ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.19, No.6, June 2008, pp.1328−1339
DOI: 10.3724/SP.J.1001.2008.01328

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

软件体系结构评估技术*

张 莉+, 高 晖, 王守信

(北京航空航天大学 软件工程研究所,北京 100083)

Software Architecture Evaluation

ZHANG Li+, GAO Hui, WANG Shou-Xin

(Software Engineering Institute, BeiHang University, Beijing 100083, China)

+ Corresponding author: E-mail: lily@buaa.edu.cn

Zhang L, Gao H, Wang SX. Software architecture evaluation. *Journal of Software*, 2008,19(6):1328−1339.
http://www.jos.org.cn/1000-9825/19/1328.htm

Abstract: Software architecture evaluation is an important technology used to assure the quality of software products early in the software lifecycle. This paper classifies three types of software architecture evaluation methods: scenario-based, metric and prediction based, and ADL-based. Software architecture evaluation method characteristics (such as method goal, quality attribute, key technique) are then combined with these classifications to produce a comparison framework. This paper utilizes this framework to analyze various existing software architecture evaluation methods and point out problems which need to be resolved. Finally, potential research directions of software architecture evaluation methods are discussed.
Key words: software architecture; software architecture evaluation; software quality

摘 要: 作为在软件生命周期早期保障软件质量的重要手段之一,软件体系结构评估技术是软件体系结构研究中的一个重要组成部分.将现有的软件体系结构评估方法划分为 3 类:基于场景的评估方法、基于度量和预测的评估方法以及特定软件体系结构描述语言的评估方法.按照软件体系结构评估技术的评价框架,分别从评估方法的目标、质量属性、关键技术等方面对这 3 类方法的特点进行介绍和对比.最后分析了现有研究中存在的不足并进一步探讨了软件体系结构评估技术的研究趋势.
关键词: 软件体系结构;软件体系结构评估;软件质量
中图法分类号: TP311 **文献标识码**: A

软件体系结构本质上并不是一个全新的概念,早在 20 世纪六七十年代,Dijkstra 和 Parnas 等人对软件设计基本原则的研究就是对软件体系结构的最初认识[1].自 20 世纪 90 年代,Perry & Wolf 和 Garlan & Shaw 分别提出了软件体系结构的概念[2,3]以来,软件体系结构得到了学术界和工业界的广泛重视.虽然 10 多年来出现了上百个关于软件体系结构的定义(详见 CMU-SEI 网站的收集),但是在以下方面的认识是一致的:1) 软件体系结构是软件系统的结构(或组织),包括构件(组成元素)、构件的外显特性和构件之间的关系;2) 软件体系结构是软件

* Supported by the Program for New Century Excellent Talents in University of China (新世纪优秀人才支持计划); the National Basic Research Program of China under Grant No.2007CB310803 (国家重点基础研究发展计划(973))
Received 2006-10-02; Accepted 2007-06-29

开发过程早期的一项软件制品,是一组使系统满足系统涉众(stakeholder)功能和非功能需求的设计决策.

Pressman 在他的书中这样写道“软件工程方法的唯一目标是:生产高质量的软件”[4].软件质量从软件工程的诞生起就一直得到广泛关注,但是人们对软件质量的认识却在发生变化,其内涵愈加丰富,从功能属性、性能属性,到可移植、可扩展、易用、易维护、安全、可靠等诸多非功能质量属性;从满足用户需求到满足系统涉众的需求.如何在一个系统中平衡诸多属性(这里,将系统对需求的满足也作为系统可表征的属性),给系统开发方提出了挑战.“缺陷放大模型”以及业界大量的统计数据表明[4]:修正软件缺陷的成本随着发现该缺陷的时间推迟而增长,而且 50%~75%的缺陷是设计阶段注入.软件体系结构评估的目的就是为了在开发过程的早期,通过分析系统的质量需求是否在软件体系结构中得到体现,识别软件体系结构设计中的潜在风险,预测系统质量属性,并辅助软件体系结构决策的制定.

20 世纪 90 年代以来,软件体系结构的评估技术一直是研究的热点问题.软件体系结构的评估技术不断出现,一些方法已经比较成熟并得到了应用和验证,如基于场景的软件体系结构分析方法——SAAM[5]、软件体系结构折中分析方法——ATAM[6]、利用软件性能工程 SPE 对软件体系结构进行评估的 PASA 方法[7,8]、软件体系结构层次可维护性预测方法——ALMA[9,10]等.近年来,随着技术和应用经验的积累,一些新的评估方法也逐渐出现,比如基于贝叶斯信念网的软件体系结构评估方法[11,12]、软件体系结构度量方法[13,14]等.

本文的主要目的就是系统和全面地分析软件体系结构评估方面的研究,并分析其发展趋势.为了便于分析和比较各种评估技术,第 1 节首先给出软件体系结构评估技术评价框架.第 2 节根据比较框架的内容分析各种软件体系结构评估技术.第 3 节总结各种评估技术的研究思路以及存在的局限性,并指出评估技术的发展方向.

1 评价框架简介

为了系统地对各种软件体系结构评估技术进行说明和比较,需要一个可操作、全面的评价框架.在对软件体系结构评估技术的分析和调查方面,下面两个工作是非常重要并且具有参考价值的:(1) Dobrica 和 Niemela 于 2002 年首次提出了对软件体系结构分析方法进行刻画和比较的框架[15],并利用该框架对当时的各种软件体系结构的评估方法进行分析;(2) Babar 等人于 2004 年扩展和细化了 Dobrica 的框架[16,17],详细描述了框架中各个元素,使分析更加具有可操作性,并在此基础上分析和比较了各种基于场景的评估方法.AliBarba 框架包括了 Dobrica 框架中的所有评价项,如图 1 所示.详细说明见文献[15,16].

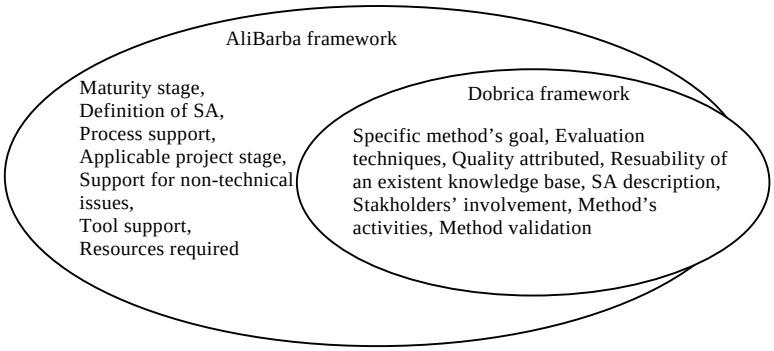


Fig.1 The AliBarba’s and Dobirca’s software architecture comparing framework

图 1 AliBarba 框架与 Dobrica 框架的比较

随着软件体系结构评估技术的发展,除了基于场景的评估方法,还出现了基于度量-预测的评估方法和基于某种特定软件体系结构描述语言(ADL)的分析方法.为了更好地比较和分析不同的软件体系结构评估技术,本文对 Dobrica 和 AliBarba 两种评价框架中的评价项进行了适当的调整和扩充.表 1 给出了本文使用的评价框架以及对各评价项的简要说明.

Table 1 The Comparing framework for the software architecture evaluation methods

表 1 软件体系结构评估方法的评价框架

Evaluation item	Description
SA description	What is the SA description requirement of the evaluation method?
Evaluation goal	The goal achieved or the output of the evaluation method
Quality attribute	What extent does the evaluation method support attributes?
Key technologies	The primary technologies or the theoretic model of the evaluation method
Experience repository support	Does evaluation method recommend any experience repository? What is the level of reuse of the knowledge and the experience?
Process support	Does the evaluation method support the development or evaluation of the process which include evaluation activities, the role of the participant involving the process?
Maturity stage	What is the level of maturity.
Applicable project stage	Which is the most appropriate development phase to apply the method?
Tool support	Does the method provide or recommend any tool for all or some of the tasks?
Resources required	Resources required by the evaluation method, including human-resource, time, and the quality of the participant

2 软件体系结构评估技术概览

在文献[15,17]中,研究人员已经对一些重要的软件体系结构评估技术进行了分析和说明.随着时间的推移,新的软件体系结构评估技术不断涌现.本文根据各种评估技术的主要特点,将评估技术分为 3 类:基于场景的评估方法、基于度量-预测的评估方法和基于特定软件体系结构描述语言的分析方法.本节分别对这 3 类方法的特点进行分析和说明.

2.1 基于场景的评估方法

该类评估方法的基本观点是,大多数软件质量属性极为复杂,根本无法用一个简单的尺度来衡量.同时,质量属性并不是处于隔离状态,只有在一定的上下文环境中才能做出关于质量属性的有意义的评判[6].利用场景技术则可以具体化评估的目标,代替对质量属性(可维护性、可修改性、健壮性、灵活性等)的空洞表述,使对软件体系结构的测试成为可能[6].所以,场景对于评估具有非常关键的作用,整个评估过程就是论证软件体系结构对关键场景的支持程度.通过对多种基于场景的评估方法的分析,我们认为该类方法具有以下重要的特征:(1) 场景是这类评估方法中不可缺少的输入信息,场景的设计和选择是评估成功与否的关键因素;(2) 这类评估是人工智力密集型劳动,评估质量在很大程度上取决于人的经验和技术.

基于场景的评估方法是研究最广泛,应用最成熟,数量最多的一类软件体系结构评估方法.本文调查了 9 种基于场景的评估方法,分别是基于场景的软件体系结构分析方法(SAAM)[5]、基于复杂场景的 SAAM (SAAMCS)[18]、基于领域的 SAAM(ESAAMI)[19]、软件体系结构折中分析方法(ATAM)[6]、针对演化和重用的 SAAM(SAAMER)[20]、设计中间产品积极评审方法(ARID)[6]、软件体系结构层次的可更改性分析方法(ALMA)[9,10]、基于模式软件体系结构评估方法(PSAEM)[21]、基于方面的 SAAM(ASAAM)[22]、软件体系结构层次的可用性分析方法(SALUTA)[23].利用第 1 节中的评价框架,针对基于场景的评估方法的特点,本文从软件体系结构描述、评估目标、质量属性、关键技术、过程支持、资源需求等几个方面对这类方法进行分析和阐述.

2.1.1 软件体系结构描述

软件体系结构描述是软件体系结构评估的前提和基础.基于场景的评估方法对软件体系结构的具体表现形式没有严格的限制,不依赖于特定的软件体系结构描述语言,但是,利用软件体系结构描述语言建立的软件体系结构模型应该为评估提供完备和准确的信息,以保证评估师能够准确理解设计师的设计策略.利用易于理解、标准的描述方法(比如 UML 等标准建模语言[24])可以促进设计师和评估师之间的交流.

2.1.2 评估目标

基于场景的评估方法主要有以下几种目标:(1) 评估软件体系结构是否满足各种质量属性的要求;(2) 比较不同的软件体系结构方案;(3) 进行风险评估.

该类方法的评估结果大多以评估报告的形式给出,根据评估目标的不同,不同的评估方法给出的报告内容各异.一般的评估报告包括软件体系结构模型对所评估质量属性的满足程度,更进一步地,通过专家的分析,在报告中还可以给出软件开发中可能存在的风险,有时甚至包括软件体系结构设计的改进建议等.

2.1.3 质量属性

适用于各种质量属性的评估是一种通用的评估方法.在理论上,只要能提供适当的场景描述,该类方法就可以评估软件的大部分质量属性.因此,场景的选择和分析成为各类质量属性评估的关键因素.

2.1.4 关键技术

场景获取技术和场景分析技术是基于场景的评估方法中最为关键的两项技术.下面分别对其进行分析和说明.

- 场景获取技术

在基于场景的评估方法中,利用场景来具体化评估目标,因此,场景获取是明确评估目标的重要环节.场景获取最基本的方法就是让在项目涉众进行头脑风暴,如在 SAAM 和 ATAM 方法中利用问题清单等方式启发评估人员获取场景.在头脑风暴的基础上,为了对场景进行积累和重用,ESAAMI 方法强调了场景的领域特性,通过领域分析增加领域知识,积累分析模板,提高在领域内对场景的重用和获取.PSAEM 方法则从系统设计的角度提出了一种基于模式的场景提取技术,将软件体系结构模式和设计模式中包含的通用行为作为评估的场景,从而得到了通用的场景模式,可以在不同项目评估中得到重用.

为了尽可能地平衡候选场景的完整性和关键性,研究人员提出了场景的等价类选择技术,该技术将所有场景划分为等价的组,然后从每组中抽取一个场景进行评估,从而避免重复评估类似的场景,减少评估成本.ALMA 首次提出并应用了该技术.

另外,针对各种质量属性的不同特点,也提出了具有一定针对性的场景获取技术.比如:ASAAM 方法借用面向方面编程技术中的“方面(aspect)”概念定义了“方面场景”,用于说明对系统中的很多构件产生“横切”影响的场景,如在窗口管理系统中“将系统移植到其他操作系统”的场景就是一个典型的关于“操作系统”方面场景.为了对这种方面场景进行提取,ASAAM 中提供了一套启发式规则在普通场景集中提取“方面场景”.SAAMCS 中,则利用软件变化类型(需求变化、质量要求变化、构件变化、技术环境变化)以及场景对软件的影响程度(无影响、影响一个构件、影响多个构件、影响软件体系结构)来指导场景的选择.

- 场景分析技术

采用评审会议的方法进行场景分析是最基本的分析方法,利用该方法评估人员可以得到软件体系结构对各场景的满足程度,可以比较多个软件体系结构方案.SAAM,ATAM 等方法都基于这种人工评审的技术.这种技术是基于场景评估方法中的主流技术.但人工评审从效率和精确性上都有一定的欠缺,所以研究人员也在利用一些自动分析的方法,对场景进行模拟执行,通过模拟数据来说明软件体系结构是否满足场景的要求[7,8].

在场景分析中,对于不同质量属性的综合分析也是一项非常重要的技术,其中最有影响的研究是 ATAM 方法中引入效用树技术来支持对多属性进行折中分析的能力.效用树描述了质量需求与设计之间的关系以及质量需求之间的优先关系,可用于划分和组织场景.

2.1.5 评估过程支持

基于场景的评估方法都定义了评估的步骤.其中两个最重要的步骤是:开发场景和评估场景.开发场景包括场景的收集和选择,其目的是为了根据不同的评估目标选择出能够体现相应软件质量属性的关键场景;评估场景则是评价软件体系结构是否能够直接或间接的支持场景所体现的质量属性.但严格地说,评估步骤的定义还不足以说明评估过程的支持.完整的过程定义应该不仅对评估步骤,而且对各个步骤的参与者、输入输出制品进行定义.目前,ATAM 方法对评估过程的描述最为完整,其评估过程定义为 4 个阶段,共 9 个活动,并详细定义了各个活动的参与者与各个活动需要的和产生的各种软件制品.

2.1.6 资源需求

基于场景的评估方法在评估期间对人力资源的要求相对较高.根据 CMU-SEI 的数据[6],采用 ATAM 进行中

等规模评估的粗略成本是70人天,进行小规模评估的粗略成本也需要32人天.该类评估涉及的人员包括:用户、软件设计开发者、软件测试者等,几乎软件开发过程中涉及到的各种角色都会参与评估,所以,组织和协调这样的评估会议的代价也是比较高的.另外,基于场景的评估要求参与人员,特别是评估师具有较高的专业素质.

2.1.7 其他方面

被调查的基于场景的评估方法都是依靠相关领域的专家完成场景抽取、软件体系结构分析等关键步骤,所以评估结果主要依赖于评估专家的实践经验和分析经验,这些方法的执行并不依赖于计算机辅助工具(虽然SAAM提供了一种SAAMTOOL[25]对评估过程进行支持).该类方法可以应用在软件体系结构设计中的各个阶段,只要该阶段能够提供评估所需的信息(主要是场景和软件体系结构模型).另外,根据成熟度,这些方法可以分为两大类:第1类方法已经基本成熟,包括ATAM,SAAM,SAAMER等,这些方法都已经过多年的应用验证;第2类是一些新出现的方法,包括PSAEM,ASAAM,SALUTA.这些方法出现在近两年的文献中,在有效性、实用性方面还需要进一步地检验和加强.

2.2 基于度量和预测的评估方法

测量对于所有科学领域的进步都是至关重要的.在软件工程领域中,软件的度量和预测技术是保证软件质量的重要技术之一.软件体系结构作为软件开发过程中一个早期的设计模型,如果能够度量并预测未来软件产品的质量,那么其预测的结果可以及时给出设计缺陷,这对于减少开发风险和提高软件质量是非常重要的.根据这一思路,出现了一类基于度量和预测的评估方法.

软件体系结构的度量是对软件中间产品的度量,可以更加精确地描述软件体系结构的各种特征;并通过预测去发现软件设计中存在的问题.通过对多种基于度量和预测评估方法的分析和比较,我们认为,该类方法具有以下的重要特征:(1) 这些方法的基本思路是将传统的度量和预测技术应用在软件体系结构层次;(2) 度量技术需要软件体系结构提供比较细粒度的信息,对模型的要求比较严格;(3) 利用度量技术对软件体系结构模型的内部特征(如复杂性、内聚度、耦合性等)进行测量;(4) 利用这些度量作为预测指标,对某些软件的外部质量(如可维护性、可演化性、可靠性等)进行预测,但由于预测模型构造的困难,所以这些预测一般只作为一种辅助评估的手段.

本文重点调查了7种基于度量和预测的评估方法,分别是软件体系结构评估模型(SAEM)[15]、软件体系结构性能评估方法(PASA)[7,8]、基于贝叶斯网的软件体系评估方法(SAABNet)[11,12]、软件体系结构度量过程[13]、软件体系结构变化的度量方法(SACMM)[14]、软件体系结构静态评估方法(SASAM)[26]、软件体系结构层次可靠性风险分析方法(ALRRA)[27](注:由于PASA中也利用了场景技术,所以有些研究人员将这种方法作为基于场景的方法进行分析[17]).利用第1节中的评价框架,针对基于度量和预测评估方法的特点,本文从软件体系结构描述、评估目标、关键技术、工具支持、资源需求这几个方面对这类方法进行了分析和比较.

2.2.1 软件体系结构描述

基于度量和预测的评估方法需要软件体系结构模型提供的信息是完整、无二义和一致的.完整性是指针对不同的度量和预测技术,软件体系结构模型提供的信息应该能够保证度量的计算,也就是说,信息是足够的;无二义性是指对软件体系结构模型的语义是清晰的,不存在不同的理解;一致性是指对于不同视图中的相同模型元素应该具有相同的性质.根据这些要求,软件体系结构描述语言最好采用形式化语言或由半形式化的UML语言扩展得到.在本文调查的方法中,大部分方法使用了UML语言并对其进行了扩展.如:PASA要求利用Kruchten提出的“4+1”视图模型[28]对软件体系结构进行描述,并对UML元模型进行扩展以适应性能评估的要求.其中,利用消息顺序图(扩展的顺序图)对性能场景进行描述;利用扩展的部署图来描述运行环境的特征.ALRRA明确提出软件体系结构描述语言必须能够描述构件间交互关系和独立构件的行为特征,ALRRA中采用了扩展UML得到的ROOM(实时面向对象建模语言)[29]中的顺序图和状态图对系统的动态行为进行描述,并且,ROOM模型可以模拟执行.SACMM则使用了形式化的方式定义了带标签的图结构,并利用这种结构给出软件体系结构,然后可以计算出变化的大小.

值得注意的是,虽然度量和预测技术本身是不依赖于语言的,只需要有合适的语言就能够给出完整、无二

义、一致的模型,但该类方法大多有自动化工具支持,而工具的实现则是需要依赖于某种具体语言的.

2.2.2 评估目标

基于度量的评估方法主要有以下几种目标:(1) 通过精确的度量,可以评估软件体系结构层次上的内部质量特征;(2) 利用预测模型可以评估软件的外部特征;(3) 可以进行风险评估.

2.2.3 关键技术

在基于度量和预测的评估方法中,度量技术解决的是各种因素的可测量问题,而预测技术解决的是各种因素之间的相关性问题,这两项技术都是评估的基础问题.

- 度量技术

根据度量对象的不同,度量技术可以分为两类:第 1 类是软件体系结构模型的度量技术;第 2 类是对各种质量属性在软件体系结构层次的度量技术.

首先,软件体系结构模型的度量技术主要是对软件体系结构模型的结构特征和行为特征进行度量,如:结构复杂度、结构形态、行为复杂度等.如 ALRRA 方法为了进行可靠性分析,提出了包括构件操作复杂度、连接件输出耦合度等一组软件体系结构模型的度量技术.

另外一种度量技术则是对性能、可靠性、可维护性等质量属性在软件体系结构层次的量化形式进行研究.如 PASA 方法就定义了计算机资源需求、作业驻留时间、利用率、吞吐量、队列长度等度量,对性能进行量化的表示;SACMM 方法为了对软件可更改性及软件体系结构的演化性进行评估,利用 graph kernel 函数定义距离度量对软件体系结构的相似性进行量化;ALRRA 方法提出与故障相关的一组度量,包括构件和连接件的故障模式、故障严重性级别.通过复杂性度量和故障严重性级别,可以计算可靠性风险因子.SACMM 方法利用距离和相对距离度量,建立软件体系结构转换模型,该模型表示了软件开发过程不同软件体系结构的差异,具有计算效率高、简单易用的特征.

- 预测技术

预测技术是在度量技术的基础上进一步研究在软件体系结构层次上各种因素之间的关系.利用这些经验关系,可以通过一些在软件体系结构层次上的特性(如结构和行为特性等)来预测未知的软件质量特性(如可靠性、可维护性等).预测技术的研究需要比较深入的理论知识,需要积累大量的经验数据,这些都是软件工程研究中的难点问题,但如果该技术得到突破,将对评估技术自动化方面的研究产生重要的影响.

目前,该技术已经取得了一些成果,如在 ALRRA 方法中,利用复杂性度量和故障严重性级别,建立了基于构件依赖图(CDG)的模型,并利用风险分析算法对 CDG 模型进行计算,可以评估软件体系结构的可靠性风险;SAABNet 方法则利用贝叶斯信念网构造了软件体系结构策略到软件质量的因果关系模型,可以通过设计策略对各种质量属性进行预测.

2.2.4 工具支持

这类方法大多数都提供了相应的工具,这些工具的主要功能有两点:通过软件模型得到相关的度量值,利用相关算法计算软件特性的预测值.在我们调研的方法中,有 PASA 方法提供的性能评估工具 SPE.ED[30],它将关键场景转化为 SPE 模型,工具 SPE.ED 读入 SPE 模型,并通过分析和模拟发现性能问题.PASA 综合利用了场景、模拟和分析的技术,对性能进行评估,是一种半自动化的评估方法.SACMM 则实现了 Kernel 计算模块和体系结构抽取模块.ALRRA 方法利用 Objectime 工具[31]对软件体系结构模型进行模拟,实现对动态复杂性的度量.SAABNet 利用通用的贝叶斯网工具 Hugin[32]对因果关系模型进行推理计算得到预测结果.没有工具的支持,度量的工作量是非常大的.工具的应用使这类方法评估效率大大提高,并且得到精确的评估结果.

2.2.5 资源需求

这类方法一般都有工具的支持,所以在评估时可以利用工具自动获取评估结果,因此,评估时的人力成本主要是将软件体系结构模型数据输入到相应的工具中,实现自动化评估.如果能将设计时的模型直接导入评估工具,则人力成本更低.这类方法实现评估时的低成本需要一个基本条件就是已经开发或购买了相关的评估工具,但现时的实际情况是,评估工具并不通用,所以在选择这类方法时常常还需要一定的开发成本.这就带来了较大

的间接成本.

2.2.6 其他方面

这类方法在工具支持下对资源的需求是较低的.这类方法试图将专家的经验都形式化为数学模型,评估实施人员只需收集数据,再利用自动化工具就可以得到评估结果.该类方法一般要求软件体系结构模型能够提供较多的信息,所以该类方法一般应用于软件体系结构设计的后期.在成熟度方法,PASA 是比较成熟的一种方法,在实时、Web、分布式应用系统中都得到应用和验证;而其他方法则还处于研究阶段,其度量指标和预测模型都需要随着应用得到验证和调整.

2.3 基于特定软件体系结构描述语言的评估方法

基于特定软件体系结构描述语言(ADL)的评估方法是一类比较特殊的方法,这类方法依赖于某种具体的软件体系结构描述语言,一般是软件体系结构语言研究的附属品.通过对多种基于特定软件体系结构描述语言的评估方法的分析和比较,我们认为该类方法具有以下的重要特征:(1) 评估技术与特定软件体系结构描述语言的定义机制和理论基础密切相关;(2) 软件体系结构描述语言的定义非常严格,通常是形式化或半形式化的描述语言.

本文调查了 3 种这类方法,分别是特定领域软件体系结构分析方法(DSSA)[33,34],Rapide 方法[35],以及基于 UML 逆向工程的软件体系结构分析方法[36].利用第 1 节中的评价框架,针对基于特定软件体系结构描述语言评估方法的特点,本文从软件体系结构描述、质量属性、关键技术、工具支持、资源需求这几个方面对这类方法进行分析和比较.其余的评价项在“其他方面”中进行简单说明.

2.3.1 软件体系结构描述

基于特定 ADL 的评估方法使用的软件体系结构描述语言通常是具有形式化基础的语言或半形式化的描述语言.所以,这类语言具有可执行、可自动分析的能力,这就为软件体系结构的评估奠定了基础.在我们调研的方法中,DSSA 方法使用了 Meta-H 语言描述软件体系结构模型.Meta-H 是由 Honeywell 公司定义的,支持对可靠性和安全性要求较高的多处理器实时嵌入式系统的创建、分析和验证,主要适用于航空电子控制软件系统.Rapide 是由美国 Stanford 大学的 Luckham 等人定义的一种可执行的软件构架定义语言,主要适用于对基于事件的、复杂、并发、分布式系统的构架进行描述.在基于 UML 逆向工程的软件体系结构分析方法中,利用 UML 外廓机制定义特定领域的软件体系结构概念模型和约束模型.

2.3.2 质量属性

在基于特定 ADL 的评估方法中,可以评估的质量属性是受到其语言特征和形式化理论基础的限制的.一般,该类方法用于评估特定领域软件系统的性能、可靠性、安全性、事务性等质量属性.

2.3.3 关键技术

各种特定的 ADL 语言具有不同的形式化理论基础,其评估技术依赖于所采用的理论模型.比如,DSSA 中的 MetaH 语言的语义主要基于形式化调度和数据流模型,所以可以通过对软件体系结构的模拟执行,得到以下分析结果:(1) 时间性能:主要分析进程时间的限制、各子系统执行时间统计和交互次数等;(2) 可靠性:通过生成马尔可夫可靠性模型对缺陷发生和传播进行分析;(3) 安全性.通过这 3 种分析,可以评估系统稳定性、性能、鲁棒性、可靠性、安全性等质量属性.Rapide 中构件的计算和交互语义通过偏序事件集(称为 posets)定义,通过抽象状态和状态转移规则来定义该类构件的行为约束,所以通过模型执行和形式化检查的方法检测体系结构参考模型的动态结构.可以分析软件体系结构的一致性,检查事务的原子性、一致性、隔离性、持久性(ACID).

另外,在基于 UML 逆向工程的软件体系结构分析方法中,通过一种结合自顶向下(已知的软件体系结构)和自底向上方法的逆向工程构造软件实现模型与软件体系结构模型之间的关系;利用概念模型和约束模型进行软件体系结构的结构分析,首先根据约束模型对软件体系结构模型进行验证,然后,利用原子操作(增加、删除和合并等)对不同的软件体系结构模型进行比较.可以对软件的可维护性进行评估.

2.3.4 工具支持

这 3 种方法都有工具的支持:由 Honeywell 公司开发的工具可以支持 DSSA 中时间性能分析、可靠性分析、

安全性分析;Rapide项目中的工具可以支持模型的执行和分析;基于UML逆向工程的软件体系结构分析方法提供的 artDECO[37,38]是一种基于 UML 外廓的软件体系结构验证工具.

2.3.5 资源需求

这类方法比较特殊,相应的方法一般都有工具支持,所以在评估时成本是很低的.但由于这些工具都没有商业化,还只是实验室产品,所以一般很难得到.

2.3.6 其他方面

这些方法成熟性较低,都只在较小范围内进行验证.DSSA 由商业机构开发,主要用在航空电子控制软件系统,所以其带有很强的领域性;而 Rapide 方法和基于 UML 逆向工程的软件体系结构分析方法都没有经过大规模应用的验证.该类方法的软件体系结构模型由特定的语言描述,所以只要能够建立模型即可进行评估.

3 研究总结

3.1 软件体系结构评估方法的技术特点比较

软件体系结构评估技术是为了在软件设计的初期发现与软件质量相关的问题.自从 1993 年 Carnegie Mellon 大学软件工程研究所提出基于场景软件体系结构分析方法以来,从学术界和商业界涌现了很多软件体系结构评估技术.从第 2 节的分析中可以看出,不同类型评估方法呈现出不同的技术特点.下面从评估方法的研究思路、主要研究内容、适用环境、典型代表这几方面对本文中的 3 类评估方法在其技术特点方面进行比较,见表 2.

Table 2 Comparison of different types of software architecture evaluation methods

表 2 不同类型软件体系结构评估方法技术特点比较

Comparison item	Scenario-Based evaluation method	Measurement-Based and forecast-based evaluation method	Evaluation method of specific ADL based on
Research approach	Enable users gain the evaluation goal through the following approach: 1. Improve or enable the operational ability by the way by the aid of scenario description. 2. Standardize the evaluation process and implement step.	Research the characteristic of SA or the intrinsic relationship between SA and software quality in order to enabling users predict the software quality based on the experience data or knowledge	Restricted analysis of SA with formal technology through the deeply research of ADL
Primary research content	1. Technologies of achieving scenario 2. Definition of evaluation process and implement step 3. Technologies of evaluating and analysis of cost-benefit	1. SA quality model 2. Architecture-Level measurement method 3. Architecture-Level quality predict-tion method 4. Automatic tool support	1. The formal base of SA 2. The formal analysis technology
Applied context	1. These methods are suitable to many kinds of project, some of them already have been applied abroad. 2. One of the key factor of these methods is the aidance of experien-tial people.	1. The kind of methods have been applied in the domain of research due to the low level of maturity of methods 2. Requiring evaluating engineering possessing well theoretic knowledge 3. Requiring supporting of the automatic tool	1. These method are restricted to specific project modeled with certain ADL and are applied in the specific domain. 2. Requiring evaluating engineering possessing well theoretic know-ledge 3. Requiring supporting of the automatic tool
Representational example	ATAM, SAAM, etc	SA assessment method based on Bayesian Network Software architecture-level reliabili-ty anal analyse method	DSSA, Rapid, etc

3.2 软件体系结构评估技术研究中存在的不足

软件体系结构是软件工程中比较新的一个领域,对其定义、描述方法等方面的研究都还处于不断发展的过程中.虽然软件体系结构评估技术在十几年的发展中取得了较大的进步,但现有的软件体系结构评估技术仍然存在一些局限性.

3.2.1 基于场景评估的方法的不足

- 评估的效果对评估师经验的依赖程度较高

这类方法的基础建立在人的智力水平上,其效果主要取决于评估师的经验和知识.在 ATAM,SAAM 及其扩展方法中,无论从场景的选取、软件体系结构的分析、评估的组织和控制都需要评估师发挥重要的作用.在评估过程中,评估师的经验对评估的效果至关重要.一个好的评估师可以准确地发现软件体系结构设计中存在问题;而当评估师经验不足时,评估则很难发挥其应有的作用.所以,评估的效果取决于评估师的经验.

- “重量级”的评估技术,成本较高

基于场景的评估方法规定的步骤比较复杂,参与人员较多,需要组织专门评估会议.在较大规模的软件项目中是比较适用的,但需要的成本也较高.如:采用 ATAM 进行中等规模评估的粗略成本是 70 人天;进行小规模评估的粗略成本也需要 32 人天.然而,对于中小规模的软件项目,由于预算和时间的限制很难按照这些“重量级”的评估方法进行实施.因此,有必要研究适合中小规模软件项目的“轻量级”评估技术.

- 没有考虑知识的积累和应用问题,造成资源的浪费

在软件工程研究中,知识的积累和应用是一个非常值得注意的问题.知识的积累和应用为软件技术的发展起着极大的推动作用.在经过一次又一次的软件体系结构评估后,是否能够积累足够的知识并让这些知识可以被重复利用是一个非常重要的问题.在基于场景的评估方法中,对知识积累问题考虑不足.已有研究表明,有一些研究人员已注意到这个问题[21],如 PSAEM 方法等.但在软件体系结构评估的知识积累和应用方面还没有形成比较系统的研究成果.

- 缺乏实用的评估信息管理工具

利用信息管理系统可以管理评估相关的各种信息,可以对评估过程进行控制和跟踪.工具还可以帮助评估人员提高评估效率,进一步规范评估过程的实施,提高评估的实施效果,并有利于积累评估的各种信息.现在仅有 SAAM 提供了 SAAMTool 对评估过程进行支持,而该工具目前只是实验室产品,还没有出现真正的商业化和实用化的评估信息管理工具.

3.2.2 基于度量和预测的评估方法的不足

- 度量技术和预测模型还不够成熟

在软件体系结构评估中,采用的度量技术和预测模型多数还来自于传统的软件度量和预测方法.由于这些技术在软件体系结构层面的应用还没有经过充分的验证,所以,这些方法和模型是否符合软件体系结构层次的评估还有待进一步研究.另外,基于贝叶斯网的评估(预测方法)在软件工程中的应用最早由 Fenton 提出[39,40],用于代替传统的回归模型解决缺陷预测问题.Fenton 从理论上比较了因果关系模型(可以贝叶斯网表示)与传统预测模型之间的优劣,说明了因果关系模型在软件度量和预测技术研究中的作用.但现在的评估技术研究中对这一新的研究成果关注不够.在我们的调研中,只有 Bosch 提出了基于贝叶斯网的软件体系评估方法(SAABNet)[12],该研究通过专家的经验,人工构造了一个表明设计策略与质量特征之间关系的模型,没有真正揭示各种软件体系结构的特征与软件质量特征之间的关系,也没有解决贝叶斯网应用中最困难和重要的问题——如何获取经验数据并且利用数据构造贝叶斯模型.

- 实用化程度低

实用化程度低表现在 3 个方面:(1) 对软件体系结构描述各不相同,如果要利用不同的方法对软件体系结构进行评估,则需要根据不同评估方法的要求构造多个不同的模型.既浪费资源,还可能出现模型不一致的情况;(2) 虽然大多数方法都说明有工具支持,但这些工具都只是实验室产品,很难在实际评估中使用;(3) 要求评估人员具备较高的理论水平.

3.2.3 基于特定 ADL 的评估方法的不足

- 领域限制和实用化程度低

该方法是 ADL 研究的进一步成果.由于目前大多数 ADL 都是面向某个特定领域的,加入了很多领域特征,应用范围比较窄;同时,基本上都采用了形式化的描述方式,对用户要求较高,目前实用化程度还较低.

3.3 软件体系结构评估技术研究的趋势分析

通过对现有软件体系结构评估技术的调研,我们发现不同类型的软件体系结构评估技术虽然有不同的研究思路和研究内容,但这些技术之间却并不矛盾,甚至存在着互补关系,如度量技术在基于场景的评估方法中可以作为评估手段来使用.在进行软件体系结构的行为度量时,场景技术则可以为度量提供信息.

对于体系结构评估技术,本文认为无论哪种方法都应该重视基础研究和应用研究.基础研究揭示规律,应用研究促进技术的推广和应用.

- 基础研究部分

基础研究的目标是揭示软件体系结构模型的特征与软件质量因素之间的关系.从软件体系结构评估的目标而言,各种评估技术,无论是利用人的经验还是自动化的预测,都需要通过在软件体系结构层次上可以观测到的现象或特征来预测未来软件产品的质量特征,如通过软件体系结构模型的结构特征(结构复杂性、结构形态与结构耦合度等)来预测软件系统的适应性.因此,软件体系结构模型的特征和软件质量因素之间的关系是所有软件体系结构评估技术的基础.

基础研究的研究思路是利用度量技术对各种特征和因素进行量化,然后选择合适的理论模型建立软件体系结构模型的特征与软件质量因素之间的关联模型,研究内容包括关联模型的表示、关联模型的构造、基于关联模型的推理.从当前软件工程领域的研究看,可选择的理论模型有表示不确定性因果关系的贝叶斯网络、回归模型等.应该注意到,贝叶斯网络作为一种新的工具,在软件工程中以及其他领域的应用都显示出了其相应的优点[11,12,39,40].对关联模型进行研究,需要分析大量的数据,而数据收集是软件工程研究领域一个公认的难题.本文认为,现在越来越多的开源项目以及软件体系结构模式的提出,为数据收集提供了一条可行的途径.

从现在的研究来看,基础研究还比较薄弱,所以,本文认为应该加强该部分的研究,研究的重点是量化体系结构特征,建立软件体系结构特征和软件质量之间的关联模型,为其他部分研究奠定基础.

- 应用研究部分

应用研究部分的研究目标是提供适合人们使用的软件体系结构评估方法及支持工具.应用研究应该从软件体系结构评估实施的角度出发,研究在实施过程中的各种问题,包括评估过程、评估资源需求、评估技巧及工具支持等.

应用研究的研究思路是现有研究成果的实用化研究,并在评估过程中积极地利用基础研究部分的研究成果,即软件体系结构的特征与软件质量因素之间的关联模型.这既能减少评估对专家经验的依赖,又有利于继续将评估的经验转化为知识积累到关联模型中.应用研究部分的另一个重点就是工具支持问题.软件工程的经验已经证明辅助工具可以提高工作效率、规范工作方式、提高工作质量,评估工具的研究将推动评估技术的进一步发展.

从现在的研究来看,应用研究已经取得了较丰富的研究成果,而且也已经有部分成果得到推广和应用.但本文认为还应该加强:

(1) 基础研究与应用研究的结合研究,减少对评估师经验的依赖,提高评估结果的准确性;

(2) 在成熟的评估过程基础上,构造商业化的软件体系结构评估支撑系统,减少评估成本;

(3) 评估技术的标准化研究,比如 CSE-CMU 在质量属性方面和软件推理框架进行的标准化研究.

References:

[1] Bass L, Clements P, Kazman R. Software Architecture in Practice. 2nd ed., Boston: Addison Wesley Professinal, 2003.

[2] Perry DE, Wolf AL. Foundations for the study of software architecture. ACM SIGSOFT Software Engineering Notes, 1992,17(4): 40–52.

[3] Garlan D, Shaw M. An introduction to software architecture. In: Ambriola V, Tortora G, eds. Advances in Software Engineering and Knowledge Engineering. World Scientific Publishing Co., 1993.

[4] Pressman RS. Software Engineering, A Practitioner’s Approach. 4th ed., McGraw-Hill, 1997.

[5] Kazman R, Bass L, Abowd G, Webb M. SAAM: A Method for Analyzing the Properties of Software Architecture. Los Alamitos: IEEE Computer Society Press, 1994. 81−90.

[6] Clements P, Kazman R, Klein M. Evaluating Software Architecture. 2nd ed., Addison Wesley, 2002.

[7] Williams LG, Smith CU. PASA^{SM}: A method for the performance assessment of software architectures. In: Proc. of the 3rd Int'l Workshop on Software and Performance. New York: ACM Press, 2002. 179−189.

[8] Williams LG, Smith CU. Performance evaluation of software architectures. In: Proc. of the 1st Int'l Workshop on Software and Performance. New York: ACM Press, 1998. 164−177.

[9] Bengtsson P, Lassing N, Bosch J, Vliet H. Architecture-Level modifiability analysis (ALMA). The Journal of Systems and Software, 2004,69(1-2):129−147.

[10] Bengtsson P, Bosch J. Architecture level prediction of software maintenance. In: Proc. of the 3rd EuroMicro Conf. on Maintenance and Reengineering (ICSE'99). Amsterdam: IEEE, 1999. 139−147. http://ieeexplore.ieee.org/xpls/abs_all.jsp?&arnumber=756691

[11] Van Gurp J, Bosch J. Automating software architecture assessment. In: Proc. of the 9th Nordic Workshop on Programming and Software Development Environment Research. Lillchammer, 2000. http://www.jillesvangurp.com/static/nwper2000_final_version.pdf

[12] Van Gurp J, Bosch J. SAABNet: Managing qualitative knowledge in software architecture assessment. In: Proc. of the 2000 IEEE Conf. on Engineering of Computer Based Systems. 2000. 45−53. http://www.jillesvangurp.com/static/nwper2000_final_version.pdf

[13] Tvedt RT, Lindvall M, Costa P. A process for software architecture evaluation using metrics. In: Proc. of the 27th Annual NASA Goddard/IEEE Software Engineering Workshop (SEW-27 2002). 2002. 191−196. http://ieeexplore.ieee.org/iel5/8545/27004/01199475.pdf?tp=&arnumber=1199475&isnumber=27004

[14] Nakamura T, Basili VR. Metrics of software architecture changes based on structural distance. In: Proc. of the 11th IEEE Int'l Software Metrics Symp. (METRICS 2005). 2005. 8−17. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1509286

[15] Dobrica L, Niemela E. A survey on software architecture analysis methods. IEEE Trans. on Software Engineering, 2002,28(7): 638−653.

[16] Babar MA, Zhu L, Jeffery R. A framework for classifying and comparing software architecture evaluation methods. In: Proc. of the 2004 Australian Software Engineering Conf. 2004. 309−318. http://ieeexplore.ieee.org/iel5/9061/28748/01290484.pdf?tp=&arnumber=1290484&isnumber=28748

[17] Babar MA, Gorton I. Comparison of scenario-based software architecture evaluation methods. In: Proc. of the 11th Asia-Pacific Software Engineering Conf. 2004. 600−607. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1371976

[18] Lassing N, Rijsenbrij D, van Vliet H. On software architecture analysis of flexibility complexity of changes: Size isn't everything. In: Proc. of the 2nd Nordic Software Architecture Workshop (NOSA'99). 1999. 1103−1581. http://www.cs.vu.nl/~hans/publications/y1999/NOSA99.arch.pdf

[19] Molter G. Integrating SAAM in domain-centric and reuse-based development processes. In: Proc. of the 2nd Nordic Workshop on Software Architecture. 1999. 1103−1581. http://www.cs.vu.nl/~hans/publications/y1999/NOSA99.arch.pdf

[20] Lung CH, Bot S, Kalaichelvan K, Kazman R. An approach to software architecture analysis for evolution and reusability. In: Proc. of the 1997 Conf. of the Centre for Advanced Studies on Collaborative Research. IBM Press, 1997. 1−11. http://portal.acm.org/citation.cfm?id=782010.782025

[21] Zhu LM, Ali Babar M, Jeffery R. Mining patterns to support software architecture evaluation. In: Proc. of the 4th Working IEEE/IFIP Conf. on Software Architecture (WICSA 2004). 2004. 25−34. http://ieeexplore.ieee.org/iel5/9167/29100/01310687.pdf?tp=&arnumber=1310687&isnumber=29100

[22] Tekinerdogan B. ASAAM: Aspectual software architecture analysis method. In: Proc. of the 4th Working IEEE/IFIP Conf. on Software Architecture (WICSA 2004). IEEE, 2004. 5−14. http://ieeexplore.ieee.org/iel5/9167/29100/01310685.pdf?tp=&arnumber=1310685&isnumber=29100

[23] Folmer E, Bosch J. Case studies on analyzing software architectures for usability. In: Proc. of the 31st EUROMICRO Conf. on Software Engineering and Advanced Applications (EUROMICRO-SEAA 2005). IEEE, 2005. 206−213. http://ieeexplore.ieee.org/iel5/10177/32500/01517744.pdf?tp=&arnumber=1517744&isnumber=32500

[24] OMG. UML 2.0 superstructure specification. http://www.omg.org/cgi-bin/doc?ptc/2004-10-02,2004-10-02/2005-03-20

[25] Kazman R. Tool support for architecture analysis and design. In: Proc. of the 2nd Int'l Software Architecture Workshop. New York: ACM Press, 1996. 94−97. ftp://ftp.sei.cmu.edu/pub/sati/Papers_and_Abstracts/ISAW-2.ps

[26] Knodel J, Lindvall M, Muthig D, Naab M. Static evaluation of software architecture. In: Proc. of the Conf. on Software Maintenance and Reengineering (CSMR 2006). IEEE Computer Society, 2006. http://ieeexplore.ieee.org/iel5/10671/33675/01602379.pdf?tp=&arnumber=1602379&isnumber=33675

[27] Yacoub SM, Ammar HH. A methodology for architecture-level reliability risk analysis. IEEE Trans. on Software Engineering, 2002,28(6):529−547.

[28] Kruchten P, Wrote; Zhou BS, Wu CY, Wang JL, Trans. The Rational Unified Process: An introduction. Beijing: China Machine Press/ Addison Wesley, 2002 (in Chinese).

[29] Selic B, Gullekson G, Ward P. Real-Time Object Oriented Modeling. New York: John Wiley and Sons, 1994.

[30] Smith CU, Williams LG. Performance engineering evaluation of object-oriented system with SPE.ED. Computer Performance Evaluation: Modelling Techniques and Tools. LNCS 1245, 1997. 135−l54. http://www.perfeng.com/~cusmith

[31] Lyons A. Developing and debugging real-time software with objecttime developer. 1999. 17−24. http://www.realtime- info.com

[32] HUGIN Expert Brochure. Hugin expert A/S. Aalborg, 1998. http://www.hugin.dk

[33] Honeywell Company. Domain-Specific software architectures for GN&C (DSSA). 1993. http://www.htc.honeywell.com/projects/dssa/

[34] Honeywell Technology Center. MetaH user's manual version 1.27. 1998.

[35] Kenney JJ, Luckham DC. Specifyng and testing conformance to reference architectures. 1993. http://pavg.stanford.edu/rapide/rapide-pubs.html

[36] Riva C, Selonen P, Systa T, Xu J. UML-Based reverse engineering and model analysis approaches for software architecture maintenance. In: Proc. of the 20th IEEE Int'l Conf. on Software Maintenance (ICSM 2004). IEEE Computer Society, 2004. 50−59.

[37] Selenen P, Xu J. Validating UML models against architectural profiles. In: Proc. of the ESEC 2003. New York: ACM Press, 2003. 58−67. http://portal.acm.org/citation.cfm?id=940081

[38] Peltonen J, Selenen P. An approach and platform for building UML model processing tools. In: Proc. of the ICSE 2004 Workshop WoDiSEE2004. 2004. 51−57. http://practise.cs.tut.fi/files/publications/inari/WoDiSee04_Peltonen.pdf

[39] Fenton N, Neil M. Software metrics: Successes, failures and new directions. The Journal of Systems and Software, 1998,47: 149−157.

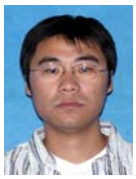
[40] Fenton N, Neil M. A critique of software defect prediction models. IEEE Trans. on Software Engineering, 1999,25(5):675−689.

附中文参考文献:

[28] Kruchten P,著;周伯生,吴超英,王佳丽,译.Rational 统一过程引论.北京:机械工业出版社/Addison-Wesley,2002.

张莉(1968－),女,北京人,博士,教授,博士生导师,CCF 高级会员,主要研究领域为软件工程,需求工程,软件体系结构,过程建模和优化.

高晖(1977－),男,博士生,主要研究领域为软件体系结构,UML 相关技术.



王守信(1979－),男,博士生,主要研究领域为需求工程,软件体系结构,MDA 相关技术.