



北京航空航天大学 计算机学院
School of Computer Science and Engineering, Beihang University



软件质量与软件质量保证方法

刘超

北京航空航天大学软件工程研究所

2015年12月

内容提要

- 软件质量
- 软件质量保证

软件开发者的目标

- 开发出好的、优质的软件产品
- 满足用户的需求，给用户带来便捷和效益
- 什么是好的软件？
 - 对用户：高性价比，即满足需求、质量高、能按时或及时交付、且价格低
 - 对开发者：满足所有用户的需求、具有市场竞争力，即软件产品质量、价格、发布时机都满足需求且优于同类产品

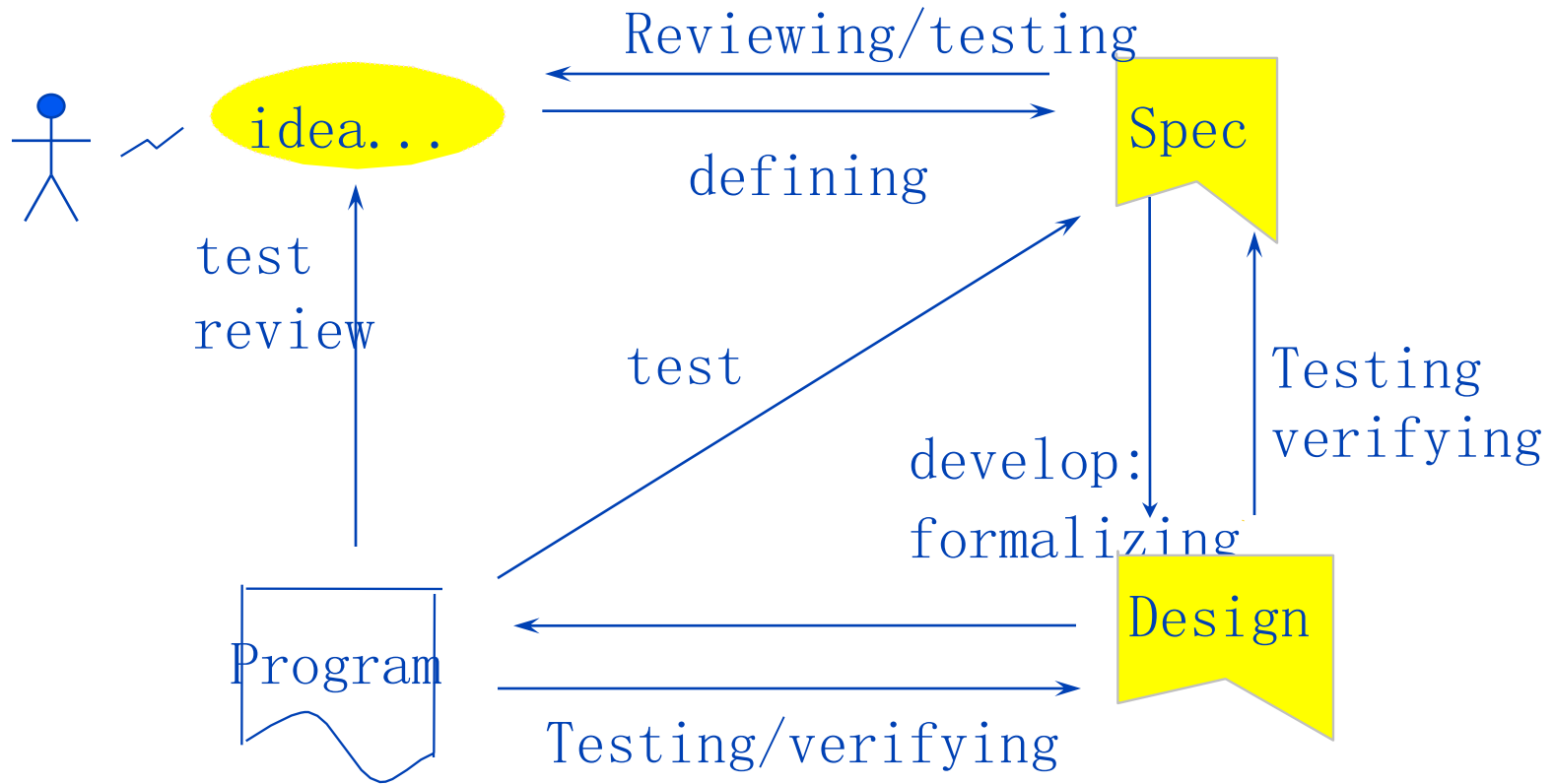


软件质量及其
保证问题！

What is Software and its Quality?

- Software = program + documents + data
 - quality of software
 - + program
 - + documents
 - + data
- Quality of software process
 - process and activities: planning, designing, ...
 - work products: documents, reports, source code, test cases,
- What Differences between Software Quality and Program Correctness?

Software Quality: Judging by?



Software Quality Definition

- ANSI/IEEE Std 729-1983
 - 与软件产品满足规定的和隐含的需求的能力有关的特征或特性的全体
 - All the features or characteristics to represent capabilities to satisfy the specified and default requirements
- M.J.Fisher
 - All the features that present extent of excellence of software
 - 所有描述计算机软件优秀程度的特性的组合
- Roger S. Pressman
 - Conformance to explicitly stated functional and performance requirements, explicitly documented development standards, and implicit characteristics that are expected of all professionally developed software.

Features of Software Quality

- McCall Quality Model
 - Three level framework model: features, evaluation criteria, metric
 - 11 features、29 evaluation criteria
- ISO software quality model
 - ISO/TC97/SC7/WG3/1985-1-30/N382
 - ISO/IEC9126
 - 6 features（特性）：functionality, reliability, usability, performance, maintainability, transferability
功能性、可靠性、易用性、效率、可维护性、可移植性
 - 21 sub-features

Effects among quality features

	Func	Reliab	Usability	Perf	Maint	Transf
Func		+			+	
Reiab				-		-
Usability				-	+	+
Perf		-			-	-
Maint		+		-		+
Trainf		-		-		

软件质量的另外一些观点

- Crosby(1979)
 - “conformance to requirements”
- Juran and Gryna(1970)
 - “fitness for use”
- In Guaspari's book **I Know It When I See It**(1985)

“Your customers are in a perfect position to tell you about quality, because that's all they're really buying. They're not buying a product. They're buying your assurances that their expectations for that product will be met. And you haven't really got anything else to sell them but those assurances. You haven't really got anything else to sell but quality.”

Definition by Companies

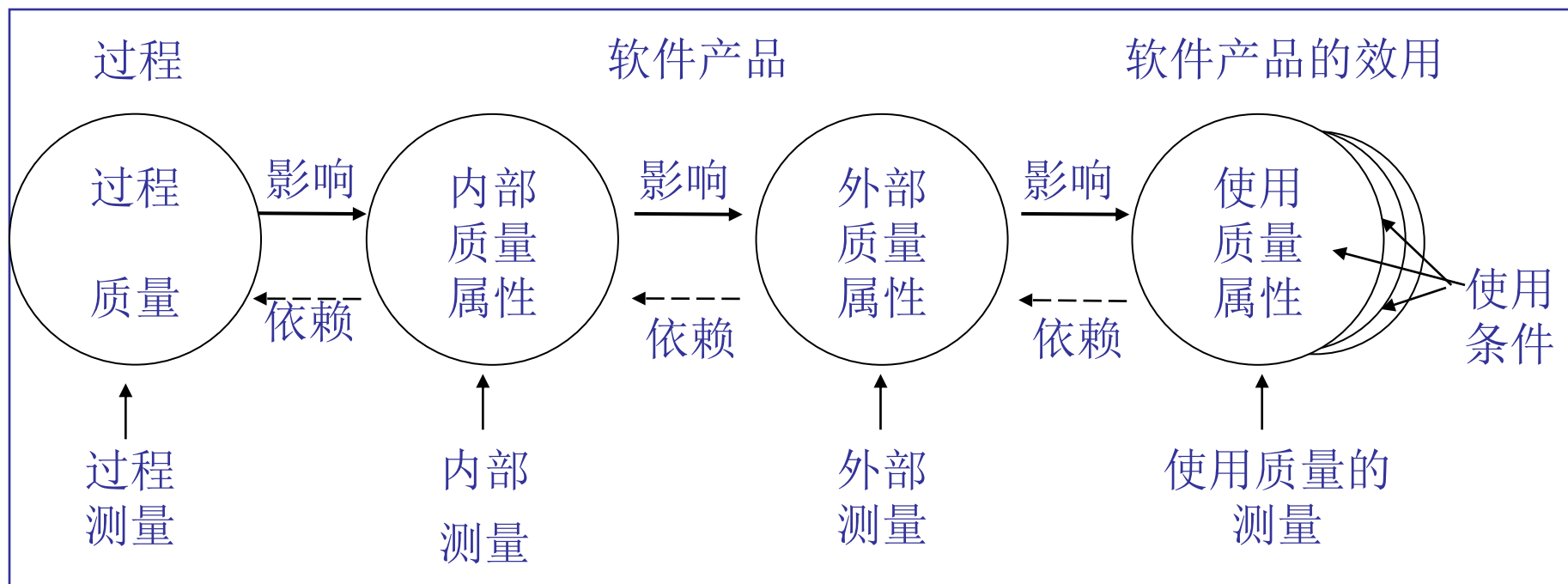
- Hewlett-Packard
 - FURPS: Functionality, Usability, Reliability, Performance, Serviceability
- IBM
 - CUPRIMDSO: Capability(Functionality), Usability, Performance, Installability, Maintainability, Documentation/Information, Service, Overall
- Lotus
 - mixture of excellence, productivity, simplicity and elegance of design that dramatically meets the user's needs.

例：某型号工程

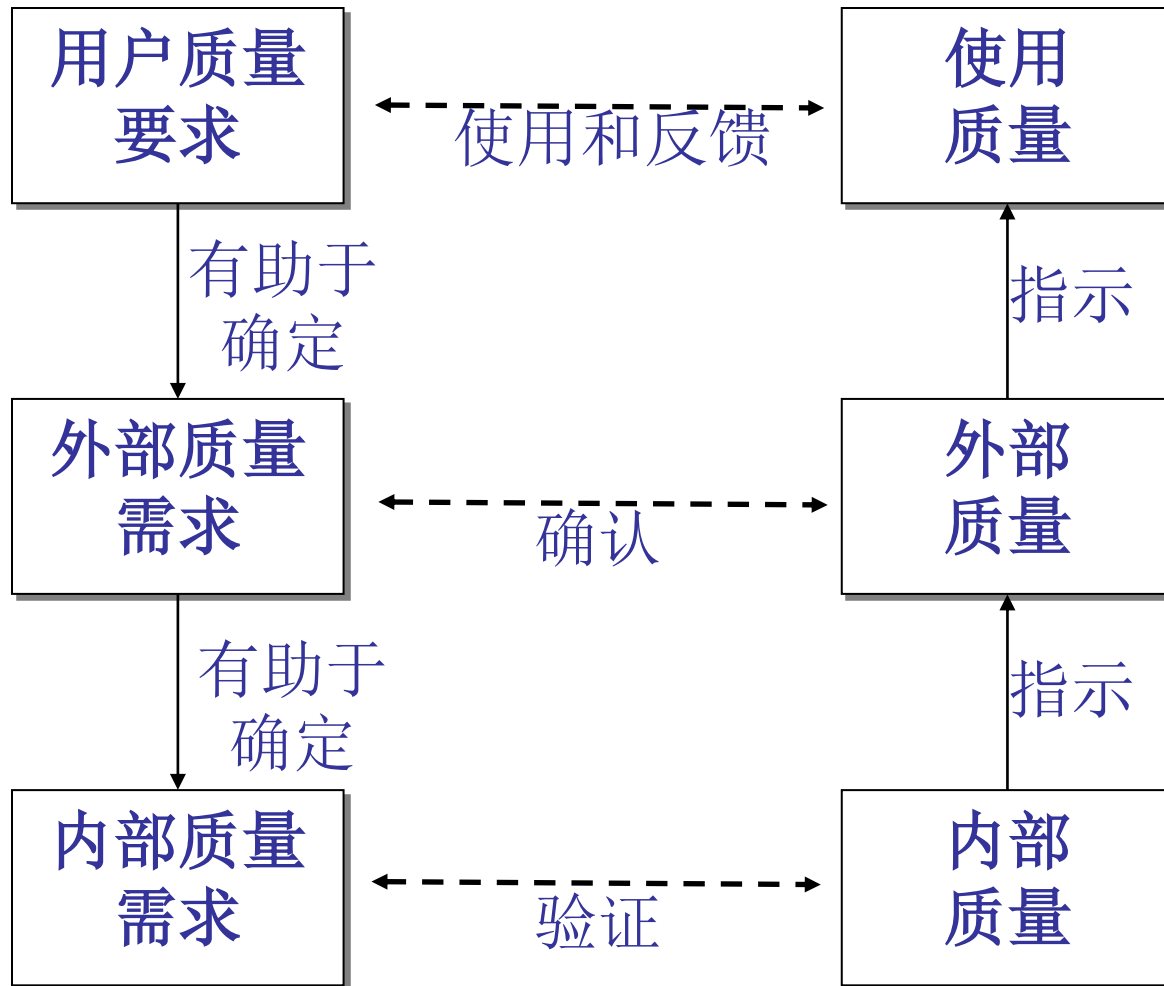
- 软件产品中能满足给定需求的性质和特性的总体
- 这些特性称为质量特性
 - 功能度
 - 可靠性
 - 易使用性
 - 时间经济性：时间效率
 - 资源经济性：资源利用率
 - 可维护性
 - 可移植性

GB/T 16260.1 -- ISO/IEC 9126:2001

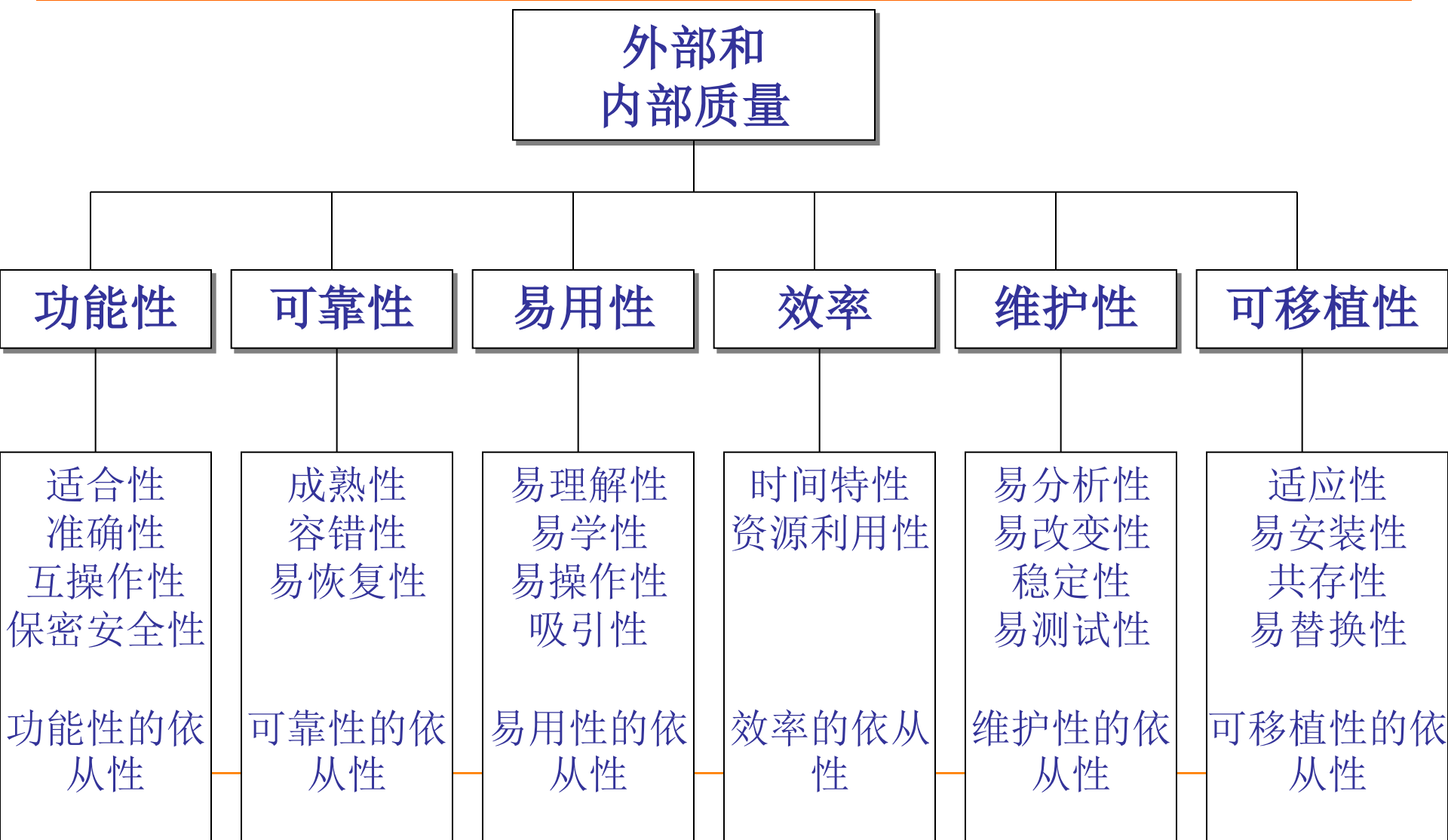
- 软件质量模型包括
 - 内部质量模型
 - 外部质量模型
 - 使用质量度量
- 过程质量模型：GB/T 8566—2001；ISO/IEC 15504



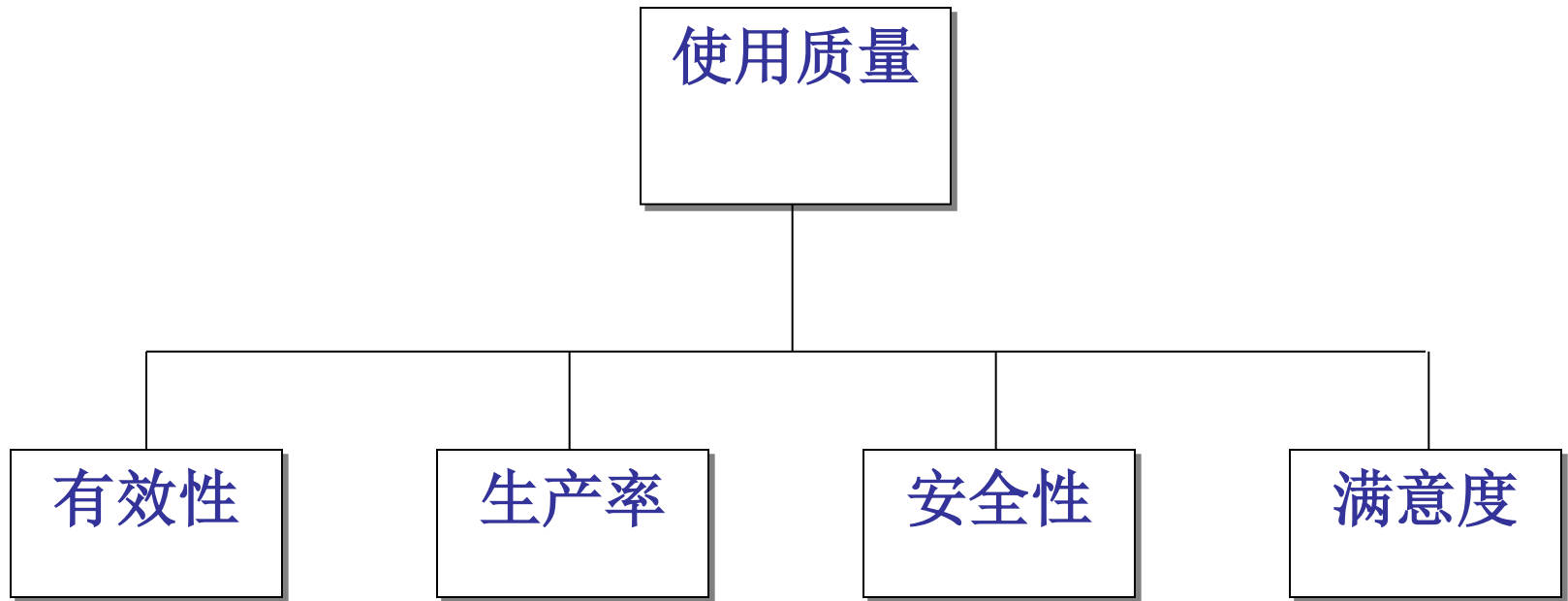
软件质量的不同观点



软件外部质量和内部质量模型



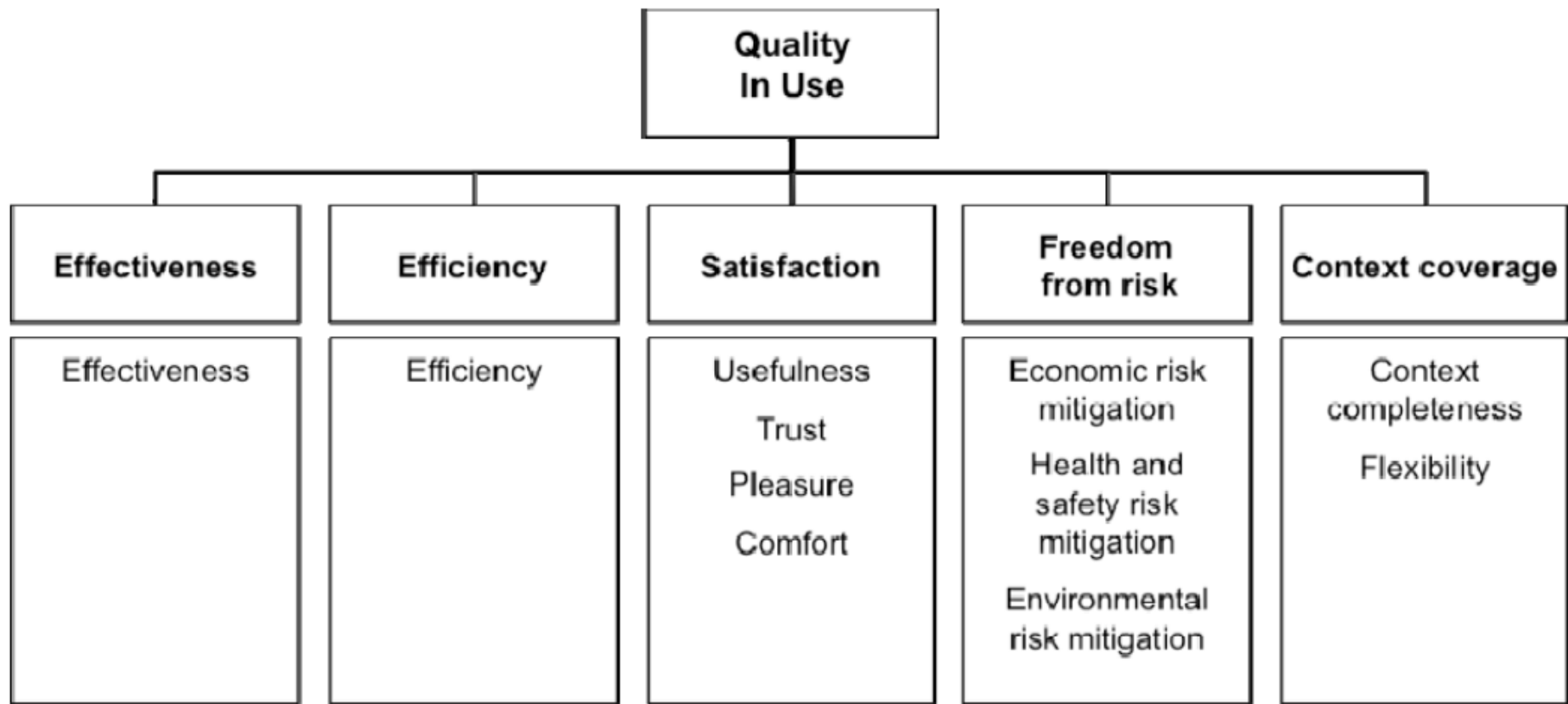
软件使用质量模型



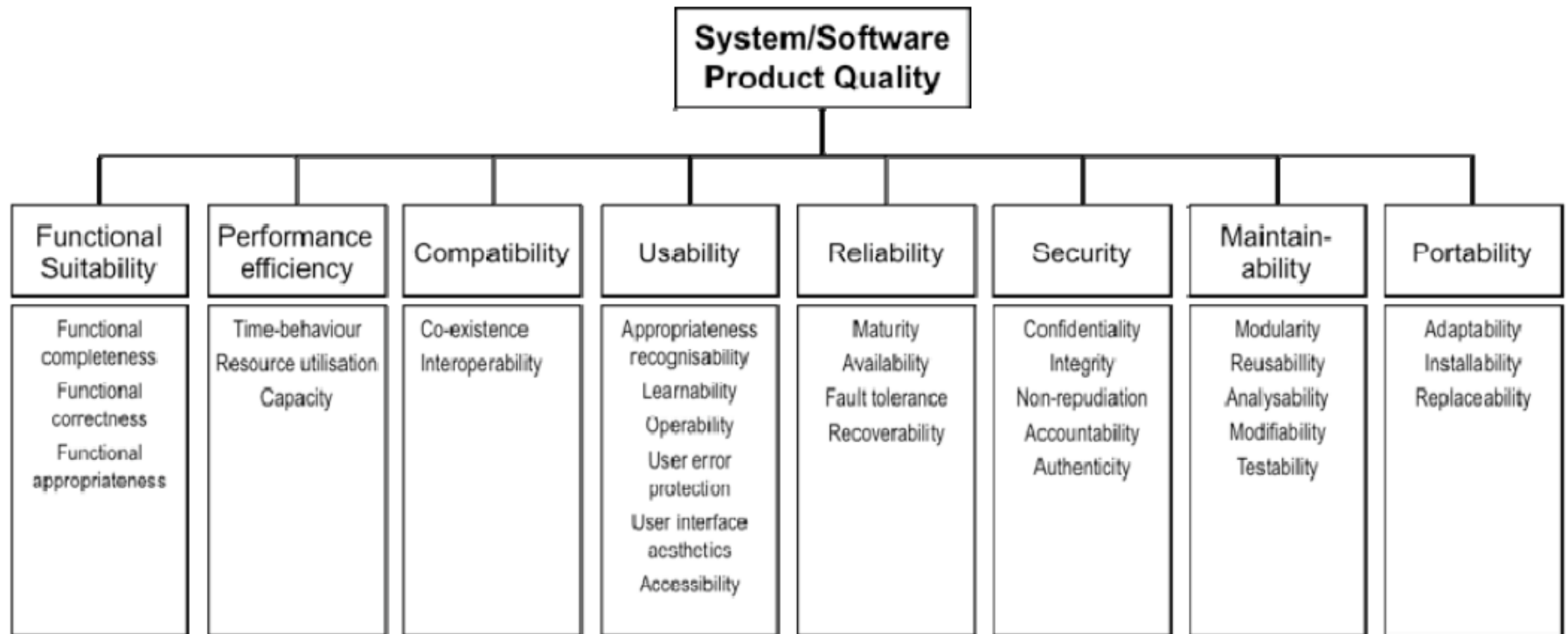
ISO/IEC 25010

- Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models
- prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software and systems engineering*
- **cancels and replaces ISO/IEC 9126-1:2001, which has been technically revised.**
- is intended to be used in conjunction with the other parts of the SQuaRE series of International Standards (ISO/IEC 25000 to ISO/IEC 25099), and with ISO/IEC 14598 until superseded by the ISO/IEC 2504n series of International Standards.

- Quality in use model



- Product Quality Model



软件质量的延伸

- 软件就是服务
 - 软件质量与服务质量(QoS)
- 软件定义一切!?

讨论

- 软件制品的质量？

特色、规范、清晰、准确、好用；

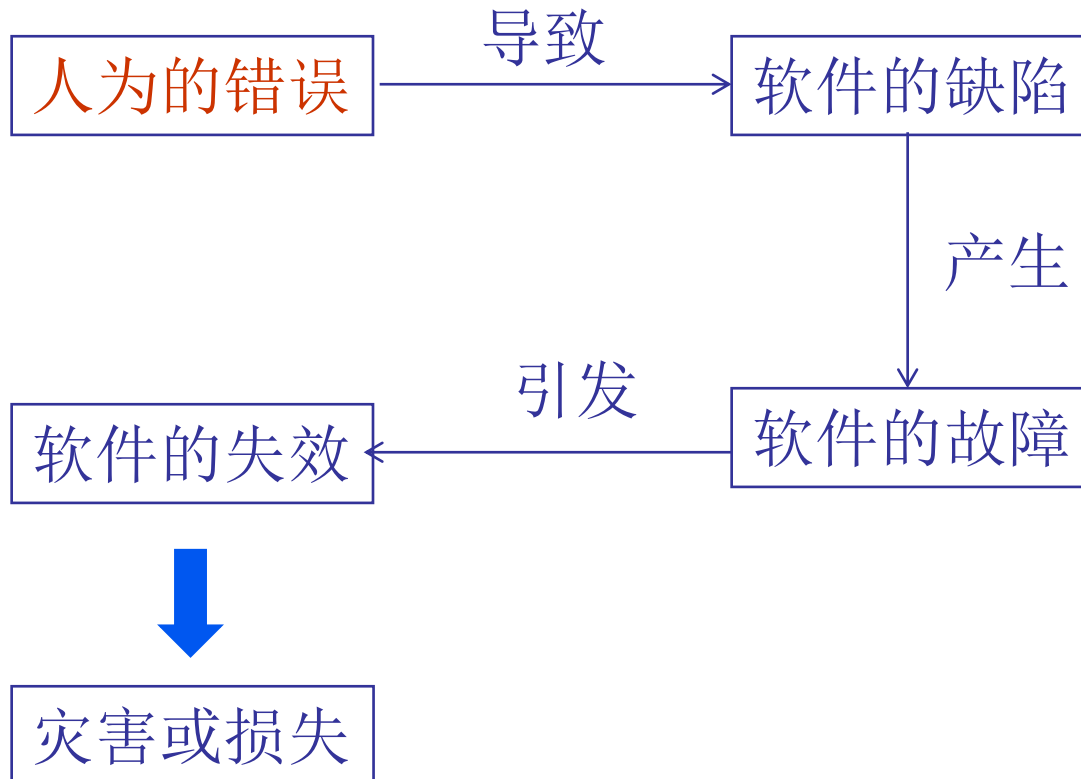
- 软件文档
 - + 完整、准确、一致、规范
- 软件模型
 - + 完整、准确、一致、规范
- 软件代码
 - + 正确、简明、易懂、规范
- 软件的运行程序
 - + 适用、易用、稳定、高效

内容完整，格式规范；
概念正确，描述准确；
文字通顺，易于理解；
面向问题，紧扣关键；
直观易懂，符合规范；
符合设计，编写规范；
功能完整，特色鲜明；
运行稳定，实例示范；

软件质量问题产生的根源

- 应用领域广 涉及问题多 难以制定统一规范
- 需求复杂 难以严格定义
- 开发周期长
- 需求变更大
- 参与人员多 流动性大
- 系统大 结构复杂 难以理解 测试和维护
- 软件与硬件不同:
 - 没有"生产" 不会"老化"
 - 需求变化大 版本更新快 开发周期长

软件质量问题的根源



错误、缺陷、故障和失效（续）

- IEEE/American National Standards Institute(ANSI/982.2)
 - Error（错误）：a human mistake that results in incorrect software
 - Defect（缺陷）：an anomaly in a product
 - Fault（故障）：an accidental condition that causes a unit of the system to fail to function as required
 - Failure（失效）：it occurs when a functional unit of a software-related system can no longer perform its required function or cannot perform it within specified limits

软件质量如何保证？

- 软件质量在很大程度上取决于软件过程的质量
- 软件过程：规范、保质、高效
- 软件能力成熟度（CMM）
 - 5个能力成熟度等级：18个关键过程域
- CMMI取代CMM

5级-优化层

- * 过程更改管理
- * 技术更改管理
- * 错误预防

4级-管理层

- * 质量管理
- * 过程量化管理

3级-定义层

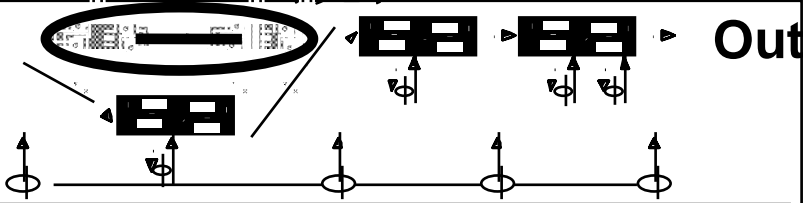
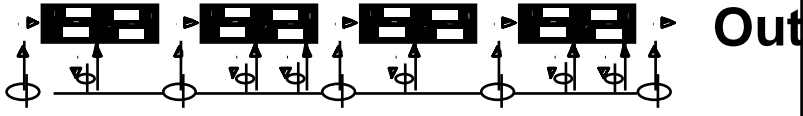
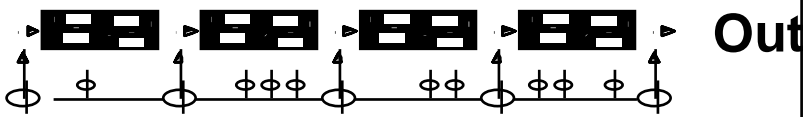
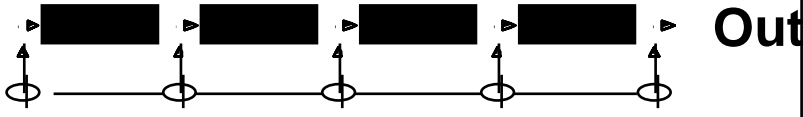

- * 同级评审
- 组间协作
- * 软件产品工程
- * 软件集成管理
- 培训计划
- * 软件过程定义
- * 软件过程要点

2级-可重复层

- 软件配置管理
- 软件质量保证
- 软件子合同管理
- * 软件项目追踪与监督
- 软件项目计划
- 需求管理

1级-初始层

- 软件过程的可见性和可控性

等级	过程特性	管理可见性
5 优化级	过程改进制度化	In  Out
4 已管理级	基于量化数据进行产品和过程的控制	In  Out
3 已定义级	技术实践与管理实践集成并制度化	In  Out
2 可重复级	项目管理实践制度化	In  Out
1 初始级	过程的随意性和不可见性	In  = Out

软件质量保证

- CMM
 - The purpose of SQA is to provide management with appropriate visibility into the process being used by the software project and of the products being built
 - SQA involves reviewing and auditing the software products and activities to verify that they comply with the applicable procedures and standards and providing the software project and other appropriate managers with the results of these reviews and audits
- 评审技术：阶段评审、评审对象、评审会参与者
- 软件质量保证体系：标准、过程、技术、组织与管理

Software Review

- Using the diversity of a group of people to:
 - point out needed improvements in the work product
 - confirm those parts of a product in which improvement is either not desired or not needed
 - achieve technical work of more uniform, or at least more predictable, quality than can be achieved without reviews, in order to make technical work more manageable.

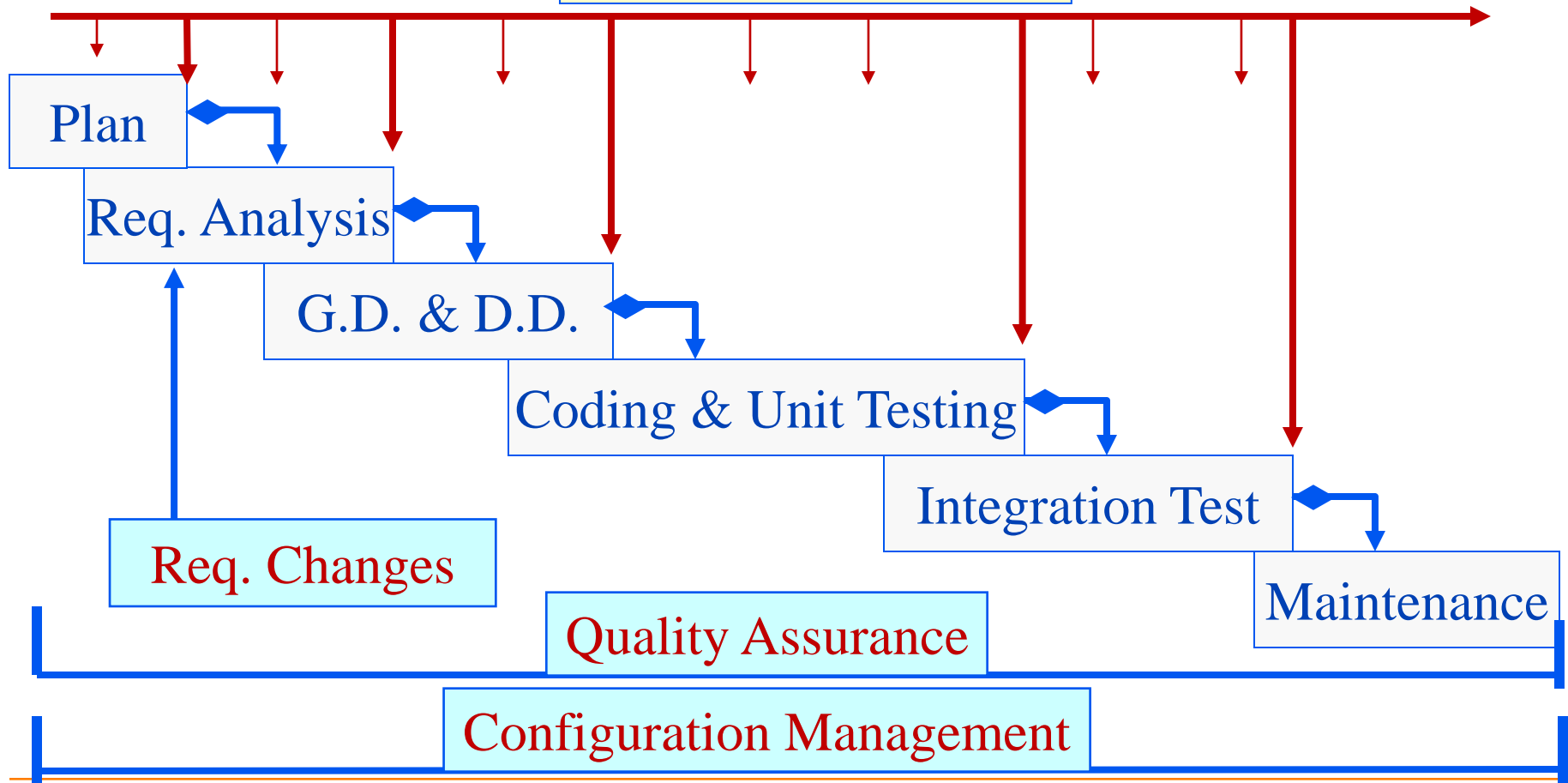
SQA Activities

- SQA Plan
 - evaluations to be performed
 - audits and reviews to be performed
 - standards that are applicable to the projects
 - procedures for error reporting and tracking
 - documents to be produced by the SQA group
 - amount of feedback provided to the software project team

Software Quality Process: Definition, Tracibility, and Monitoring

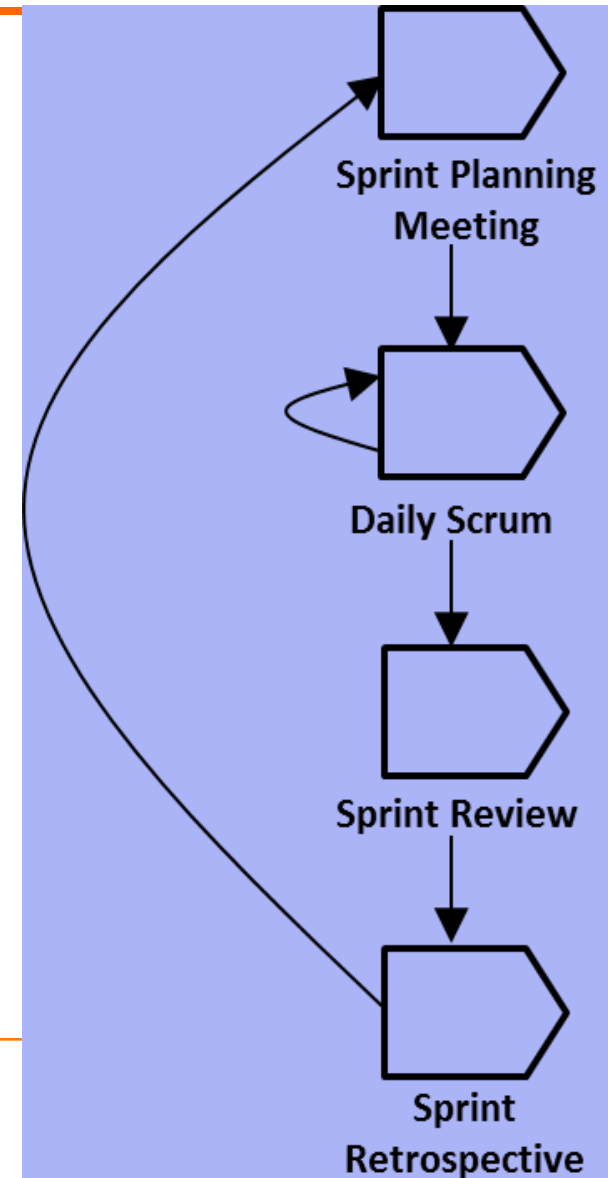
Water Fall Model:

Milestones: Reviews



Agile Process? ----紧密的讨论和协调

- **Scrum** ---- **问题**的识别，重要程度的标识，任务协调
- **Sprint** ---- 任务的细分和计划，进展的评审



评审、审查、走查、审计（审核）

Review, Inspection, Walkthrough, Audit

- 对象：
 - 代码、规格说明、设计、计划等
 - 影响大、复杂、不熟悉的部分
- 方法：“专家会诊”
 - 领域、软件开发、质量、管理、市场
 - 正式/非正式： 3-5人/1-2小时
 - 问题单
 - 开发者介绍 -> 大家提问 -> 发现问题 -> 总结
- 时机：
 - 工作期间：非正式/正式
 - 阶段（节点）：正式

Reviewing purpose

- Lotus:
 - The purpose of a review is to satisfy ourselves that the thing we're reviewing does what we said it should, and does it using generally accepted practices.
 - It is not a tool by which to measure or grade our developers.
 - The review is conducted not by managers, but for managers, by professionals who are technically competent in the area of concern

What is the Review's Capability?

- Finding 30-70% logical problems in design and coding
- Lotus
 - using code inspections to find and fix bugs costs just 10% of the cost of finding and fixing them in the usual SPR cycle, and we believe that number might fall to just 2% when we use spec inspections to find bugs in the specs. **More than 60% of all bugs occur before coding starts.**
 - **The sooner you find those bugs, the cheaper it is to fix them, and it's a better-than-linear relationship.**

Code inspections

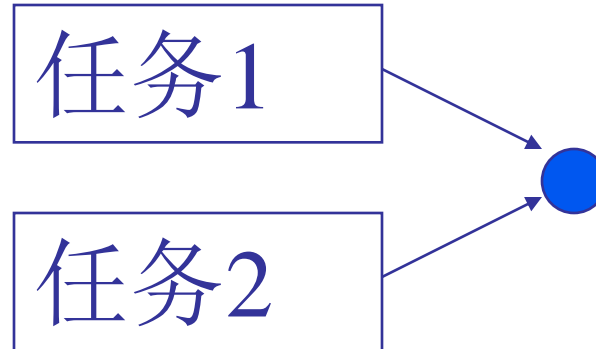
- code inspections
 - Expect to spend roughly 3 developer days per 1000 non-comment lines of code, and find approximately 100 issues
 - Without the code inspection, 30 - 50 SPRs are expected per 1000 lines of code, and that's 10 to 25 QA days!

Reports from companies

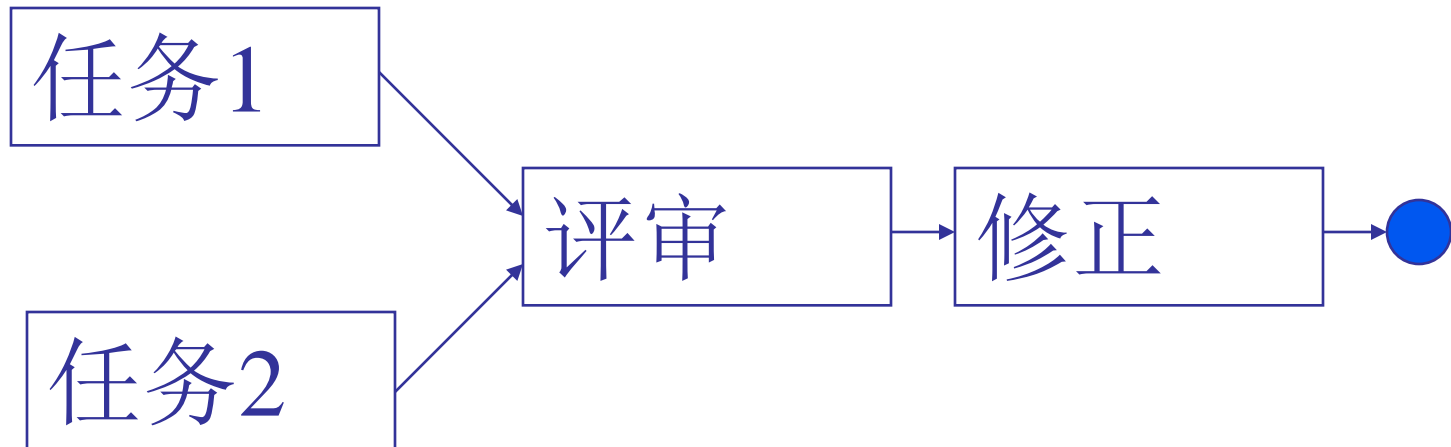
- Average cost in person hours to find/fix a significant problem during an inspection versus after release

Company	Inspection hours	After release
Jet Propulsion Labs	1.8	100
Bell Northern	1	36
Shell Research Lab	1.8	30

里程碑的设置



加入评审：



例1：需求评审

- 需求过程与方法论（软件工程规范）的一致性
- 功能规格说明的定义
- 可用性规格说明的确定
- 维护规格说明的确定
- 需要确定可移植性
- 系统接口的定义
- 性能标准的确定
- 可操作性的确定
- 偏差的确定
-

例1.1需求过程与方法论的一致性

- 收集的应用程序所需的数据是否能够达到期望的可靠性水平
- 是否可以在指定时间段内收集数据
- 用户需求是否已经以书面形式定义
- 需求的描述是否可度量
- 项目解决方案是否强调了用户需求
- 制定的测试数据是否能测试目标的实现程度
- 规程是否指明了要评价已实现的系统，以保证实现了需求
- 是否将可度量的目标应用系统的人工部分和自动部分

例1.2：需求说明的审查

- 需求描述
 - 图像采集的分辨率分为高、中、低三种
 - 前端获得的图像可以实时地显示在后端监控屏幕上
 - 在后端机上，可以对采集到的多个实时图像文件拼接成一个文件
 - ...

例2

- 系统接口的定义
 - 是否识别了将从其他应用程序接受的数据
 - 是否识别了将要用于其它应用程序的数据
 - 是否定义了接口数据的可靠性
 - 是否定义了数据传送的时间
 - 是否定义了接口方法
 - 是否文档化了接口需求
 - 是否考虑了接口系统日后的潜在需求

设计评审

- 数据完整性控制的设计
- 授权规则的设计
- 文件完整性控制的设计
- 审计追踪的设计
- 应急计划的设计
- 达到预定服务水平的设计
- 访问规程的定义
- 设计过程与方法论的一致性
- 设计与需求的一致性
- 设计利于使用
-

例2：某信息系统设计评审检查表

商业信息系统设计评审检查表							
表项				检查结果			注释
	类别	序号	检查项	是	否	不确定	
1	系统概述						
		1	是否有与其他系统接口的简要描述				
		2	是否有系统主要功能性需求的提纲				
		3	是否将主要功能定义到了没有边界覆盖的离散步骤中				
		4	是否定义了人工和自动的步骤				
		5	是否说明了执行各步骤需要的数据及期货的方式				
2	系统描述						
		1	是否制定了系统结构图表来说明其逻辑结构及其接口				
		2	是否定义了主要的输入输出及所需的功能性处理				
		3	是否有对主要功能的准确、直观的描述				
		4	是否画出了各子系统功能流程图（或其他特征模型图）				
		5	是否有对子系统的准确、直观的描述				
						

例3：编程评审

- 数据完整性控制的实现
- 授权规则的实现
- 文件完整性控制的实现
- 审计追踪的实现
- 意外计划的编写
- 系统达到服务水平的设计
- 安全规程的实现
- 程序编制与方法论的一致性
- 程序符合设计
- 程序可维护性
-

4.1

错误分类	编号	检查项（问题描述）	示例
变量	1	变量未初始化	<pre>char *s; cin>>s;</pre>
	2	应当使用常量处,使用了变量	<pre>int n = 100; double a[n];</pre>
	3	给变量的赋值被截断	<pre>int number=3.14;</pre>
	4	变量声明关键字次序错	<pre>double long a;</pre>
	5	变量有相似的名称	<pre>int count; char Count;</pre>
函数	6	程序某条路径缺少返回值	<pre>int main(){ if(cin) return 0; }</pre>
	7	函数实参类型是否与形参完全匹配	<pre>extern int f(int); double x =1.0; f(x);</pre>
表达式	8	计算中使用了不同类型的变量	<pre>int i= 1; result= x/3.14+10;</pre>

条件表达式	9	对浮点类型的表达式使用“==”操作	if(x==0.0)
	10	if的条件表达式中有“=”操作符	if(I=1)
	11	循环的条件表达式中存在“=”操作符	while(n=10)
	12	if条件语句中存在恒真	if(a>=b a<b)
	13	if条件语句中存在恒假	if(a<b&& a>b)
语句	14	不适当地使用goto	
	15	声明语句与执行语句混用	
类	16	类构造函数是否初始化了所有成员	
编程风格	17	嵌套语句没有适当的缩进、对齐	
	18	函数/方法没有注释	
	19	变量/形参没有注释	
	20	条件转移/循环体没有注释	

例4

- 程序可维护性
 - 程序是否符合维护规格说明
 - 程序文档是否完整可用
 - 程序中是否包括相当数量的注释
 - 程序中是否充分避免了复杂了逻辑
 - 是否将在编程过程中所做得改变合并到了设计文档中
 -

审查所需文档

- 审查目的(组织方针、项目计划)
 - 会议通知
 - 审查资料包
 - 需求说明书
 - 缺陷检查表、规则集
 - 勘误（**typo**）表/问题清单/议题记录
 - 更正后的工作产品
 - 基线产品
 - 审查总结报告、经验教训分析报告、过程改进
- ➡ 归档并建立追踪关系！
- 标识：编码
 - 追踪表

例：代码走查

➤ 求三个数中的最大值

```
main(int a, int b, int c)
{
    int m;
    m = a;
    if ( b > m )
        m = b;
    if ( c > m )
        m = c;
    return (m);
}
```

问题：

1. 需求：

三个数的类型；
函数/程序？

2. 设计与实现：

变量、函数及其参数的说明；
返回值类型；
源文件、包含文件？

3. 文档：

使用说明/手册

4. 需求/设计的扩展：

求n个数的最大值

例：模型检查

- Queue10:

- 要求Queue类对象是一个由不超过10个Element类对象组成的队列，其所有Element类对象的值均不相同，
- 且所有元素的排列顺序始终按照其值从小到大排列。

addElement(value: int):
增加一个值为value的元素
且返回true;

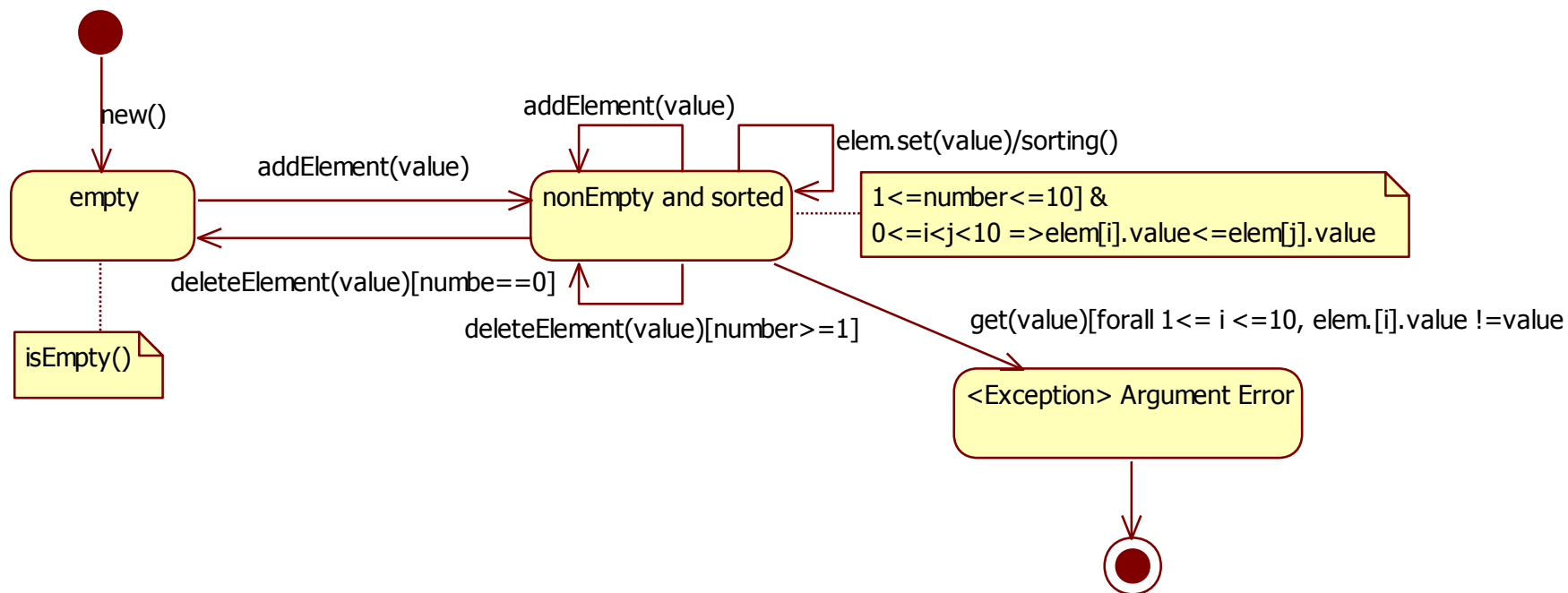
+set(index: int, value: int):
+index(value: int): int
+addElement(value: int): Boolean
+deleteElem(value: int): Boolean
+sorting(): Boolean
+numberOfElements(): int
+isEmpty(): Boolean
+isSorted(): Boolean

addElement(value: int): 如果向量中的元素个数小于10且没有一个元素的值等于value，则增加一个值为value的元素并保持Queue满足排序要求，返回true，否则返回false

+equal(value: int): Boolean
+lessThan(value: int): Boolean

$(1 \leq \text{index} \leq 10) \ \& \ (1 \leq i < j \leq 10) \Rightarrow \text{elem}[i].\text{value} < \text{elem}[j].\text{value}$

- 模型的一致性



评审效率

- 源代码：150-200 LOC/hour（不计算空行、注释行）
- 文档：3-4pages/hour
- 个人准备：0.5-1.5pages/hour
- 评审会： ≤ 2 hours/meeting

审查勘误表（Inspection Typo List）

- 审查员
- 计划审查日期：
- 工作产品标识：

页号	节号/行号	勘误说明

项目计划审查检查单

- 清晰性
- 完整性
- 一致性
- 正确性

问题记录

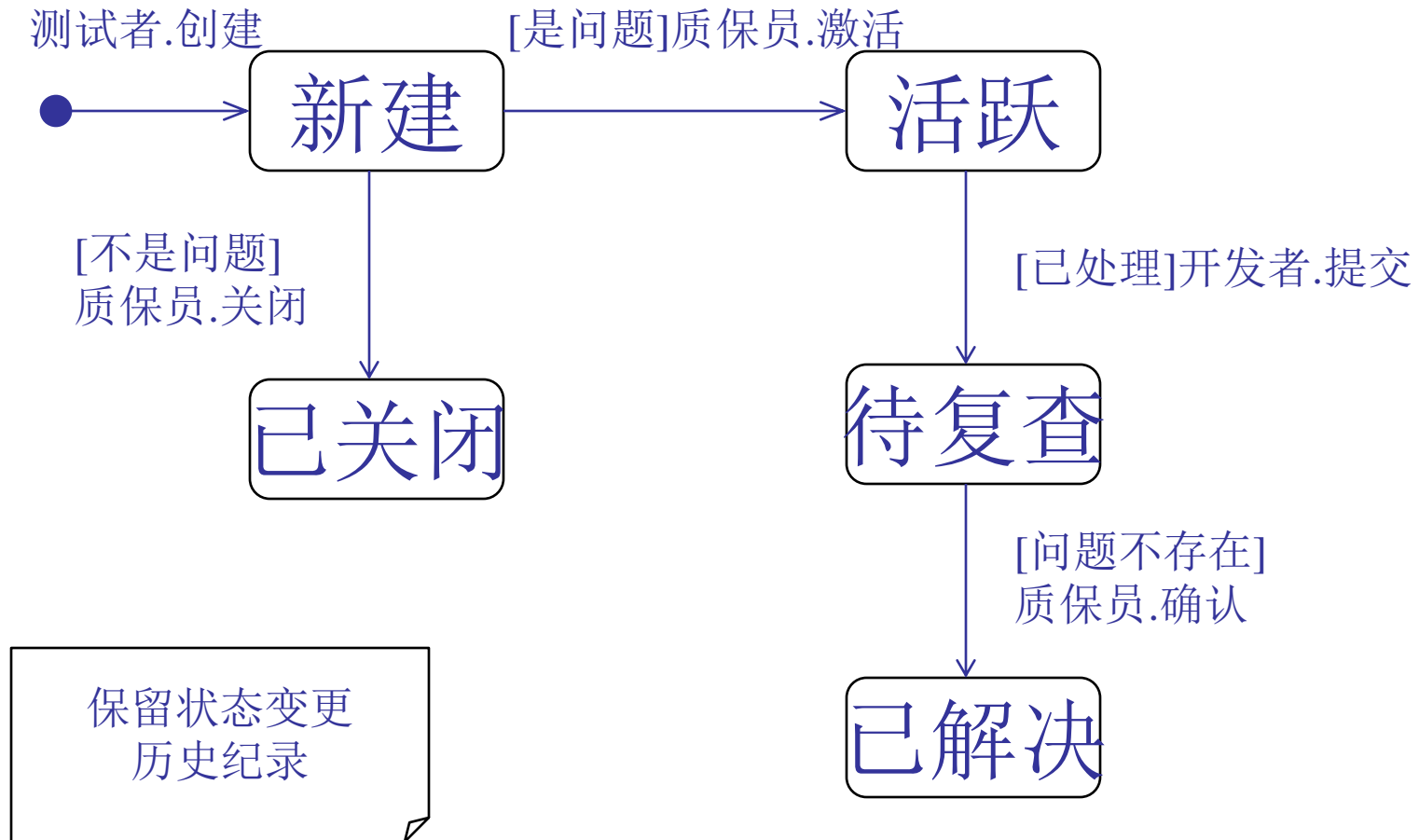
- 项目名称
- 审查代码:
- 审查日期:
- 审查员:

- 问题代码:
- 问题类型:
- 问题严重性:
- 问题位置:
- 问题描述:

问题驱动的软件质量过程控制

- 问题
 - 过程的问题：规程的依从性、计划的符合性、产品的质量等
 - 产品的问题：系统、数据、功能、性能、界面、维护、文档等
- 软件问题报告（**SPRs**）
 - 问题描述、测试用例、重现步骤
 - 问题分类、严重性
 - 注入过程、发现过程、剔除过程

软件问题报告（SPR）的生命周期



软件度量 (Software Metrics)

- 软件度量可划分为三个范畴：
 - 产品度量：产品特性，以帮助对其质量状态进行直接或间接的判断
 - + 规模、复杂度、设计特性、性能、质量水平...
 - 过程度量：开发过程和维护过程的基本特征，以便对其实施有效的追踪、监控和改进
 - + 纠正缺陷的效率、纠错过程的响应时间
 - 项目度量：项目特征
 - + 开发人员数量、整个软件生命周期中的人力分布、费用、进度、生产率

软件质量度量 (Software Quality Metrics)

- 软件质量度量：软件产品、过程和项目的质量特性和特征
- 软件产品的质量度量：
 - 最终产品的质量度量
 - 中间产品的质量度量
- 软件质量工程的一个基本课题就是研究中间过程（产品）、项目特性和最终产品质量之间的关系

常用的度量

- 缺陷发现数
- 缺陷密度
- 缺陷改正数
- 审查人力花费
- 发现每个缺陷的人力花费
- 改正每个缺陷的人力花费

- 各阶段缺陷发现率：各阶段缺陷改正的人力花费比率

软件产品质量度量

- 平均失效等待时间 (MTTF--Mean time to failure)
- 缺陷密度 (Defect density) :
 - defects/KLOC
 - + defects/KSSI: defects per thousand shipped source instructions
 - + defects/KCSI: defects per thousand new and changed source instructions
 - Functions points: 类别->加权求和
- 客户问题 (Customer problems) :
 - PUM: problems per user month
- 客户满意度 (Customer satisfaction) :
 - 非常满意、满意、一般、不满意、极不满意

软件复杂性的度量

- 度量元
 - 源代码行数: **KLOC**
 - 注释行数
 - 操作数个数、操作符个数
 - **功能点**
- 度量
 - 复杂度
 - + **McCabe: 基本路径数**
 - 分支点数 + 1
 - $e - n + 2p$, 其中 e : 边数; n : 节数; p : 相邻的组件数
 - + **Halstead: 基于操作符和操作数的个数**

参考书籍

- GB/T 16260
- ISO/IEC 9126
- 杨海燕等译，软件度量，（英）Norman E. Fenton等著，Software Metrics, Rigorous and Practical Approach, 2nd Edition, 机械工业出版社，2004年9月

参考书籍

- IEEE Std. 1028-1997, "IEEE Standard for Software Reviews"
- 兰雨晴等译，软件测试的有效方法，（美）William E Rerry著，Effective Methods for Software Testing, second edition, 机械工业出版社，2005年3月
- 唐云深等译，走查、审查与技术复审手册，（美）Daniel P. Freedman, etc, Handbook of Walkthroughs, Inspections, and Technical Reviews: Evaluating Programs, Projects, and Products, 3rd Edition, 清华大学出版社，2003年10月

例：小组项目实践

- 关注点
 - 规模**Scale**（有效的更新）
 - + 代码行数（KLOC）
 - + 文档页数（POD）
 - + 模型（元素）数量（NOM or NOME）
 - 质量**Quality**
 - + 评审、测试：缺陷数（NOD），缺陷率(ROD)，及其修正率
 - 生产率**Productivity**
 - + 人工量（men-hour, men-day, men-month）
 - + 实际进度与计划进度的符合度

练习：代码走查

- 评审对象：
 - 程序：计算程序的源代码行数(LOC)
 - 扩展：计算注释行、空行；每个函数或对象类的行数
- 两个小组
 - 开发小组：3-5人
 - 走查小组：3-5人

练习：代码走查（续1）

- 开发小组：
 - 问题说明
 - 开发计划（任务、进度、分工）
 - 程序设计
 - 源程序
 - 执行程序
 - 实际执行情况记录
 - + 任务与进度；问题、原因、修改情况

练习：代码走查（续2）

- 走查小组
 - 评审计划：目的、任务、分工、对象、时间
 - 检查表（**Checklist**）：目标、类别、检查要项、结果
 - 评审记录格式设计：
 - + 基本信息：走查对象、参加人员、时间等
 - + 问题描述、问题分析、解决方案/建议
 - 评审记录
 - 问题记录

练习：代码走查（续3）

- 总结
 - 任务与进度分析：计划与实际执行情况的差异分析
 - 缺陷数及分布情况分析
 - 对评审练习的看法与建议