如何保证软件测试的质量

软件质量问题由来已久。早在二十世纪六十年代，人们就发现由于软件质量低下、可靠性差所带来的不仅是维护费用高昂，而且还会造成严重的后果。随着计算机技术的发展，软件的规模正变得越来越大、复杂性越来越高，而软件的开发能力却远远跟不上应用需求的迅速扩展，出现了所谓的“软件危机”。时至今日，计算机在工业控制、医疗、通信、交通、航天、航空、经济、金融等领域的应用，对软件的质量提出了前所未有的挑战。计算机科学家们认为人类目前所掌握的软件开发技术尚不能满足在这些应用领域中所提出的极高的质量要求。软件失效逐渐成为系统瘫痪的主要原因。

软件测试是提高软件质量的有效途径。根据Boehm的统计，在软件开发总成本中，用在测试上的开销要占30％到50％。但是，就目前软件工程发展的状况而言，软件测试又是相对于其他研究方向而言较为薄弱的一个方面。所以有必要开展软件测试质量度量理论的研究，以使之成为选择高效软件测试方法的指导。

软件测试是为了查找软件错误而执行某个程序的过程，目的是尽可能发现并改正被测软件的错误，提高软件的可靠性。在查找软件错误的过程中，对过程的管理犹为重要。一个成功的软件测试项目，离不开对软件测试过程科学的组织和监控，对软件测试过程进行管理已成为软件测试成功的重要保证。然而，有效地管理软件测试过程的质量状态离不开对软件测试过程的度量。我们通过使用一些度量方法来量化软件测试过程，并利用度量数据结果对测试项目进展做出评估，从而实现对软件测试过程的有效管理，在尽可能节约资源的情况下提高软件测试的效率。

目前，软件度量已经发展成为一门至关重要的软件工程学科，随着人们对软件度量认识的逐步深入，度量在软件测试中应用的广度和深度也得到了前所未有的展。从宏观的角度来看，应当有一个全局的框架模型对过程度量结果进行分析，为测试管理提供客观标准。所以，利用基于软件测试过程度量结果，可以实现工作量／成果模型(effort／outcome model)在软件测试过程中的质量管理。

首先，要进行过程度量的构造。在测试阶段，主要的过程度量有缺陷密度、测试用例深度、有效性、测试执行的效率、缺陷报告的质量、测试覆盖度、测试环境的稳定性或有效性等。在对测试过程进行度量时，不能只用度量数据来控制过程。度量数据必须恰当地代表所要控制的过程，而且它们必须被充分地、很好地定义，能够反映实际的变化，以便作为决策的基础。过程的度量对于项目开发的逐步改进是非常必要的，必须仔细地计划和准备，否则结果将会不尽人意。使用良好定义的度量有如下好处：度量记录了满足需要的信息；使用具有多种用途的基本度量可以减少冗余信息；充分定义度量方法的所有基本方面可以增加度量的准确性和重复性；通过度量能够最大化其价值；度量的各个层次(基本度量、导出度量和指示器的模式)易于标志、重用和修改。

如前所述，软件缺陷是软件产品的固有成分，它的存在对软件功能的实现是一个潜在威胁。为了保证软件质量，对软件缺陷进行分类管理能使所有类型的缺陷得到适当的处理。软件缺陷的分类方法众多，各种分类方法目的不同、观察问题的角度和复杂程度也不一样。由IBM 公司提出的缺陷正交分类 0DCL (Orthogonal Defects classification)对缺陷的类型分类细致，适用于缺陷的定位、排除、缺陷原因分析和缺陷预防活动。它用多个属性来描述缺陷特征，包括：发现缺陷的活动、缺陷影响、缺陷引发事件、缺陷载体(Target)、缺陷年龄、缺陷来源、缺陷类型和缺陷限定词。这 8 个属性对缺陷的消除和预防起到关键作用。0DC 对缺陷类型分为 7 大类：赋值、检验(Checking)、算法、时序、接口、功能、关联(Relationship)。缺陷严重等级与缺陷所属类型有一对多的对应关系，缺陷严重等级分为严重缺陷、较大缺陷、较

小缺陷和轻微缺陷 4 种。综上所述，根据缺陷类型和缺陷严重等级定义，前者是对缺陷的类型进行划分，后者是给缺陷一个定位，以便于给缺陷赋予权值进行相关度量操作。对缺陷的度量一般以每千行代码的缺陷数(Defects／KL0C)进行测量，称为缺陷密度。缺陷密度仅仅能反映每千行代码的缺陷数量， 并不能反映缺陷的严重程度。在本文中，结合工作量／成果模型，按照测试效率／缺陷程度对测试质量进行管理。测试效率的度量有多种方法，如单位时间内所执行的测试用例数、单位时间所发现的缺陷数、单位时间内所测试的代码量等。本文采用单位时间内所测试的代码量进行测试效率的度量，将测试阶段所发现的缺陷与其权值关联起来以反映缺陷的严重程度。所需要的度量数据有测试人员在测试阶段发现的各种类型的缺陷数 N、缺陷类型、测试周期丁以及代码行数 L。利用以上基本度量数据，可以方便计算缺陷严重程度和测试效率 TE (Testing Effectiveness)：

TE =一定周期内测试的代码行数 L／测试周期 T

其次，为了进行良好的测试管理，可以使用有效的度量和工作量／成果模型(Effort／Outcome Paradigm．)进行有效的测试过程质量管理。在软件测试过程中，度量分为两大类：第一类是衡量测试效率和测试工作量的工作量指标，如测试效率评价、测试进度 S 曲线等；第二类是从质量的角度表明测试结果的结果指标，如缺陷数量、到达模式、系统崩溃和挂起次数等。在工作量／成果模型 L7 中，过程度量中某些度量一起使用可以充解释过程中质量的状态，利用这些度量构造软件测试质量管理模型的目的就是监督和管理软件测试的质量。如图 1 所示的 2×2 矩阵，它是关于缺陷和测试效率的度量，通过这些度量实现软件测试的管理。如何使用质量管理模型进行软件测试质量管理呢?软件质量管理模型必须提供警告或改进的早期标志，以便可以计划和实施及时的措施。对于过程中质量管理，数据点可以表明当前版本的质量趋势，从而可以采取及时措施。当追踪模型的缺陷时，较低的真实缺陷级别可能是较高的代码质量或测试工作不好的结果，较高的真实缺陷可能是较低质量的代码或测试工作较好的结果。如何知道是哪种情况呢(更好的测试工作、更高的错误植入、较低的错误植入或较差的测试工作)?需要其他指标结合到模型的上下文中，从而更好地解释数据，这个指标就是测试工作的度量，用来作为测试过程执行严格度的一个代替指标。这个度量与测试缺陷级别相结合，可以提供有用的缺陷模型的解释。真实数据和模型数据进行比较，或比较产品的当前和以前版本，模型数据可以采用历史数据。

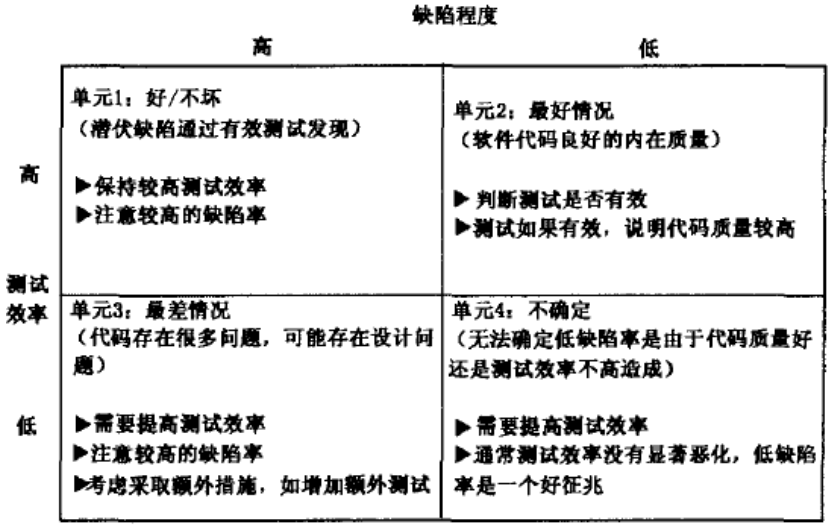

图 1 缺陷级别/缺陷效率矩阵

在测试效率／缺陷程度的工作量／成果模型中，单元 2 和单元 1 是属于较好的情景。单元 2 是最好的情况：较低的缺陷级别和较高的测试效率，说明软件代码有着良好的内在质量。

单元 1 是一种好的或者说不坏的情况，虽然其测试效率较高但是缺陷级别也较高。负面情景是单元 3 和单元 4。单元 4 有着较低的缺陷级别和较低的测试效率， 单元 3 是最差的情况，缺陷级别较高、测试效率较低。单元 3 和单元 4 从质量管理的角度来讲都是不可接受的，无论是在项目全局的层次上还是项目过程中状态层次上讲都需要改善这两种情况，需要采取相应的质量管理措施。从项目全局的层次上来讲，假设项目仍然处于开发早期，可以适当提高测试效率。而当测试已接近结束时，则需要增加额外的测试移除代码潜在的缺陷。

本文中将软件过程度量引入测试质量管理中， 为考查软件代码质量和项目的进展情况提供参考信息。在软件测试过程的度量中，重要的是度量的有效性而非度量的数量。度量虽然易于建立，但是，不好的度量不仅没有用处，实际上还会导致相反的效果，并增加项目开销。所以，在建立度量的过程中要注意每个度量都要遵循测试理论的基本原则，并能反映经验值。需对概念、执行中的定义、测量单位、有效性和可靠性问题进行认真地考虑。本文对缺陷权值的确定可能带有一定的主观因素，这就要求我们必须对度量和数据有一个良好的理解，才能做出正确的决策。当然，在软件测试质量管理中的数据分析部分还有待进一步挖掘，我们还可以提出更多与软件测试质量有关的问题，并找到它们的答案，这也是以后的工作重点。

刘晗旭　　14126116