文章编号:1001 - 9081(2006)07 - 1497 - 03

基于模糊层次分析法的集成软件质量评估模型

熊鹏程,范玉顺
(清华大学 自动化系,北京 100084)
(xpc03@ mails. tsinghua. edu. cn)

摘　要:通过建立软件质量点模型和软件质量度量模型,并运用模型集成的方法,构建了软件质量的集成评估模型。通过模型转换方法将其转换为层次分析模型,并针对模型中权重指标难以确定的问题引入了模糊分析的方法。最后通过一个实例验证了该方法的可行性。

关键词:软件质量保证;层次分析法;权重;模糊分析
中图分类号: TP311.5　　**文献标识码**:A

Integrated software quality evaluation model based on fuzzy analytic hierarchy process

XIONG Peng-cheng, FAN Yu-shun
(*Department of Automation, Tsinghua University, Beijing* 100084, *China*)

Abstract: An integrated software quality evaluation model combining software quality model with software quality evaluation model was proposed. Model transformation methods were adopted to translate this model to a Fuzzy Analytic Hierarchy Process (FAHP) model in order to gain a better evaluation result. Finally, an example was presented to prove the correctness and operability of the model.

Key words: software quality assurance; analytic hierarchy process; weight; fuzzy analysis

0　引言

能力成熟度模型(Capability Maturity Model, CMM)[1~3]是美国 Carnegie Mellon 大学软件工程研究中心 SEI 推出的一套后来成为国际主流的软件开发质量评估标准。CMM 包括 6 个模型、5 个等级、18 个过程域、52 个目标和 300 多个关键实践。在 5 个级别中,除了初始级外,每个成熟度级别都包含若干个关键过程域(Key Process Area, KPA)。软件质量保证(Software Quality Assurance, SQA)是 CMM 可重复级的一个 KPA。软件质量保证的整个过程[4]包括:制定软件质量保证活动计划;检验软件产品和活动是否按照合适的标准、步骤和需求运作;通过实际软件产品和活动的相关指标和理想的指标的比较来监控及调整软件开发活动和软件质量目标以满足用户及最终用户对软件产品的需要和期望。软件质量保证过程与软件开发过程是同步的,并且与软件开发过程具有双向反馈信息。软件质量保证活动贯穿于整个 SQA 软件生命周期,使软件开发或维护过程不仅是一个技术开发过程,而且是一个软件工程过程。

由于软件质量是软件产品中能满足规定需求的性质和特性的总体,这不仅取决于与明确确定的功能和性能的需求的一致性程度,还取决于专业开发隐含标准的一致性程度。文献[5]中提出了软件质量保证管理生命周期的概念,将软件质量管理划分为软件质量保证目标确定、软件质量保证操作、软件质量保证评估和反馈 3 个阶段。在 SQA 的实施方面,文献[6]中给出了基于 SQA 的软件开发过程管理的详细描述,该实施过程与开发小组同步。在对软件产品的评估方法方面,Marcello[7]提出了软件开发过程中的量化分析方法,他对关键指标的度量采取具体的数字测量方法并将其开发成称为 SQUID 的软件,但是在具体实施时这些具体指标很难获得。Bud[8]提出了一种短期软件质量的跟踪办法,虽然实施起来较方便,但对软件的评估停留在文档层次。Katsumi Honda[5]按照 division 等方法来划分评估内容层次,评估指标等级为 good 和 no good,评价结果很粗略。刘琳琳[9]对软件的质量提出了专家评分的方法,但是由于评分的过程带有很大的主观性,其准确度很难保证。在具体实例实施方面,Takeshi[10]以 Toshiba 公司为实例来介绍了实施软件质量分析和评估的过程。上述这些文章虽然各有侧重,从某种程度上指导了软件质量保证的实施过程,但是仅仅只有少数企业能够跨越从诊断、规划到具体实施的鸿沟。这一方面由于软件开发的实施过程多样,软件质量度量点的选取存在难度;另外一方面由于软件质量具有多方面的很难将其考虑周全的特点;再就是对这些特性度量指标的相关定义以及判断很大程度上依据用户以及 SQA 小组人员的主观因素,带有很大的误差。

层次分析法(Analytic Hierarchy Process, AHP)[11]是一种在多指标情况下决策的方法,常被用于按照相关目标来分析计算各指标的权重。为了对具有多特征的软件的质量进行评估,可以使用 AHP 的方法来平衡各个软件评价指标的权重最后获得相关的质量数据。然而,由于软件指标评价的模糊性和不确定性,按照传统 AHP 方法获取的权重数据不全面而且不准确。因此,可以引入模糊规则,用模糊区间来替代具体的

收稿日期:2006 - 04 - 30

作者简介:熊鹏程(1982 -),男,湖北荆州人,博士研究生,主要研究方向:企业建模与仿真、Petri 网及性能分析、网络化制造及企业集成技术; 范玉顺(1962 -),男,江苏扬州人,教授,博士生导师,博士,主要研究方向:企业建模方法与优化分析技术、企业经营过程重组与工作流建模、仿真、实施技术、系统集成与集成平台技术、面向对象与柔性软件系统技术、Petri 网建模与分析技术、车间管理与控制技术.

权重数值,通过相关计算最后获得相应权重,这样能提高获取相关指标的准确度。

1 软件质量保证模型

1.1 软件质量点模型

软件产品不仅仅包括最后提交给用户的程序,而且包括整个开发过程阶段中的各种成果。因此,软件质量保证的目的在于实时地探测和反馈软件生命周期中对软件质量有影响的质量点的观测值,保证软件开发过程的顺利进行。文献[12]中将软件的质量点分为技术、管理、组织、经济 4 个方面来考察,文献[13]中将软件质量点区分为可视性、可测性和可维性的。软件开发模型是软件开发的全部过程、活动和任务的结构框架,能清晰、直观地表达软件开发全过程,是软件项目开发的基础。目前常见的软件工程模型有:瀑布模型、原型模型、演化模型、螺旋模型、喷泉模型等。在瀑布模型中,软件开发从需求分析到最后交付给用户通常包括如下五个阶段:需求分析、总体设计、详细设计、代码编写和系统测试。在需求分析阶段,形成的主要成果是需求说明书,在总体设计阶段形成的主要成果是总体设计说明书,在详细设计阶段形成的主要成果是详细设计说明书,在代码编写阶段形成的主要成果是模块测试用例及经过模块测试的程序,在系统测试阶段形成的主要成果是经过测试的可交付的软件。在交付给用户软件产品之后,还要对软件进行相关的维护和售后服务。演化模型采取的是分批循环递增的开发模式,每次循环开发一部分的功能,成为产品原型的新增功能,从而不断地演化成为新的系统。实际上,这个模型可看作是重复执行的多个“瀑布模型”。螺旋模型将瀑布模型与原型模型结合起来,并且加入风险分析,弥补了前两种模型的不足,是演化模型的一种具体形式。可以看出,无论软件的开发采取哪种模型,其各个阶段总是通过瀑布模型的各个阶段组合而成的。因此,瀑布模型的软件开发方法中定义的提交成果的质量可以作为一种基本而且通用的软件质量评价点。当然,为了克服瀑布模型缺乏灵活性,不能适应用户需求改变的缺点,需要对这些质量点进行循环反复的评估。

表 1 软件质量点模型

阶段名称	软件工程活动	主要质量点
第一阶段	需求分析	需求说明书
第二阶段	总体设计	总体设计说明书
第三阶段	详细设计	详细设计说明书
第四阶段	代码编写	模块测试用例,模块测试的程序
第五阶段	系统测试	可交付的软件

1.2 软件质量度量模型

软件质量度量模型主要对前面所述的质量点按照层次来划分它们关于软件质量的特征、子特征和可以度量的属性。按照 ISO9126[14] 中对质量的定义,这里的特征主要包括以下六个大类:功能性(Functionality)、可靠性(Reliability)、可用性(Usability)、有效性(Efficiency)、可维护性(Maintainability)和可移植性(Portability)。功能性描述了软件产品是否满足了用户的功能需求。可靠性描述了软件产品是否能够一直在一个稳定的状态。可用性衡量了使用软件产品所需要的努力。有效性衡量了软件产品正常运行需要耗费的物理资源。可维护性衡量了对已经完成的软件产品进行调整需要多大的工作量。可移植性衡量了软件产品是否能够方便地部署。为了考虑经济因素,还需要加上获得软件产品的成本标准。将这些特征因素加以层次分解获得这些特征下的子特征,以及组成子特征的属性如表 2 所示。

表 2 软件质量度量模型

特征	子特征	属性
功能性	适应性	满意度,有用性
	完整性	遗漏率,剩余率,覆盖率
	一致性	错误率
	无歧异性	标准度,合法性
可靠性	可靠性	顺应性,正确性,准确性
可用性	可理解性	时间,复杂度,清晰度
	可学习性	时间,复杂度,清晰度,标准度
有效性	有效性	时间复杂度,空间复杂度
可维护性	可分析性	时间,规模,复杂度
	可变性	时间,规模,复杂度
	可测试性	时间,覆盖度
	模块性	耦合度,内聚度
可移植性	适应性	独立性
	互操作性	兼容性
成本标准	成本	人力,时间

1.3 软件质量模型合成

软件质量点模型和软件度量模型的模型合成过程主要通过度量活动来实现。度量活动首先从软件质量点模型中选取定义的一个质量点作为度量对象,然后从软件度量模型中选取和前面质量点对象相关的特征作为度量指标,对每个相关特征细化到其包含的属性层次,为每个属性列出计算规则或者评价结果显示单位。由于篇幅关系,下面以“可交付的软件”质量点为例来展示模型的合成:如图 1 所示,对“可交付的软件”质量点可以按照 6 个特征来度量,而第一个特征“功能性”又包含 4 个子特征,而其第一个子特征“适应性”取决于软件产品的“满意度”,其具体计算规则是通过“80% 用户满意分数 + 20% 开发人员满意分数”获得的。以此类推其他的特征和子特征、属性以及相关的软件度量活动,从而获得关于该质量点的集成质量评估模型。

表 3 集成质量评估模型(以“可交付的软件”质量点为例)

软件度量模型			软件度量活动
特征	子特征	属性	计算规则
功能性	适应性	满意度	80% 用户满意分数 + 20% 开发人员满意分数
	完整性	覆盖率	覆盖的功能目标/定义的功能目标
	一致性	错误率	检测出的程序前后不一致地方的数目
	无歧异性	标准度	与所用标准不一致的形式化错误数目
		合法度	语法错误数目 + 拼写错误数目
可用性	此处略		
有效性			
可维护性			
可移植性			
成本标准	成本	人力	人 × 小时
		时间	天数

1.3.1 集成质量评估模型转换为层次分析模型

尽管获得了软件质量的集成评估模型,但是由于软件产品的类型和用户的需求不同,软件质量衡量标准的侧重点也不同。例如,对于一些需要独立于操作系统的应用软件系统而言,高可移植性会是衡量软件质量的首要要素;而对于需要

用户与软件本身进行大量交互的系统,软件可用性的要求较高。另外,对一些小型企业,主要以实用为主,对相关的成本考虑较多。因此,需要对软件质量集成评估模型中的各个特征、子特征、属性给予一定的权重,再按权重将各个具体的评估值组合起来,得到软件产品质量的最终评价结果,从而消除单一特征或者属性的片面性,更加贴切地衡量软件产品的质量。将上面的软件质量的集成评估模型转换为层次分析模型的结果如图 1 所示,特征、子特征、属性分别对应了 AHP 的第一、第二、第三层次。

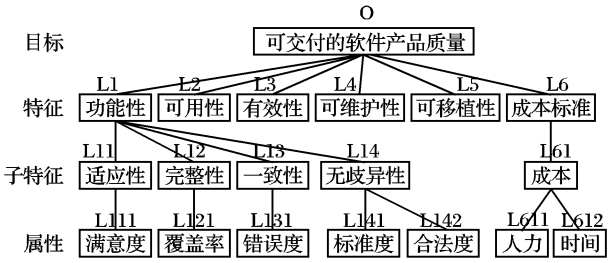

图 1　集成质量评估模型转换为层次分析模型

1.3.2　模糊层次分析模型

在传统 AHP 的方法[11]中确定同一级别的特征或者属性权重时,常常先找出它们之间的相对重要性,然后再用归一化的方法给出权重。比较的过程中常用比例等级表例如 1 到 5 的数字来度量它们之间的相对重要性。虽然这些离散数字比较简单易用,但是没有考虑到将个人的主观意志映射到具体数字时的不确定性。为此,可以在软件质量模型中用平等(E)、强(S)、很强(V)来表示同级的两个指标之间的 3 种相对重要性,采用常用的三角型模糊隶属度图如图 2。

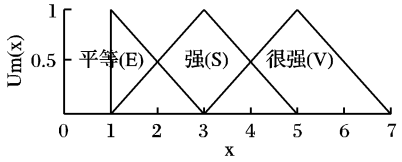

图 2　软件评价指标相对重要性的模糊隶属度

获得同级之间的权重比较模糊矩阵 $\boldsymbol{A}$:

$$\boldsymbol{A} = \begin{bmatrix} 1 & a_{1,2} & \cdots & a_{1,n-1} & a_{1,n} \\ a_{2,1} & 1 & & a_{2,n-1} & a_{2,n} \\ \vdots & \vdots & & \vdots & \vdots \\ a_{n-1,1} & a_{n-1,2} & \vdots & 1 & a_{n-1,n} \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n-1} & 1 \end{bmatrix}$$

其中, $a_{i,j} = \begin{cases} 1, & i = j \\ E,\ S,\ V,\ E^{-1},\ S^{-1},\ V^{-1}, & i \neq j \end{cases}$

设置信度为 ζ,则 $a_{i,j}$ 在 ζ 下的上下界表示为 $[a_{i,j(l)}(\zeta), a_{i,j(u)}(\zeta)]$,且有:$\boldsymbol{E}_\zeta = \boldsymbol{E}_\zeta^{-1} = [1,\ 3-2\zeta]$, $\boldsymbol{S}_\zeta = [1+2\zeta,\ 5-2\zeta]$, $\boldsymbol{S}_\zeta^{-1} = \left[\frac{1}{5-2\zeta}, \frac{1}{1+2\zeta}\right]$, $\boldsymbol{V}_\zeta = [3+2\zeta,\ 7-2\zeta]$, $\boldsymbol{V}_\zeta^{-1} = \left[\frac{1}{7-2\zeta}, \frac{1}{3+2\zeta}\right]$,设优化系数为 ψ,按照文献[15]中提出的凸线性组合关系则有:$a_{i,j}(\zeta,\psi) = \psi * a_{i,j(l)}(\zeta) + (1-\psi) * a_{i,j(u)}(\zeta)$,此时,模糊矩阵 $\boldsymbol{A}$ 也转换为 $\boldsymbol{A}(\zeta,\psi)$。通过 $\boldsymbol{A}(\zeta,\psi) * \boldsymbol{X} = \lambda * \boldsymbol{X}$ 解得 $\boldsymbol{A}(\zeta,\psi)$ 的最大特征值 $\lambda_{\max}$ 和其对应的特征向量 $\boldsymbol{X}_{\max}$,按照文献[16]中的理论算法,将 $\boldsymbol{X}_{\max}$ 归一化即可得到各个评价指标的权重。

2　应用实例

实例取为为某中小企业而开发的某软件提供质量保证。首先建立集成软件质量评估模型如表 3 所示,其经过转换后的 AHP 模型如图 1 所示。就 AHP 的第一层次的各个特征比较而言,该企业主要关注软件的功能性特征,其次关心成本特征。因此有:$L_1 = V, L_6 = S, L_{2,3,4,5} = E$,第一层次的权重比较模糊矩阵:

$$\boldsymbol{A}_1 = \begin{bmatrix} 1 & V & V & V & V & S \\ V^{-1} & 1 & E & E & E & S^{-1} \\ V^{-1} & E & 1 & E & E & S^{-1} \\ V^{-1} & E & E & 1 & E & S^{-1} \\ V^{-1} & E & E & E & 1 & S^{-1} \\ S^{-1} & S & S & S & S & 1 \end{bmatrix},$$

选定置信度为 $\zeta = 0.5$,优化系数为 $\psi = 0.5$,则有:

$$\boldsymbol{A}_1(0.5,\ 0.5) = \begin{bmatrix} 1 & 5 & 5 & 5 & 5 & 3 \\ 0.2083 & 1 & 1.5 & 1.5 & 1.5 & 0.375 \\ 0.2083 & 1.5 & 1 & 1.5 & 1.5 & 0.375 \\ 0.2083 & 1.5 & 1.5 & 1 & 1.5 & 0.375 \\ 0.2083 & 1.5 & 1.5 & 1.5 & 1 & 0.375 \\ 0.375 & 3 & 3 & 3 & 3 & 1 \end{bmatrix},$$

通过 Matlab[17]计算获得其特征值分别为 $\lambda_1 = 7.2121$, $\lambda_2 = 0.1439 + 0.3348\mathrm{i}$, $\lambda_3 = 0.1439 - 0.3348\mathrm{i}$, $\lambda_{3,4,5} = 0.5$。取最大的特征值 λ_1 对应的特征向量 $\boldsymbol{X} = [0.8218,\ 0.1921,\ 0.1921,\ 0.1921,\ 0.1921,\ 0.4208]^{\mathrm{T}}$,将其归一化后得到第一层次的权重分别为 $W_{L_1} = 0.4086$, $W_{L_{2,3,4,5}} = 0.0955$, $W_{L_6} = 0.2094$,同理可得第二层次和第三层次的权重。

3　结语

目前,越来越多的企业开始进行软件过程的改进,但是大部分的软件尚不成熟,主要原因在于软件质量保证 SQA 的实施方面较为欠缺。本文首先总结了各种软件工程模型,提炼出了软件质量点模型,同时在软件质量相关的国际标准下扩展出了软件的质量度量模型。通过统一的软件测量活动将这两种模型集成起来并转换为利于分析的模糊层次分析模型。这样建立的基于模糊层次分析法的集成软件质量评估模型不仅很容易实施,而且保证了各级指标权重的正确性和合理性。

参考文献:

[1]　卡耐基梅隆大学软件工程研究所. 能力成熟度模型(CMM):软件过程改进指南[M]. 刘孟仁, 等译. 北京:电子工业出版社, 2001.

[2]　龚波. 能力成熟度模型集成及其应用[M]. 北京:中国水利水电出版社, 2003.

[3]　PAULK MC, CURTIS B, CHRISSIS MB, *et al*. CMU/SEI-93-TR-24, Capability Maturity Model for Software, Version 1. l[R], 1993.

[4]　BAKER E, FISHER M. Software Quality Program Organization, Handbook of Software Quality Assurance[M]. Prentice Hall, 1999. 115 - 145.

[5]　HONDA K, MINOMURA K, KOMIYAMA T. Meta-SQAP: Meta-Methodology for Software Quality Assurance[A]. Proceedings of the 13th Annual International Computer Software and Applications Conference[C], 1989. 509 - 515.

[6]　苟娟琼. 第三方软件质量保障实施研究[J]. 计算机工程, 2002, 28(4).

(下转第 1506 页)

3.3.2 结构性测试方法的相关知识

目前比较成功的结构性测试方法比较少,比较典型测试的有 DD-路径测试法和数据流测试法。功能性测试离代码过远,是一个极端;路径测试则是测试方法的另一个极端:将代码采用有向图表示和程序路径公式化,掩盖了代码中的可行路径和不可行路径的区别。数据流测试技术可以克服这个不足。

一般讲,结构性测试指标越精细,给定功能性测试方法通过更严格的结构性测试指标评估时,功能性测试用例覆盖结构性测试指标的覆盖比例 S 就变得越低。这与直观相一致。

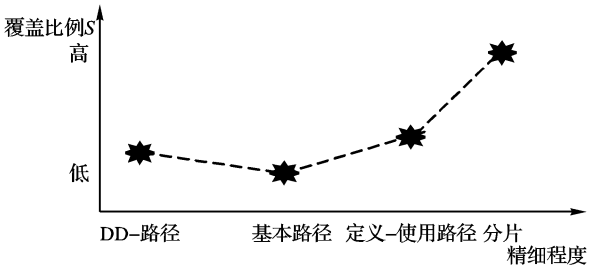

图 5 覆盖比例 S 和不同精细程度的对应方法间的关系

可以通过分析已成功开发的软件开发项目历史数据,得到不同测试方法的覆盖比例 S—— 精细程度系数 $Ks\text{-}e$。这些数据,包括系数 $Ks\text{-}e$ 以及对应的关系图,都可以作为经验知识被以后的项目开发所参考。

了解功能性测试方法存在漏洞和冗余后,可以开发一些将功能性测试技术的有效性和结构性测试进展指标关联的指标。功能性测试技术始终可以产生一组测试用例,结构性测试指标始终以可计算的内容表示。

4 ITIT 技术的实现与应用

ITIT 技术已经在"VOCAL 策略下软件测试与分析原型系统"[11]中得到验证。限于篇幅,本文只介绍该技术实现的步骤:

首先,知识库中知识的搜集和当前软件开发项目基本属性数据的准备。知识库采用二维关系数据库的形式存放知识,一个记录存放一个案例或一条规则。特别地,对于规则库,每条规则都有一个可信度属性值。

第二,根据当前软件开发项目的关键属性值,使用基于反馈式权重相似性算法的 CBR,从软件测试方案范例库中确定当前软件开发项目有关软件测试可以参考的初步方案。

第三,对初步方案中的具体各项,使用 RBR,根据软件测试规则库中的相关规则,得到这些方案项的参考意见和相关的参考数据,用户根据这些参考意见和参考数据,修改初步方案得到一个新的测试方案。

第四,在新测试方案执行过程中,根据规则库中的相关规则和范例中的相关经验知识,利用测试分析工具对测试过程中得到的测试数据进行动态分析,得到分析数据,为测试决策提供依据。

第五,如果该新方案具有代表性,且执行效果好,则可以作为一个新范例保存到范例库中。

该原型系统的实验数据表明,在 ITIT 技术的支持下,软件测试过程中,各种测试方法综合应用后测试效率有较大提高。但是,分析这些实验数据,发现 ITIT 技术在实际软件测试过程中的效率,明显受到下列三个因素的影响:

1) 测试知识库中范例和规则的正确度、数量和类型;

2) 范例描述软件测试方案的精细度(软件测试过程中测试活动项覆盖程度)和规则的可信度;

3)搜集到的测试数据的量和准确性。

这将是我们下一步需要研究的问题。

参考文献:

[1] MCGREGOR JD, SYKES DA. 面向对象的软件测试[M]. 杨文宏, 李新辉, 杨洁, 译. 北京: 人民邮电出版社, 2002.

[2] JORGENSEN PC. Software Testing – A Craftsman's Approach (Second Edition)[M]. 韩柯, 杜旭涛, 译. 北京: 机械工业出版社, 2003.

[3] 许静, 陈宏刚, 王庆人. 软件测试方法简述与展望[J]. 计算机工程与应用, 2003, 39(13): 75 – 78.

[4] ROBERT P. A complete toolkit for the software tester[J]. American Programmer, 1991, 4(4): 28 – 37.

[5] MILLER Jr EF, Automated software design: A technical perspective [J]. American Programmer, 1991, 4(4)4: 38 – 43.

[6] 孔繁胜. 知识库系统原理[M]. 杭州: 浙江大学出版社, 2000.

[7] 高济. 基于知识的软件智能化技术[M]. 杭州: 浙江大学出版社, 2001.

[8] 马廷淮. 基于粗糙集理论的数据挖掘方法研究[D]. 中国科学院, 2003.

[9] 李少波. 基于知识的产品开发集成系统关键技术的研究[D]. 中国科学院, 2003.

[10] 张海盛, 古乐野, 眭俊华, 等. "基于知识的 CAD/CAE/CAPP/CAM 集成技术(2001AA412260)"课题技术报告[R], 2004.

[11] 眭俊华. "VOCAL 策略下软件测试与分析工具的研制"课题技术报告[R], 2004.

(上接第 1499 页)

[7] VISCONTI M , GUZMAN L . A Measurement - Based Approach for Implanting SQA & SCM Practices [A]. Proceedings of the XX International Conference of Computer Science Society[C], 2000. 126 – 134.

[8] GLICK B. An SQA quality tracking methodology[A]. Proceedings of International Conference on Software Maintenance[C], 1990.

[9] 刘琳琳, 吴永英. 一个 CMM 自评估系统的研究与开发[J]. 计算机工程, 2004,30(16).

[10] TANAKA T, AIZAWA M, OGASAWARA H, *et al.* Software quality analysis and measurement service activity in the company [A]. Proceedings of the 1998 (20th) International Conference on Software Engineering[C], 1998. 426 – 429.

[11] SATTY TL. The analytic hierarchy process [M]. New York: McGraw-Hill, 1980.

[12] RAI A . Software quality assurance : an analytic survey and research prioritization[J]. Journal of Systems and Software, 1998, 40(1): 67 – 83.

[13] 吴超,林家骏,俞玲,等. 军用软件质量特性和设计属性的研究[J]. 计算机工程, 2005,31(12).

[14] ISO/IEC9162-1991, Information Technology Software Product Evaluation Quality Characteristics and Guidelines for Their Use [S], 1991.

[15] LEE AR. Application of modified fuzzy AHP method to analyze bolting sequence of structural joints [Z]. UMI Dissertation Services, A Bell & Howell Company, 1995.

[16] CHENG CH, MON DL. Evaluating weapon systems by analytical hierarchy process based on fuzzy scales [J]. Fuzzy Sets and Sytems, 1994,63(1):1 – 10.

[17] 王家文, 王皓, 刘海. MATLAB 7.0 编程基础[M]. 北京:机械工业出版社, 2005.