西安科技大学

**毕业设计(论文)文献翻译**

题 目 基于项目管理流程的校园软件项目协同平台

院、系(部) 计算机科学与技术学院

专业及班级 软件工程1303班

姓 名  任勃

指 导 教 师 史晓楠

日 期 2017.2.26

**中文译文**

**题 目：**软件开发分析

**摘 要：**

尽管有大量的数据和多种类型的指标，但是软件项目还是很难进行预测和风险推断。在本文中，我们提出软件分析，它承诺帮助软件项目的管理者将他们由现有工具产生的丰富信息资源，转变成他们可以执行的见解。 我们讨论分析如何工作，为什么它是一个很好的软件工程，以及为了实现它的承诺必须克服的研究问题。

类别和主题描述符

D.2.9 [Management]

**一般条款：**

管理，人为因素，测量

**关 键 字：**

分析，项目管理

**正 文：**

**1．引言**

软件工程是一个数据非常丰富的活动。 几乎每个项目开发的工件，从代码仓库到测试框架到缺陷数据库，都可以用自动化程度，效率和粒度进行衡量。 项目可以在其整个生命周期内进行测量：从规格到维护。 研究界已经为复杂性，可维护性，可读性，故障倾向和其他有关软件质量和开发过程健壮的重要因素提出了许多指标和模型（例如[7,16]）。

尽管如此，软件开发仍然是有风险的，不可预测的。 重要的开发工作经历大的延迟或故障是很常见的[1]。 此外，软件缺陷每年使美国经济损失数十亿美元[19]。 总之，这些观察意味着，（A）项目经理作出良好决策所需的信息和（B）现有工具提供的信息之间仍然存在实质性的脱节。 从根本上说，问题在于研究界对项目经理的现实世界信息需求没有很好的理解。 事实上，研究已经在很大程度上忽略了管理者的需求，而是专注于开发者的信息需求（例如，[3,14,20]）。

当不能满足数据需求时，由于工具不可用，太难以使用，太难解释，或者它们根本不提供有用或可行的信息，管理者必须主要依赖过去的经验和直觉来进行关键决策。 这种基于直觉的决策有时可以很好地工作，但它们时常是不必要地次优或者甚至是破坏性的[23]。 随着软件项目规模和复杂性的不断增长，决策制定将变得更加困难。 软件工程的信息密集型特性以及我们稍后将讨论的其他特性表明，软件项目管理具有强大的潜力，可以充分利用分析，数据和系统推理来作出决策。 这种以数据为中心的决策风格称为分析。 分析的思想是将潜在的大量数据用于真正的和可操作的洞察。

分析的协作使用已经改变了许多领域的决策[9]。 许多技术社群可能熟悉网络分析[13]。 Web分析利用大量的点击流数据来帮助网站管理者对其业务的许多方面做出明智的决策，从广告到内容布局到投资。 今天，大型网站不仅彻底测试了所有提出的传统意义上的变化，而且还进行详细的分析实验，旨在精确量化任何拟议变化的净效益。 分析对技术，零售和金融等业务产生了深远的影响。

我们相信分析技术对软件开发有很大的希望。 然而，软件提出了许多与代码本身的复杂性相关的新挑战，也涉及项目管理的复杂性。 在本文中，我们讨论了一些重要的研究挑战，必须满足分析之前可以成功应用在这种情况下。

分析共同使用具有革命性的决策在许多领域[ 9 ]。在技术社区许多人可能熟悉网站分析[ 13 ]。网络分析利用大V 点击流数据来帮助网站管理者的容积做出明智的决定，作为广告内容布局投资业务方面。今天，大型网站不仅雷神 全面测试所有提出改变传统意义上的，但他们也进行详细的分析实验的目的是精确地量化提出的任何变化的净效益。分析已经有 深刻的E等从技术到零售金融业务。

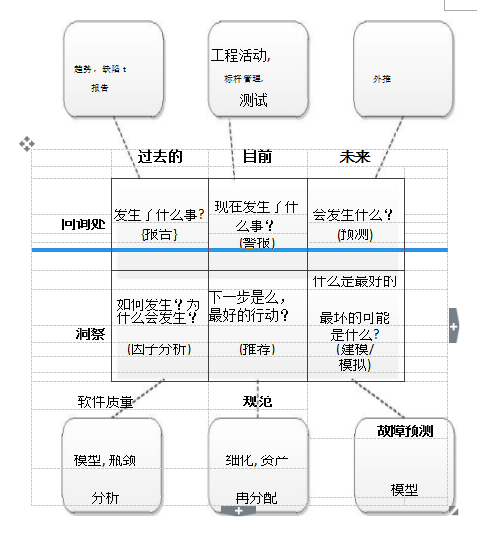
我们相信，分析技术软件开发有很大的希望。软件，但是，提出了一些新的挑战相关的代码本身的复杂性，但也 项目管理的复杂性。在本文中，我们讨论了一些重要的研究挑战，这些研究挑战必须满足在分析之前就可以成功地应用在这种情况下.

首先，我们认为软件项目管理鲜为人知。在第2节中，我们主张对那些在软件项目中做出关键决策的人的信息需求进行新的深入研究。为了帮助管理者，首先了解他们如何工作是至关重要的：他们寻找什么条件？他们可以采取什么行动？第二，我们需要新的和有原则的数据汇总和分析工具，适合项目经理使用。在第4节中，我们讨论当前工具的局限性和未来可能性的一些。最后，在第5节中，我们主张改变发展管理过程本身。特别是我们提出软件项目分析师的职位。这样的位置分享超出管理者和工具范围之外的强大分析所需的分析技能和领域知识。我们首先讨论当前对项目管理的理解，以及为什么仍然不能有效地将分析应用于软件开发。

**2. 项目管理**

项目管理

软件项目管理是一个复杂和广泛定义的职位。项目经理监督和指导一些设计师，开发人员和软件测试人员的工作，同时有时参与这些活动。当工程师专注于代码，架构和性能时，管理人员专注于高级关注：项目的方向，资源的分配，在某些情况下，用户体验，功能集，产品使用方式的细节。管理人员致力于同时满足客户，开发人员，测试人员，维护人员和管理人员提出的（潜在冲突）约束。 Microsoft的经理Steven Sinofsky指出，“记录项目经理的职责的完整列表几乎是不可能的”[22]。项目经理的工作的复杂性导致研究界对设计和评估工具的不便。尤其是管理者的信息需求不是很好理解。 Boehm和Ross提出了一个项目管理理论，其中包括“软件项目风险的十大主要来源和解决这些风险的最有效方法”[6]。虽然这样的前十个列表可以是有启发性的，但问题是所呈现的管理技术（例如，基准测试，组织分析，技术分析等）不是特别足够有用。许多关键问题仍然没有得到答复：哪些可以自动执行？哪些是最重要的？如何呈现结果？他们能做出什么决定？如何评价成功？最近，为了开始回答这些问题，Wallace et al对507个项目经理进行了调查[26]。使用聚类分析来试图识别项目中的风险因素。例如，这项研究发现，“即使是低风险项目也具有高度的复杂性。”这项研究没有产生关于软件指标管理者应该关注的问题的实际答案。

Komi-Sirvio et al注意到软件管理员通常太忙于日常工作，花费大量时间执行测量活动[15]。 通常，数据驱动任务被降级到辅助工作。 我们认为这应该改变。 然而，研究团体需要更多的研究来了解项目经理足够好，以原则的方式为他们寻找新的过程模型和工具。 在下一节中，我们将描述分析的应用如何弥补管理人员与他们作出明智决策所需的数据之间的差距。**图1：分析问题（改编自[ 9 ]）：我们区分信息可直接测量的问题，从了解仔细分析分析出现的问题 并为管理者提供行动依据。**

3. **软件分析**

分析学可以帮助回答管理者提出的他们的项目的重要问题。分析的目标是帮助决策者从数据集中提取重要的信息和见解，否则这些数据集将被隐藏。图1，我们根据Davenport等人[9]确定六个问题领域分析可以帮助回答按时间框架和信息与见解组织。这个想法是区分一些工具已经提供的信息问题（例如，错误数据库中有多少个错误）从洞察问题中提取，这些问题为管理者提供了对项目动态的理解以及作出决策的依据（例如，项目会延迟吗？）。分析的首要目标是帮助管理人员超越信息和洞察力。然而，这种转变不容易。洞察力需要领域的知识，加上识别涉及多个指标的模式的能力。例如，要了解为什么错误报告的数量增加，如果它是一个问题，重要的是要了解其他活动正在发生：下一个版本很快？该团队是否正在处理错误修补或功能添加？分析技术可以帮助管理人员在大规模干草堆数据中快速找到重要的针头。图1中的圆形框提供了分析如何能够很好地适应传统软件工程度量和概念的示例。事实上，软件工程有许多暗示一个适合分析的业务流程的品质，使其更好的应用于分析学：

•数据丰富。当大量的数据可用于分析时，分析运行效果最好。

• 劳动密集型。分析可以利用专业知识，特别是在人才供应短缺，需求周期性，培训时间长的情况下[9]。软件工程特别是劳动密集型。此外，许多研究发现开发商之间的生产力差距的数量级，包括[5,25]。

•时序相关。在业务产品必须满足特定计划的情况下，分析可能很有用;分析使决策者能够查看上游和下游。

•取决于一致性和控制。分析帮助实现一致的决策，即使在异常情况下[9]。

•取决于分布式决策。分析可以帮助决策者理解项目的集体状态，即使面对巨大的地理分布。许多软件项目都具有高度分布式开发，特别是在开源项目中。

•平均成功率低。具有高故障率的域最有可能从分析中获益。软件项目失败率高达33％[1]。

将严格的分析用于软件项目管理有很多潜在的优势（从[9]的启发）。分析可以帮助管理者：  
•监视项目。分析提供了帮助管理员了解复杂项目动态的工具。

•知道什么是真正的工作。分析可以帮助评估对开发过程的变更的效率。例如，它可以测量某些效果是否具有统计意义。

•提高效率。基于分析的技术可以帮助管理者将资源和人才分配到最需要的地方，并在发生利用时识别。

•管理风险。风险模型的效率可以随着数据的数量和精度的增加而改善。分析可以提供数据通道到风险模型和用于以直观的方式传递这样的模型的输出的车辆。

•预测更改。分析可帮助管理员检测和预测数据中的趋势。

•评估过去的决定。基于数据的逻辑和一致决策比以直觉为基础的决策更适合以后的审查和评估。

分析帮助描述一个我们观察到的推理框架有可能适合软件工程。 然而，为了实现这种潜力，显然在研究之外，工具支持也是必要的。 在下一节中，我们将讨论一些目前可用的工具，以及为什么它们还不足以完成此任务。

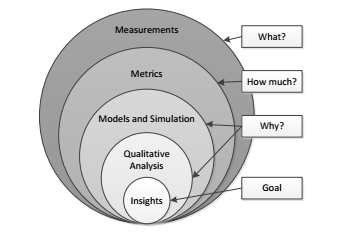


图2：分析范式适应于[ 13 ] ]：分析员的作用是组成多种类型的分析，以制定更完整的景点.

**4. 分析工具**

许多现有的工具为支撑管理而设计。例如，PROM [21]和Hackystat [12]都能够监视和报告大量的软件项目统计。然而，经过七年和十年的发展，在学术界以外，他们都没有显着的采纳水平[10]。一个解释可能是这些工具根本不回答正确的问题[8]。这些工具中的每一个主要关注数据收集，而本身的具有挑战性的问题可能不再是最关键的问题。最近的工具，集中在数据呈现而不是收集，已经得到一些采纳。例如，Microsoft的Team Foundation Server和IBM的Jazz开发环境[11]提供了仪表板视图，旨在使开发人员了解各种事件（如修改，错误和构建结果）的最新状态。然而，最近的一项研究得出结论，虽然这种类型的综合工具可以帮助支持发展过程，但“高层次和低层次意识之间的区别通常不清楚”[24]。虽然现代工具可以呈现来自各种来源的大量数据，但大多数数据集中于数据收集或开发者意识;因为他们没有一个好的模型来满足实际产品经理的需求，实际产品经理通常不使用它们。此外，管理人员可能太忙或可能只是缺乏定量技能或分析专业知识来充分利用先进的分析应用程序，可能包括趋势分析，分类算法，预测建模，统计建模，优化和模拟，以及数据和文本挖掘[9]。一种可能性是，工具应该考虑到这一点：它们将由具有很少的专业知识和巨大的时间约束的管理者消费。然而，另一个有趣的可能性是为软件开发团队增加了一个分析专家。

**5. 软件分析**

分析师是数据创建，收集，解释和使用方面的专家，但也受过相关业务流程的工作培训。考虑我们从Kaushik改编的图2 [13]。在当前的实践中，管理者有时仅仅从测量或度量收集稀疏的见解。其他经理可能主要依赖于定性讨论。分析师的作用是使用定量技能和领域知识来组合许多类型的定量和定性信息，并形成最完整的见解。软件分析师可能被要求对管理者甚至复杂的工具进行过多的研究。例如，Bird et al发现分布式开发在Windows Vista的实现中并没有明显提高软件质量[4]。 Mockus等人的另一项研究枚举了一组可以高度预测缺陷的度量[17]。分析人员可以进行类似的研究，并根据结果规定纠正措施。此外，软件工程研究中的许多发现依赖于大量的上下文变量[2]。因此，这些发现可能无法概括[18,27]。软件分析师对于确定研究团体提供的哪些重要结果适用于特定项目至关重要。我们相信，分析师可以作为软件工程课程的一部分进行培训，修改以强调量化技能，或通过硕士的分析程序，如在北卡罗来纳州立大学推出的软件工程课程。1本文提倡的软件分析研究可能构成在这方面学习的无价指南。

**6.结论**

所有资源，特别是人才，总是受到限制。 这暗示了软件项目经理仔细和仔细决策的重要性。 观察到软件项目仍然是风险和不可预测的，尽管是高度可测量的，意味着更多的分析信息应该用于决策。 在本文中，我们描述了软件分析如何帮助管理者从低级测量转移到对复杂项目的高层次见解。 我们主张对管理者的信息需求和决策过程进行更多研究，以便我们可以构建能够满足这些需求的新工具，并显示关于哪些管理者可以做出更好决策的信息。 最后，我们讨论了软件开发的复杂性如何表明具有定量技能和领域知识的专业分析专业人员可能为未来的项目提供巨大的好处。

**7. 参考文献**

* 1. T. Addison and S. Vallabh. Controlling software project risks: An Empirical Study of the use of empirical project managersIn saicsit 02 page128 { 140，2002。
  2. V. R. Basili, F. Shull, and F. Lanubile. Building knowledge through the experimental family.IEEE software engineering,25:456 { 473，1999。
  3. J. T. Biehl, M. Czerwinski, G. Smith, and G. G. Robertson. fastdash：Develop team awareness of the visual dashboard software。在07，1313页{ 1322，2007。

[4] C. Bird, N. Nagappan, P. Devanbu, H. Gall, andB. Murphy. Does distributed development a ect softwarequality: an empirical case study of windows vista. Commun. ACM, 52(8):85{93, 2009.

1. B. Boehm. Software Cost Estimation with Cocomo II. Addison Wesley, Boston, MA, 2000.
2. B. Boehm and R. Ross. Theory-w software project management principles and examples. IEEE TSE, 15(7):902 {916, jul 1989.
3. R. P. L. Buse and W. R. Weimer. A metric for software readability. In International Symposium on Software Testing and Analysis, pages 121{130, 2008.
4. I. D. Coman, A. Sillitti, and G. Succi. A case-study on using an automated in-process software engineering measurement and analysis system in an industrial environment. In ICSE '09, pages 89{99, 2009.
5. T. Davenport, J. Harris, and R. Morison. Analytics at Work. Harvard Business School Publishing Corporation, Boston, MA, 2010.
6. G. Gousios and D. Spinellis. Alitheia core: An extensible software quality monitoring platform. In ICSE '09, pages 579{582, 2009.
7. IBM Corporation. Jazz. http://www.ibm.com/software/rational/jazz/.
8. P. Johnson, H. Kou, M. Paulding, Q. Zhang, A. Kagawa, and T. Yamashita. Improving software development management through software project telemetry. IEEE Software, 22(4):76 { 85, july-aug. 2005.
9. A. Kaushik. Web Analytics 2.0. Wiley Publishing, 2010.
10. A. J. Ko, R. DeLine, and G. Venolia. Information needs in collocated software development teams. In ICSE'07: Proceedings of the International Conference on Software Engineering, pages 344{353, 2007.
11. S. Komi-Sirvi, P. Parviainen, and J. Ronkainen. Measurement automation: Methodological background and practical solutions-a multiple case study. In IEEE International Symposium on Software Metrics, page 306, 2001.
12. T. J. McCabe. A complexity measure. IEEE Trans. Software Eng., 2(4):308{320, 1976.
13. A. Mockus, P. Zhang, and P. L. Li. Predictors of customer perceived software quality. In ICSE '05, 225{233, 2005.
14. I. Myrtveit, E. Stensrud, and M. Shepperd. Reliability and validity in comparative studies of software prediction models. IEEE Trans. Softw. Eng., 31(5):380{391, 2005.
15. National Institute of Standards and Technology. The economic impacts of inadequate infrastructure for software testing. Technical Report 02-3, Research Triangle Institute, May 2002.
16. J. Sillito, G. C. Murphy, and K. De Volder. Questions programmers ask during software evolution tasks. In

SIGSOFT '06/FSE-14, pages 23{34, 2006.

1. A. Sillitti, A. Janes, G. Succi, and T. Vernazza. Collecting, integrating and analyzing software metrics and personal software process data. In EUROMICRO, page 336, 2003.
2. S. Sinofsky. Steven sinofsky's microsoft techtalk / pm at microsoft. http://blogs.msdn.com/b/techtalk/archive/ 2005/12/16/504872.aspx, Dec. 2005.
3. L. Strigini. Limiting the dangers of intuitive decision making. IEEE Software, 13:101{103, 1996.
4. C. Treude and M.-A. Storey. Awareness 2.0: staying aware of projects, developers and tasks using dashboards and feeds. In ICSE '10, pages 365{374, 2010.
5. J. D. Valett and F. E. McGarry. A summary of software measurement experiences in the software engineering laboratory. Journal of Systems and Software, 9(2):137 { 148, 1989.
6. L. Wallace, M. Keil, and A. Rai. Understanding software project risk: a cluster analysis. Inf. Manage., 42(1):115{125, 2004.
7. T. Zimmermann, N. Nagappan, H. Gall, E. Giger, and

B. Murphy. Cross-project defect prediction. In Symposium on the Foundations of Software Engineering, August 2009.