软件工程中软件质量管理分析

冯越鑫
北方信息控制研究院集团有限公司　江苏　南京　211153

摘　要：软件工程不断的开发和运营过程中不可避免会出现软件质量问题，这就要求在软件开发过程进行中要对软件进行严格管理把控，制定一套严格的软件工程质量标准，认识到软件工程管理的重要性，加强软件配置的管理。软件质量是衡量软件产品是否满足用户需求、具备稳定性和可靠性的关键指标。本文分析了质量管理的存在的问题，同时，还提出了培养质量文化、建立激励机制等软性措施，以全面推动软件质量管理水平的提升。通过综合这些策略，可以有效确保软件产品的质量和市场竞争力。

关键词：软件工程；软件质量；管理

引言：在当今信息化快速发展的时代，软件已成为各行各业不可或缺的重要工具。然而，随着软件规模的扩大和复杂度的提升，软件质量管理面临着前所未有的挑战。软件质量不仅直接关系到用户体验和满意度，还影响着企业的声誉和市场竞争力。因此，深入探讨软件工程中软件质量管理的有效措施和方法，对于提高软件产品质量、降低维护成本、增强企业竞争力具有重要意义。通过对软件质量管理进行分析，以此为软件工程实践提供参考和借鉴。

1 软件工程中软件质量管理的重要性

软件工程中软件质量管理直接关系到软件产品的成败以及企业的长远发展。第一，在软件开发过程中，质量管理贯穿始终，从需求分析、设计、编码、测试到维护，每一个环节都离不开质量管理的把控。一个优秀的软件产品，除了满足用户的功能需求外，还必须具备高可靠性、易用性、可维护性和安全性等质量属性。这些属性的实现，离不开严格的质量管理。第二，在激烈的市场竞争中，软件产品的质量往往成为企业赢得客户信任和市场份额的关键因素。通过加强质量管理，企业能够培养出一支具备高度质量意识的团队，从而在软件开发过程中不断追求卓越，提升产品的竞争力[1]。高质量的软件产品能够降低企业的售后维护成本，提高客户满意度，进一步巩固企业在市场中的地位。第三，对于软件从业人员而言，掌握质量管理的知识和技能，不仅能够提升个人的综合素质，还有助于在职业生涯中取得更好的发展前景。具备质量管理能力的软件工程师，在求职市场上更具竞争力，能够获得更多的就业机会和晋升空间。

2 软件工程中软件质量管理存在的主要问题

2.1 需求管理不清晰

需求管理不清晰通常表现为需求定义模糊、不完整或频繁变更。在项目初期，如果未能与用户或相关利益方进行充分的沟通，就可能导致需求收集不全面，关键功能或性能要求被遗漏，由于技术、市场或业务环境的变化，需求在项目开发过程中可能会发生变更。然而，如果缺乏有效的需求变更管理机制，这些变更可能无法及时、准确地反映到项目计划中，从而导致开发团队与用户需求之间的脱节。另外，需求管理不清晰还会引发一系列连锁反应。例如，由于需求不明确，开发团队在设计和编码阶段可能会遇到困惑，导致开发进度延误和成本超支。同时，不清晰的需求也增加了测试的难度，因为测试团队需要基于不完整或模糊的需求文档来制定测试计划，这可能导致测试不充分，遗漏重要的测试场景。

2.2 质量度量和评估不充分

在实际项目中，质量度量和评估往往被忽视或执行不到位，这可能是因为项目团队缺乏明确的质量度量标准，或者没有采用科学、全面的评估方法。质量度量和评估不充分会导致一系列问题。首先，项目团队可能无法准确了解软件产品的当前质量状态，从而难以制定有效的改进措施。其次，缺乏质量度量和评估的支持，项目团队可能无法及时发现和纠正潜在的质量问题，导致这些问题在后续的开发和测试阶段被放大，甚至影响到最终产品的发布和使用。

2.3 缺乏自动化工具支持

在软件工程实际项目中，由于资源限制、技术难度或认知偏差等原因，往往缺乏足够的自动化工具支持。缺乏自动化工具支持会导致多个问题。手动执行质量检查和分析任务不仅耗时费力，而且容易出错[2]。这可能导致质量问题的遗漏或误判，进而影响软件产品的整体质量。其次，没有自动化工具的支持，项目团队可能难以实时监控软件质量的变化趋势，从而无法及时采取纠正

措施，缺乏自动化工具还限制了项目团队对大规模数据集进行质量分析的能力，使得一些潜在的质量问题难以被发现和解决。

2.4 质量保证体系不完善

质量保证体系的不完善主要体现在缺乏明确的质量标准和规范，导致在软件开发和测试过程中无法形成统一的质量评判依据。这可能导致团队成员对质量的理解存在差异，进而影响项目的整体质量。另外，质量监控机制不健全，无法对软件开发过程进行实时、全面的监控。这可能导致一些潜在的质量问题被忽视或遗漏，增加了软件产品的风险。还有质量保证体系的不完善还可能导致质量反馈机制不畅，使得项目团队无法及时获取用户对软件产品的反馈意见，从而无法对软件进行及时的优化和改进。这不仅会影响软件产品的用户体验，还可能降低软件的市场竞争力。

3 加强软件工程中软件质量管理的有效措施

3.1 实施严格的需求管理

在软件工程中，加强软件质量管理的首要有效措施是实施严格的需求管理。（1）明确需求定义与优先级。在项目启动阶段，应与客户或用户进行深入沟通，明确软件的需求细节，包括功能需求、性能需求、安全需求，要对需求进行优先级排序，确保关键和紧急的需求得到优先处理。这有助于团队在有限的资源和时间内，集中力量满足最重要的需求。（2）建立需求变更控制流程。需求变更在软件开发过程中是不可避免的，但应建立有效的变更控制流程来管理这些变更。任何需求变更都应经过严格的审批流程，确保变更的合理性和可行性。同时，变更应被及时、准确地记录，并通知到所有相关团队成员，以避免信息不一致和误解。（3）采用需求管理工具。利用专业的需求管理工具，如需求跟踪矩阵、需求管理工具软件等，来跟踪和管理需求的状态、变更历史以及与其他项目元素的关联。这些工具能够帮助团队更有效地管理需求，确保需求的完整性和一致性。（4）加强需求评审与验证。在项目的各个阶段，都应进行需求评审和验证。通过组织专家评审、用户测试等方式，对需求进行严格的审查和验证，确保需求的准确性和合理性。同时，通过验证，可以及时发现并纠正需求中的错误和遗漏，降低后续开发的风险。（5）培养需求管理意识。通过培训、分享会等方式，提高团队成员对需求管理重要性的认识，让他们了解如何更好地进行需求管理。这有助于形成全员参与、共同负责的需求管理氛围，提高软件项目的整体质量。

3.2 引入敏捷开发方法

在软件工程中，加强软件质量管理的一个重要且有效的措施是引入敏捷开发方法。敏捷开发以其灵活性、高效性和用户导向性，为软件质量管理提供了新的思路和手段。

3.2.1 迭代式开发与快速反馈

敏捷开发强调迭代式开发，即将软件开发过程划分为多个短周期（如两周一个迭代），每个迭代结束时都交付可用的软件产品。这种方式使得团队能够迅速获取用户反馈，并根据反馈进行及时调整。快速反馈机制有助于团队及时发现并纠正质量问题，从而确保软件质量的持续改进。

3.2.2 自组织团队与高度协作

敏捷开发鼓励自组织团队，即团队成员根据自身技能和兴趣选择任务，并在迭代过程中进行灵活调整。这种高度协作的模式有助于团队成员之间建立紧密的联系，共同解决问题。同时，团队成员之间的紧密协作也有助于提高代码质量和测试覆盖率，从而降低软件缺陷率。

3.2.3 用户故事与持续交付

敏捷开发使用用户故事作为需求描述方式，强调以用户为中心，确保软件功能满足用户需求。通过持续交付可用的软件产品，团队能够不断获取用户反馈，并根据反馈进行迭代改进。这种用户导向的开发方式有助于确保软件产品的质量和用户体验。

3.2.4 自动化测试与持续集成

敏捷开发强调自动化测试和持续集成，通过自动化测试工具（如Selenium、JUnit等）和持续集成工具（如Jenkins、GitLabCI等），实现代码的自动化构建、测试和部署。这有助于提高测试覆盖率，降低人为错误，确保软件质量的稳定性和可靠性。

3.2.5 培养敏捷文化

引入敏捷开发方法还需要培养敏捷文化。通过培训、分享会等方式，提高团队成员对敏捷开发方法的理解和认识，让他们了解敏捷开发的优点和操作方法。同时，鼓励团队成员积极参与敏捷实践，形成全员参与、共同进步的敏捷文化氛围。

3.3 加强代码审查和测试

为了加强代码审查，团队应建立规范的审查流程，明确审查的标准、方法和责任。审查流程应包括代码提交、审查、反馈和修正等环节，确保每一行代码都经过严格的质量把关。同时，团队还应鼓励成员间进行交叉审查，以提高审查的多样性和全面性。接着，团队应引入自动化测试工具，如单元测试框架、集成测试工具、性能测试工具等，对代码进行全面的测试。自动化测试

不仅能够帮助团队快速发现代码中的缺陷，还能降低人为错误的风险，提高测试的准确性和稳定性。另外，测试覆盖率是衡量测试全面性的重要指标。团队应努力提高测试覆盖率，确保所有关键功能和边界条件都得到充分的测试[3]。这可以通过编写更多的测试用例、采用不同的测试方法（如黑盒测试、白盒测试、灰盒测试等）以及利用测试管理工具来跟踪和管理测试用例等方式来实现。

3.4 建立持续集成和持续部署（CI/CD）流程

在软件工程中，建立持续集成（Continuous Integration，CI）和持续部署（ContinuousDeployment，CD）这一流程旨在自动化软件构建、测试和部署过程，从而确保软件质量的持续提升和快速响应市场变化。（1）明确CI/CD流程的目标和原则。建立CI/CD流程的首要步骤是明确其目标和原则。目标通常包括提高软件构建和部署的效率、降低人为错误的风险、加快产品上市时间以及提升软件质量。原则则包括自动化、频繁集成、快速反馈和持续改进等。（2）选择合适的CI/CD工具。选择合适的CI/CD工具是实现流程自动化的关键。这些工具应支持代码管理、构建、测试、部署和监控等功能，并具有良好的可扩展性和易用性。常见的CI/CD工具包括Jenkins、GitLabCI、CircleCI等[4]。团队应根据自身需求和资源选择合适的工具，并进行必要的配置和定制。（3）设计合理的CI/CD流水线。CI/CD流水线是自动化构建、测试和部署的核心。团队应设计合理的流水线，确保每个阶段都经过严格的质量把关。流水线应包括代码提交触发、构建验证、单元测试、集成测试、安全扫描、部署到测试环境、自动化验收测试以及最终部署到生产环境等环节。（4）实施持续监控和反馈。建立CI/CD流程后，团队应实施持续监控和反馈机制。这包括监控构建和部署过程的状态、收集和分析测试结果、跟踪缺陷的修复情况等。通过持续监控和反馈，团队可以及时发现并解决问题，确保软件质量的持续提升。（5）培养团队对CI/CD流程的认知和接受度。团队应培养对CI/CD流程的认知和接受度。通过培训、分享会等方式，让团队成员了解CI/CD流程的重要性和操作方法。同时，鼓励团队成员积极参与CI/CD流程的实践和改进，形成全员参与、共同进步的良好氛围。

3.5 培养质量文化

3.5.1 树立质量意识

质量文化的培养始于团队成员对质量的深刻认识。团队应明确质量是软件产品的生命线，是赢得用户信任和市场竞争力的关键。通过培训、讲座、内部宣传等方式，强化成员对质量重要性的认识，树立“质量第一”的价值观。

3.5.2 建立质量导向的激励机制

为了激发团队成员对质量的追求，应建立质量导向的激励机制。将质量指标纳入绩效考核体系，对在质量提升方面做出突出贡献的团队成员给予奖励。同时，鼓励团队成员主动发现并修复质量问题，形成全员参与质量管理的良好氛围。

3.5.3 强化质量培训和知识分享

团队应定期组织质量相关的培训和知识分享活动，如软件测试技术、代码审查技巧、质量管理工具使用等。通过培训和学习，提升团队成员的质量管理能力和技能水平。

3.5.4 实践质量原则和方法

团队应在实际工作中践行质量原则和方法，如敏捷开发中的持续集成和持续部署、代码审查和测试驱动开发等。通过实践，让团队成员深刻体会到质量原则和方法带来的好处，从而更加坚定地遵循这些原则和方法。

结语

总之，通过实施一系列有效的质量管理策略和方法可以显著提升软件产品的质量和稳定性。同时，培养质量文化、建立激励机制等软性措施也是提升软件质量管理水平不可或缺的一环。未来，随着技术的不断进步和需求的不断变化，软件质量管理将面临更多的挑战和机遇，我们需要持续关注和研究软件质量管理的新方法、新技术，以适应不断变化的市场环境，推动软件工程事业的持续发展。

参考文献

[1]吴文庆,修雅慧.软件测试在软件开发中应用的探讨[J].职业技术,2019,17(11):83-85.

[2]焦胜男.软件测试在软件开发过程中的应用研究[J].硅谷,2020,7(07):42-43.

[3]朱寅非.软件测试在软件开发过程中的应用探析[J].无线互联科技,2020(05):48+50.

[4]潘祯,吴永强,王艳华.软件测试在软件开发中应用的探讨[J].硅谷,2019(22):156-157.