内部(过程)	内部(产品)	外部(质量)	外部(效率)	
度量评价	〈C〉*＋	〈C〉*	〈C〉*	〈C〉＋	〈C〉＋	难度大 规范 客观
特征分析	＝	〈C〉*＋	〈C〉*＋	＝	＝	难度小 不规范 主观
本体评价	不适用	＋	＋	不适用	不适用	难度大 规范主观

注：〈C〉：本文提出的评价框架；*：本文介绍的方法；＋：以前的评价方法；＝：需进一步研究。

3　框架概述

本文所提框架包括一套五点李克特量表定义的定性特征和可量化指标。基于这些特征和指标的基础，对相关资料进行全面分析，然后提取一组特征扩展现有的方法。为了给框架推导和特定指标提供基础，并支持更有针对性的评价过程，该框架结构依据已有软件标准和质量模型，这样的层次结构可以支持更有针对性的评价过程，只需评价人员专注于具体评价资源和整体评价的目标和要求。此外，分层模型更容易分析和维护。为此，该框架的结构是一个质量方面的分层模型(如图 2 所示)。该模型关注 3 个质量特征：结构、过程、产品，包括 7 个高级别质量特征(C1－C7)，细分为 22 个子特征(SC1.1－SC7.4)，这些特征可以采用特征顺序或度量比例进行评价。

从“结构”的评价角度看，评价方法构建的实体是过程类结构化集合和软件开发方法元模型包含的它们之间的关系。具体而言，是通过量化普通软件开发方法的可理解性(特征 C1)、工作安排(T)和工作产品(WP)的复杂性进行评价。

从“过程”的评价角度来看，其涵盖了规定的开发过程和相关方法指导的各方面特征，并设计为适合特定发展模式。

从“产品”的评价角度看，其主要有 3 个部分组成：产品文档、配套的软件工具、相关培训。这些部分的可用性和有效性是评价框架的重点(特征：C5－C7)。

本文提出的评价机制分为具体质量子特征，可以应用在两个不同的层级：个人任务/工作产品或方法整体。低层级(个人任务/工作产品)机制的目标是支持针对性的方法，评价是通过分析特定的优缺点，提供方法论所需的特征；高级别(评价方法整体)的机制包括聚合低层次特征的价值，以及更广泛的覆盖通用方法(如成熟度作为 SC5.1－5.3 的反映)的评价机制，并可以被设计成具体工具，用于客观全面地比较不同的方法。

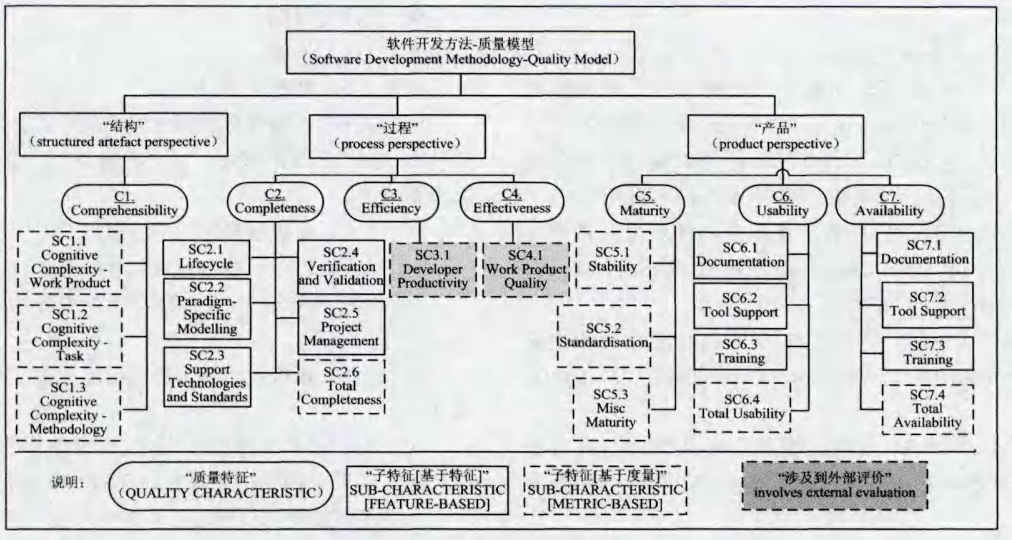

图 2　评价框架-质量模型

3.1 "结构"

本节将从结构化角度讨论产品质量的特征——C1 可理解性(C1 Comprehensibility)以及相关子特征(由 SC1. n 标记)和度量(SC1. n-Mn)。相关子特征分别描述了框架背后包含的基本原理、度量特征、粒度,以及度量的定义。需注意的是规模和粒度的所有度量都包含在一个子特征中,所包含的度量是内部类型。

3.1.1　子特征 C1 可理解性(C1. Comprehensibility)

在本文所述框架内,可理解性是指用户对方法内部组成的理解,以及对总体过程结构的分析。具体而言,该框架通过量化粒度的 3 个(产品、任务、方法)不同层面认知的复杂性作为整体(子特征 SC1. 1－SC1. 3)。

子特征 SC1. 1 复杂度-产品

一个工作产品(work product,WP)概念上的复杂性反映了封装了对象、属性和关系之间的数量。因此,本文所述框架中相关度量标准的定义如下。

度量特征:度量比例表(WP 粒度)

SC1. 1:M1. n(O):每个 WP 类型的对象类型的数量。

SC1. 1:M2. n(P):每个 WP 类型的属性类型的数量。

SC1. 1:M3. n(R):每个 WP 类型的关系类型的数量。

SC1. 1:M4. P_o:每个对象类型的属性数量。

SC1. 1:M5. P_r:每个关系的属性数量。

SC1. 1:M6. C′:WP 型的复杂性集合。

子特征 SC1. 2 复杂度-任务。

任务(T)的复杂性是对产品(WP)的输入输出和技术(Te(T))的反映。本文提出 4 个专用度量用来量化复杂性。

度量特征:度量比例表(T 粒度)。

SC1. 2:M1. BTI:基本任务交互(basic task interaction)。

$BTI=NWP_{in}(T)+NWP_{out}(T)$,$NWP_{in}$ 和 NWP_{out} 是指一个任务(T)操作产品输入输出的数量。BTI 很容易在最初的评价方法中被计算出来,它代表了任务复杂性的粗略估计。与此相反,度量 WTI 需要更多的细节工作,在涉及 WP 的复杂性估计上也更精确。

SC1. 2:M2. WTI:加权任务交互(weighted task interaction)。

$WTI=\Sigma C'(wp_{in})+\Sigma(C'(wp_{out})*F1)$,$wp_{in}\in WP_{in}(T)$,$wp_{out}\in WP_{out}(T)$。$C'(wp)$是输入输出产品的复杂性乘以(可选)任务的独特因素(F1),这个因素是必要的,因为输入(WP_{in})和输出(WP_{out})的产品类型可以被给定的任务交叉操作。这可能会降低生产者分析结构和交叉输入输出工作产品类型时的复杂性。在这种情况下,任何输出产品类型 WP_{out} 的复杂性应该匹配一个输入产品类型 WP_{in},WP_{in} 应该人为减少任务的独特因素(F1),以提供对任务复杂性更准确的评估。注意,在这个阶段的独特因素的实际值被设置为 0. 5(即 F1=0. 5)。但是这个值是暂时的,应在今后的工作实验中进行评估。

SC1. 2:M3. N(Te(T)):任务的支持技术数量(number of supporting techniques for a task)。

SC1. 2:M4. WN(Te(T)):任务的支持技术加权数(weighted number of supporting techniques for a task),$WN(Te(T))=N(Te_{qn}(T))+(N(Te_{ql}(T))*F2)$,$N(Te_{qn}(T))$是定量技术任务的数量,$(N(Te_{ql}(T))*F2)$是定性技术任务的数量乘以权重因素(F2),从而区分技术之间定性和定量的性质。这是因为定量技术适用于整个开发过程,因为它允许生产者(P)执行任务(T)和用客观统一的方式评价产品(WP)。与此不同,定性技术会对认识任务的复杂性产生负面影响,也可能对产品内部矛盾难以预测。为此,定性技术应该加权高于定量指标,以提供更精确的对任务复杂性的估计,权重目前设置为 2(F2=2)。同样,这个值是暂时的,应在今后的工作中进行评估。

子特征 SC1. 3 复杂度-评价方法

评价方法的复杂性是通过 3 个互补的度量方法:1)使用软件过程建模和评价框架提出的验证和实证评价度量,其目的是量化基于产品数量、工作产品类型和任务的软件开发方法的结构特性,以及所有任务和工作产品之间的依赖关系;2)推导复杂度综合性评价是通过整合产品类型和任务,对提出的新指标分别量化;3)使用一套新指标 SC1. 3:M11－M14 对定量和定性进行量化。

度量特征:度量比例表(M 粒度)

SC1. 3:M1. NP:生产者数量(number of producers)。

SC1. 3:M2. NWP:工作产品类型的数量(number of work

product types)。

SC1.3:M3. NT:任务数量(number of tasks)。

SC1.3:M4. $NDWPT_{in}$:所有产品任务的输入依赖数量(number of input dependencies of all WP with all T)。

SC1.3:M5. $NDWPT_{out}$:所有产品任务的输出依赖数量(number of output dependencies of all WP with all T)。

SC1.3:M6. NDWPT:所有工作产品和任务之间依赖关系数量(number of dependencies between all WP and all T),$NDWPT=NDWPT_{in}+NDWPT_{out}$。

SC1.3:M7. C′(M):评价工作产品复杂性方法的集合(total WP complexity of a methodology),$C'(M)=\Sigma C'(wp)$,$wp\in WP$。

SC1.3:M8. MWPC:平均工作产品复杂性的一种方法(mean WP complexity of a methodology),MWPC=C′(M)/NWP。

SC1.3:M9. TTI:任务交互的方法集合(total T interaction of a methodology),$TTI=\Sigma WTI(t)$,$t\in T$。

SC1.3:M10. MTI:平均任务交互的方法(mean T interaction of a methodology),MTI=TTI/NT。

SC1.3:M11. NTe:技术总数(定性和定量)(total number of(quantitative and qualitative) techniques)。

SC1.3:M12. WNTe:技术加权数(weighted number of techniques),$WNTe=\Sigma WN(Te(t))$,$t\in T$。

SC1.3:M13. TTR:技术任务比例(technique to task ratio),TTR=NTe/NT。

SC1.3:M14. TQD:技术量化程度(technique quantification degree),TQD=NTeqn / NTe,N(Teqn)是定量技术的总数。

3.2 "过程"

本节讨论软件开发方法的过程问题,提出了3个高质量的逻辑子特性:完整性(C2)、效率(C3)和有效性(C4)。

3.2.1 子特征C2 完整性(C2. Completeness)

在本文所述框架中,完整性是指为过程提供的具体范式,主要为以下阶段提供支持:1)核心开发周期阶段(SC2.1);2)范式特定建模(SC2.2);3)相关的支持技术和标准(SC2.3);4)核查和验证机制(SC2.4);5)项目管理活动(SC2.5)。注意,该特征的目的是概述一个过程评价方法,而不是提供一个详尽的所有可能功能的集合。

子特征SC2.1 生命周期

根据软件工程中关于软件生命周期的定义,软件开发的核心过程可分为若干阶段,其中包括:需求分析、架构设计、细节设计。因此,SC2.1子特征的评价支持上述3个阶段的范式。

度量特征:度量表[1(不支持)-5(全支持)](T粒度)

SC2.1:F1. 为生命周期核心任务提供支持(Tx)。此特征应用在服务识别、服务规范和服务实现阶段的每个核心任务(Tx)。

子特征SC2.2 范式特定建模

度量特征:度量表[1(不支持)-5(全支持)](M粒度)

SC2.2:F1. 提供明确的不同类型的面向对象的系统。

SC2.2:F2. 提供明确的不同类型的面向对象的关系。

根据以下特征提供明确的不同类型的服务:

SC2.2:F3. 目标。

SC2.2:F4. 构成。

SC2.2:F5. 功能范围/粒度。

SC2.2:F6. 执行服务的理念始终贯穿在开发阶段和各种活动中,包含了:概念服务->分析服务->设计服务->解决方案服务。

SC2.2:F7. 促进服务之间的松散耦合。

SC2.2:F8. 结合了技术评价和管理服务粒度。

为核心任务的定义、发布以及维护服务约定提供方法支持,包括:

SC2.2:F9. 通过使用规定格式的注册表实现服务约定的定义。

SC2.2:F10. 外部注册服务约定的发布和维护。

SC2.2:F11. 内部注册服务约定的发布和维护。

为主要服务研究和集成任务提供方法支持,包括:

SC2.2:F12. 使用语义语言描述服务接口。

SC2.2:F13. 静态评价、选择和外部服务软件集成。

SC2.2:F14. 动态发现、选择和外部服务软件集成。

子特征SC2.3 支持技术和标准

支撑现代范式的具体技术和标准可以提高软件开发方法的实用性和效率。在SOC的情况下,这样的技术和标准可以包括,但不限于ESB架构。

子特征SC2.4 核查和验证机制

评价框架系统采用正确的形式验证分析技术可以减少可靠性方面的问题。根据软件工程中关于软件生命周期的定义,开发过程应该通过需求追踪和统一设计标准保持架构的一致性。

子特征SC2.5 项目管理活动

为质量管理提供范式支持。对基于SOA的项目有3个重要的意义:1)持续集成和验证不断发展的服务系统;2)提升交付能力;3)早期检测和减少缺陷。

子特征SC2.6 总的完整性

子特征SC2.6的整体过程量化是根据子特征SC2.1-SC2.5实现的。

度量特征:度量表(M粒度)

SC2.6:M1. LSC:生命周期覆盖范围(lifecycle coverage),LSC=NST / NIT,NST和NIT的数值分别来自生命周期任务(T),一个任务被分配给相应的特征SC2.1的值,值的范围包括:[0(覆盖不足)到1(全覆盖)]。

SC2.6:M2. PSC:具体范式支持范围(paradigm-specific support coverage),PSC=NSF / NIF,NSF的值反映了评价分数4(同意)或5(强烈同意);NIF数量特征包含在子特征SC2.2(如当前NIF=14)中。

SC2.6:M3. STC:支持技术范围(support technologies coverage),STC=NSF / NIF,NSF和NIF的值包含在子特征SC2.3中。

SC2.6:M4. VVC:核查和验证范围(verification and validation coverage),VVC=NSF / NIF,NSF和NIF的值包含在子特征SC2.4中。

SC2.6:M5. PMC:项目管理覆盖范围(project management coverage),PMC=NSF / NIF,NSF和NIF的值包含在子特征SC2.5中。

3.2.2 子特征 C3 效率(C3. Efficiency)

在本文所述框架中,将方法的效率定义为多大程度上生产者的生产效率得以提高。同时,提高生产效率也是软件开发方法的核心目标之一。

子特征 SC3.1 开发人员生产力

度量特征:外部度量表(T 粒度)

SC3.1:M1. PE:生产效率(producer efficiency),PE=Σ(Time(t) /WPi_{out}(t)),t$\in$T(sp)。其中 Time(t)是表示生产一个完整软件产品分解任务所用时间的总和,WPi_{out}(t)是单位时间内产品数量的总和。PE 较低的值表示更高的效率。注意,这是为了确保 PE 测量同等规模系统时,分析和比较不同的方法值非常重要,同时应考虑工作产品质量和生产的复杂性。

3.2.3 子特征 C4 有效性(C4. Effectiveness)

在本文所述框架中,方法的有效性被定义为对生产高质量软件产品的基本过程的支持程度,以便满足用户需求和其它预定义功能和非功能约束。

子特征 SC4.1 产品质量

软件产品的质量特征显示了其结构特性的影响,因此,这有利于评价产品内外部的质量。

度量特征:外部度量表(WP 粒度)

SC4.1:M1. EQ:产品外在质量(external quality of a produced product),EQ=Σ(ExQualityFactor(wp)),wp$\in$$WPi_{out}$,ExQualityFactor(wp)是产品实例的质量特征,评价者应根据目标要求选择质量特征。

SC4.1:M2. IQ:产品的内部质量(internal quality of a produced product),IQ = Σ(InQualityFactor(wp)),wp$\in$$WPi_{out}$,InQualityFactor(wp)是工作产品实例的结构属性的测量标准,WP$\in$$wpi_{out}$是已有的软件度量。注意,WP$\in$$wpi_{out}$评价应该选择基于评价目标和要求的结构属性。

3.3 "产品"

本节将介绍质量特征及其子特征,并设计评价方法的特定产品特征,比如成熟度(特征 C5)、可用性(特征 C6)和可靠性(C7)。所有产品的相关特征都是内部类型。

3.3.1 子特征 C5 成熟度(C5. Maturity)

在本文所述框架中,方法的成熟度是指已开发的基本模型(即任务 T 和 工作产品 WP)的适用范围,包括:1)标准化(子特征 SC5.1);2)稳定性(子特征 SC5.2);3)表示各种成熟度的度量(子特征 SC5.3)。成熟度是一个非常重要的评价特征,成熟的基本模型有利于降低成本和开发难度。

子特征 SC5.1 标准化

标准化的任务(T)和工作产品(WP)可以反映一个方法的成熟度,也会提高工作质量。

度量特征:度量表(M 粒度)

SC5.1:M1. SD:标准化程度(standardisation degree),SD=(NST/NT+NSWP/NWP)/2,NST 和 NSWP 是任务 $T_{st}\subseteq$T 和工作产品类型 $WP_{st}\subseteq$WP 分别的标准化数值。NT 和 NWP 定义为子特征 SC1.3 的一部分,值为 0(缺乏标准化)或 1(完全标准化)。

子特征 SC5.2 稳定性

稳定性评价体现各评价方法之间的变化程度,稳定的方法表现出较低的变化程度和频率。

度量特征:度量表(M 粒度)

SC5.2:M1. LRCD:最新版本的变化程度(latest release change degree),LRCD=(NCT/NT+NCWP/NWP)/2,NCT 和 NCWP 分别是任务变化的数值 $T_{ch}\subseteq$T 和工作产品类型 $WP_{ch}\subseteq$WP,NT 和 NWP 被定义为子特征 SC1.3 的部分,可能的值是:0(缺乏变化)、1(充分的变化)。

SC5.2:M2. MCD:平均变化程度(mean change degree),$MCD=\sum_{i=0}^{NRV}LRCD_i/NRV$,LRCD 是子特征 SC5.2:M1 中的定义,NRV 是评价方法版本的总数量(见 SC5.3:M2),可能的值为:0(在每个版本之间锁定改变)、1(允许在每个版本之间的完全改变)。

子特征 SC5.3 混合成熟度

本项子特征不包含之前子特征的成熟度特性。

度量特征:外部度量表(WP 粒度)

SC5.3:M1. NYSR:方法论第一次公布至今的年数(number of years since the first public release of the methodology)。

SC5.3:M2. NRV:方法论版本的数量(number of released versions of the methodology)。

3.3.2 子特征 C6 可用性(C6. Usability)

在本文所述框架中,可用性指的是用户学习和应用方法的范围。本框架旨在从生产者的角度量化可用性,主要从软件开发文档、软件支持工具和培训 3 个方面进行。

子特征 SC6.1 文档

一个合适的文档可以促进方法论应用广泛影响潜在用户。文档分正式和非正式,正式文档里包含正式的语法、图形符号等。非正式文档可以是自然语言描述的形式(例如屏幕截图、图表等)和实际的例子。

度量特征:度量表[1(完全不同意)-5(完全同意)](T 和 WP 粒度)

为任务(T)和工作产品类型(WP)提供合适的文档。

SC6.1:F1. 文档的一致性。

SC6.1:F2. 文档的全面性。

SC6.1:F3. 文档的可理解性。

SC6.1:F4. 文档技术中立。

SC6.1:F5. 文档包含适当说明。

SC6.1:F6. 文档包含具体实例。

SC6.1:F7. 正式文档提供目标说明。

上述特征应单独应用到每个包含 T 和 WP 的方法,且容易被分解,并在正式和非正式的文档中根据需求重新定义。

子特征 SC6.2 支持工具

集成的支持工具(如集成开发环境、项目管理和质量保证工具)可以提高生产力,并保证任务和产品的有效性。所以,支持工具是软件开发的重要组成部分。

度量特征:度量表[1(完全不同意)-5(完全同意)](T 粒度)

SC6.2:F1. 支持工具的一致性。

SC6.2:F2. 支持工具的综合性。

SC6.2:F3. 支持工具的可理解性。

SC6.2:F4. 支持工具包含实例。

SC6.2:F5. 支持工具的目标说明。

SC6.2:F6. 支持工具可以根据评价方法选择。

子特征 SC6.3 培训

度量特征：度量表[1(完全不同意)－5(完全同意)](T 粒度)

SC6.3：F1.为每项任务提供专业的现场培训。

SC6.3：F2 为每项任务提供公众研讨。

SC6.3：F3.为每项任务提供自我引导训练。

SC6.3：F4.为每个任务提供正式的验证。

上述特征分别适用于每个方法，并且可以细化评价一致性、全面性和可理解性 3 个重要方面的培训模式。

子特征 SC6.4 可用性集合

度量特征：度量表(M 粒度)

SC6.4：M1. DUC：文档可用性覆盖(documentation usability coverage)，DUC＝(Count(NDT)/NT＋Count(NDWP)/NWP)/NSF 2，NT 和 NWP 被定义为子特征 SC1.3 的一部分，NSF 是子特征 SC6.1 评价特性的总数(目前 NSF＝7)，计数(NDT)和计数(NDWP)表明适当的文档数量。对于给定的 T 或者 WP 分别体现在 4 到 5 的值分配给子特征 SC6.1：F1－F7 的相应功能。可能的值为：0(缺乏文档)、1(完全文档)。

SC6.4：M2. TSUC：支持工具的可用性覆盖(tool support usability coverage)，TSUC＝Count(NST)/NT/NSF，NT 被定义为子特征 SC1.3 的一部分，NSF(目前 NSF＝6)是子特征 SC6.2 评价特征的总数。NSF 计数的值反映了支持工具的情况，通过分配给子特征 SC6.2：F1－F6 相应特征 4 或者 5 的值，可能的值为：0(缺乏工具支持)、1(完全工具支持)。

SC6.4：M3. TUC：培训可用性覆盖(training usability cover)，TUC＝Count(NTT)/NT/NSF，NT 被定义为子特征 SC1.3 的一部分，NSF(目前 NSF＝4)是子特征 SC6.3 评价特征的总数。NTT 计数的值反映了培训的情况，通过分配给子特征 SC6.3：F1－F4 的值 4 或者 5，可能的值是：0(完全缺乏培训)、1(完全支持培训)。

3.3.3 子特征 C7 可靠性(C7. Availability)

在本文所述框架中，可靠性是指在多大程度上评价方法被自由引用。评价方法的可靠性是基于文档、支持工具和培训的可用性量化。

子特征 SC7.1 文档

度量特征：度量表[1(完全不同意)－5(完全同意)](M 粒度)

SC7.1-F1：文档的公开和发布是非常有价值的。

SC7.1-F2：文档提供免费升级。

子特征 SC7.2 支持工具

度量特征：度量表[1(完全不同意)－5(完全同意)](M 粒度)

SC7.2-F1：公开的支持工具是非常有用的。

SC7.2-F2：支持工具提供免费升级。

子特征 SC7.3 培训。

度量特征：度量表[1(完全不同意)－5(完全同意)](M 粒度)

SC7.3-F1：免费现场培训。

SC7.3-F2：提供免费公开讨论。

SC7.3-F3：提供公开的培训资料。

SC7.3-F4：免费认证。

子特征 SC7.4 可靠性集合

本特征反映了可靠性的整体，依据是子特征 SC7.1－SC7.3。需注意度量的计算公式和容易被忽略的细节，因为它们遵循相同的模式和度量范围。

度量特征：度量表(M 粒度)

SC7.4-M1：DAC：文档可靠性覆盖(documentation availability coverage)。

SC7.4-M2：TSAC：支持工具可靠性覆盖(tool support availability coverage)。

SC7.4-M3：TAC：培训的可靠性覆盖(training availability coverage)。

4 案例应用

本节通过一个探索性的案例来说明本文所述框架的适用性，并概述了今后的研究方向。案例研究涉及了一个面向服务的建模和体系结构评价，使用了一个子集的度量特征和指标。SOMA 是分析设计和构建基于 SOA 的解决方案，在 2004 年提出的一个端到端的软件开发方法，SOMA 具有全面的文档和支持工具。

本案例研究的目的是验证本文所述框架选定部分的应用，即提供一个有意义的典范，而不是进行 SOMA 或者框架的整体综合评价。本节论证了评价子特征 SC1.2 采用两个比规模复杂性度量，并提供了一个简单例子的特征分析，采用适用于子特征 SC2.2 的 11 个建议的功能。

4.1 基于度量的评价

SOMA 的任务，“识别业务流程的能力”是服务识别阶段的一部分，也是演示目标选择一个任务的认知复杂性的量化过程，包括应用程序相关的重要因素。这项任务的特殊之处在于选择，因为它必须是易于管理，能够提供结构覆盖范围并建立计算任务复杂性的值，特别是以下构造覆盖：

〈输入〉工作产品类型(WP_{in})：

· BPMN 过程模型

· 服务依赖图(SoaML[8])

〈输出〉工作产品类型(WP_{out})：

· 服务依赖图

〈定量〉技术(TE_{qn})：

· 将给定业务分解为过程、子过程、叶级子过程 3 个层次。

〈定型〉技术(TE_{ql})。

· 消除高度抽象的过程

· 消除细粒度任务

度量计算：SC1.2：M2. WTI：加权任务交互(weighted task interaction)

$WTI=\Sigma C'(wp_{in})+\Sigma(C'(wp_{out})*F1)$，$wp_{in}\in WP_{in}(T)$，$wp_{out}\in WP_{out}(T)$。WTI 值计算的第一步是确定所有输入和输出的工作产品类型的复杂程度，这个特定的任务有两个输入 WP 型和一个输出 WP 型。

BPMN 工作产品类型概念的复杂性之前已经评估，$C'(BPMN)=93.6$。

计算服务依赖图的工作产品类型的复杂性，其元模型所有可能的对象(O)、属性(P)、关系(R)确定如下：

O＝{功能，服务接口}

P＝{功能名称，操作，操作限定符，服务接口名称}

R=｛运用，公开｝

根据以上所述，服务依赖图的工作产品类型的复杂度值计算为：

SC1.1：M1. $n(O_{ServiceDependenciesDiagram})=2$

SC1.1：M2. $n(P_{ServiceDependenciesDiagram})=4$

SC1.1：M3. $n(R_{ServiceDependenciesDiagram})=2$

SC1.1：M4. $P_o(_{ServiceDependenciesDiagram})=2$

SC1.1：M5. $P_r(_{ServiceDependenciesDiagram})=2$

SC1.1：M6. $C'(_{ServiceDependenciesDiagram})=\sqrt{(n(O)\wedge 2+n(P)\wedge 2+n(R)\wedge 2)}=4.9$

可以看到，服务依赖图概念的复杂性（C=4.9）要比BPMN的复杂性要低得多（C=93.6）。这是因为服务依赖关系图相比BPMN涵盖了数量有限的对象、属性和关系，这也符合之前由其它研究人员评价的工作产品类型复杂性计算。

总的输入和输出工作产品复杂性的计算是：$\Sigma C'(wp_{in})=C'(_{BPMN})+C'(_{ServiceDependenciesDiagram})=93.6+4.9=98.4$，$\Sigma C'(wp_{out})*F1=C'(_{ServiceDependenciesDiagram})*0.5=2.45$。

注意：权重系数作为一个输入和输出工作产品类型应用在服务依赖关系图中。加权任务交互（WTI）SOMA的任务“识别业务流程服务能力”是：WTI=98.4+2.45=100.85。

请注意，由于本研究的目的是展示选定的部分框架，我们无法得出关于复杂度值大小的比较或相对结论。

SC1.2：M4. WN(Te(T))：任务的支持技术加权数（weighted number of supporting techniques for a task），$WN(Te(T))=N(Te_{qn}(T))+(N(Te_{ql}(T))*F2)$，SOMA任务“识别从业务流程服务能力”是被一个定量（Teqn）和两个定性（TEql）技术支持，为此，WN(Te(T))的值可以计算为：1）N(Teqn)=1；2）N(Teq1)*F2=2*2=4；3）最后，SOMA的“服务功能的业务过程识别”的技术支持加权数的任务是：WN(Te(T))=1+4=5。与WTI一样，这个值的相对大小没有得出结论。

4.2 特征分析

提供完整的基于特征及子特征的分析超出了本文的范围。本文所展示的特征分析应用选择了子特征“范式的具体建模”和其相应特征（SC2.2：F1－F11）。评价结果表明整体SOMA似乎对SOC的基本概念提供了强有力的支持，然而，SOMA缺乏方法论的支持，这也说明方法改进可以对SOMA进行更好的支持。

结束语 本文提出一种全面和新颖的评价分析框架，针对SO开发方法使用定性和一组在质量模型分层结构的定量指标，其覆盖了普通的软件开发方法，主要基于3个角度：1）“结构化产品”的角度，是指作为一个构造实体结构方法过程类集合和它们之间的关系；2）“基础过程”的角度设计调整，使之适应于特定的开发范例；3）从“交付产品”的角度来看，是把产品、文档、软件支持工具和培训作为一个方法。基于一个探索性的案例，通过所提出的指标和功能，选择一个子集，通过SOMA例子来证明该框架的实用性。具体来说，具体度量的评价说明了任务复杂性的量化过程，它可以替代其他SOMA里的任务方法和其他特殊的方法。此外，结果表明，SOMA缺乏对服务发现任务方法论上的支持，从而提示可能的方法改进的领域。在可预见的未来，重要的利益相关者将对该框架进行一个详尽的评价和比较。例如，工程师可以根据理解，针对方法的优缺点选择框架的特定方法。同时，可以根据项目需求和约束来选择最合适的方法。在今后的工作中，一些目前建议的功能可以被一个更客观的评价机制所取代。

参考文献

[1] Bell M. Service-Oriented Modeling：Service Analysis，Design，and Architecture[M]. Hoboken，NJ，USA：John Wiley & Sons，Inc.，2008

[2] OMG. The Software and Systems Process Engineering Meta-Model 2.0(SPEM 2.0)[Z]. 2008

[3] ISO/IEC. ISO/IEC TR 9126-1：2001 Software Engineering：Product quality -Quality model[C]// International Organization for Standardization / International Electrotechnical Commission. Geneva，2001

[4] ISO/IEC. ISO/IEC TR 9126-3：2003 Software Engineering：Product quality -Internal metrics[C]// International Organization for Standardization / International Electrotechnical Commission. Geneva，2003

[5] ISO/IEC. ISO/IEC TR 9126-2：2003 Software Engineering：Product quality -External metrics[C]// International Organization for Standardization / International Electrotechnical Commission. Geneva，2003

[6] ISO/IEC. ISO/IEC 25000：2005 Software Engineering：Software product Quality Requirements and Evaluation(SQuaRE)—Guide to SQuaRE [C] // International Organization for Standardization/International Electrotechnical Commission. Geneva，2005

[7] ISO/IEC. ISO/IEC 12207：2008 Systems and Software Engineering-Software life cycle processes[C]// International Organization for Standardization/International Electrotechnical Commission. Geneva，2008

[8] 安金霞，王国庆，李树芳，等. 基于多维度覆盖率的软件测试动态评价方法[J]. 软件学报，2010(9)：2135-2147

[9] 胡红雷，毋国庆，梁正平，等. 软件体系结构评估方法的研究[J]. 计算机应用研究，2004(6)：11-15

[10] 马飒飒，陈自力，赵守伟. 软件可靠性定量评估过程研究[J]. 计算机测量与控制，2005(5)：503-505

[11] 余为峰，黄松. 软件质量度量分析与研究[J]. 电脑知识与技术，2010(6)：5106-5108

[12] 宋丹辉. 本体评价若干问题研究[J]. 图书馆学研究，2011(9)：6-10

[13] 赵巾帼，徐德智，罗庆云. 本体论及其应用研究[J]. 四川理工学院学报，2007(6)：102-106

[14] 袁玉宇. 一个实用的软件质量评估模型[J]. 计算机工程，2003(5)：32-34

[15] 徐鹏，杨放春. 建模样式：一种评估软件体系结构非功能属性的方法[J]. 软件学报，2006(6)：1318-1327

[16] 蔡旭辉，晏海华，柳永坡. 一个改进的软件过程工程元模型ISPEM [J]. 沈阳航空工业学院学报，2005(1)：41-47

[17] 黄斌，张佳祺. 面向对象的软件过程SPEM模型的地位与作用[J]. 现代计算机：专业版，2005(4)：12-18

[18] 田立，曾光裕，于磊，等. 支持CMMI的软件过程元模型的研究与实现[J]. 计算机工程与设计，2010(18)：3983-3988