第十五章 质量保证

王美红

主要内容

- 什么是软件质量保证
- 软件质量保证的要素
- 软件质量保证的任务、目标和度量
- 软件质量保证计划

实现软件质量的四大管理和实践活动

- 软件工程方法
- 项目管理技术
  - 交付日期可达、进度依赖关系清楚、进行了风险规划……
- 质量控制活动
  - 包括一套软件工程活动，帮助确保每个工作产品符合其质量目标，如检查代码，应用一系列的测试步骤
- 软件质量保证
  - 建立基础设施，以支持坚实的软件工程方法，合理的项目管理和质量控制活动。

15.1什么是软件质量保证（SQA）

- 是适用于整个软件过程的一种普适性活动
- 是为了保证软件高质量而必需的“有计划的、系统化的行动模式”
- 各个参与者都对软件质量负有责任—包括软件工程师、项目管理者、客户、销售人员和SQA小组成员。
- SQA小组充当客户在公司内部的代表，也就是说SQA小组成员必须从客户的角度来审查软件。

15.1什么是软件质量保证（续）

- 软件质量保证(SQA)包括: (1)SQA过程，(2)具体的质量保证和质量控制任务（包括技术评审和多层次测试策略）；(3)有效的软件工程实践（方法和工具）；(4)对所有软件工作产品及其变更的控制；(5)保证符合软件开发标准的规程;(6)测量和报告机制。

15.2 软件质量保证的要素

- **标准**：IEEE、ISO及其他标准化组织制定了一系列广泛的软件工程标准和相关文件。软件质量保证的任务是要确保遵循所采用的标准，并保证所有的工作产品符合标准。

- **评审和审核**：技术评审是由软件工程师执行的质量控制活动，目的是发现错误。审核是一种由SQA人员执行的评审，意图是确保软件工程工作遵循质量准则。如对评审过程审核。

15.2 软件质量保证的要素（续）

- **测试**：软件测试是一种质量控制功能，它有一个基本目标——发现错误。SQA的任务是要确保测试计划适当和实施有效，以便最有可能实现软件测试的基本目标。

- **错误/缺陷的收集和分析**：改进的唯一途径是衡量如何做。软件质量保证人员收集和分析错误和缺陷数据，以便更好地了解错误是如何引入的，以及什么样的软件工程活动最适合消除它们。

15.2 软件质量保证的要素（续）

- **变更管理**：变更是对所有软件项目最具破坏性的一个方面。如果没有适当的管理，变更可能会导致混乱，而混乱几乎总是导致低质量。软件质量保证确保进行足够的变更管理实践。
- **教育**：每个软件组织都想改善其软件工程实践。改善的关键因素是对软件工程师、项目经理和其他利益相关者的教育。软件质量保证组织牵头软件过程改进，并是教育计划的关键支持者和发起者。

15.2 软件质量保证的要素（续）

- 供应商管理：可以从外部软件供应商获得三种类型的软件：（1）简易包装软件包(例如微软Office)；（2）定制外壳(通过可以根据购买者需要进行定制的基本框架结构)；（3）合同软件(按客户公司提供的规格说明定制设计和构造)。软件质量保证组的任务是，通过建议供应商应遵循的具体的质量做法，并将质量要求作为与任何外部供应商签订合同的一部分，确保高质量的软件成果。

15.2 软件质量保证的要素（续）

- 安全防卫：软件质量保证确保应用适当的过程和技术来实现软件安全。
- 安全：软件质量保证可能负责评估软件失效的影响，并负责启动那些减少风险所必需的步骤。
- 风险管理：软件质量保证组应确保风险管理活动适当进行，且已经建立风险相关的应急计划。

15.3 软件质量保证的任务、目标和度量

- 软件质量保证是由与两个不同人群相联系的多种任务组成：
  - 技术工作的软件工程师
  - 负有质量策划、监督、记录、分析和报告责任的软件质量保证组。
- 软件工程师通过采用可靠的技术方法和措施，进行技术评审，并进行周密计划的软件测试来获得质量。

软件质量保证任务

- 编制项目质量保证计划。
- 参与项目的软件过程描述的编写。
- 评审软件工程活动，以验证是否符合规定的软件过程。
- 审核指定的软件工作产品以验证是否遵守作为软件过程一部分的那些规定。
- 确保根据文档化的规程记录和处理软件工作和工作产品中的偏差。
- 记录各种不符合项并报告给高层管理人员。

目标、属性和度量

- 需求质量: 软件质量保证必须确保软件团队严格评审需求模型，以达到高水平的质量。
- 设计质量:软件团队应该评估设计模型的每个元素，以确保设计模型显示出高质量，并且设计本身符合需求。SQA寻找能反映设计质量的属性。

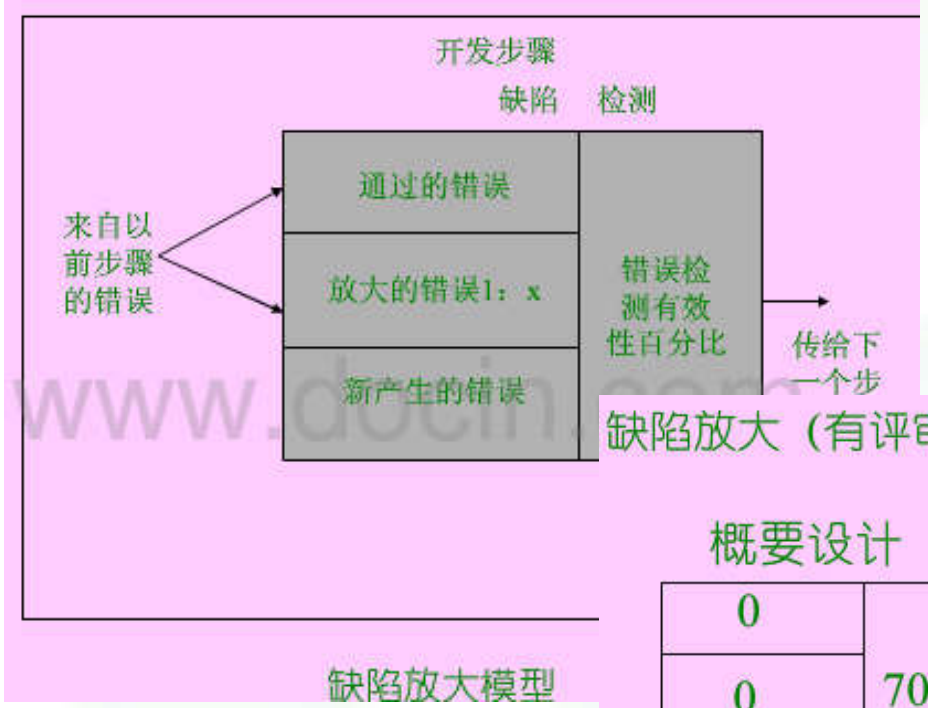
目标、属性和度量

- 代码质量。源代码和相关的工作产品必须符合本地的编码标准，并显示出易于维护的特点。SQA应该找出那些合理分析代码质量的属性。
- 质量控制有效性。软件团队应使用有限的资源，在某种程度上最有可能得到高品质的结果。SQA分析在评审和测试上的资源分配，评估是否以最有效的方式进行分配的。

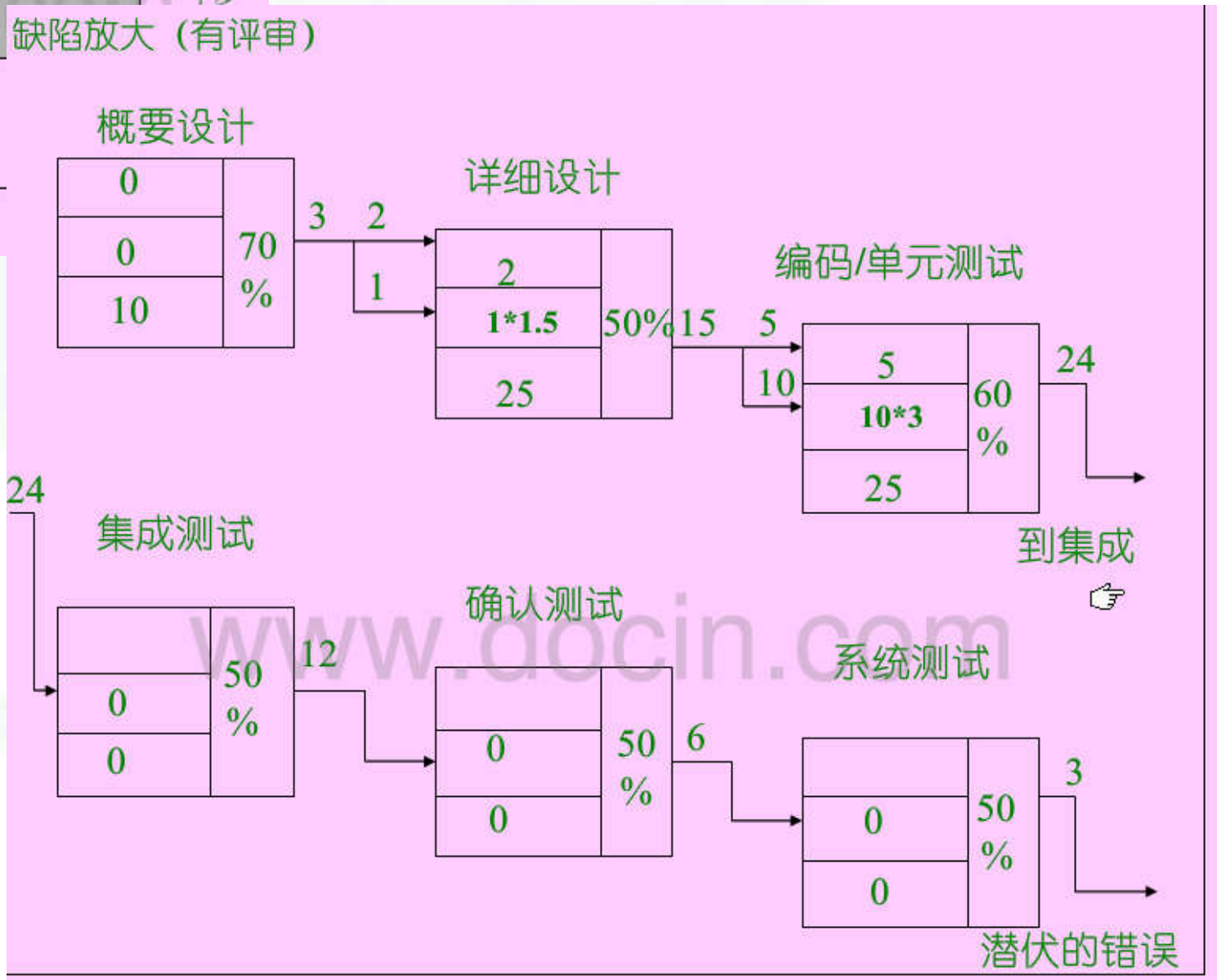
目标	属性	度量
需求质量	歧义	含糊修饰词的数量（例如：许多、大量、与人友好）
	完备性	TBA 和 TBD 的数量
	可理解性	节 / 小节的数量
	易变性	每项需求变更的数量
		变更所需要的时间（通过活动）
	可追溯性	不能追溯到设计 / 代码的需求数
	模型清晰性	UML 模型数
		每个模型中描述文字的页数
		UML 错误数
设计质量	体系结构完整性	是否存在现成的体系结构模型
	构件完备性	追溯到结构模型的构件数
		过程设计的复杂性
	接口复杂性	挑选一个典型功能或内容的平均数
		布局合理性
	模式	使用的模式数量
代码质量	复杂性	环路复杂性
	可维护性	设计要素（第 23 章）
		内部注释的百分比
	可理解性	变量命名约定
	可重用性	可重用构件的百分比
	文档	可读性指数
质量控制效率	资源分配	每个活动花费的人员时间百分比
	完成率	实际完成时间与预算完成时间之比
	评审效率	参见评审度量
	测试效率	发现的错误及关键性问题数
		改正一个错误所需的工作量
		错误的根源

图 16.1 软件质量的目标、属性和度量 [Hya96]

缺陷的方法和消除



缺陷放大模型



技术评审：
正式、非正式

15.4 统计软件质量保证

- 统计质量保证反映了一种在产业界不断增长的趋势：质量的量化。对于软件而言，统计质量保证包含以下步骤：

  1.收集软件的错误和缺陷信息，并进行分类。

  2.追溯每个错误和缺陷形成的根本原因。

  3.使用Pareto原则(80%的缺陷可以追溯到所有可能原因的20%)，将这20%原因分离出来。

  4.一旦找出这些重要的少数原因，就可以开始纠正引起错误和缺陷的问题。

15.4 统计软件质量保证

- 将所有发现的问题追溯原因：
- 不完整或错误的规格说明（IES）。
- 与客户交流中所产生的误解（MCC）。
- 故意违背规格说明（IDS）。
- 违反程序设计标准（VPS）。
- 数据表示有错（EDR）。
- 构件接口不一致（ICI）。
- 设计逻辑的错误（EDL）。
- 不完整或错误的测试（IET）。
- 不准确或不完整的文档（IID）。
- 将设计转换为程序设计语言实现时的错误（PLT）。
- 不清晰或不一致的人机界面（HCI）。
- 其他（MIS）。

	总计		严重		中等		微小	
错误	数量	百分比	数量	百分比	数量	百分比	数量	百分比
IES	205	22%	34	27%	68	18%	103	24%
MCC	156	17%	12	9%	68	18%	76	17%
IDS	48	5%	1	1%	24	6%	23	5%
VPS	25	3%	0	0%	15	4%	10	2%
EDR	130	14%	26	20%	68	18%	36	8%
ICI	58	6%	9	7%	18	5%	31	7%
EDL	45	5%	14	11%	12	3%	19	4%
IET	95	10%	12	9%	35	9%	48	11%
IID	36	4%	2	2%	20	5%	14	3%
PLT	60	6%	15	12%	19	5%	26	6%
HCI	28	3%	3	2%	17	4%	8	2%
MIS	56	6%	0	0%	15	4%	41	9%
总计	942	100%	128	100%	379	100%	435	100%

图 16-2　统计 SQA 数据收集

表中显示IES、MCC和EDR是造成所有错误的53%的“重要的少数”原因。但是需要注意，在只考虑严重错误时，应该将IES、EDR、PLT和EDL作为“重要的少数”原因。一旦确定了这些重要的少数原因，软件开发组织就可以开始采取改正行动了。

- 已经证明统计软件质量保证技术确实使质量得到了提高。在某些情况下，应用这些技术后，软件组织已经取得每年减少50%缺陷的好成绩。
- 统计SQA及Pareto原则的应用可以用一句话概括：将时间用于真正重要的地方，但是首先你必须知道什么是真正重要的！
- 六西格玛是目前产业界应用最广泛的基于统计的质量保证策略。

15.5 软件可靠性

- 软件可靠性是指在特定环境和特定时间内，计算机程序正常运行的概率。
- 失效意味着与软件需求的不符。失效可能仅仅是令人厌烦的，也可能是灾难性的。
- 让问题更加复杂的是，纠正一个失效事实上可能会引入其他的错误，而这些错误最终又会导致其他的失效。

可靠性和可用性的测量

- 考虑基于计算机的系统时，可靠性的简单测量是“平均失效间隔时间”MTBF：

  MTBF = MTTF + MTTR

  – 其中MTTF和MTTR分别是“平均失效时间” 和“平均维修时间”。

- MTBF可能会产生问题的原因有两个：(1)它突出了失效之间的时间跨度，但不会为我们提供一个凸显的失效率；(2)MTBF可能被误解为平均寿命，即使这不是它的含义。

可靠性和可用性的测量（续）

- 可靠性另一个可选的衡量是失效率（FIT）
  - 一个部件每10亿机时发生多少次失效的统计测量。1FIT相当于每10亿机时发生一次失效。
- 软件可用性是指在某个给定时间点上程序能够按照需求执行的概率。定义为：
  - 可用性 = MTTF / (MTTF + MTTR) * 100%

安全性

- 软件安全是一种质量保证活动，它主要用来识别和评估可能对软件产生负面影响并促使整个系统失效的潜在灾难。
- 如果能够在软件过程的早期阶段识别出这些灾难，就可以指定软件设计特性消除或控制这些潜在的灾难。
- 建模和分析过程可以视为软件安全的一部分。开始时，根据危险程度和风险高低对灾难进行识别和分类。

安全性（续）

- 一旦识别出这些系统级的灾难，就可以运用分析技术来确定这些灾难的严重性和概率。为了达到高效，应该将软件置于整个系统中进行分析。
- 例如，一个微小的用户输入错误(如踩下刹车）有可能会被软件错误放大，产生将机械设备置于不正确位置的控制数据，此时当且仅当外部环境条件满足时，机械设备的不正确位置将引发灾难性的失效。

安全性（续）

- 一旦完成了灾难识别和分析，就可以进行软件中与安全相关的需求规格说明了。在规格说明中包括一张不希望发生的事件清单，以及针对这些事件所希望产生的系统响应。这样就指明了软件在管理不希望发生的事件方面应起的作用。

15.6 ISO 9000质量标准

- ISO9001:2008认证：
  - 描述的质量要求涉及管理者责任、质量体系、合同评审、设计控制、文件和资料控制、产品标识与可追溯性、过程控制、检验和试验、纠正及预防措施、质量记录的控制、内部质量审核、培训、服务以及统计技术等主题。
  - 软件组织要登记为ISO 9001:2008认证，就必须针对上述每个方面的质量要求制定相关的政策和规程，并且有能力证明组织活动的确是按照这些政策和规程实施的。

15.7 软件质量保证计划

- SQA计划为软件质量保证提供了一张路线图。该计划由SQA小组制定，作为各个软件项目中SQA活动的模板。
- IEEE公布的标准建议SQA计划应包括：(1)计划的目的和范围；(2)SQA覆盖的所有软件工程工作产品的描述；(3)应用于软件过程中的所有适用的标准和习惯做法；(4)SQA活动和任务（包括评审和审核）以及它们在整个软件过程中的位置；(5)支持SQA活动和任务的工具和方法；(6)软件配置管理的规程；(7)收集、保护和维护所有SQA相关记录的方法；(8)与产品质量相关的组织角色和责任。

15.8 产品度量框架

- 软件业界的一些人始终在争论：软件是不可测量的，或者说，测量的尝试应该推迟，直到我们对软件以及用于描述软件的属性有较好的理解。这种说法是错误的！

- 测量是任何工程过程的一个关键环节。
- 使用测度以较好地理解所创建模型的属性，评估所制造工程产品或系统的质量。
- 与其他工程学科不同，软件工程并不是建立在基本的物理定量定律上。直接测度在软件世界是不常见的。由于软件测度与度量经常是间接得到的，因此有广泛的争论空间。
-

- “测度”一词可用作名词，也可用作动词。在软件工程中，“测度”为产品或过程的某些属性的程度、数量、维数、容量或大小提供量化的指示。
- “测量”是确定测度的动作。
- 度量在《IEEE标准词汇表》中定义为：度量是一个系统、构件或过程具有给定属性的量化测量程度。

- 当收集了一个数据点，就已建立了一个测度。收集一个或多个数据点，由此产生测量。软件的度量以某种方式与单个测度相关。
-

产品度量的挑战

- 在过去的30年中，许多研究人员试图开发一种能为软件复杂性提供全面测量的度量。尽管已提出了许多复杂性测量，但每种方法都对什么是复杂性以及哪些系统属性导致复杂性持有不同的看法。
- 然而，仍有必要去测量和控制软件的复杂度。若这个质量度量的单一值难以获取的话，针对不同内部程序属性开发测度应该是可能的。这些测量和由此产生的度量可用作分析模型和设计模型的独立指标。

测量的原则

- 度量应该具有引人的数学特性。
- 当度量代表一个软件特征时，当正向品质出现时特征值提高，当不理想品质出现时特征值下降。度量值应该以同样的方式增加或减小。
- 每种度量在发布或用于做决策之前，应该在广泛的环境中根据经验加以确认。

- 尽管公式化、特征化和确认是关键的，但收集和分析是驱动测量过程的活动。[ROC94]为这些活动提供了以下指导原则：(1)只要有可能，数据的收集与分析应能自动化地进行；(2)应该使用有效的统计技术以建立内部产品属性与外部质量特性之间的关系；(3)应该为每个度量建立解释性指导原则和推荐建议。
-

面向目标的软件测量

- 目标定义模块可用于定义每个测量目标。模板采取以下形式：

  在…{进行测量的环境}…环境中，从…{对测量感兴趣的人}…的角度，关于…{活动或属性被考虑的方面}…的方面，为…{分析的总体目标}…目的，分析…{将要测量的属性和活动名}…。

-

- 考虑SafeHome软件体系结构，目的是评估体系结构构件，涉及方面为使SafeHome具有较强可扩展性的能力，视角为完成该工作的软件工程师，环境为后续三年的产品改进。
- 明确定义了测量目标之后，形成一组问题。回答这些问题有助于软件团队(或其他共利益者)确定是否已达到测量目标。可能会问到的问题如下：

  Q1：体系结构构件是否将以功能与数据分开的方式描述？

  Q2：每个构件的复杂性是限定在一定的范围内以便于修改与扩展？

- 每个问题都应该利用一个或多个测度和度量以量化的方式回答。

有效软件度量的属性

- [Ejl91]定义了一组有效软件度量所应具有的属性。导出的度量及导出度量的测度应该是：
  - 简单的和可计算的。
  - 在经验上和直觉上有说服力。
  - 一致的和客观的。
  - 单位与量纲的使用是一致的。
  - 编程语言的独立性。
  - 高质量反馈的有效机制。
- 一些常用的软件度量可能不满足其中某些属性，如功能点方法。不同人计算结果可能不同，但是仍然很有用。

作业

- 1. 为什么软件工程小组和独立的软件质量保证小组之间的关系经常是紧张的？这种紧张关系是否是正常的？
- 2. 除了可以统计错误和缺陷之外，还有哪些可以统计的软件特征是具有质量意义的？他们是什么？是否可以直接测量？