软件质量保证与测试

Software Quality Assurance and Testing

第 9 章 软件质量与软件质量管理

金陵科技學院

主要内容

- 软件错误分类与预测
- 软件质量的基本概念
- 软件质量模型与度量
- 软件质量保证活动

软件质量问题引发的挑战

软件质量问题引发的事故屡见不鲜

① ATM 机“新年红包”

2009年英国曼彻斯特郡，有一台ATM 机，实际出款额是取款额的2倍。许多人闻讯赶来排队取款。6小时内被取走了1万英镑。

故障原因：ATM机程序错误。

软件质量问题引发的挑战

软件质量问题引发的事故屡见不鲜

②1999年全球计算机发生“千年虫”问题。

问题的原因是采用2位数表示年份，当进入新的世纪时，无法有效区分年份。

00年= { 1900 年 / 2000 年


惠州破解“千年虫”
计算机终端设备扩展卡横空出世

据美国一家顾问公司估计，全球花在防备千年虫发作上的费用高达6000亿美元。

软件质量问题引发的挑战

软件质量问题引发的事故屡见不鲜

③ 2007年美国洛杉矶机场航空管理软件故障，导致1.7万人滞留机场。

④ 1996年欧洲宇航局阿丽亚娜 5 型火箭发射中爆炸，发射失败，并炸死 2 名法国士兵，后经查明为控制软件设计存在缺陷。

…… ……

程序正确性的不同标准

① 程序编写得无语法错误

② 程序执行中未发现明显的运行错误

③ 程序中无不适当的语句

④ 程序运行时能通过典型的有效测试数据，而得到正确的预期结果

⑤ 程序运行时能通过典型的无效测试数据，而得到正确的结果

⑥ 程序运行时能通过任何可能的数据，并给出正确结果

程序正确性的不同标准

标准不同，要求就不同

① 程序编写得无语法错误

——程序通过编译即可达到

…… ……

⑥ 程序运行时能通过任何可能的数据，并给出正确结果

——要达到很不容易！

软件错误的分类

按照错误的来源分类：

① 软件需求错误

② 功能和性能错误

③ 软件结构错误

④ 数据错误

⑤ 软件实现和编码错误

⑥ 软件集成错误

⑦ 软件系统结构错误

⑧ 测试定义与测试执行错误

软件错误的分类

为了分析软件错误的来源，有人做了大量的统计工作。例如，对某一个包含有687, 7000行代码的软件，在进行完单元测试、集成测试和系统测试之后，经统计共发现错误1, 6209个，平均每千行代码有错误2.36个，将发现的错误进行分类，具体情况如下表。

错误分类	错误数	百分比
软件需求错误	1317	8.1
功能和性能错误	**2624**	**16.2**
软件结构错误	**4082**	**25.2**
数据错误	**3638**	**22.4**
软件实现和编码错误	1601	9.9
软件集成错误	1455	9.0
软件系统结构错误	282	1.7
测试定义与测试执行错误	447	2.8
其他类型错误	763	4.7

软件错误的分类

按照错误后果的严重程度：

① 较小错误

② 中等错误

③ 较严重错误

④ 严重错误

⑤ 非常严重错误

⑥ 最严重错误

软件错误的分类

注意：错误本身的大小与其所导致后果的严重程度并不成比例，例如1963年美国金星探测火箭的飞行控制软件中，有一段FORTRAN语句，其中：

DO 5 I=1，3

被误写成：

DO 5 I=1.3

这里只是将“，”错写成“.”，结果这一疏忽竟造成极为严重的后果，火箭发射失败，损失1千万美元！

程序中隐藏错误数量估计

由于我们无法对软件进行穷举测试，所以即使是经过测试的软件，其中也会有隐藏的错误。但是利用测试的统计数据，我们可以对软件中隐藏的错误数量进行估计，当然这种估计是建立在合理的模型基础上的。

程序中隐藏错误数量估计

1. 撒播模型（Seeding Models）

问：如何估计一个池塘中鱼的总数？

答：撒一网捕捞上来Nt尾，作上标记，放入池中，等其与未作标记的鱼充分混合，几天以后，再从池中任意取出一些鱼样，得到带标记者nt尾，无标记者n尾。

- 则有等比关系：

$$\frac{N_t}{N+N_t}=\frac{n_t}{n+n_t}$$

- 池中鱼总数N的估计值

$$\widehat{N}=\frac{n}{n_t}N_t$$

程序中隐藏错误数量估计

1. 撒播模型（Seeding Models）

模仿上述方法，假设在开始测试以前，软件中的错误数为 N，首先，往程序中人为插入N_S个错误，经过一段时间的测试工作以后，发现的错误可以分为两类，一类属于人为插入的错误n_s，另一类是软件中原来就有的错误n，则软件中错误数的估算值为：

$$\widehat{N} = \frac{n}{n_s} N_s$$

程序中隐藏错误数量估计

1. 撒播模型（Seeding Models）

这一方法，在应用时存在两个实际困难：

① 人为植入错误很困难

② 错误的难易程度不一样，
估算结果不一定准确。

程序中隐藏错误数量估计

2. Hyman分别测试法

张三和李四两人各自都到南京来玩，他们每人各选取了5个景点参观游览，如果他们各自选取的 5 个景点重合度高说明什么问题？重合度低又说明什么问题？

程序中隐藏错误数量估计

2. Hyman分别测试法

由两个测试员同时互相独立地测试同一程序的两个副本，用 t 表示测试时间，记 t = 0时，程序中原有错误总数是B_0；记 t = t1时，

B_1：第一个人发现的错误

B_2：第二个人发现的错误

b_c：两人都发现的错误

b_1：第二个人发现的不在B_1中的错误

$B_2=b_c+b_1$

程序中隐藏错误数量估计

2. Hyman分别测试法

由比例式：

$$\frac{B_1}{B_0} = \frac{b_c}{b_c + b_1}$$

推算可得程序中原有错误总数B_0：

$$\widehat{B}_0 = \frac{B_2}{b_c} B_1$$

程序中隐藏错误数量估计

3. 回归分析

有变量 X 和 Y，假设我们事先并不知道它们之间的定量关系，但有几组数据， X =1时 Y=2，X =3时 Y=6，X =5时 Y=10，根据这些数据，我们很容易推出 Y=2X。

这就是一种最为简单的回归分析。

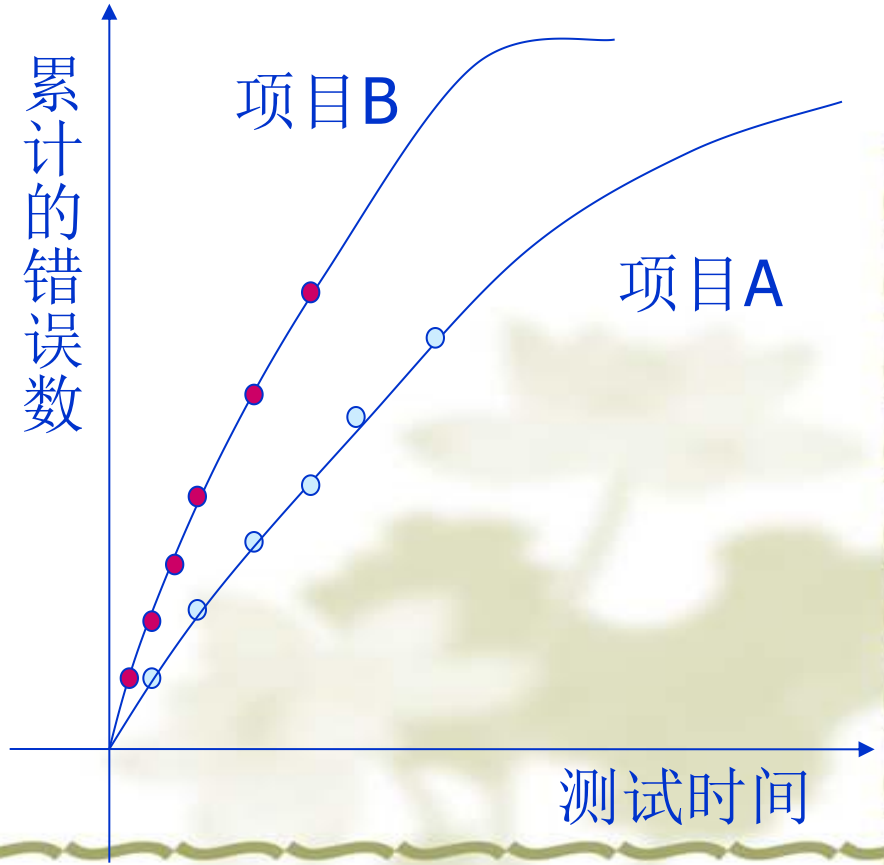
回归分析（regression analysis）指的是确定两个或两个以上变量间定量关系的一种统计分析方法。

程序中隐藏错误数量估计

3. 回归分析

① 得出回归曲线

② 预测任何时刻的错误数



软件质量的基本概念

软件质量的概念

《美国传统字典》对软件质量的定义：软件的特征或属性。GB 软件质量定义：软件产品中能满足给定需求的性质和特性的总体。

软件质量包括：

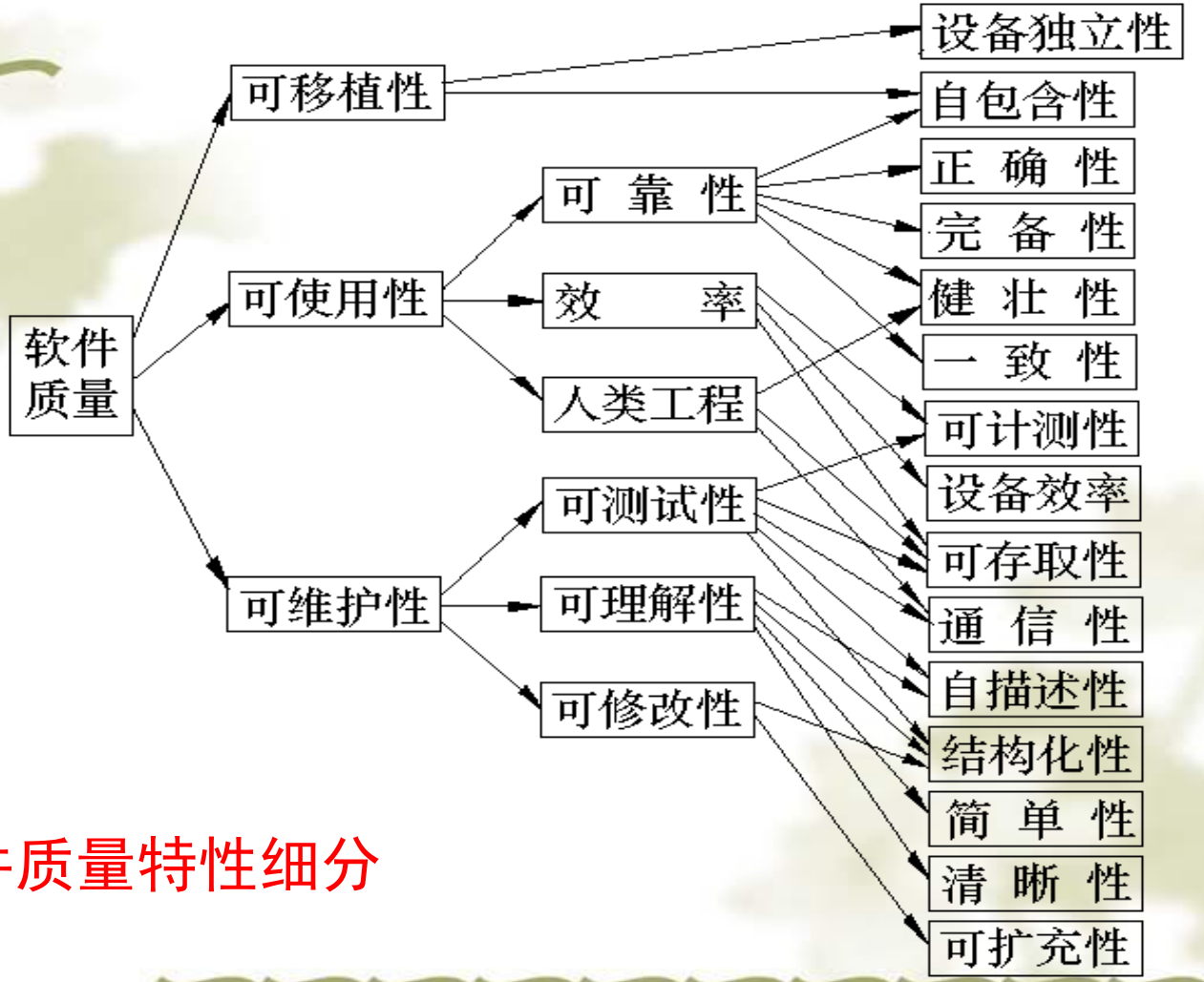
设计质量：指设计者为一件软件产品规定的特征。

符合质量：是指软件符合设计规格的程度。

软件质量特性

软件质量特性，反映了软件的本质，基本的质量特性包括：

- 功能性（Functionality）
- 可靠性(Reliability)
- 易使用性(Usability)
- 效率(Efficiency)
- 可维护性(Maintainability)
- 可移植性(Portability)


软件质量
可移植性
可使用性
可维护性
可靠性
效率
人类工程
可测试性
可理解性
可修改性
设备独立性
自包含性
正确性
完备性
健壮性
一致性
可计测性
设备效率
可存取性
通信性
自描述性
结构化性
简单性
清晰性
可扩充性

软件质量特性细分

软件质量模型

软件质量模型就是软件质量评价的指标体系。关于软件质量模型，业界已经有很多成熟的模型定义，比较常见的质量模型有：

- Jim McCall 软件质量模型（1977 年）
- Barry W. Boehm 软件质量模型（1978 年）
- FURPS/FURPS+ 软件质量模型
- R. Geoff Dromey 软件质量模型
- ISO/IEC 9126 软件质量模型（1993 年）
- ISO/IEC 25010 软件质量模型（2011 年）

McCall软件质量模型

Jim McCall 的软件质量模型，也被称为 GE 模型（General Electrics Model）。其最初起源于美国空军，主要面向的是系统开发人员和系统开发过程。

McCall 试图通过一系列的软件质量属性指标来弥补开发人员与最终用户之间的沟壑。

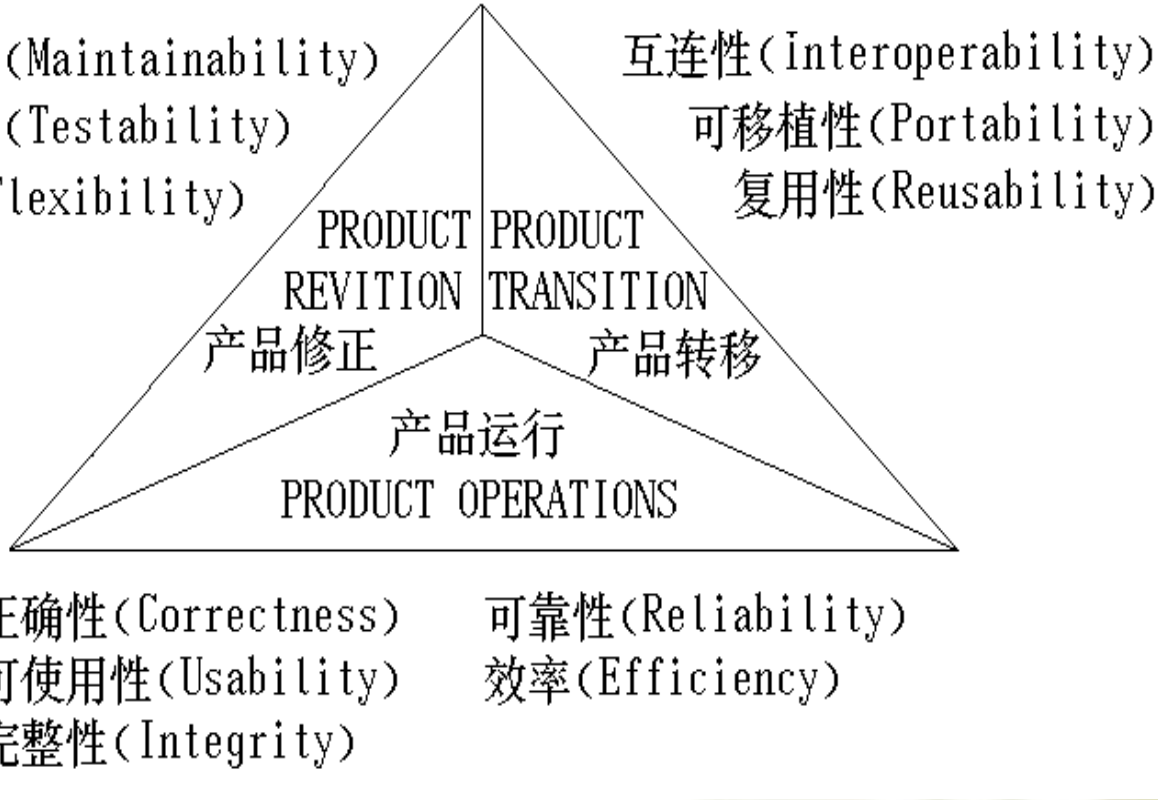
McCall软件质量模型

McCall 质量模型使用 3 种视角来定义和识别软件产品的质量：

1. Product revision (ability to change).
2. Product transition (adaptability to new environments).
3. Product operations (basic operational characteristics).

如图所示：

McCall软件质量模型

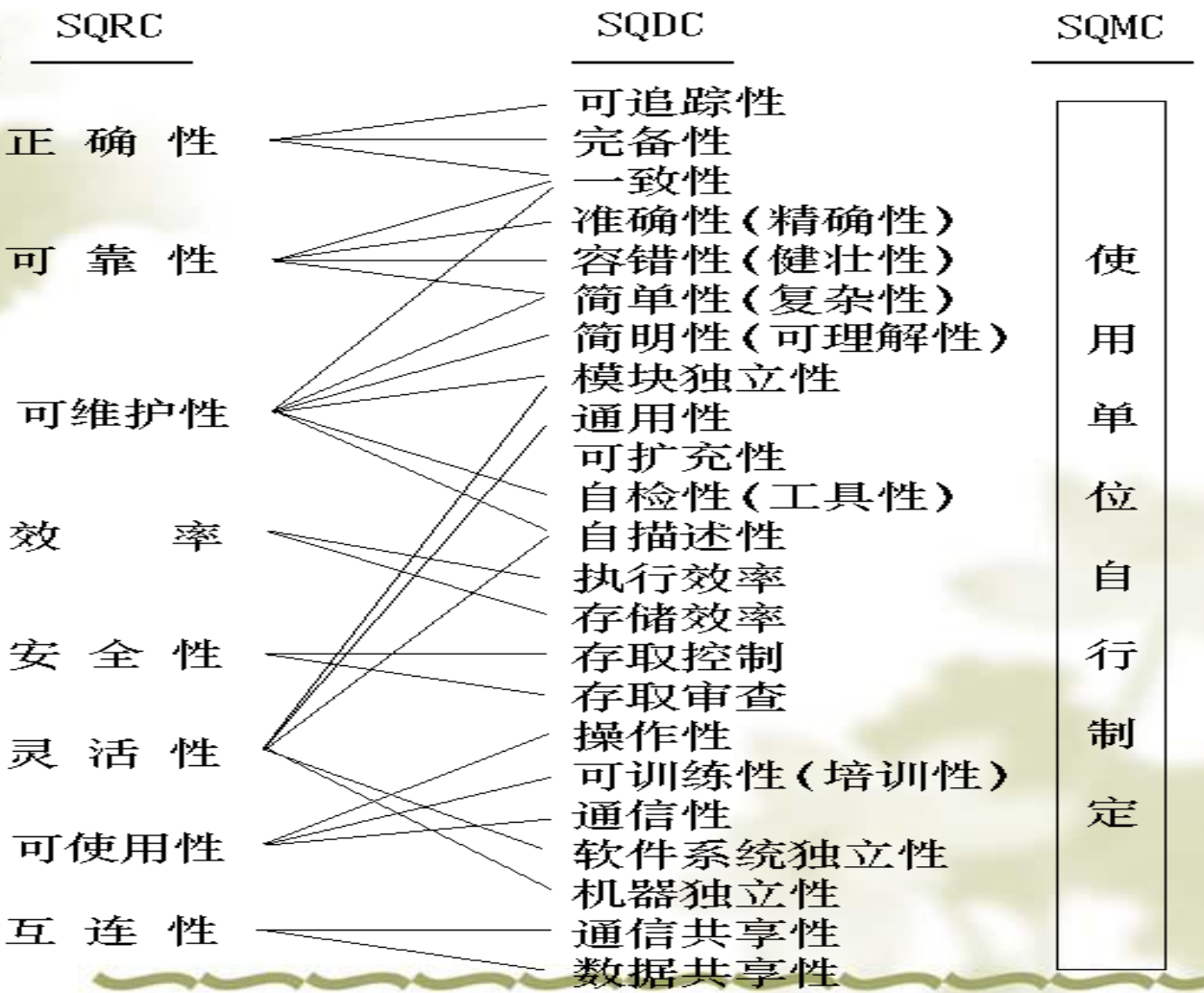


ISO/IEC 25010 软件质量模型（2011 年）

这一软件质量模型包含 8 个特征，并且被进一步分解为可以度量的内部和外部多个子特征。

这一软件质量度量模型由三层组成：

- 高层（top level）：
  软件质量需求评价准则（SQRC）
- 中层（mid level）：
  软件质量设计评价准则（SQDC）
- 低层（low level）：
  软件质量度量评价准则（SQMC）


SQRC
SQDC
SQMC
正 确 性
可 靠 性
可维护性
效 率
安 全 性
灵 活 性
可使用性
互 连 性
可追踪性
完备性
一致性
准确性（精确性）
容错性（健壮性）
简单性（复杂性）
简明性（可理解性）
模块独立性
通用性
可扩充性
自检性（工具性）
自描述性
执行效率
存储效率
存取控制
存取审查
操作性
可训练性（培训性）
通信性
软件系统独立性
机器独立性
通信共享性
数据共享性
使用单位自行制定

软件质量的度量

- 软件质量特性度量方法有两类：预测型和验收型。
- 预测度量是利用定量或定性的方法，估算软件质量的评价值，以得到软件质量的比较精确的估算值。
- 验收度量是在软件开发各阶段的检查点，对软件的要求质量进行确认性检查的具体评价值，它是对开发过程中的预测进行评价。

度量方式有两种

- 第一种叫做尺度度量，这是一种定量度量。它适用于一些能够直接度量的特性，例如，出错率。

- 第二种叫做二元度量，这是一种定性度量。它适用于一些只能间接度量的特性，例如，可使用性、灵活性等等。

尺度度量检查表（定量指标）

评价准则	度量	需求		设计		编码	
		是／否	值	是／否	值	是／否	值
程序复杂性	每一模块的复杂性度量(McCabe)		□		□		□
	系统复杂性度量 $= \dfrac{\text{各模块复杂性度量之和}}{\text{系统模块数}}$		□		□		□

填具体的数据值

二元度量检查表（定性指标）

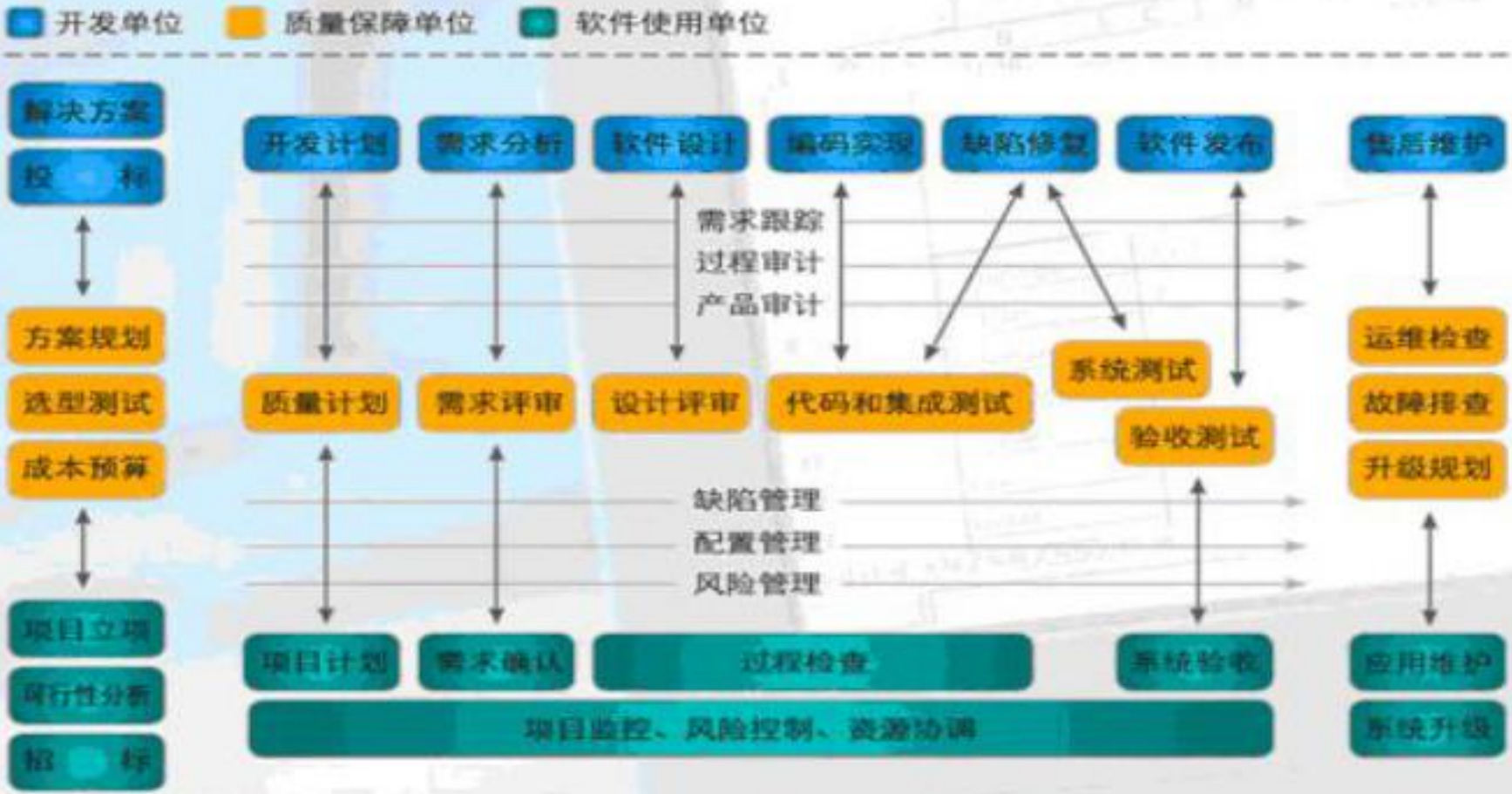
评价准则	度量	需求		设计		编码	
		是／否	值	是／否	值	是／否	值
设计文档的完备性	(1) 无二义性引用（输入／功能／输出）	□		□		□	
	(2) 所有数据引用都可以从一个个外部源定义、计算和取得	□		□		□	
	(3) 所有定义的功能都被使用	□		□		□	
	(4) 所有使用的功能都被定义	□		□		□	
	(5) 对每一个判定点，所有的条件和处理都已被定义	□		□		□	
	(6) 所有被定义、被引用的调用序列的参数一致	□		□		□	

填定性结论，如：是/否

软件质量保证活动

是为保证软件产品和服务充分满足消费者要求的质量而进行的有计划、有组织的活动，包括：

- 软件工程管理方法和工具
- 在整个软件过程中采用的正式技术复审
- 多层次的测试策略
- 对软件文档及其修改的控制
- 保证软件遵从软件开发标准的规程
- 度量和报告机制


开发单位
质量保障单位
软件使用单位
解决方案
投 标
方案规划
选型测试
成本预算
项目立项
可行性分析
招 标
开发计划
需求分析
软件设计
编码实现
缺陷修复
软件发布
售后维护
需求跟踪
过程审计
产品审计
质量计划
需求评审
设计评审
代码和集成测试
系统测试
验收测试
运维检查
故障排查
升级规划
缺陷管理
配置管理
风险管理
项目计划
需求确认
过程检查
系统验收
应用维护
项目监控、风险控制、资源协调
系统升级
规划阶段
实施阶段
交付阶段
运维阶段

标准的质量保证体系

- ISO 9000 （通用质量标准体系）

ISO9000质量体系认证是由国家或政府认可的组织以ISO9000系列质量体系标准为依据进行的第三方认证活动。

ISO9000：2008族标准核心标准为下列四个:

1）ISO9000：2005《质量管理体系一基础和术语》

2）ISO9001：2008《质量管理体系一要求》

3）ISO9004：2009《质量管理体系一业绩改进指南》

4）ISO19011 ：2011《质量和环境管理体系审核指南》

标准的质量保证体系

- CMM（软件能力成熟度模型）

它是对于软件组织在定义、实施、度量、控制和改善其软件过程的实践中各个发展阶段的描述。

CMM的核心是把软件开发视为一个过程，并根据这一原则对软件开发和维护过程进行监控和研究。

CMM是一种用于评价软件承包能力，以改善软件质量的方法，侧重于软件开发过程的管理及工程能力的提高与评估。

标准的质量保证体系

目标：

- 明确划分各开发过程，通过质量检验的反馈作用确保差错及早排除并保证一定的质量。
- 在各开发过程中实施进度管理，产生阶段质量评价报告，对不合要求的产品及早采取对策。

提高软件可靠性的方法和技术

- 建立标准的质量保证体系
- 使用开发管理工具（如版本管理）
- 加强测试（及早和不断的测试）
- 容错设计（预见到差错，并使其影响减至最小）
- 软件复用（构件化，如微信小程序）