# 软件质量度量的方法与实践

摘 要

软件开发的管理贯穿软件开发的整个生命周期，而软件的质量是对软件开发效果的评价。本文选择了软件的三个特性对软件开发进行了评价，中间结合了软件开发过程中管理的思路，准则，计划，通过建立直接或简介评价机制对软件的整体质量进行了评分。结合实施的情况对软件的三个特性以及管理和质量保证等进行了评价和描述。

关键字：软件质量度量、项目管理、项目计划

## 1．引言

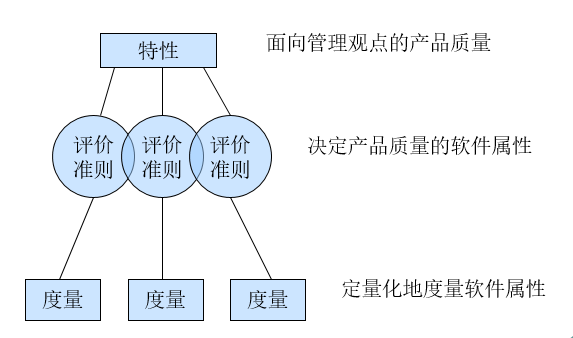
在60年代末期的大型软件所面临的软件危机反映了软件开发中管理的重要性。而对于管理层人员来说，对所见的事物没有适当的度量或适当的准则去判断、评估和决策，就无法进行优秀的管理。因此软件开发需要使用度量。软件度量是对软件开发项目、过程及其产品进行数据定义、收集以及分析的持续性定量化过程。软件度量的根本目的是为了项目管理，即利用度量来改进软件过程。通过软件度量可以促进项目成功，从而开发出高质量的软件产品。

考虑到本软件产品（“笔画印象”素描图像处理软件）的开发实践过程及软件类型，我们参考McCall质量模型的框架，并选取了三个重要的质量特性——正确性、可靠性和可使用性，针对本软件产品质量进行度量。

软件正确性是指在预定环境下，程序按照预期执行的能力，即软件满足设计规格说明及用户预期目标的程度。这是对软件产品最基本的要求，因此是最重要的质量指标之一；软件的可靠性反映软件在稳定状态下，维持正常工作的能力，即按照设计要求，在规定时间和条件下不出故障、持续运行的程度。此外，还需考虑在意外环境下，软件能否做出恰当的响应；“笔画印象”素描图像处理软件是一款适用于安卓系统的移动应用，对于此类应用来说，可使用性也是衡量软件质量优劣的重要因素。一款优秀的移动应用要求软件在基本覆盖功能需求的同时，学习成本和使用成本不能过高，对不同的用户群体来说，可使用性差别不能太大。

## 2．所参照的模型、方法及综述

本项目中软件质量的度量参考McCall三层质量模型来确定相应的软件度量方法。



图：McCall质量模型的三层次框架

#### 2.1参照McCall模型，三层次模型：

在此模型中，规定了11个面向软件产品的运行、修正和转移的反映软件质量的特性：

|  |  |
| --- | --- |
| **软件特性** | **说明** |
| 正确性 | 在预定环境下，软件满足设计规格说明及用户预期目标的程度。它要求软件没有错误。 |
| 可靠性 | 软件按照设计要求，在规定时间和条件下不出故障，持续运行的程度。 |
| 效率 | 为了完成预定功能，软件系统所需的计算机资源的多少。 |
| 完整性 | 为了某一目的而保护数据，避免它受到偶然的、或有意的破坏、改动或遗失的能力。 |
| 可使用性 | 对于一个软件系统，用户学习、使用软件及为程序准备输入和解释输出所需工作量的大小。 |
| 可维护性 | 为满足用户新的要求，或当环境发生了变化，或运行中发现了新的错误时，对一个已投入运行的软件进行相应诊断和修改所需工作量的大小。 |
| 可测试性 | 测试软件已确保其能够执行预定功能需工作量的大小。 |
| 灵活性 | 修改或改进一个已投入运行的软件所需工作量的大小。 |
| 可移植性 | 将一个软件系统从一个计算机或环境移植到另一个计算机系统或环境中运行时所需工作量的大小。 |
| 复用性 | 一个软件（或软件的部件）能再次用于其他应用（该应用的功能与此软件或软件部件的所完成的功能有联系）的程度。 |
| 互操作性 | 连接一个软件和其他系统所需工作量的大小。 |

#### 2.2用作评价规则的软件属性；

在进行软件质量度量的过程中，针对各个质量特性进行直接度量略有困难，McCall定义了下列软件属性，以便于评价软件质量：可跟踪性、完备性、一致性、安全性、容错性、准确性、简单性、执行效率、存储效率、存取控制、存取审查、操作性、易训练性、简明性、模块独立性、自描述性、结构性、文档完备性、通用性、可扩充性、可修改性、自检性、机器独立性、软件系统独立性、通信共享性、I/O容量、I/O速率、通信性。

针对本款软件，以“可追踪性”、“完备性”和“一致性”作为软件正确性的主要评价准则；以“一致性”、“安全性”、“容错性”、“准确性”、“简单性”、“简明性”、“模块独立性”、“自描述性”和“通用性”作为考察软件可靠性的主要评价标准；以“操作性”和“易培训性”作为考察软件可使用性的主要评价标准。

#### 2.3观察软件质量的软件属性度量。

##### 2.3.1可靠性

采取间接度量法：

|  |  |  |  |
| --- | --- | --- | --- |
| **质量特性：可靠性** | | | |
| **子特性** | **质量名称** | **评价准则** | **评价结果得分**  **或满足与否** |
| 一致性 | 前后台 | 一致 | 满足 |
| 安全性 | Post方法 | 安全 | 满足 |
| 容错性 | 接口 | 可容错 | 满足 |
| 准确性 | 参数调节 | 准确 | 满足 |
| 简单性 | 基础代码 | 简单 | 满足 |
| 简明性 | 注释 | 简明 | 满足 |
| 模块独立性 | package | 模块独立 | 满足 |
| 自描述性 | 无 | 无 | 满足 |
| 通用性 | 系统复用 | 普适 | 满足 |

##### 2.3.2正确性

正确性是指在预定环境下，软件满足设计规格说明及用户预期目标的程度。它要求软件在执行所设定的功能时没有错误，程序必须正确地运行，并为它的用户提供某些输出。

软件的正确性采用直接度量法，主要通过评估其可追踪性、完备性以及一致性这三种评价准则来进行度量。其中，可追踪性是指对软件进行正向和反向追踪的能力，主要是指在软件开发过程中建立和维护软件制品之间的关联关系，并利用这些关系对软件项目进行一系列分析的能力；一致性是指设计文档与系统实现的一致性，该项准则保证软件的开发过程与软件设计文档的契合性；而完备性是指所需功能的完全实现程度，它需要度量整个软件产品的功能完成度，同时也需要度量已完成功能的正确性。

针对每一项评价准则设计具体的度量指标对其进行质量评估，加权处理所有的评估准则后得到该软件质量特性的质量度量结果，该计算过程可以表达为下式：

其中，M为质量特性的整体度量结果，为第k个评估准则的加权系数，为第k个评估准则的度量结果。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **质量特性： 正确性** | | | | |
| **评价准则** | **加权系数** | **度量指标** | **度量指标说明** | **评价结果得分或满足与否** |
| **可追踪性** |  | 开发文档的齐全度 | 是否依据开发流程产生相应的开发文档 |  |
| 模块的可复查性 | 各设计模块是否能够在相应开发文档中找到细节说明 |  |
| **一致性** |  | 软件与开发文档的内容匹配度 | 阶段性开发文档是否符合系统开发实现过程 |  |
| **完备性** |  | 系统功能完成度 | 系统实现的功能是否匹配软件需求规格说明的功能需求 |  |
| 系统缺陷密度 | 评估系统的正确性，采用程序复杂性度量，即每千代码行(KLOC)或FP的差错数，其中将差错定义为已被证实是不符合需求的缺陷。计算软件缺陷密度（缺陷数量/软件规模） |  |

##### 2.3.3可使用性

|  |  |  |  |
| --- | --- | --- | --- |
| **质量特性：可使用性** | | | |
| **序号** | **质量名称** | **评价准则** | **评价结果得分**  **或满足与否** |
| 1 | 基础性 | 为学习系统所需要的体力上的和智力上的技能较低 |  |
| 2 | 学习性 | 为达到适度有效使用系统所需要的时间（min）<3 |  |
| 3 | 流畅性 | 应用操作是否流畅 |  |
| 4 | 友好性 | 界面美观整洁 |  |
| 5 | 完整性 | 用户认为功能齐全 |  |
| 6 | 吸引度 | 应用不会令人沮丧 |  |
| 7 | 需求性 | 满足用户拍照等需要 |  |
| 8 | 分享性 | 能够分享给用户好友 |  |
| 9 | 优质性 | 图片处理效果较好 |  |

## 3．提出在应用及实践中的具体方法

可靠性30%，正确性30%，可使用性40%

|  |  |  |  |
| --- | --- | --- | --- |
| **质量特性：可靠性** | | | |
| **子特性** | **质量名称** | **评价准则** | **评价结果得分**  **或满足与否** |
| 一致性 | 前后台 | 一致 | 满足 |
| 安全性 | Post方法 | 安全 | 满足 |
| 容错性 | 接口 | 可容错 | 满足 |
| 准确性 | 参数调节 | 准确 | 满足 |
| 简单性 | 基础代码 | 简单 | 满足 |
| 简明性 | 注释 | 简明 | 满足 |
| 模块独立性 | package | 模块独立 | 满足 |
| 自描述性 | 无 | 无 | 满足 |
| 通用性 | 系统复用 | 普适 | 满足 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **质量特性： 正确性** | | | | |
| **评价准则** | **加权系数** | **度量指标** | **度量指标说明** | **评价结果得分（1-10分）** |
| **可追踪性** | 0.3 | 开发文档的齐全度 | 是否依据开发流程产生相应的开发记录文档 | **8分**  整个开发过程依次产生：1.需求分析规格说明书 2.系统原型设计图 3.类图及其关系图 4. 软件接口说明表 5.技术风险分析书 |
| 模块的可复查性 | 软件产品各个设计模块是否能够在相应开发文档中找到细节说明 | **8分**  程序整体逻辑体现在《类图及其关系图》中；模块间接口参数体现在《软件接口说明表》中 |
| **一致性** | 0.3 | 软件与开发文档的内容匹配度 | 阶段性开发文档是否符合系统开发实现过程 | **7分** 符合 |
| **完备性** | 0.4 | 系统功能完成度 | 系统实现的功能是否匹配软件需求规格说明的功能需求 | **10分** 完成软件《软件需求规格说明书》中定义的全部需求 |
| 系统缺陷密度 | 评估系统的正确性，采用程序复杂性度量，即每千代码行(KLOC)或FP的差错数，将差错定义为已被证实是不符合需求的缺陷。计算软件缺陷密度（缺陷数量/软件规模） | **8分** 总共代码行数42971  缺陷总数4  1，历史图片的加载每点击一次就下载6张图片；2，历史图片需要下载完成后才能显示，且间隔数秒时间…  FP = 4/42.97 < 1 |

|  |  |  |  |
| --- | --- | --- | --- |
| **质量特性：可使用性** | | | |
| **序号** | **质量名称** | **评价准则** | **评价结果得分**  **或满足与否** |
| 1 | 基础性 | 为学习系统所需要的体力上的和智力上的技能较低 | 满足 |
| 2 | 学习性 | 为达到适度有效使用系统所需要的时间（min）<3 | 满足 |
| 3 | 流畅性 | 应用操作是否流畅 | 满足 |
| 4 | 友好性 | 界面美观整洁 | 满足 |
| 5 | 完整性 | 用户认为功能齐全 | 不满足 |
| 6 | 吸引度 | 应用不会令人沮丧 | 满足 |
| 7 | 需求性 | 满足用户拍照等需要 | 满足 |
| 8 | 分享性 | 能够分享给用户好友 | 满足 |
| 9 | 优质性 | 图片处理效果较好 | 满足 |

（10分制）

## 4．软件质量度量在软件质量控制和质量保证中的应用

#### 4.1内部质量保证：

在组织内部，“质量保证”使管理者建立信心。

##### 4.1.1组织结构

项目经理：黄俊萌

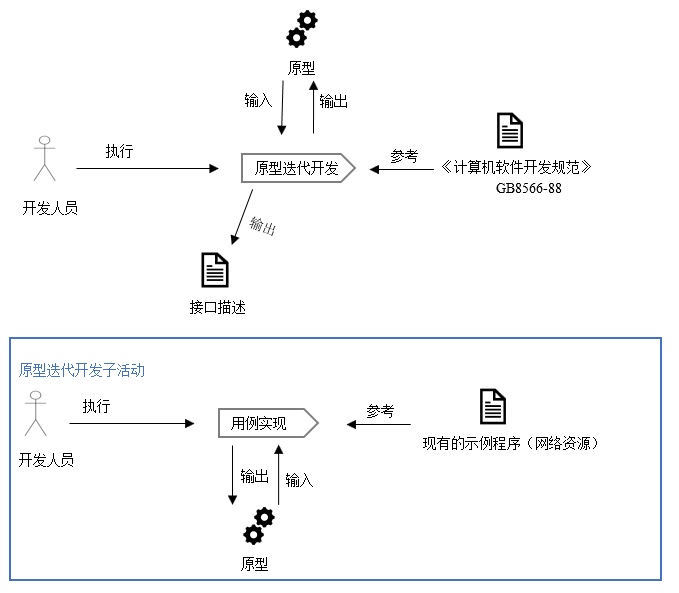
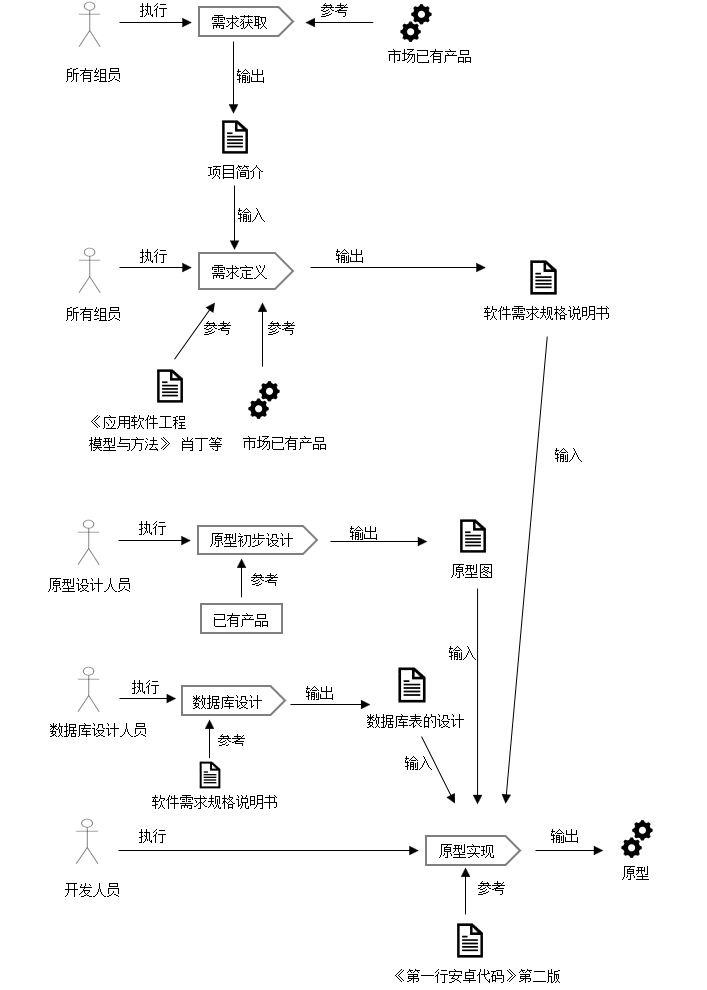
产品经理：张晓

界面开发工程师：李梦宇

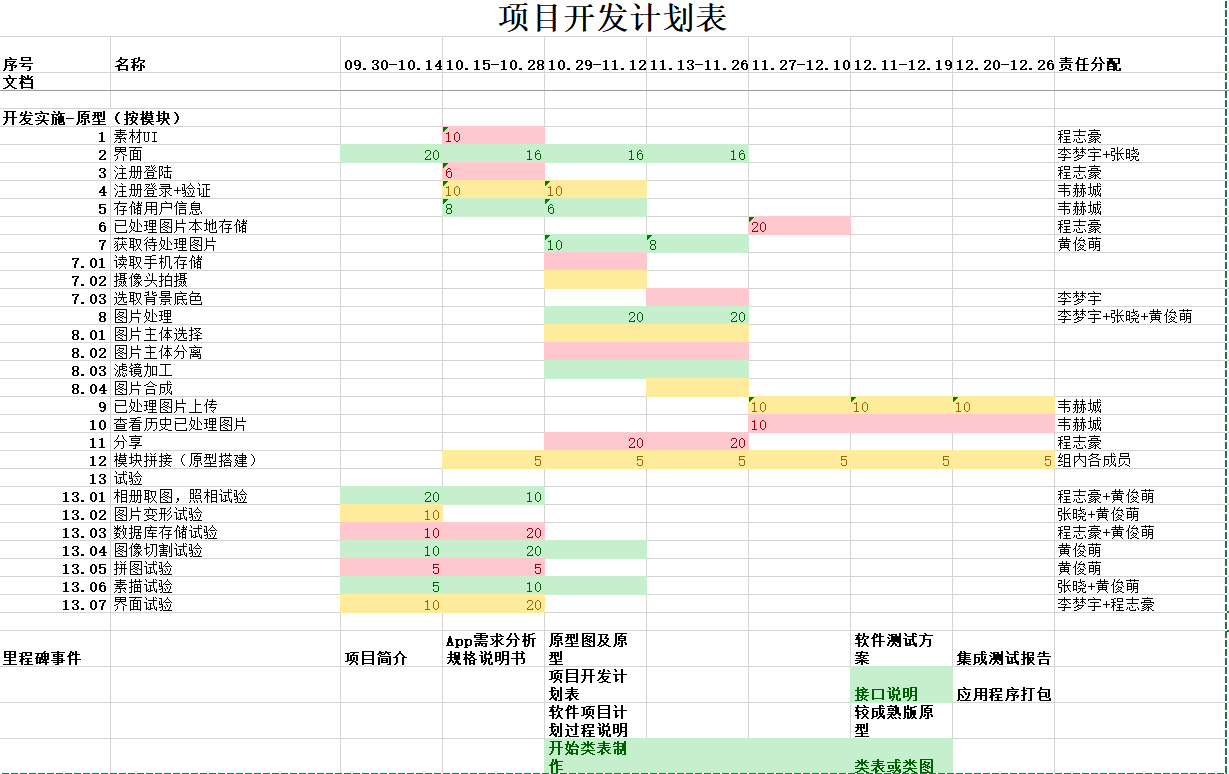
前端开发工程师：程志豪

后台开发工程师：韦赫城

##### 4.1.2计划过程图



##### 4.1.3进度计划



##### 4.1.4交流和计划

学习讨论，每周进行会议讨论，并且得到个人的学习情况记录和任务完成情况。

个人填写跟踪表格，制定个人详细的细分计划，从而按步骤完成任务。保证项目总体计划里的工件的完成。

## 4.2外部质量保证：

在合同或其他环境中，“质量保证”使顾客或其他人建立信心。

#### 4.2.1用户要求定义：

1. 熟练掌握正确定义用户要求的技术：不仅依赖用户的直接描述，还需深入解读用户意图，通过设计原型探索挖掘用户的真实需求；
2. 熟练使用和指导他人使用定义软件需求的支持工具：利用Staruml软件绘制用户交互用例图，辅助进行需求分析；
3. 重视领导全体开发人员收集和积累有关用户业务领域的各种业务的资料和技术技能。

#### 4.2.2力争不重复劳动

1. 编写PhotoUtils类里面的getImageUri等的方法，实现功能复用；使用github进行版本管理，在原型上添加功能，实现扩展复用性。统一Android Studio的版本号。
2. 建立imageUriSet类管理不同存储位置图片信息，避免创建多种类型存储对象。使用一个Adapter适配器完成历史记录展示和主页面推荐功能，避免代码冗余。
3. 编写JDBCUtils等工具包，实现功能复用；使用网上的开源jar包，略去许多底层代码的编写。
4. 权限管理申请，用户持续登录，页面设计的简洁明了，不做无用跳转

#### 4.2.3，掌握开发新软件的方法

1. 学会使用面向对象技术，网络化技术上传下载，github和Android studio；
2. 面向对象编程技术，Android Studio开发工具的使用
3. 学会使用面向对象技术编程，让代码可用性和可修改性更强。使用了OkHttp框架，编写符合自己需求的功能函数。
4. 使用Android Studio 对软件进行开发，整体框架逐步细化，使用模块化编程

#### 4.2.4，排除无效劳动

1. 开始学习同样的原型，从原型中找可用的模块改进；
2. 设计和实现分离。在程序实现之前进行界面UI设计，避免多词更改降低效率
3. 通过每周会议交流各成员工作情况，信息流通，避免无效工作。
4. 使用GitHub的merge与fork功能对代码进行更新迭代

#### 4.2.5，发挥每个开发者的能力

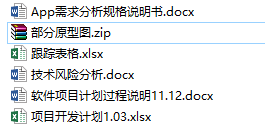
1. 开发过程中查阅文件，寻找别人的例子；
2. 学习android界面设计相关知识
3. 通过自己学过的后台技术，完成App的注册、登陆和图片的上传、下载功能。
4. 学习了Android系统的动态权限调用，摄像及相关文件的读写与传输。学习了微信平台相关的分享技术与应用包。学会使用guthib进行项目进度控制与更新。

#### 4.2.6提高软件开发的工程能力

1. 培训github管理项目
2. 提高了测试软件的效率，简化了测试软件的过程，明确了自己的目标与在组内的任务，大幅度提高了

#### 4.2.7提高计划和管理质量能力

（1）项目开发初期的项目计划



1. 完成《App需求分析规格说明书》，项目计划模板，跟踪表格模板，简单的技术风风险列举（未量化）及《软件计划过程说明》，原型图
2. 制定管理计划：

* 平台，建立学习资料共享平台，以文档方式交流，形成工件，有统一格式，保持对需求理解的一致性。
* 项目整体采用敏捷模型，用例驱动，每两周一次迭代。项目定义了需求，得到UML图，根据UML分配任务，制定计划。
* 迅速学习安卓开发的书籍，找到网上用例测试相应功能难度，直接实施出几个有用的例子，并且列举技术风险，给予用户信心。
* 小组成员功能开发者统一学习几个模型，迅速掌握安卓开发的要点。（sqlite增删改查，调用相机相册等例子，学习了安卓权限和布局问题）

1. 评价

对可靠性的考虑，统一的平台和学习模型使得小组成员具备初步的开发能力，采用统一模板和敏捷模型开发，有了一致的目标。

对正确性的考虑，采用敏捷开发，可以对bug进行检测。有了计划模板，可以对进度进行判断。

对易使用性的考虑，采用敏捷开发，可以随时对原型进行使用，得出需要改进的地方。

1. 计划执行过程中完成的报告
2. 完成《项目开发计划表》，概要设计（接口说明），原型图及原型，会议纪要，及个人任务计划。
3. 评价

* 对可靠性的考虑，原型图指明了将要形成原型的样子，开发计划表记录了要使用功能的具体责任人，明确了开发目标。
* 对正确性的考虑，有了开发计划表，会议纪要和个人任务计划，可以对个人要完成的目标和个人实际完成情况进行对比，开会研究要解决的bug和要协助完成的事项。
* 对易使用性的考虑，原型使得每个小组成员都可以对原型功能进行测试，检查，会议上进行评论和改进。

1. 评价评审工作的计划

* 产生工件记录，产生会议纪要和跟踪表格。每次迭代进行原型的展示，在小组间讨论问题和遇到的bug，提升队伍水平，引导队伍开发的方向。
* 小组成员各自指定计划，细化自己的工作，从什么出发得到什么，然后得到最终成果，并将最终成果合并入项目的计划成果中。
* 代码量的工作交给github管理，可以实时看到每个人的代码贡献度。

（4）提高软件开发项目管理的精确度：

细化工作任务，明确个人职责。

明确开发的模型和开发成果的提交平台，对个人的成果进行细化记录。

参考文献：

|  |
| --- |
| [1] 简阳,闫然,沈晓美.基于TestBed的舰船装备软件质量度量[J].船舶标准化与质量,2018(04):47-51+46. |
| [2] 徐鑫.基于SaaS模式的医疗设备软件质量评价研究与应用[J].中国医疗设备,2018,33(04):134-138. |
| [3] 王丽丽.手机软件质量度量研究[J].长春理工大学学报(自然科学版),2017,40(05):115-118+124. |
| [4] 郑慧君.中小型软件质量度量应用与研究[J].现代计算机(专业版),2012(09):20-23. |

[5] 刘莉芳. APP软件质量度量的研究[J]. 价值工程, 2016, 35(22):236-238.