第4章 工程过程

- 4.1 需求开发RD
- 4.2 需求管理RM
- 4.3 技术解决方案TS
- 4.4 产品集成PI
- 4.5 验证VER
- 4.6 确认VAR
- 4.7 软件测试
- 4.8 同行评审

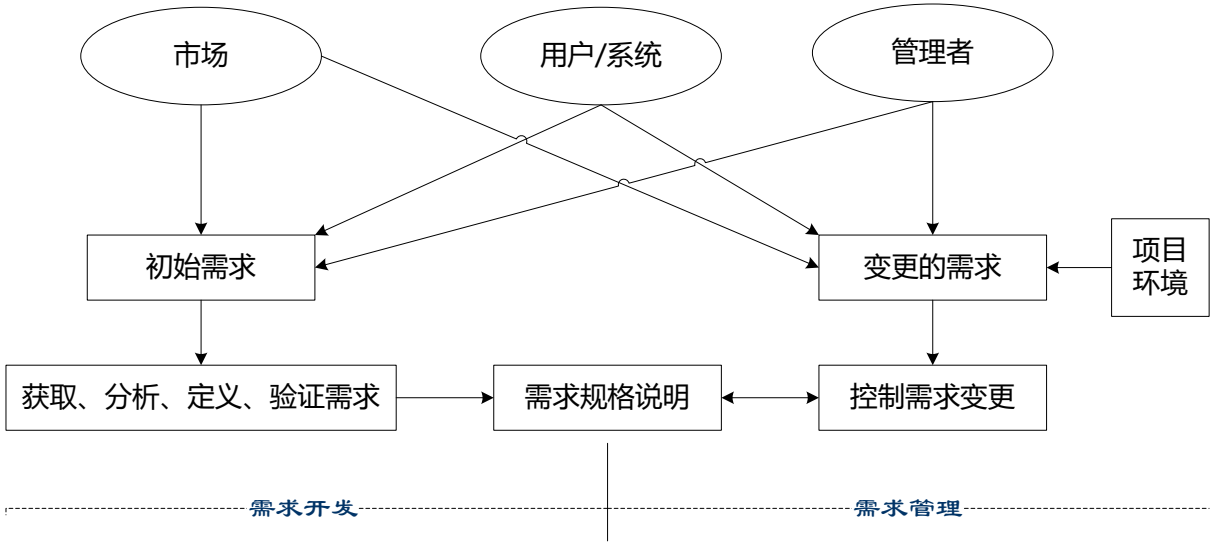
4.1 需求开发

需求工程是系统工程或软件工程中解决需求问题的一个崭新领域，目标在于使得到的产品能够准确、真实地体现客户的需要，令客户满意。

需求工程是运用已证实有效的技术和方法，通过合适的模型、工具、图表、记号，完整准确地描述软件需求规格，并对用户不断变化的需求演进给予支持。

需求是针对特定产品/项目的，受合同额、工期等因素约束，切忌被用户带得不着边际。

4.1 需求开发



需求工程包括需求开发、需求管理二个方面。

4.1 需求开发

需求开发目的：是通过调查与分析，获取用户需求并定义产品需求 。

需求管理目的：在用户与开发之间建立对需求的共同理解，维护需求与其他工作成果的一致性，并控制需求的变更；

需求管理强调的是需求的确认和需求变更控制，需求开发讲究的是通过系统的方法获取真正的、全面的、能实现的需求。

需求规格说明作为需求基线，设计对需求负责，开发对设计负责。

4.1 需求开发

什么是需求开发

需求开发是为实现项目目标，完成合同规定任务，通过调查分析，获取用户需求，确定软件产品或产品组件需求规格的过程，通常包括需求获取、需求分析、需求定义、需求验证等阶段，形成文档化的需求分析规格说明书。

4.1 需求开发

特定目标和特定实践

SG1	开发客户需求	
	SP1.1	引导需求
	SP1.2	将相关干系人的需要转化为客户需求
SG2	开发产品需求	
	SP2.1	建立产品或产品组件需求
	SP2.2	分配产品组件需求
	SP2.3	识别接口需求
SG3	分析并确认需求	
	SP3.1	建立操作概念和场景
	SP3.2	建立必要的功能和质量属性定义
	SP3.3	分析需求
	SP3.4	分配需求以取得平衡
	SP3.5	确认需求

（1）业务需求：反映了客户对系统、产品的概括的目标要求，他在项目目标与范围文档中予以说明。主要目的是对企业目前的业务流程进行评估，得出一个业务愿景。

（2）用户需求：描述了用户使用产品必须要完成的任务。用户需求在《用户需求说明书》中予以说明。

（3）功能（产品）需求：是满足用户需求，并对软件产品规格进行了详细描述的需求，定义了产品必须实现的软件功能。

4.1 需求开发

需求可以分为功能性需求和非功能性需求。功能性需求是系统要具备怎么的功能，能做什么事情。非功能性需求是指系统应具备怎样的性能，安全等级等方面的需求

软件需求规格说明书是开发、测试、质量保证、项目管理的基线

4.1 需求开发

需求的层次

- 组织机构、用户群
- 规模范围
- 业务目标、处理流程
- 使用场景与用例
- 功能需求
- 信息需求
- 接口需求
- 性能需求
- 技术需求
- 开发与运行环境需求
- 工期与项目管理需求
- 验收需求
- 运行维护需求
- 隐形需求

需求获取：收集用户对系统的需求陈述，确定软件需求。

需求分析：运用程式化、模型化方法分析软件需求。

需求定义：以规范化、模板化、文档化的形式完整准确地描述软件需求，产出物是需求分析规格说明书。

需求确认：反讲、验证、评审、确认软件需求规格。

4.1 需求开发

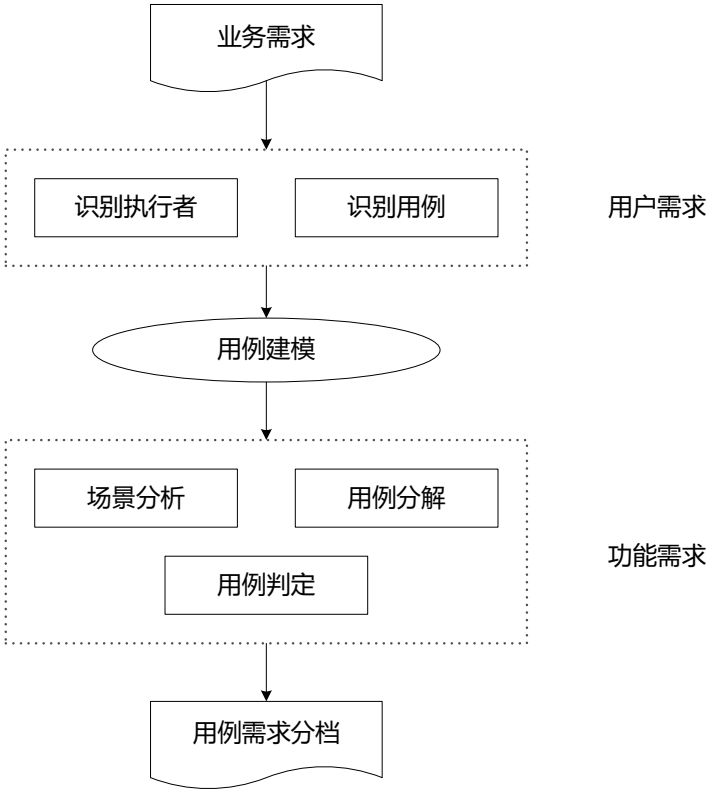
需求获取是通过一个诱导过程来收集客户需求；从交流到完整的需求理解，是一个复杂、反复的过程。对需求工程师的表达、理解、把控等能力要求较高。要求：

- 应确定产品的客户群或产品的服务对象；
- 要对需求分析人员进行业务技能培训；
- 对用户代表进行需求阶段相关知识培训，让他们了解需求阶段的工作性质和要求；培训为双方相互沟通和密切配合做准备，减少误会和返工，降低风险；

需求获取方法

书面调查	调研提纲、调查分析
访谈座谈	简单、直接的需求获取方法，一对一、一对多
焦点小组	选定的业务技术骨干和主题专家集中互动讨论
头脑风暴	开放式自由讨论，需要甄别裁剪
需求研讨	相关干系人尽量聚集到一起开会，讨论并达成需求共识
角色扮演	跟班操作、直接体验
用例模型	描述用户和系统之间的交互场景，逐层分解
原型方法	通过原型展示、直观理解需求

4.1 需求开发



基于用例的需求获取过程模型

需求获取过程：

导出干系人对整个产品生命周期的需要、期望、约束和接口要求。

转化干系人的需要、期望、约束、接口等为用户需求（用户需求说明书）

4.1 需求开发

需求分析是分析人员分析用户提供的需求信息，区分业务需求、功能需求、非功能需求和质量属性等，并且弄清楚每项需求的必要性。接着要分析这些需求实现的可行性。出现观点差异时，必须充分交换意见、认真讨论后取得共识，从而确定可以接受的和准备加以实现的需求。然后分配需求并提炼出软件功能（产品）需求。

需求分析方法

结构化分析方法	业务处理流程、数据流图、实体关系图、功能分解
面向对象分析方法	对象、属性、方法、类、消息通信
用例驱动分析方法	场景分析、识别执行者、识别用例
基于构件的分析方法	构件技术建模、软件总线、提高软件复用的层次和流程重组的灵活性
面向服务的分析方法	SOA、微服务、中台技术

需求分析的内容

建设目标	约束条件	假定条件	规模范围	组织机构	用户角色	业务流程	场景用例	功能需求	信息需求	接口需求	技术需求	资源需求	质量需求	性能需求	安全需求	管理需求	交付需求	验收需求	运维需求	扩展需求	关联影响

需求分析引导表示例

总体	明确的需求分析方法
	专业的领域知识、术语概念
	使用主动语态明确表示是什么、不是什么、做什么，不做什么
	厘清的关键任务、重要任务、次要任务，二八定律
	合理使用图、表、模型化、形式化、规约化、数学方法等描述形式
功能	针对每一项业务问题，明确定义业务目标、协同关系、功能清单、功能关联
	明确每一项功能的输入输出信息，如来源、传输方法、格式、规约
	明确每一项功能的业务操作规程、事件顺序、处理逻辑、数据加工、非正常响应方式

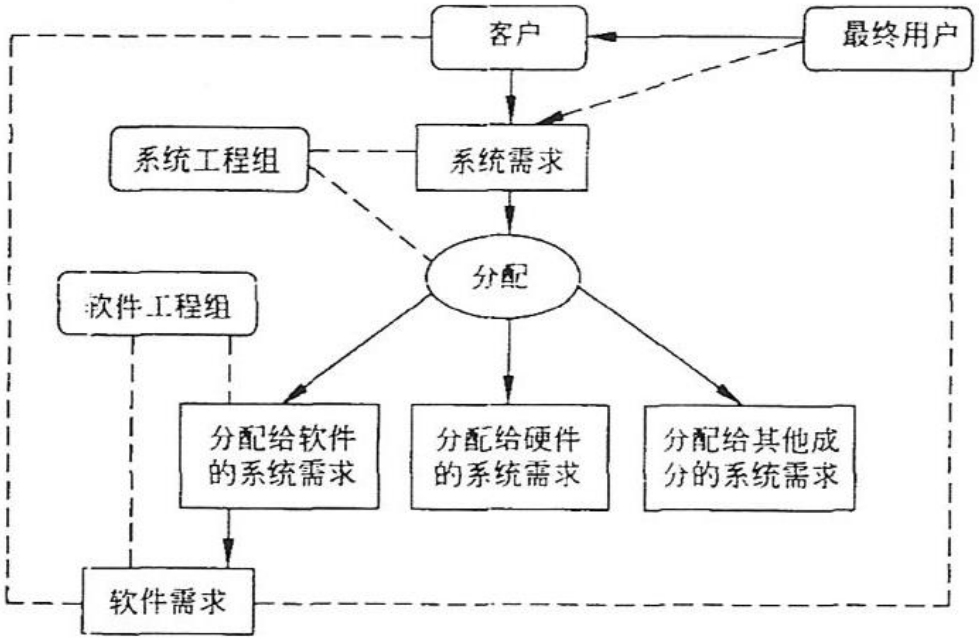
需求分析引导表示例

数据	定义全部实体数据，建立数据字典，如类型、取值范围、约束条件
	规定了系统的数据存储、数值计算、数据汇总、数据采集、数据交换的精度要求，
接口	定义软件与内部软件、组件、自主设备的接口关系和规约
	定义软件与第三方软件、组件、设备的接口关系和规约
性能	规定容量要求，如数据累计增长、数据容量、连接数、并发数
	规定事件特性要求，如主功能响应时间、数据更新时间、大数据查询时间、峰谷值
	规定可靠性要求，如平均故障间隔时间、故障响应时间、故障修复时间、回退操作

需求分析引导表示例

安全	规定数据的安全措施、保密要求、备份要求
	规定系统应具有的特殊要求或应急措施
	规定系统的操作日志、数据日志
界面	规定人机交互界面的风格、布局、交互操作和使用要求
	规定引导、帮助、提示等要求
环境	规定DBMS、中间件等软件支持环境
	规定计算资源、存储资源等设备及技术参数

需求分配



系统需求分配

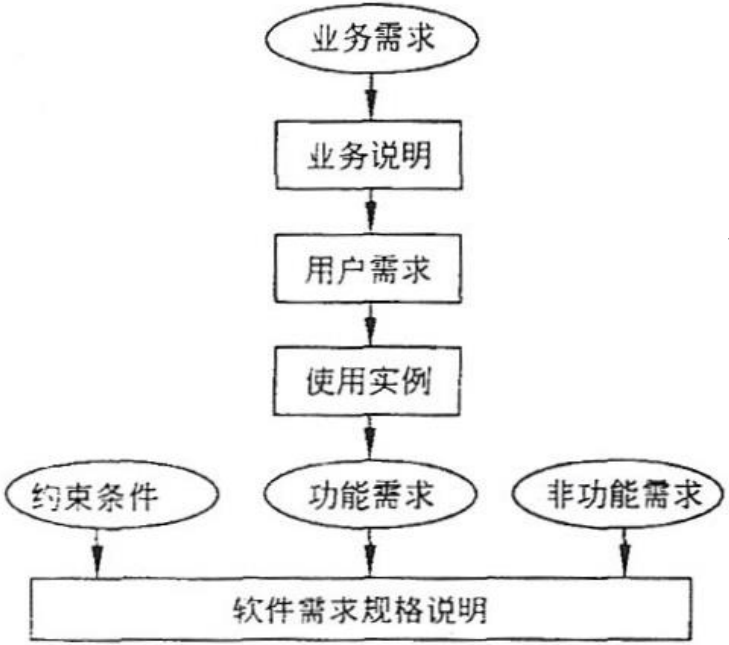
从用户那里取得需求以后，应将系统需求进行分解，与技术解决方案联系起来，所有的需求应该与设计的产品组件对应起来。保证需求驱动后续的设计工作，同时保证设计是为需求服务。

需求定义

需求定义是指确定下来的需求必须以正式文档形式表达，这就是需求规格说明。

详细描述软件与用户是怎么交互的，用户需要输入什么，系统会输出什么。

需求规格说明是对软件规格的描述，是可以用来做验收标准。



软件需求的层次

软件需求规格说明

用户使用软件必须达到的要求和完成的任务；通常使用实例、UML用例、方案脚本等加以说明。

过程需求：交付需求、实现需求、遵循标准

接口需求：互操作性、安全性、机密性

性能需求：响应时间、并发数、峰谷值

环境需求：应用范围、组织机构、用户群、开发环境、运行环境、技术体系

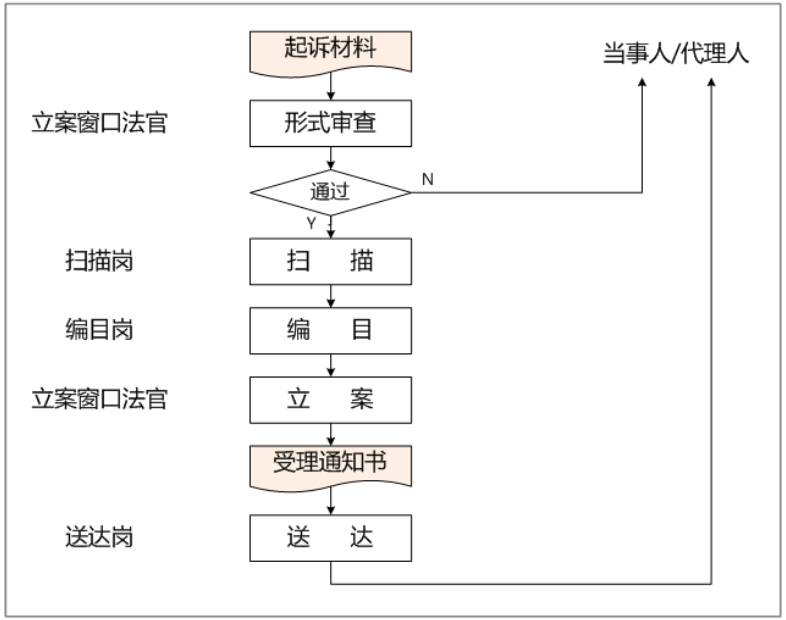
软件需求规格说明的内容

1	需求分析阶段使用的方法
2	分清软件的关键任务、重要任务、次要任务、其他有关任务
3	所使用的术语是明确的、定量的、可度量的
4	使用主动语态精确地表示软件必须做什么，不做什么
5	给出了软件的每一项功能及其目的
6	用文字、图形、逻辑或数学方法描述了每一项功能的特性
7	确定了与功能有关的所有输入信息，包括其来源、意义、格式、规约、接收方法、数量、输入范围及换算方法，并说明了时间要求、优先顺序、操作控制要求和所有的输入媒体
8	确定了从输入数据到中间数据，直到获得预期输出结果的全部过程、操作的准确顺序、对需求规格说明规定范围内的非正常情况的正确响应
9	确定了与功能有关的所有输出信息，包括信息的传送方法、意义、格式、数量、输出范围及换算方法
10	规定了与功能有关的事件要求、优先顺序和输出形式
11	确定了系统应具有的特殊要求或应急措施
12	规定了系统的精度要求，如数据的精度要求、数值计算的精度要求、数据汇总的精度要求、数据传送交换的精度要求

13	规定了系统的容量要求，如处理的记录数、表单显示的记录数、处理数据的最大容量
14	规定了系统的时间特性要求，如主要功能的响应时间、更新数据时间、大数据量查询时间以及峰值、谷值要求
15	指明了反映系统环境变化和系统适应能力的各项参数
16	说明了当需求发生某些变化时系统的适应能力
17	指出了为适应这些变化而需要设计的软件成分和过程（隐形需求）
18	定量地描述了软件系统应满足的具体可靠性要求，如平均故障间隔时间、故障响应时间、故障修复时间
19	指明了软件与内部软组件的各种接口关系，并说明了各个接口的特性和规约
20	指明了软件与系统需要使用的外部设备的各种接口关系，并说明了各个接口的特性和规约
21	规定了系统需要使用的每种外部设备对软件的要求、设备的类型号、功能、控制方法
22	指明了软件与第三方设备和软件系统的各种接口关系，并说明了接口的特性和规约
23	指明了软件的人机界面、交互操作和使用要求
24	指明软件人机界面采用的交互方式（菜单、图形），并在运行的每一步都提供使用的提示，以便操作者易学易用

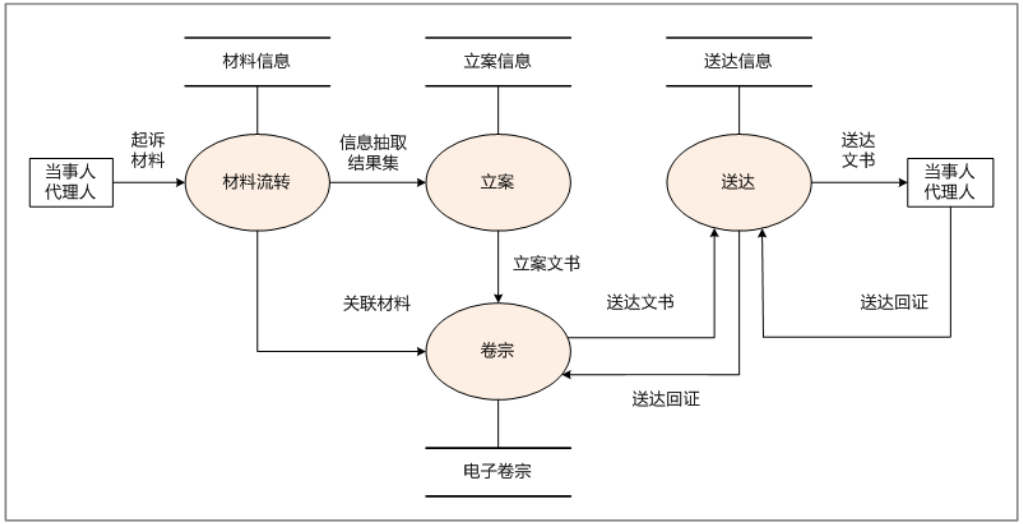
25	定义了系统使用的各种实体数据，并规定了静态数据、动态输入输出数据及内部生成数据的逻辑结构，列出了这些数据清单，建立了数据字典，说明了数据元素的约束
26	规定了数据采集的要求，被采集数据的特性、要求和范围
27	规定了开发和运行软件系统所需的硬件环境和软件支持环境
28	规定了开发软件系统所需要的系统资源，包括计算资源、存储资源、处理时间、产出物
29	规定运行软件系统所需的硬件设备、当前可用的设备、要求新采购的设备及技术参数
30	根据应用和数据累计增长情况，给出了这些设备的余量要求
31	指定了与软件开发和运行所需要的全部支撑软件，如数据库管理系统、应用服务器中间件、可视化工具、GIS、PORTAL、...)
32	利用功能分析、原型、模拟试验等方法对功能和性能需求进行了验证，与用户要求一致
33	规定了反映系统环境、能力以及用符号定义的各种参数
34	确定了软件的数据安全措施和保密要求
35	确定了哪些软件功能可能发生变动、功能变动后修改软件所需要的时间和范围
36	规定了影响软件开发和运行环境的一些假设、约束条件、风险分析，以及影响系统能力某些限制

4.1 需求开发

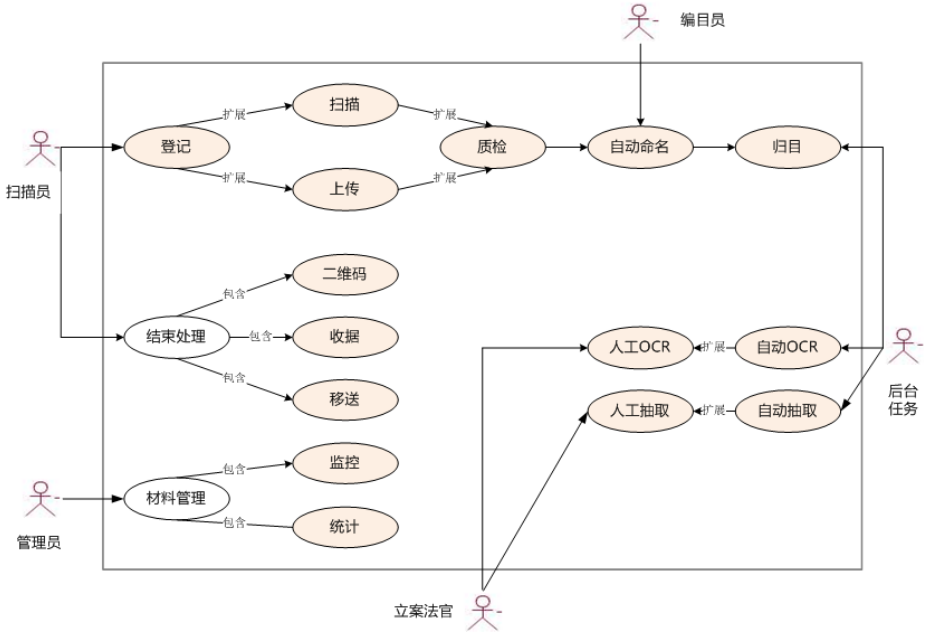


业务流程图

数据流图

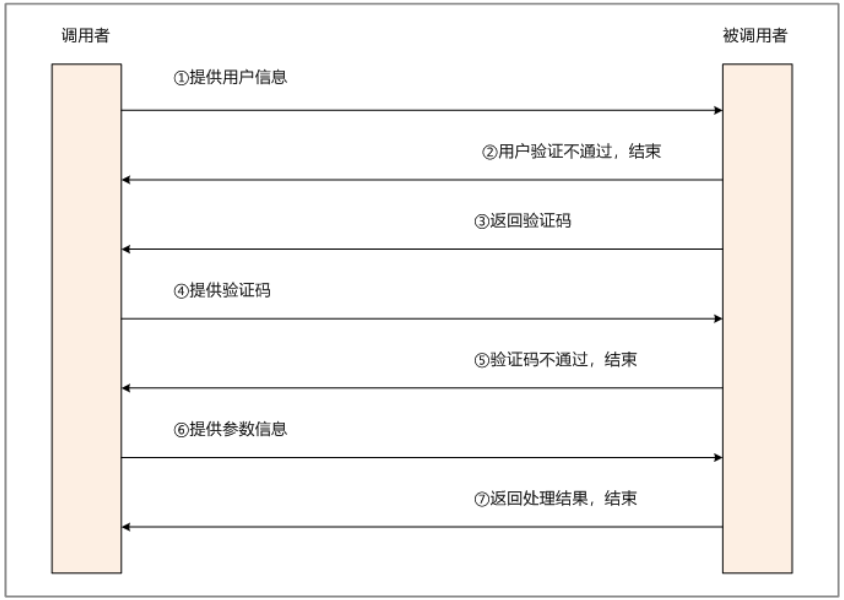


4.1 需求开发

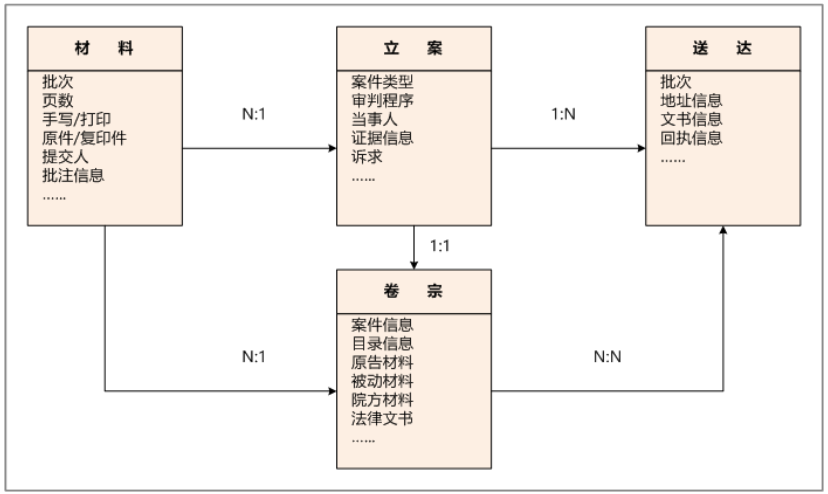


用例图

时序图

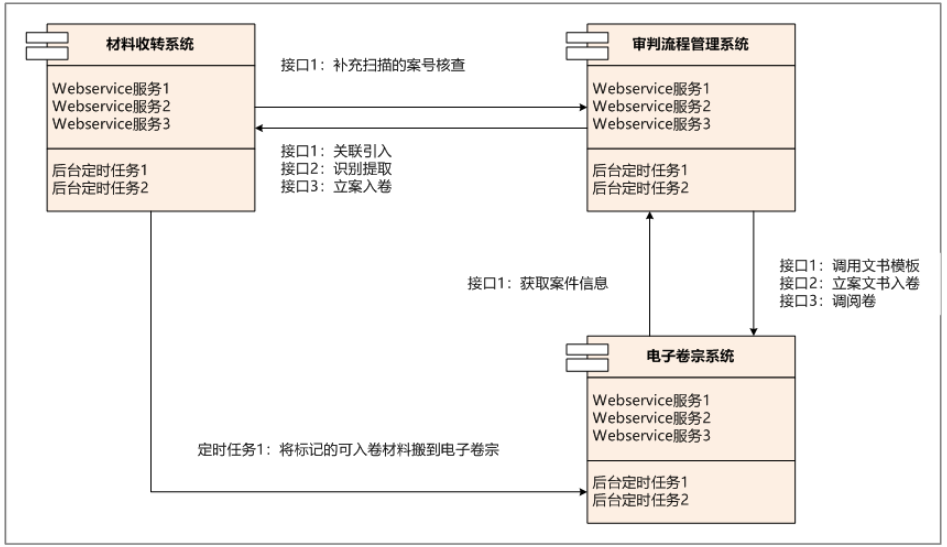


4.1 需求开发



实体关系图

接口关系图



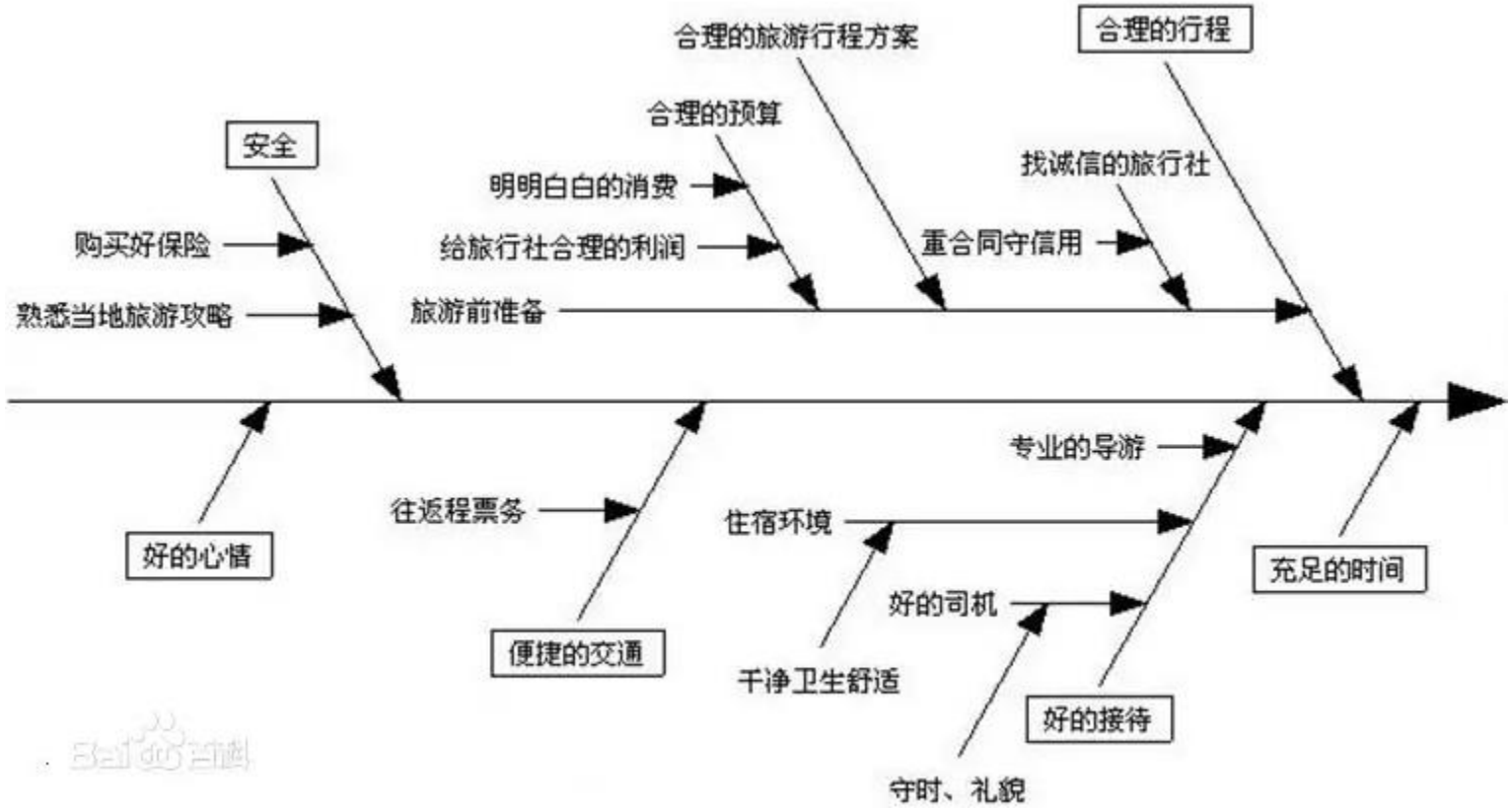
4.1 需求开发

思维导图

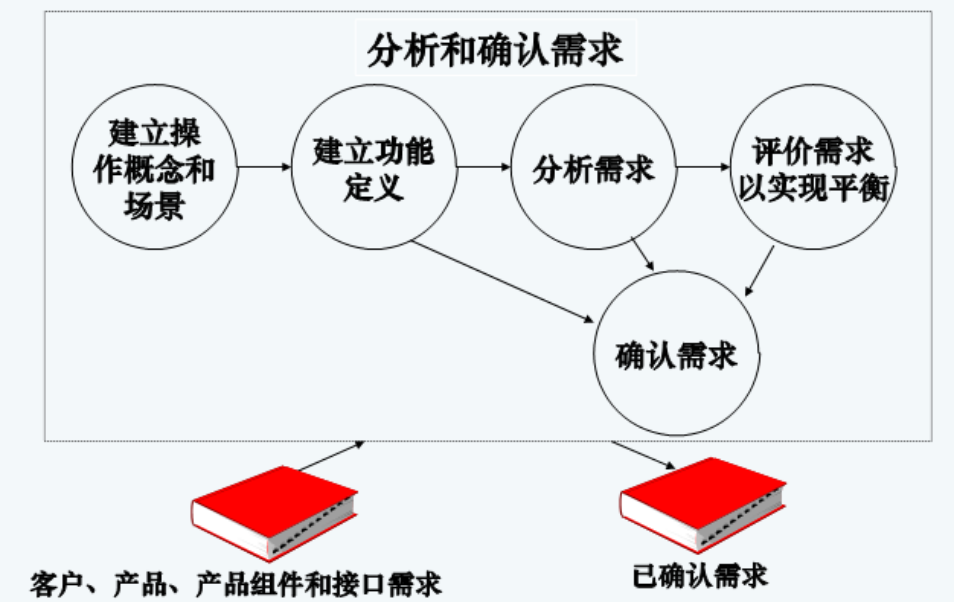

赡养纠纷
当事人信息
原告
被告（多被告可多组）除被告多组外，其他同离婚案件当事人信息
诉讼请求
支付赡养费（勾选赡养费这项，链接事实和理由中的“原告主张赡养费”相关事实与理由）
共同居住生活（勾选共同生活这项，链接事实和理由中“共同居住生活”相关事实与理由）
定期探望（勾选该项，链接定期探望的事实和理由部分）
诉讼费负担
事实与理由
原、被告的身份关系
父母与子女
祖父母与孙子女
外祖父母与外孙子女
其他关系
原告主张赡养费
主张赡养费数额（）元/月
原告日常花费
生活费（）元/月
医疗费（）元/月
护理费等其他费用（）元/月
原告收入
有固定收入（）元/年
无固定收入（）元/年
被告收入（多被告可多组）
有固定收入（）元/月
无固定收入（）元/月
共同居住生活
原告主张与被告共同居住
居住在原告处
居住在被告处
其他住处
原告的身体状况（勾选）
其他需要说明的情况
定期探望
原告主张探望的时间和方式（文本填写，例：每月探视一次，到原告住处探望）
诉讼费负担
备注
《中华人民共和国老年人权益保障法》第十八条第一款、第二款规定：家庭成员应当关心老年人的精神需求，不得忽视、冷落老年人。与老年人分开居住的家庭成员，应当经常看望或者问候老年人。


赡养纠纷
当事人信息
原告
被告（多被告可多组）除被告多组外，其他同离婚案件当事人信息
诉讼请求
支付赡养费（勾选赡养费这项，链接事实和理由中的“原告主张赡养费”相关事实与理由）
共同居住生活（勾选共同生活这项，链接事实和理由中“共同居住生活”相关事实与理由）
定期探望（勾选该项，链接定期探望的事实和理由部分）
诉讼费负担
事实与理由
原、被告的身份关系
父母与子女
祖父母与孙子女
外祖父母与外孙子女
其他关系
原告主张赡养费
主张赡养费数额（）元/月
原告日常花费
生活费（）元/月
医疗费（）元/月
护理费等其他费用（）元/月
原告收入
有固定收入（）元/年
无固定收入（）元/年
被告收入（多被告可多组）
有固定收入（）元/月
无固定收入（）元/月
共同居住生活
原告主张与被告共同居住
居住在原告处
居住在被告处
其他住处
原告的身体状况（勾选）
其他需要说明的情况
定期探望
原告主张探望的时间和方式（文本填写，例：每月探视一次，到原告住处探望）
诉讼费负担
备注
《中华人民共和国老年人权益保障法》第十八条第一款、第二款规定：家庭成员应当关心老年人的精神需求，不得忽视、冷落老年人。与老年人分开居住的家庭成员，应当经常看望或者问候老年人。

鱼骨图


合理的旅游行程方案
合理的行程
合理的预算
安全
找诚信的旅行社
明明白白的消费
购买好保险
给旅行社合理的利润
重合同守信用
熟悉当地旅游攻略
旅游前准备
专业的导游
往返程票务
住宿环境
好的心情
充足的时间
好的司机
便捷的交通
干净卫生舒适
好的接待
守时、礼貌

需求规格说明必须具有完整性、正确性、可行性、无歧义性和可验证性。能否符合上述要求，需要经过评审。评审后应对发现的问题做出必要的跟踪和修改。



- 4.1 需求开发RD
- 4.2 需求管理RM
- 4.3 技术解决方案TS
- 4.4 产品集成PI
- 4.5 验证VER
- 4.6 确认VAR
- 4.7 软件测试
- 4.8 同行评审

4.2 需求管理

需求管理的目的是在客户与开发方之间建立对需求的共同理解，维护需求与其他工作的一致性，并控制需求的变更。

软件需求是软件产品的源驱动力，需求管理以需求开发过程确认的需求规格为基础，实现需求理解、需求承诺、需求基线、需求变更、需求一致性追溯等管理过程。

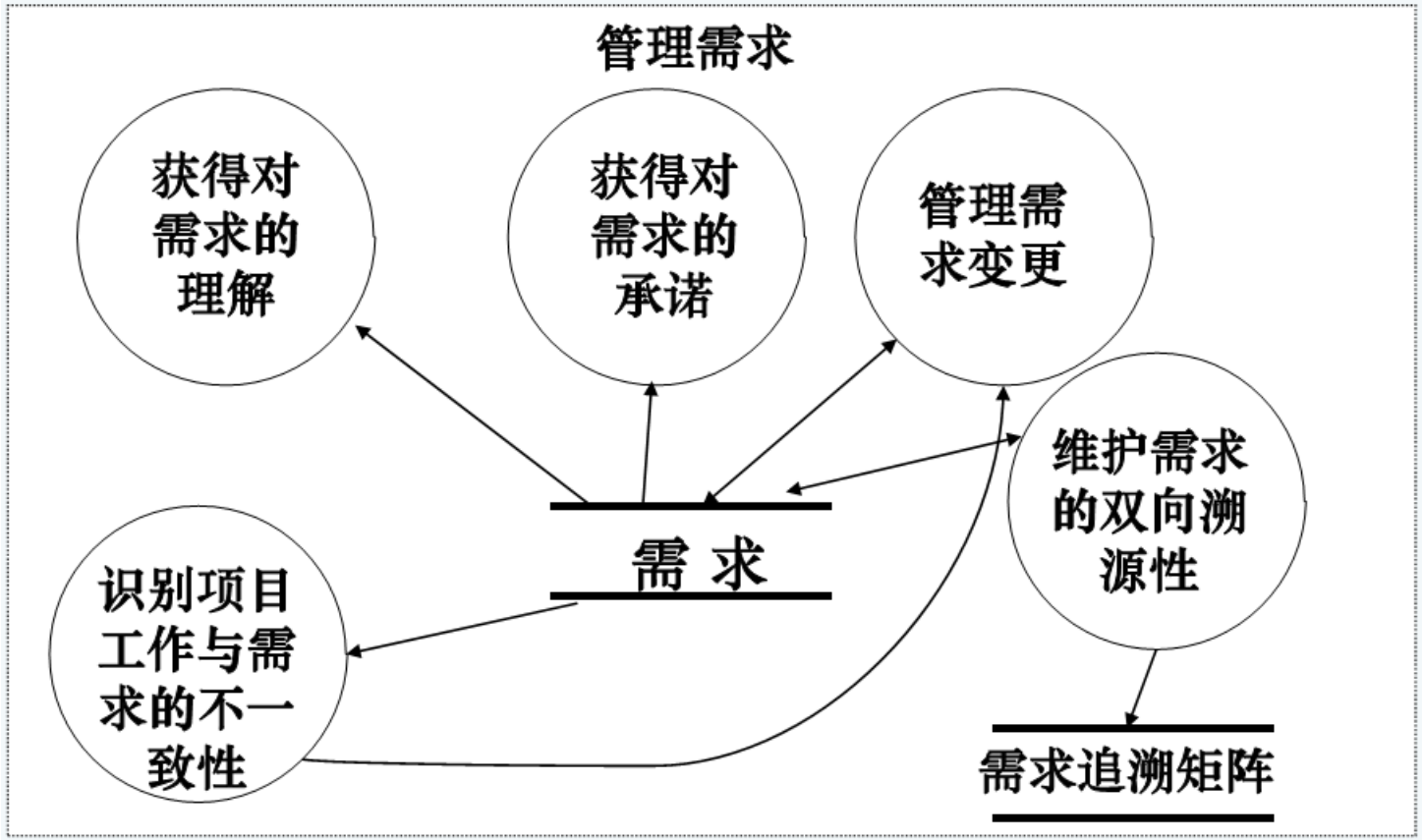
4.2 需求管理

特定目标和特定实践

SG1	管理需求	
	SP1.1	理解需求
	SP1.2	获得对需求的承诺
	SP1.3	管理需求的变更
	SP1.4	维护需求的双向可追溯性
	SP1.5	确保项目工作和需求之间的一致性

4.2 需求管理

特定目标和特定实践



4.2 需求管理

需求管理的内容

（1）需求确认。开发方和客户共同对需求文档进行评审，双方达成共识后做出书面承诺。

（2）需求跟踪。需求跟踪使得每一项需求均能追溯到相应的设计、代码和测试用例。同时，各阶段的工作产品也能反向追溯到初始的需求；

（3）需求变更控制。是指需求从提出变更申请到变更的具体实施进行控制的过程，通过对需求基线实施配置管理来实现。

4.2 需求管理

（1）获得对需求的理解。与需求提供方一起对需求进行分析，以确保双方对需求含义的一致性理解。

（2）获得对需求的承诺。是在执行必要的需求实现活动的人员之间达成一致与承诺。确保项目参与者随着需求的演变，对当前的、已批注的需求及其导致的项目计划、活动与工作产品的变更作出承诺。

正式需求评审：邀请同行专家和用户一起评审需求文档，尽最大努力使需求文档能够正取无误地反映用户的真实意愿。

（1）用户因素：用户对需求有了新的认识，提出了新的、变更了的需求；用户要求工期紧，匆忙上马。

（2）需求开发缺陷：需求分析、定义和评审工作不够充分，导致需求规格说明中隐含着问题，如未能获得用户的潜在需求，事后才有所发现；

（3）市场因素：市场形势变化引发的需求变更；

（4）系统因素：在系统内部，如计算机硬件、系统软件或数据等的变更要求软件与其相适应；

（5）环境因素。与软件运行相关的工作制度或政策法规的变更，或是业务要求变更导致的需求变更；

4.2 需求管理

（1）所有需求变更必须遵循需求变更控制规程实施变更。若某一需求变更提出请求后未获准采纳，则后续过程中不再考虑；

（2）需求变更提出后是否被接受，应由专门的变更控制委员会（change control bord，CCB）审查决定;

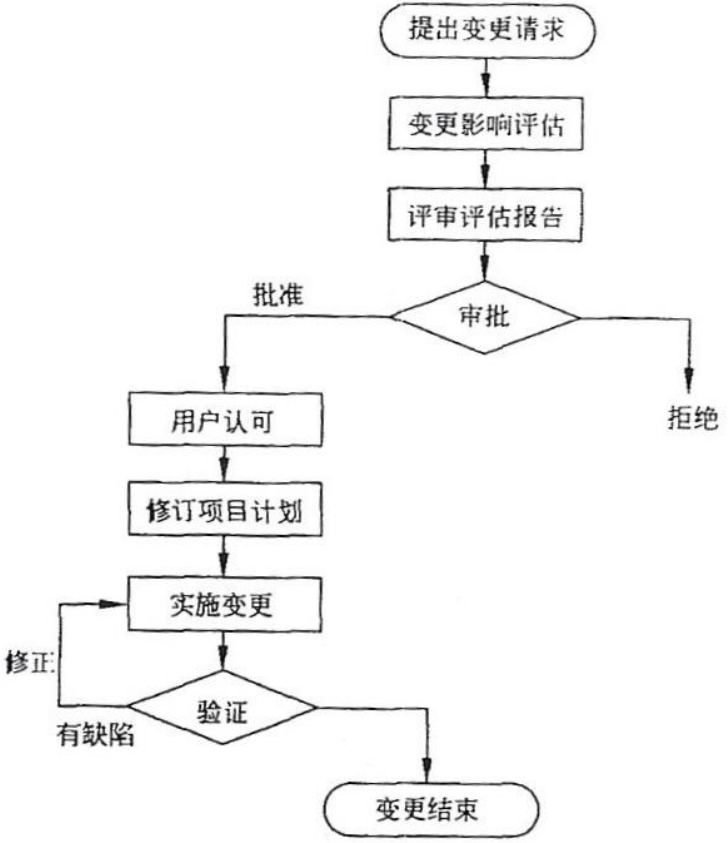
（3）不得以任何理由删除或修改需求变更的原始文件；

（4）应将已接受的需求变更通知到所有相关干系人；

（5）已接受的需求变更应能追溯到批准的变更请求；

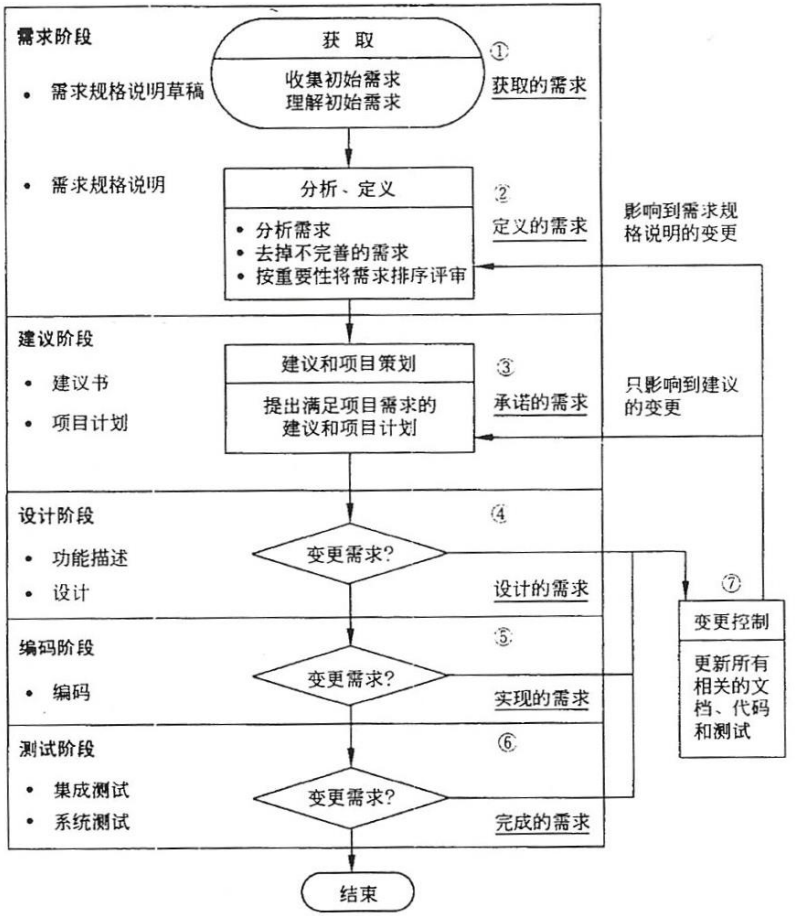
（6）对项目的需求赋予状态属性，以利于需求变更控制。

4.2 需求管理



需求变更处理流程

需求变更控制



需求变更控制流程

4.2 需求管理

需求可追溯性管理

需求可追溯性管理应使每一项需求均能追溯到：对应的设计、实现该项需求的代码、测试此项实现的用例。这样便可做到确保软件产品满足所有需求，并已测试了所有需求，从而使需求的前后集成关系脉络清楚。

建立与维护需求跟踪矩阵

追溯矩阵实例

①	②	③	④	⑤	⑥	⑦	⑧
需求号	需求描述	概要设计文档索引号	对应的设计（功能、结构、数据库）	实现（程序、类、继承类）	单元测试用例	集成/系统测试用例	验收测试用例
1.1.2	利用收集的数据实现亮点的实时集成	5.3.2	数据采集与亮度控制器接口	PB405 数据采集	#12	#46	#11
				CICS203 亮点控制器启动	#1	#47	#11

4.2 需求管理

需求可追溯性管理

向前追溯：沿生存期从需求规格到设计、编码、测试等后继阶段输出产品的相关元素。

向后追溯：从各阶段工作产品的元素反向追溯，直至追溯到初始需求。

确保项目工作与需求间的协调一致。

- 4.1 需求开发RD
- 4.2 需求管理RM
- 4.3 技术解决方案TS
- 4.4 产品集成PI
- 4.5 验证VER
- 4.6 确认VAL
- 4.7 软件测试
- 4.8 同行评审

4.3 技术解决方案

项目工期紧，常常成为很多事情的理由。因为赶时间，拿到需求后，不考虑哪种设计方案更合适，想到什么办法就用什么办法来做，甚至是没有设计可言，直接编码，写设计文档变成了浪费时间的一个事情。不少人进行设计的时候，眼光都放得不够开，不知道公司提供了很多可重用的代码或第三方产品。

TS重点强调的是：技术解决方案及其设计、实现过程，即整个软件开发周期的技术解决路径，强调了从粗到细的演变过程。

4.3 技术解决方案

目的

技术解决方案(Technical Solution, TS) 的目的是设计、开发及实现需求的解决方案。简单说TS就是评估与选择解决方案，对选定的解决方案进行详细设计、开发、实现。

4.3 技术解决方案

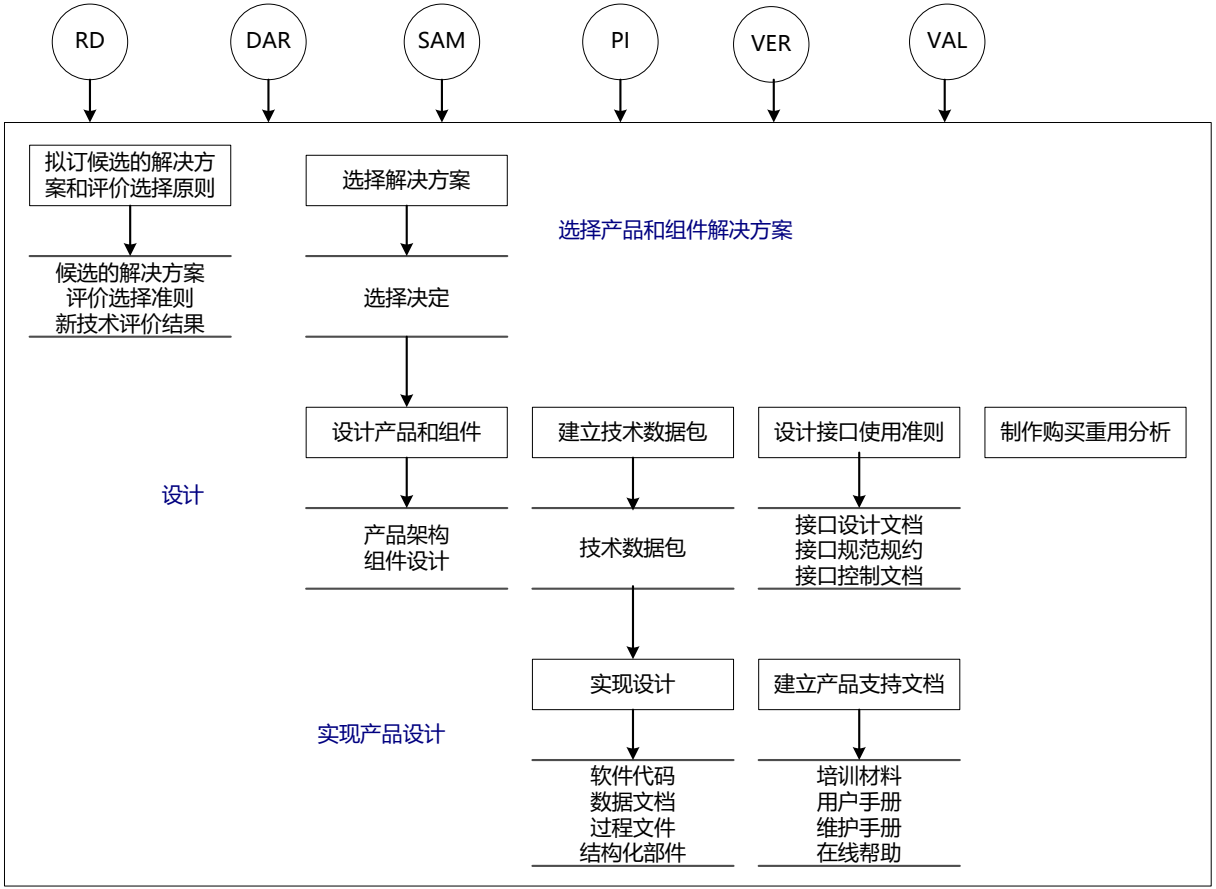
特定目标和特定实践

关键过程域——技术解决方案TS

SG1	选择产品和产品组件解决方案	
	SP1.1	开发备选解决方案及评选准则
	SP1.2	选择产品组件解决方案
SG2	开发设计：设计产品和产品组件	
	SP2.1	设计产品或产品组件
	SP2.2	建立技术相关数据
	SP2.3	使用准则设计接口
	SP2.4	执行自制、购买或重用分析
SG3	实现产品设计	
	SP3.1	实现设计
	SP3.2	建立产品支持文件

4.3 技术解决方案

特定目标和特定实践



技术解决方案的选择标准

- 可用性：系统的顺畅运行、可靠稳定，对用户的商业经营和业务管理极为重要。
- 可扩展性：新需求、新技术易导入，适应用户数、连接数、频率、数据的增长。
- 可维护性：排除缺陷、优化升级、移植迁移时，较少的运维成本和技术难度。
- 可定制化：对行业软件来说，可根据市场需求、个性化需求进行适度调整。
- 用户体验：易学易用好用。
- 市场时机：面对同行竞争，以最快速度推出伸缩性好的产品，占领市场。

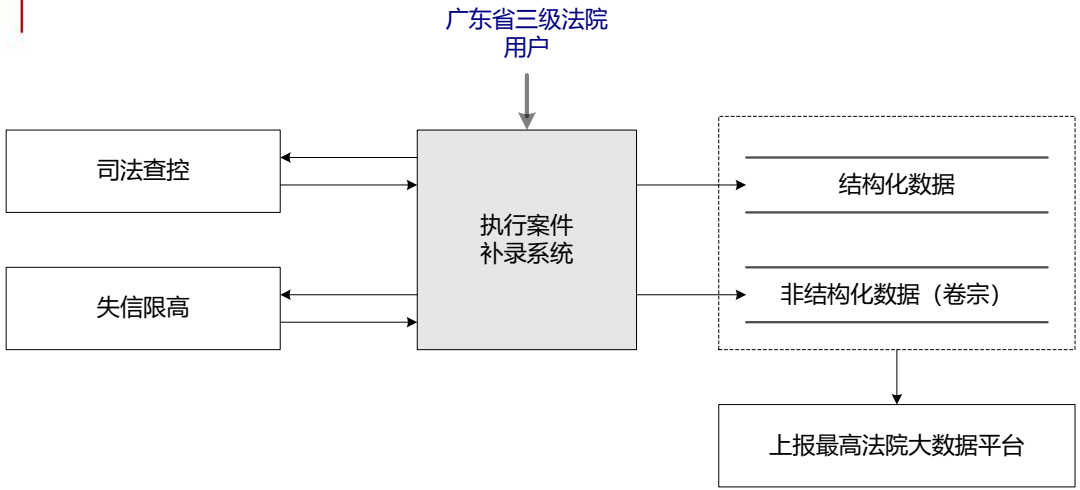
4.3 技术解决方案

充分考虑成本、进度、效益、风险、技术性能等因素，针对产品需求规格说明，依据组织技术方案选择标准从候选方案中选出合适的解决方案（也称“总体设计”“概念设计”），含系统架构所需的全部信息。如：

- 开发技术路线：C/S、B/S、J2EE、SOA
- 数据库：Oracle、Mysql、人大金仓、达梦
- 应用服务器：Tomcat、Weblogic、金蝶；
- 业务和数据管理模式：分布式、大集中；
- 第三方组件（控件）：文字编辑、文件批注、网页浏览；
- 系统支撑平台：硬件、操作系统等；

4.3 技术解决方案

技术解决方案选择



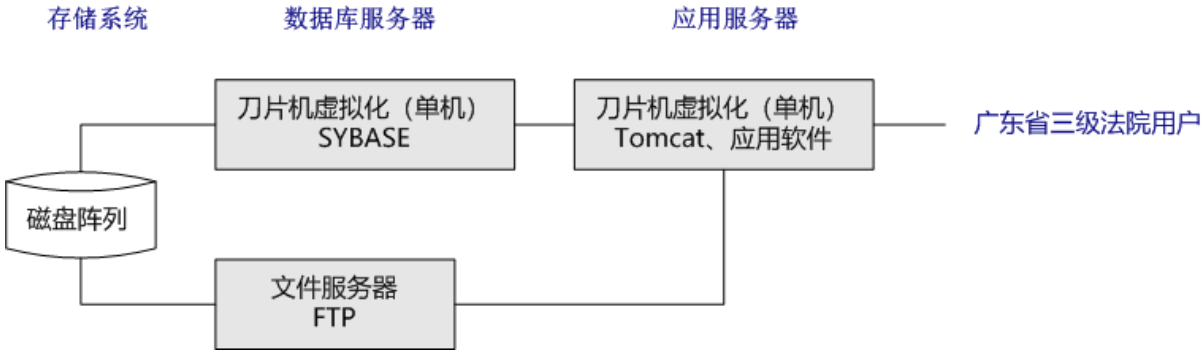
1、数据库卡慢瘫、应用服务器CPU爆满，系统无法使用；

2、分库、分应用、法院用户错峰使用：上班时间不准补录、18:00-23:00/23:00-04:00/04:00-09:00；

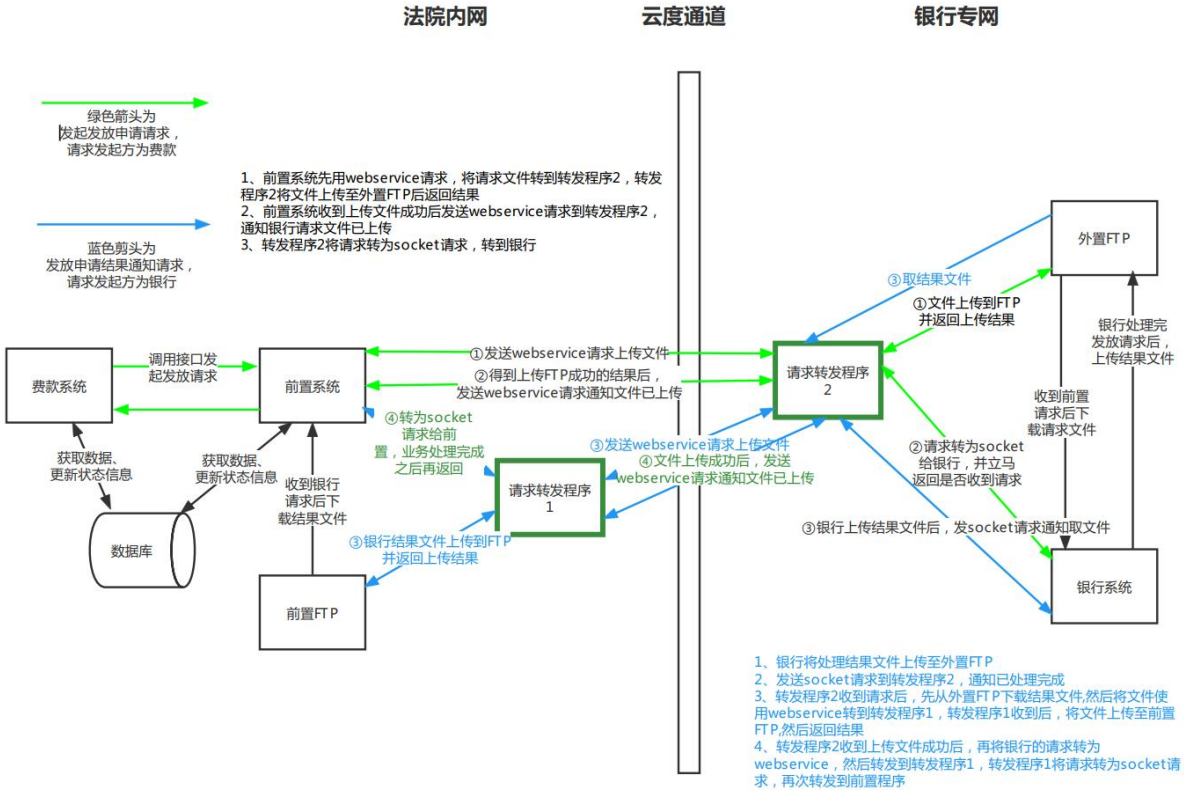
1、规模估计

2、技术方案

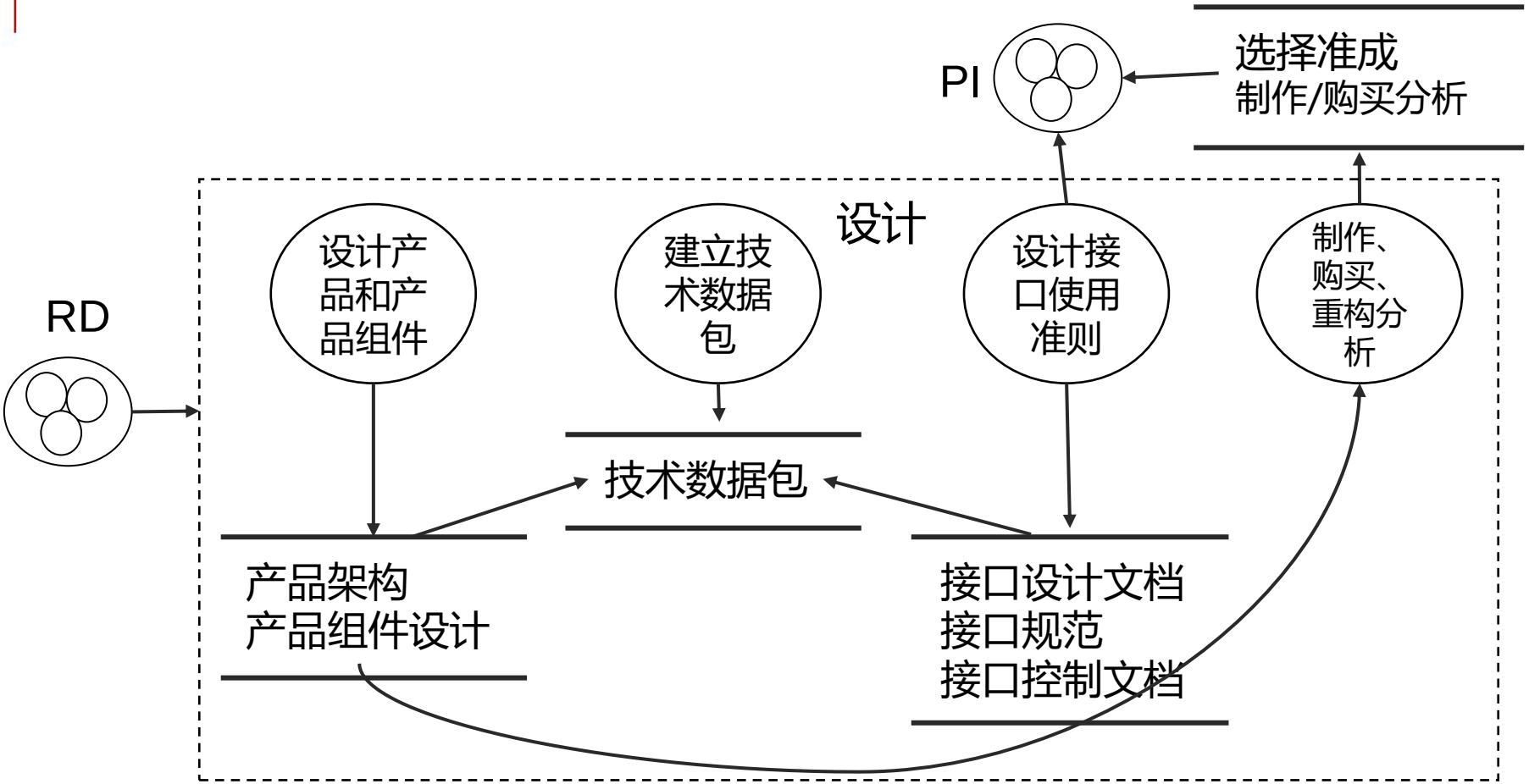
3、大字段处理



4.3 技术解决方案


法院内网
云度通道
银行专网
绿色箭头为
发起发放申请请求，
请求发起方为费款
蓝色剪头为
发放申请结果通知请求，
请求发起方为银行
1、前置系统先用webservice请求，将请求文件转到转发程序2，转发程序2将文件上传至外置FTP后返回结果
2、前置系统收到上传文件成功后发送webservice请求到转发程序2，通知银行请求文件已上传
3、转发程序2将请求转为socket请求，转到银行
费款系统
调用接口发起发放请求
前置系统
①发送webservice请求上传文件
②得到上传FTP成功的结果后，发送webservice请求通知文件已上传
④转为socket请求给前置，业务处理完成之后再返回
获取数据、更新状态信息
获取数据、更新状态信息
收到银行请求后下载结果文件
数据库
前置FTP
③银行结果文件上传到FTP并返回上传结果
请求转发程序1
③发送webservice请求上传文件
④文件上传成功后，发送webservice请求通知文件已上传
请求转发程序2
外置FTP
③取结果文件
①文件上传到FTP并返回上传结果
银行处理完发放请求后，上传结果文件
收到前置请求后下载请求文件
②请求转为socket给银行，并立马返回是否收到请求
③银行上传结果文件后，发socket请求通知取文件
银行系统
1、银行将处理结果文件上传至外置FTP
2、发送socket请求到转发程序2，通知已处理完成
3、转发程序2收到请求后，先从外置FTP下载结果文件,然后将文件使用webservice转到转发程序1，转发程序1收到后，将文件上传至前置FTP,然后返回结果
4、转发程序2收到上传文件成功后，再将银行的请求转为webservice，然后转发到转发程序1，转发程序1将请求转为socket请求，再次转发到前置程序

4.3 技术解决方案


PI
选择准成
制作/购买分析
设计
RD
设计产品和产品组件
建立技术数据包
设计接口使用准则
制作、购买、重构分析
技术数据包
产品架构
产品组件设计
接口设计文档
接口规范
接口控制文档

4.3 技术解决方案

产品设计传统上划分为概要设计和详细设计二个阶段。

概要设计通常包括业务逻辑设计（再造业务处理流程）、技术架构设计、数据结构设计、功能结构设计、交互接口设计、系统性能设计、系统安全设计、运行环境设计（硬软件支撑环境）、应急保障设计等内容。

详细设计则根据概要设计所做的模块划分，实现各模块的算法设计、用户界面设计、数据结构设计的细化等工作。

4.3 技术解决方案

（1）原型设计方法。推出探索原型、实验原型，积极鼓励用户改进需求，逐步演进原型，直至用户满意。

（2）结构化设计方法。把系统作为一系列数据流的转换，输入数据被转换为期望的输出值，通过模块化来完成自顶向下实现的文档化，并作为一种评价标准在软件设计中起指导性作用，通常与结构化分析方法衔接起来使用，以数据流图为基础得到软件的模块结构。模块分解与功能抽象。

（3）基于信息隐蔽原则的parnas设计方法。在概要设计时列出将来可能发生变化的因素，并在模块划分时将这些因素放到个别模块内部，从而提高软件的可维护性、避免错误的蔓延。

（4）面向数据结构的设计方法。从目标系统的输入输出数据结构入手，导出基本的处理框，分析这些处理框之间的先后关系，导出程序框架结构，再补充其他细节，就可以得到完整的程序结构。

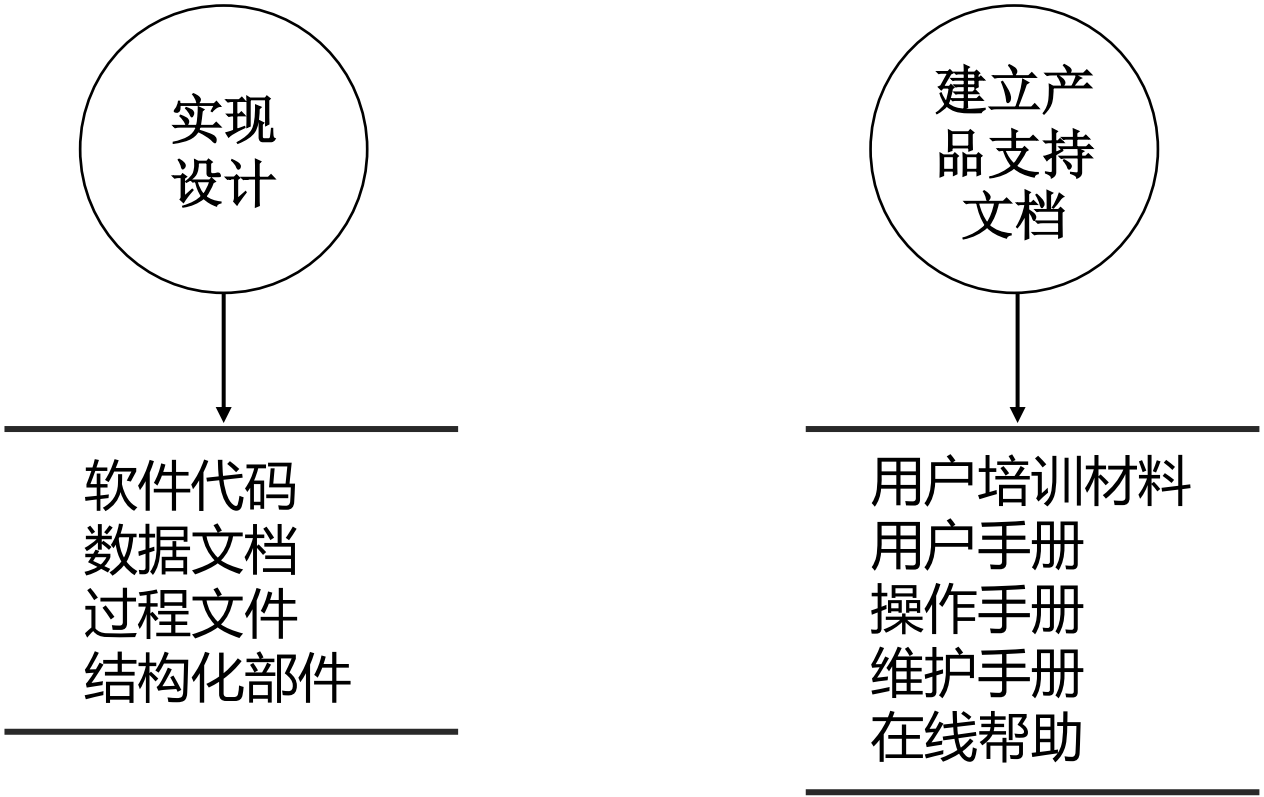
（5）面向对象的设计方法。以对象建模为基础，若干个对象类可以组成一个系统（层次结构），不同的对象类之间仅能通过传递消息来互相联系。其本质是抽象化定义和组织现实世界中的概念的过程，使开发软件的方法和过程与人类发现问题、解决问题的方法和过程尽可能地接近，使叙述问题的问题空间（问题域）和实现解决问题的方法的解空间（求解域）在结构上尽可能地达成一致。

（6）面向构件的设计方法。用“构件”取代“代码”，构件成为软件产品或系统的基本结构单元。构件可以完成一个或多个功能的特定服务，并为用户提供多个接口，复杂的业务需求可以在基础构件和行业构件上可视化组装完成。

（7）面向工作流的设计方法。从分析业务的流转过程入手，导出流转环节，然后分析业务和组织机构关系，导出软件结构。

4.3 技术解决方案

实现产品设计



养成良好的开发习惯

1、开发前要与需求分析人员或设计人员沟通你所理解的功能是否正确；

2、懂得测试技术，撰写单元测试用例，实现自动化测试；

3、20%的代码实现主要的业务逻辑，80%的代码在处理异常，保证程序的健壮性；

4、开放封闭的原则，尽量少改前人的代码，最好重构以防止修改而带来的不可估计的错误；

5、开发过程有问题时，不能自己做决定，而是应该与需求人员或设计人员沟通，或是提供自己的建议。

编程理念

1、极限编程

2、测试驱动编程

3、采用成熟技术，不追求新技术

4、遵守已定义的编程准则和规范

5、经常进行代码审查、走查

6、代码提交测试前充分单元测试

7、每日构建，确保能成功的构建软件包

8、定期代码重构

4.3 技术解决方案

规范遵循

1、结构化、模块化、层次清楚

2、清晰性、简易性、易读

3、编程语言标准、规范

4、变量命名的约定

5、足够的注释行

6、特定规则，如：指针使用、内存释放

编程方法

1、结构化程序开发

2、面向对象程序开发

3、自动代码生成

4、软件代码复用

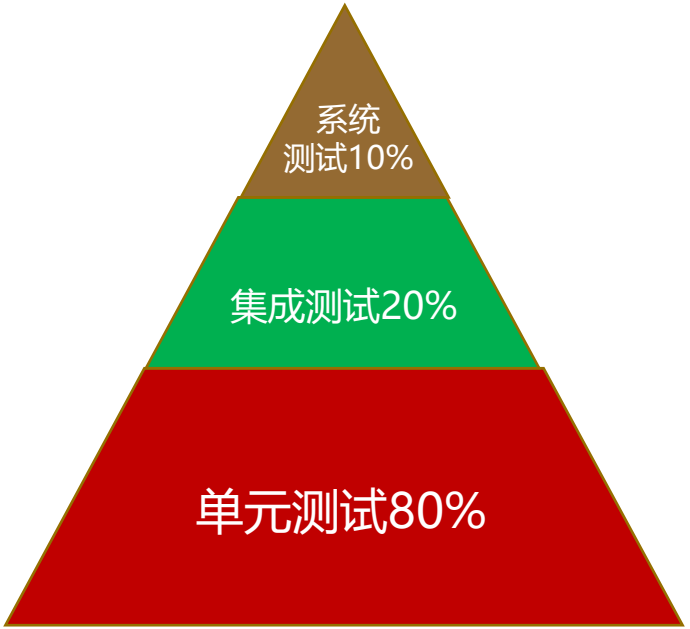
5、参考实用的程序范例

单元测试

测试依据：组件的规格说明或软件设计或数据模型

测试方法：

- 使用单元测试用例进行检验
- 借助于开发环境支持的单元测试框架
- 测试驱动开发（借助自动化测试工具）
- 静态分析工具，如FindBug，PMD
  - 产生代码测试覆盖率
  - 找到不可达的死代码
  - 语法错误，使用了未定义的变量
- 同行评审（代码复查）



单元测试技术

1、语句覆盖：所以语句至少覆盖一次的目的；

2、判定覆盖（分支覆盖）：对每个判定的分支至少覆盖一次；

3、条件覆盖：每一个判定语句中每个逻辑条件的可能值至少满足一次；

4、判定-条件覆盖：判定中每个条件的所有可能至少出现一次，并且每个判定本身的判定结果也至少出现一次；

5、条件组合覆盖：每个判定中条件可能取值的各种可能组合至少出现一次；

4.3 技术解决方案

开发和维护用于产品安装、操作及维护的相关文档。如

1、用户使用手册

2、终端用户培训教材

3、系统安装手册

4、系统管理维护手册

5、在线求助

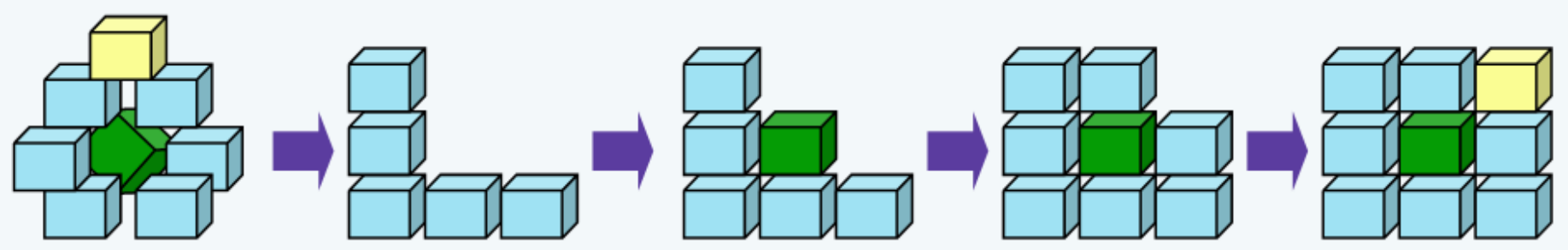
……

- 4.1 需求开发RD
- 4.2 需求管理RM
- 4.3 技术解决方案TS
- 4.4 产品集成PI
- 4.5 验证VER
- 4.6 确认VAL
- 4.7 软件测试
- 4.8 同行评审

4.4 产品集成

目的

产品集成就是把组成产品的所有软件组件组装起来，包括软件组件之间的集成、软件与硬件的集成、软件基础数据的配置等，目的是将产品组件组合为产品、并确保已集成的产品能在目标环境上适当地运行及交付。



4.4 产品集成

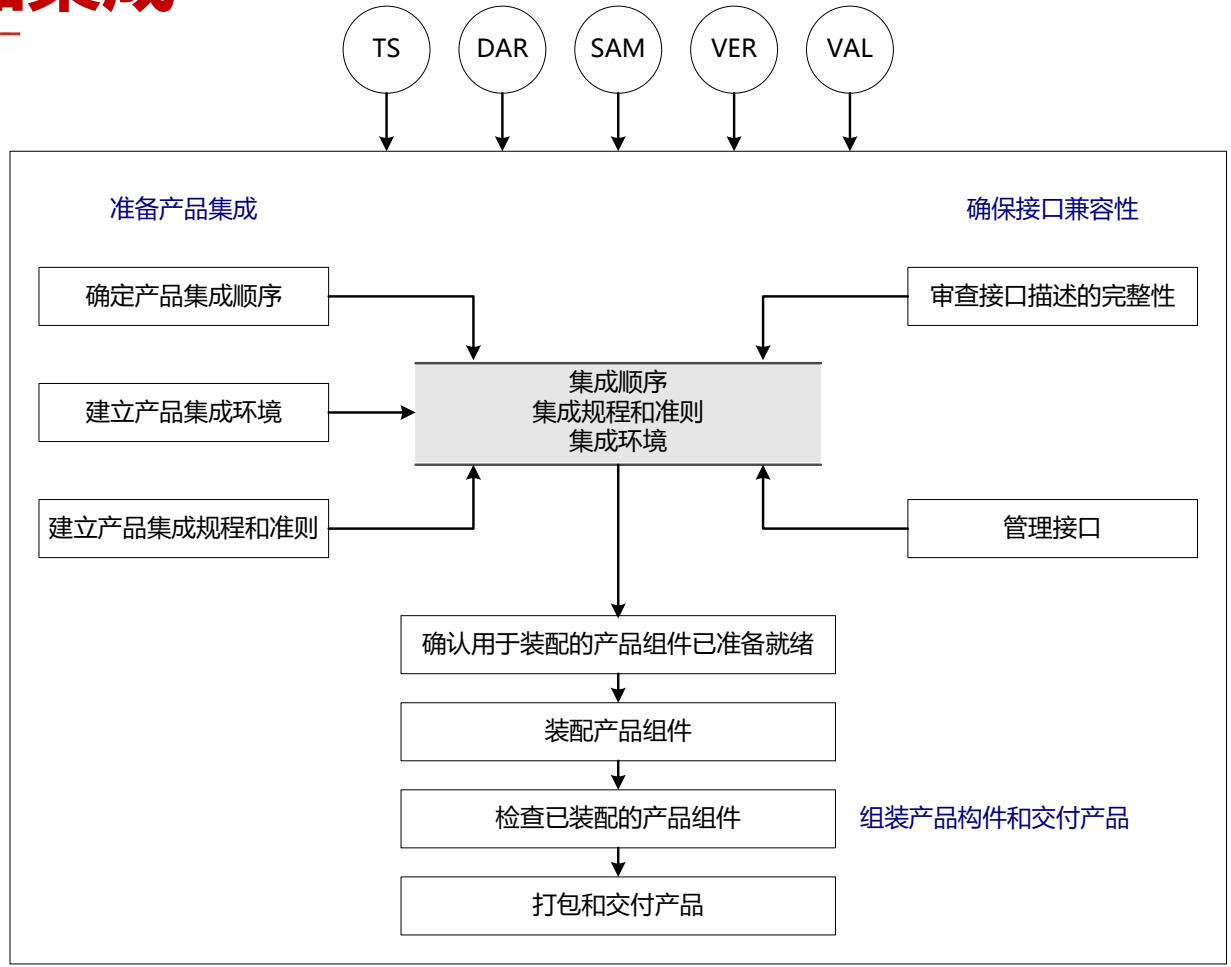
特定目标和特定实践

关键过程域——产品集成PI

SG1	准备产品集成	
	SP1.1	建立产品集成步骤
	SP1.2	建立产品集成环境
	SP1.3	建立产品集成规程与准则
SG2	确保接口兼容性	
	SP2.1	审查接口描述的完备性
	SP2.2	管理接口
SG3	组装产品构件和交付产品	
	SP3.1	确认用于装配的产品组件已准备就绪
	SP3.2	装配产品组件
	SP3.3	检查已装配的产品组件
	SP3.4	打包和交付产品

4.4 产品集成

特定目标和特定实践

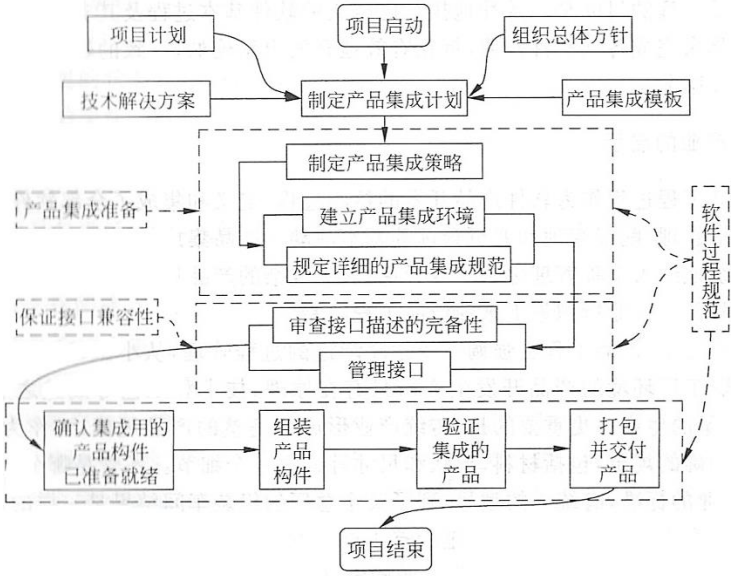


4.4 产品集成

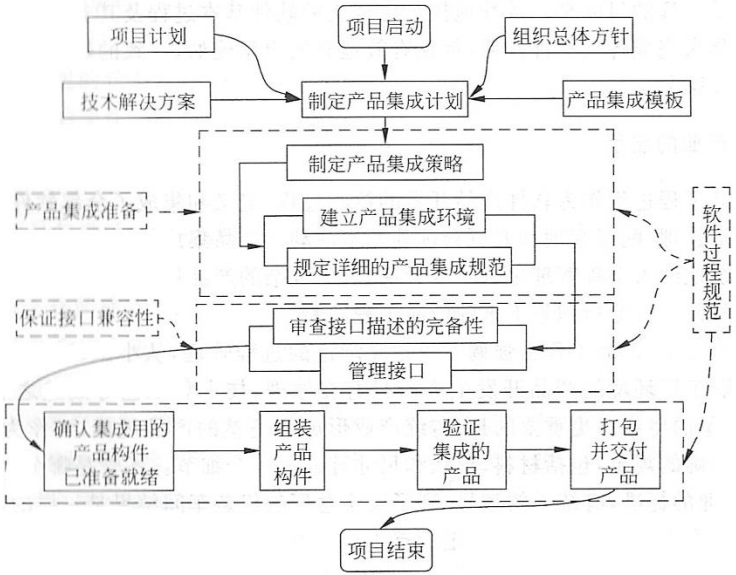
产品集成的任务

（1）制定软件产品集成计划，它是基于企业总体方针、产品集成模板、项目计划和软件产品的技术解决方案制定出来的，对软件产品集成过程的具体任务描述与进度计划；

（2）产品集成的准备工作，包括制定产品集成的策略、建立产品集成环境和详细的产品集成规范。在这过程中，受到组织已定义的软件过程规范的指导和约束；



4.4 产品集成



（3）保证产品集成接口的兼容性，审查接口描述的完备性，并能很好地管理这些接口，包括配置项管理、变更管理，确保接口变更的一致性；

（4）实施集成，包括一系列的软件活动，确认产品集成构件已准备就绪（进行了充分的单元测试并通过测试标准）、组装构件、验证已集成的产品、打包并交付完整的产品。

4.4 产品集成

（1）建立并维护产品集成的组织策略和组织方针，集成策略主要是确定要使用产品构件间接口定义、集成顺序规则，选择最佳集成顺序，并支持产品构件的渐进式集成和测试、协调产品构件的设计解决方案和选择。对于复杂的企业级产品，集成策略运用“集成-验证/测试-集成”的不断迭代模式；

（2）具体产品的集成策略，根据组织策略和方针，进一步完善产品的集成策略和环境、产品构件接口的兼容性、集成次序和方法以及集成验证标准和方法；

4.4 产品集成

（3）确定产品集成需要使用的资源；

（4）确定产品集成相关角色的责任、权限和人选；

（5）产品集成技能培训；

（6）确定产品集成的干系人及介入时机；

（7）建立和维护产品集成过程的详细描述；

（8）产品集成计划的审批规程。

4.4 产品集成

产品集成的规程即产品集成与测试的具体方法与步骤，包括手工集成的步骤，自动集成的脚本，集成测试的步骤与用例。产品集成的准则包括集成准备就绪的准则、集成测试的用例与通过准则等。

产品集成需要详细的规程、输入、输出、预期的结果和判定进展的准则。产品构件集成的详细规范应包括所要执行的迭代次数、预期的测试要求以及不同阶段的评估标准。

（1）构件的测试要求，包括允许替代的构件；

（2）接口验证，包括外部接口的需求和标准；

（3）每类构件的性能指标；

（4）集成测试环境参数和条件限制；

（5）集成操作的质量成本权衡；

（6）验证产品构件预期功能的方法和要求；

（7）产品交付标准；

（8）人力资源和硬软件环境资源的可用性。

4.4 产品集成

产品集成的一个重要方面是管理产品与产品组件的内部与外部接口，以确保接口间的兼容性。这些接口并不局限于用户界面，也适用于产品组件之间的接口，包括内部与外部的数据源、中间件以及其它组件——它们可能受到或不受到开发组织的控制，但是产品会依赖着它们，因此，每次集成后都要同步进行测试接口，包括外部接口，内部接口。

同时，项目的全生命周期内要进行接口的管理，有接口需求、接口设计，要评审接口需求、接口设计，发生变更时，要保持各描述的一致

4.4 产品集成

深度优先：关键业务流程（主控路径）上涉及到的模块先集成到一起，然后再集成辅助业务模块。

自上而下：首先集成一个稳定的顶层架构，然后不断地向下细化。

自下而上：已实现的较底层的功能先集成，然后逐层上升，形成整个系统。

持续集成：尽可能将产品集成的工作日常化、自动化，便于尽早发现产品集成时由于各种接口不匹配所带来的风险，使项目团队、干系人尽早对项目的进展有直观的了解。敏捷开发模型的“持续集成”，微软MSF开发模型倡导的“日构建”。

- 4.1 需求开发RD
- 4.2 需求管理RM
- 4.3 技术解决方案TS
- 4.4 产品集成PI
- 4.5 验证VER
- 4.6 确认VAL
- 4.7 软件测试
- 4.8 同行评审

4.5 验证

验证 verification是按照既定的标准，检查工作产品是否符合要求，目的在于确保选定的工作产品满足其规定的需求。

对于软件开发来讲，验证就是要用数据证明我们是不是在正确地开发产品，强调的是过程的正确性，如：单元测试验证产品每一模块是否符合详细设计规格说明书的要求、集成测试验证产品接口是否符合概要设计的要求。

验证方法：同行评审（需求评审，设计评审，代码评审等）或软件测试（单元测试，集成测试，系统测试）等。

4.5 验证

目的

验证过程是希望软件企业或组织在软件开发整个过程中，做好相应的检查工作，尽量把问题发现在前面，保证项目的可控性，降低开发成本。

“验证”本身是一个增量式的迭代过程，因为它贯穿产品与工作产品开发的整个过程，始于对需求的验证，终于对完整产品的验证。

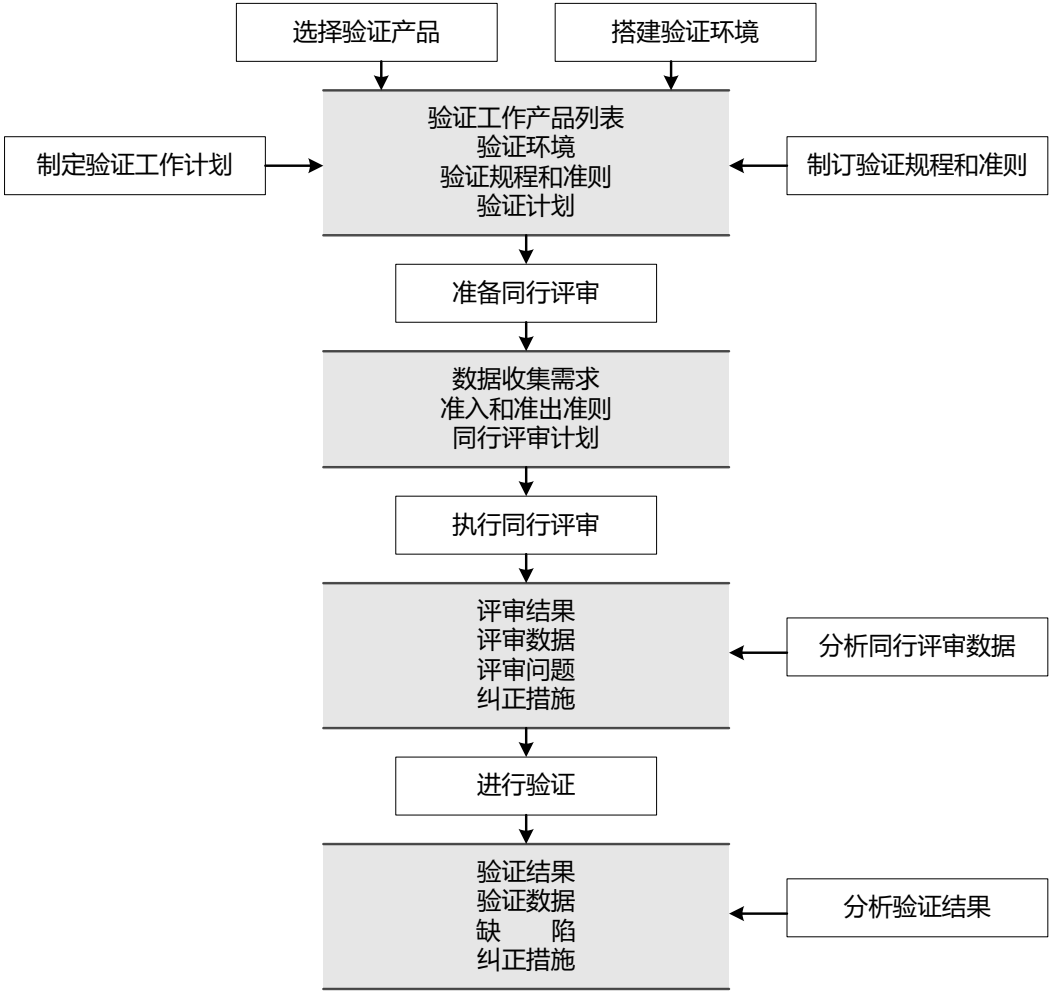
4.5 验证

关键过程域——验证VER

SG1	准备验证	
	SP1.1	选择验证产品
	SP1.2	建立验证环境
	SP1.3	建立验证规程与准则
SG2	进行同行审查	
	SP2.1	准备同行审查
	SP2.2	执行同行审查
	SP2.3	分析同行审查数据
SG3	验证所选择的工作产品	
	SP3.1	进行验证
	SP3.2	分析验证结果并确定纠正措施

4.5 验证

验证过程


选择验证产品
搭建验证环境
验证工作产品列表
验证环境
验证规程和准则
验证计划
制定验证工作计划
制订验证规程和准则
准备同行评审
数据收集需求
准入和准出准则
同行评审计划
执行同行评审
评审结果
评审数据
评审问题
纠正措施
分析同行评审数据
进行验证
验证结果
验证数据
缺　　陷
纠正措施
分析验证结果

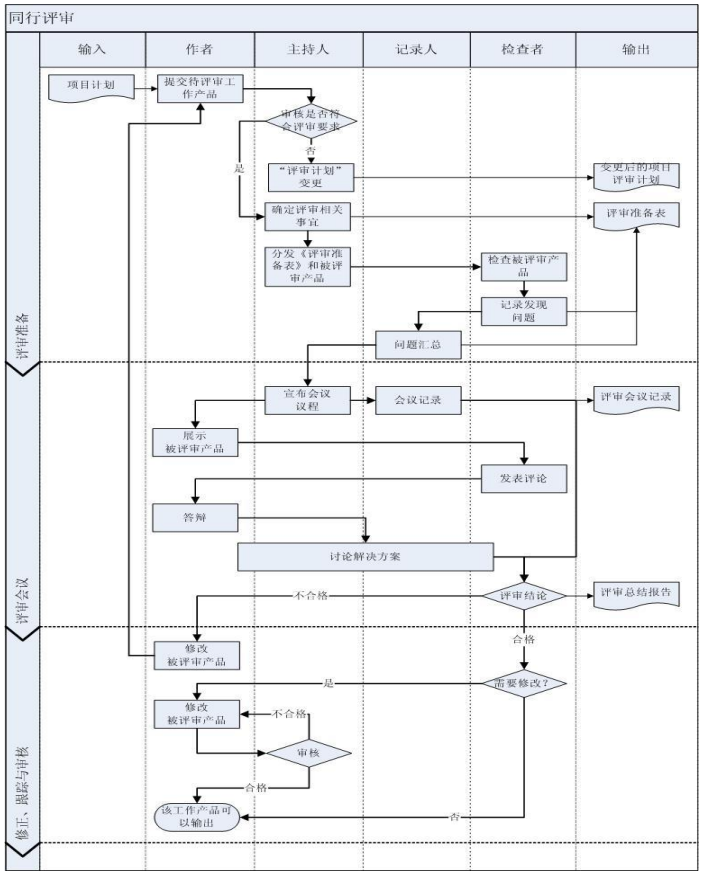
4.5 验证

验证准备内容	同行评审	测试
建立验证的工作产品及验证方法	工作产品：计划文档、需求文档、设计文档、代码等； 验证方法：规定每种文档的评审办法	工作产品：源程序 验证方法：单元测试、集成测试、系统测试等。
建立验证所需要的环境	支持环境：会议室、投影、电脑、事先准备好的文档等	支持环境：就是测试的软件环境、数据环境、硬件环境
建立验证过程及准则	过程：如事先发资料、通知大家到会、会议的组织、会议记录等： 准则：每个工作产品的评审标准	过程：测试过程的相关规定 准则：就是需求规格说明书（或测试通过的标准）

4.5 验证

同行评审（Peer Review）是验证工作的重要组成部分，是一种有效消除缺陷的机制。由产品开发者的同行（具有同等背景和能力）对工作产品（包括相关文档）进行系统性检查，以便发现缺陷和其他需要更改之处，提供必要的修改意见和纠正措施，应以报告形式提交给项目或产品管理者，并对纠正措施的实施进行跟踪直至问题解决（符合退出准则）。

4.5 验证



1、评审准备

（1）根据《项目计划》确定需要评审工作产品。

（2）确定：评审方式，评审人员，记录人，时间地点等

（3）通知相关评审人员

2、评审会议

（1）宣布会议议程：会议议程、重点、原则、时间限制等。

（2）展示工作产品：作者简要介绍工作产品。

（3）识别缺陷和答辩：

（4）讨论解决方案：作者和评审员共同讨论缺陷解决方案。

（5）评审结论：评审小组给出评审结论和意见：合格、不合格、

3、修改、跟踪与审核

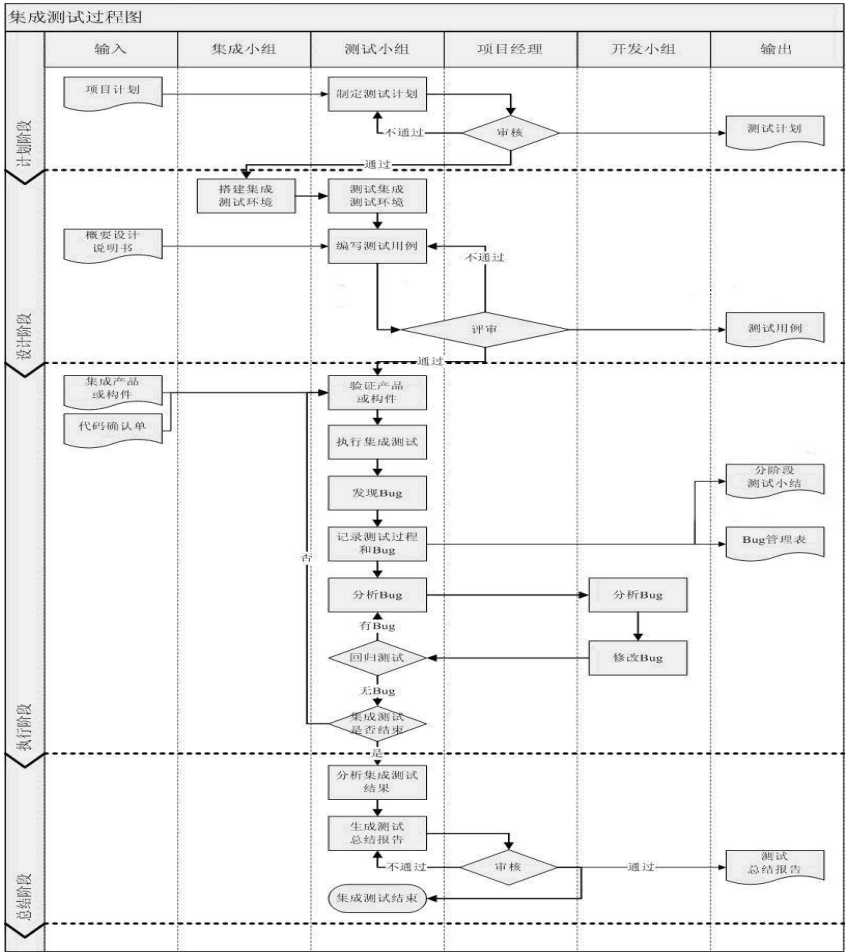
（1）修改与跟踪：作者对缺陷进行修正。QA跟踪缺陷修正情况。

（2）审核：主持人审核作者消除缺陷后的工作产品。

4.5 验证

软件测试是为了发现错误而执行程序的过程。或者说，软件测试是根据软件开发各阶段的规格说明和程序的内部结构而设计一批测试用例（即输入数据及其预期的输出结果），并利用这些测试用例去运行程序，以发现程序错误的过程。

4.5 验证



1、计划阶段

（1）测试小组根据《项目计划》制定测试计划

2、设计阶段

（1）搭建测试环境

（2）编写测试用例并评审

3、执行阶段

（1）执行测试用例并记录测试结果。

（2）测试小组将Bug反馈开发组修改

（3）回归测试确保Bug已修复

4、总结阶段

（1）测试小组分析Bug分布和原因、测试过程和结果，生成《测试总结报告》

- 4.1 需求开发RD
- 4.2 需求管理RM
- 4.3 技术解决方案TS
- 4.4 产品集成PI
- 4.5 验证VER
- 4.6 确认VAL
- 4.7 软件测试
- 4.8 同行评审

4.6 确认

确认是通过提供客观证据对特定的预期用途和应用要求已得到满足的认定。目的是展示完全置于预期环境中的产品或产品组件可以满足预期的使用需求。

通常确认活动包括了最终使用者及相关干系人，以及实际或模拟的使用环境中的调试、试验、试用、测试、评审、验收等方法。

4.6 确认

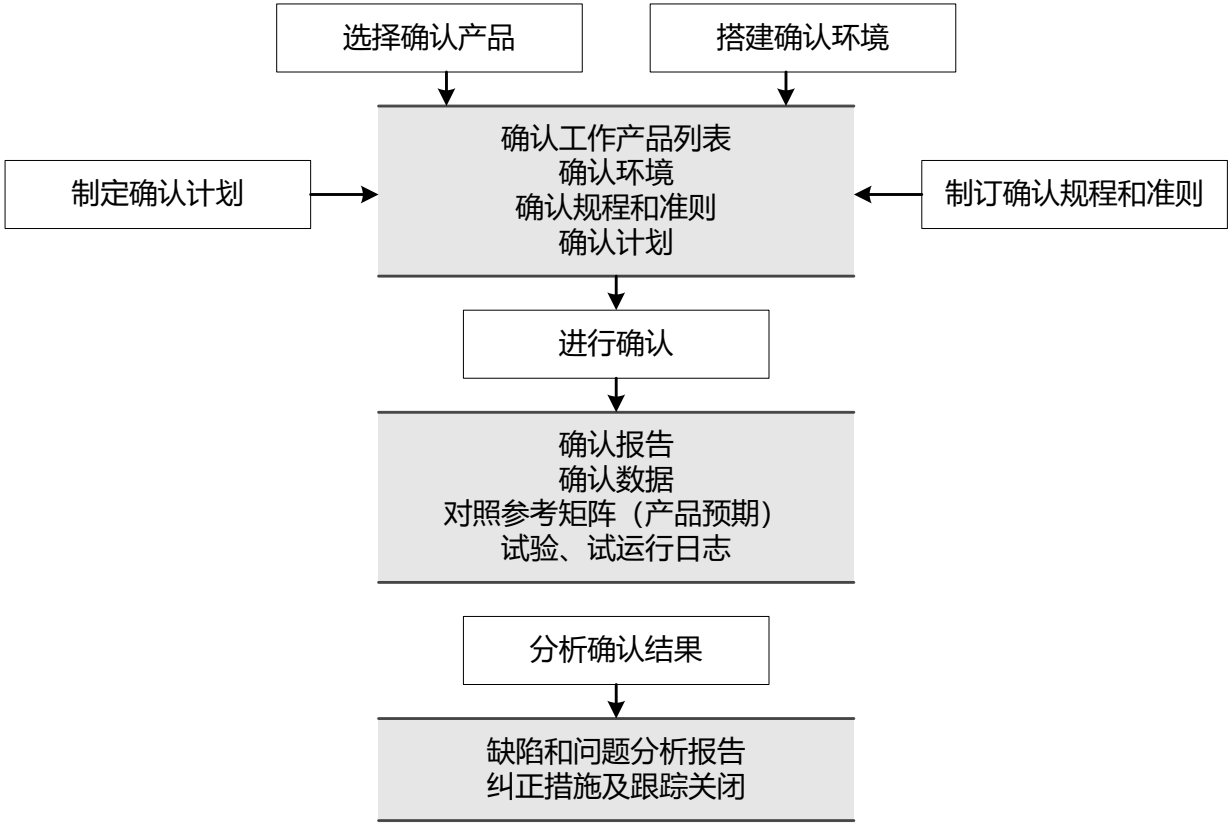
特定目标和特定实践

关键过程域——确认VAL

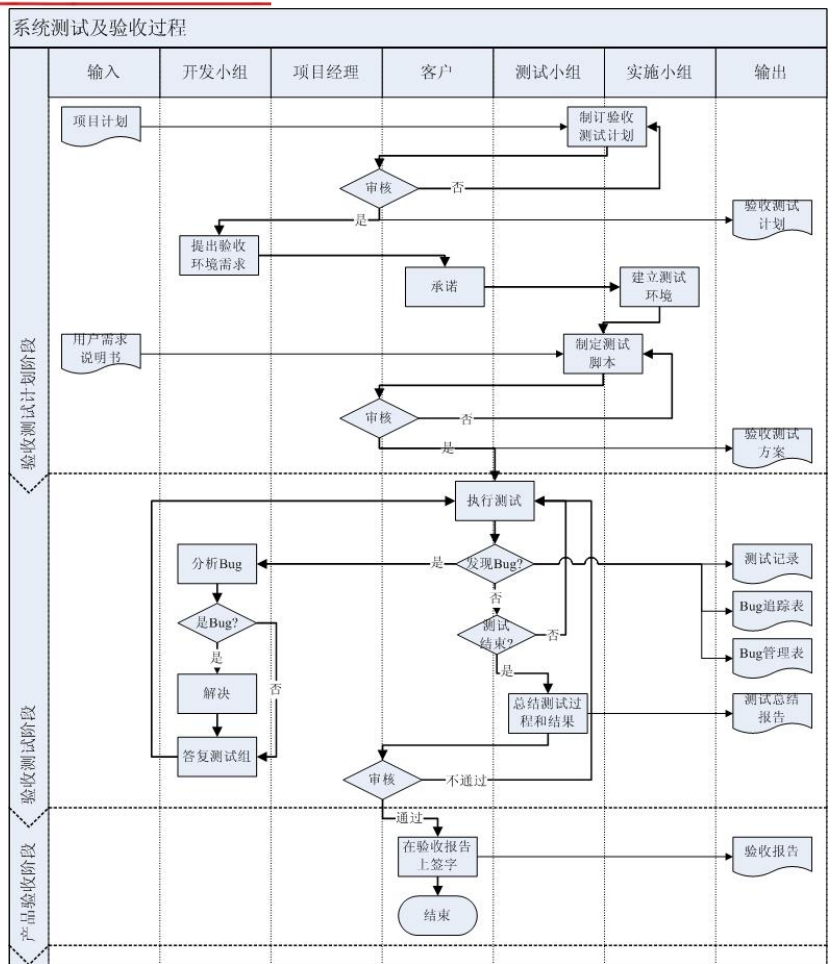
SG1	准备确认	
	SP1.1	选择确认产品
	SP1.2	建立确认环境
	SP1.3	建立确认规程与准则
SG2	确认产品或产品组件	
	SP2.1	进行确认
	SP2.2	汇集并分析确认结果

4.6 确认

确认过程


选择确认产品
搭建确认环境
确认工作产品列表
确认环境
确认规程和准则
确认计划
制定确认计划
制订确认规程和准则
进行确认
确认报告
确认数据
对照参考矩阵（产品预期）
试验、试运行日志
分析确认结果
缺陷和问题分析报告
纠正措施及跟踪关闭

4.6 确认



1、验收测试计划阶段

（1）制定验收测试计划

（2）验收环境的需求与承诺

（3）建立验收测试环境

（4）制定系统测试方案

2、验收测试阶段

（1）执行验收测试

（2）发现bug

（3）总结测试过程和结果

3、产品验收阶段：验收活动

4.6 确认

验证与确认的比较

论题	验证	确认
目标	保证开发过程种工作产品和产品设计规格的一致性，遵守生命周期过程种已定义的标准、实践和约定。	保证产品功能的有效性和可追溯到用户需求，即保证产品尽可能满足用户的实际的、真正的需求
范围	建立并维护验证策略和环境，进行同级评审和验证工作产品、最终产品和过程的有关活动。	最终产品，或可能包含对用于构造该产品的适当层次的产品构件。
知识领域	（1）应用领域； （2）验证原则、标准和方法，如分析、证明、检验和测试； （3）验证工具和设施； （4）同级评审准备和规程； （5）会议技巧。	（1）应用领域； （2）确认原则、标准和方法，如分析、证明、检验和测试； （3）预定的适用环境。
流程	制定验证计划、选择验证工作产品和建立验证环境； 建立验证过程和准则、准备同级评审； 进行同级评审、分析同级评审数据； 执行验证、分析验证结果和确定纠正措施等。	制定确认计划、选择确认产品和建立确认环境； 建立确认过程和准则； 执行确认、分析确认结果等。

4.6 确认

验证与确认的比较

论题	验证	确认
配置项	（1）验证策略； （2）同行评审规程； （3）同行评审资料； （4）验证报告。	（1）确认策略； （2）确认规程； （3）确认报告。
进入准则	产品设计规格说明书、不同产品（构件）特性的验证方法以及核查表模板等文档已完成。	需要确认的产品（构件）清单、每个产品（构件）的需求和确认方法以及对每个产品（构件）的确认约束等文档已经完成。
环境	（1）测试工具和验证的产品的接口； （2）暂时涉及的软件； （3）为将来分析和重放的记录工具； （4）仿真（模拟的）系统或部件； （5）仿真或真实接口系统； （6）专用计算机和网络测试环境。	基本同“验证”
主要活动	（1）制定和维护验证策略； （2）进行同行评审； （3）验证所选择的工作产品； （4）跟踪所发现的问题及不符合项。	（1）制定和维护确认策略； （2）对产品（构件）确认结果进行审查； （3）与顾客或最终用户一起解决问题； （4）偏离合同或约定的内容； （5）附件纵深研究或试验、复试和评价； （6）合同或协议的变更。

4.6 确认

验证与确认的比较

论题	验证	确认
活动准则	（1）产品（构件）设计； （2）验证标准； （3）组织方针； （4）测试类型； （5）测试参数； （6）测试质量与成本的平衡参数； （7）工作产品类型。	（1）产品（构件）需求； （2）确认标准； （3）客户可接受的准则； （4）环境性能； （5）性能指标。
度量项目	（1）计划和实施的所有验证活动和发现的缺陷数量、等级、分布情况； （2）已报告的缺陷的跟踪（进展、处理时间等）。	（1）计划和实施的所有确认活动和发现的缺陷数量、等级、分布情况； （2）已报告的缺陷的跟踪（进展、处理时间等）。
方法	检验、同行评审、审核、走查、分析、仿真、测试和证明等方法： （1）路径覆盖测试； （2）负载、压力和性能测试； （3）基于决策表的测试； （4）基于功能分解的测试； （5）测试用例复用； （6）可接受的其他测试。	检验、评审会议、审核、走查、测试和证明等方法： （1）用户场景、用例确认； （2）用户界面、操作性等适用性测试； （3）系统集成功能测试； （4）部署测试； （5）预定环境下性能测试； （6）预定环境下的验收测试； （7）预定环境下的第三方测试。

4.6 确认

验证与确认的比较

论题	验证	确认
工具	（1）测试管理工具； （2）测试用例生成程序； （3）测试工具； （4）测试覆盖分析工具； （5）仿真程序。	基本同“验证”
输出	（1）验证策略； （2）同行评审核查表； （3）验证报告； （4）分析报告，包括缺陷统计、性能统计、不符合项原因分析； （5）缺陷及跟踪报告； （6）方法、准则、基础设施和环境条件变更； （7）验证方法、准则和基础设施的纠正措施。	（1）确认策略和方法； （2）确认规范和准则； （3）确认结果报告； （4）确认缺陷、问题、不符合项的报告； （5）确认情况对照表； （6）确认规范执行日志； （7）产品运行演示材料； （8）相关规程变更。

4.6 确认

验证与确认的区别

“验证”与“确认”都是确定软件产品是否满足其预期要求和条件的检查。验证可适用于分析、设计、编码、测试等众多的过程，而确认通常用于最后的验收过程。通常我们选择验证和确认的产品包括需求、设计、代码、测试等文档。

验证是确保“正确地做了事”，而确认是确保“做了正确的事”。验证确认有助于确保采用系统化的方法来选择将要测试的工作产品、将要搭建环境、将要管理的接口，有助于确保尽早识别并处理各类缺陷。确保需求与解决方案之间的相容性，以确保项目最终满足最终用户的使用需求。

- 4.1 需求开发RD
- 4.2 需求管理RM
- 4.3 技术解决方案TS
- 4.4 产品集成PI
- 4.5 验证VER
- 4.6 确认VAL
- 4.7 软件测试
- 4.8 同行评审

4.7 软件测试

——什么是软件测试

软件测试是根据开发各个阶段的规格和程序的内部结构而精心设计一批测试用例（即输入数据及其预期的输出结果），并利用这些测试用例人工或自动去运行程序，以发现程序错误和缺陷的过程。

4.7 软件测试

——软件测试的目的

目的在于检验它是否满足规定的需求或弄清预期结果与实际结果之间的差别。

测试在于发现尽可能多的、迄今为止尚未发现的程序错误和缺陷；

一个好的测试用例在于能发现至今未发现的错误和缺陷；

一个成功的测试是发现了至今未发现的错误和缺陷。

1、尽早地和不断地进行软件测试。需求规格确定后，就可以开始测试方案、测试计划；设计规格确定后，就可以开始测试用例编写。

2、测试用例应由输入数据和与之对应的预期输出结果组成；

3、测试用例应包括有效及合理的输入条件、无效及不合理的输入条件；

4、程序员（开发小组）应尽可能避免测试自己编写的程序；

5、充分注意测试中错误和缺陷的群聚现象；

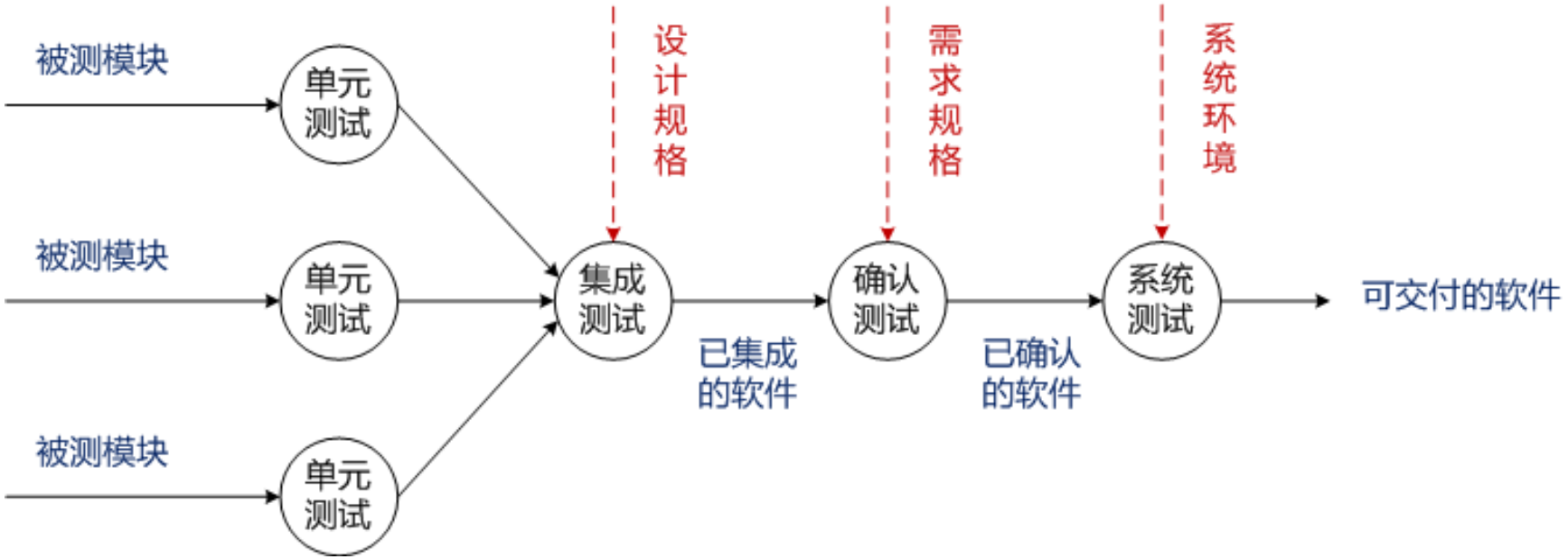
6、严格执行测试计划，排除测试的随意性；

7、应当对每一个测试结果做全面检查；

8、妥善保存测试计划、测试用例、错误和缺陷记录、分析报告，为回归测试和运行维护提供方便 。

4.7 软件测试

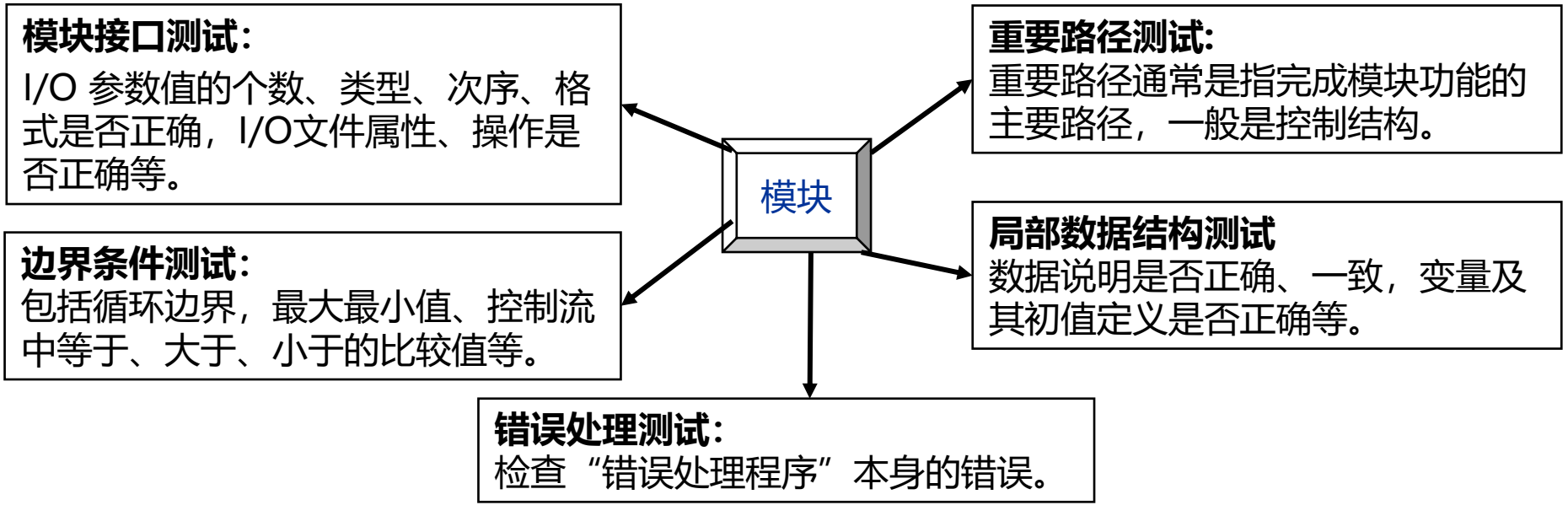
——软件测试的阶段



4.7 软件测试

——软件测试的阶段

单元测试包括模块接口测试、局部数据结构测试、路径测试、错误处理测试、边界测试、性能测试等内容。



集成测试在单元测试基础上，将所有模块按照设计的软件结构组装成一个完整的系统而进行的测试，也称为联合测试、组装测试。重点测试模块的接口部分。

集成测试的方式：一次性集成方式、自底向上的渐增集成方式、自顶向下的渐增值集成方式、混合增值集成方式、回归测试。

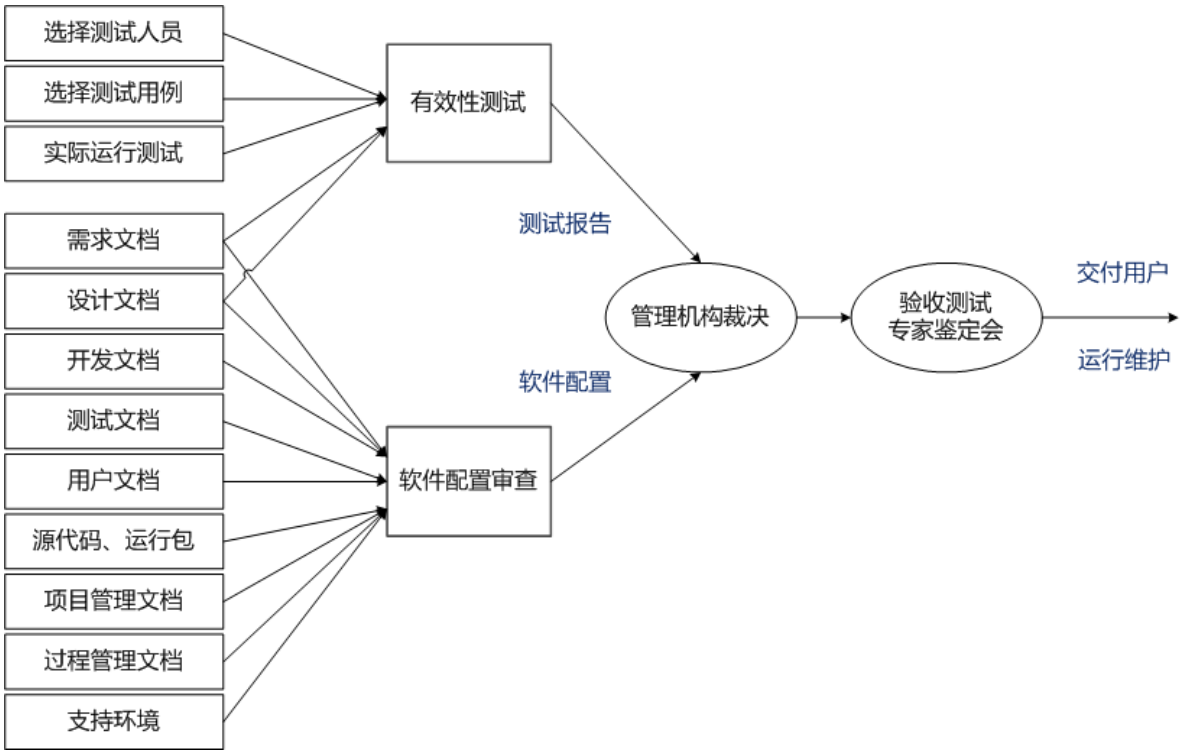
集成测试的任务：

- 在把各个模块连接起来时，穿越模块接口的数据是否出错；
- 一个模块的功能是否会对另一个模块的功能产生不利的影响；
- 各个子功能组合起来，能否达到预期要求的父功能；
- 全局数据结构是否有问题；
- 单个模块的累计误差是否会放大，从而达到不能接受的程度；
- 单个模块的错误是否会导致其他模块、数据结构的错误；
- 数据格式变换、代码转换是否正确。

4.7 软件测试

——软件测试的阶段

确认测试是验证系统的功能、性能等特性是否符合需求规格说明。



系统测试是将通过确认测试的软件，作为整个系统的一个元素，与计算机硬件、设备、外设、支持条件、数据和人员等其它系统元素结合在一起，在实际运行、使用环境下，对系统进行的测试。

包括：功能测试、性能测试、安全测试、恢复测试、压力测试、文档测试

4.7 软件测试

——软件测试的方法

静态测试	人工测试	桌前检查：程序员检查代码、静态分析。 交叉阅读代码：同行检查代码、静态分析。 代码走查：扮演计算机角色，沿程序逻辑一步一步的“走”，记录踪迹并分析。 代码会审：会审小组通过集体阅读、讨论，开展程序静态分析。 一致性检查：软件配置成分、文档和软件产品的一致性。
	源程序静态分析	生成引用关系：如程序结构、组件调用、接口调用、变量调用、方法调用。 静态错误分析：如数据类型分析、表达式分析、接口分析、调用分析。 代码规范性检查：如程序风格、开发规范、SQL性能、连接申请释放、企业财富使用。
动态测试	白盒测试	基于程序内部结构设计测试用例，如语句覆盖、判定覆盖、条件、路径覆盖。
	黑盒测试	基于程序功能、性能、接口等设计测试用例。
其他测试	压力测试、安全测试、性能测试、α测试、β测试、回归测试	

4.7 软件测试

静态测试是不执行程序代码而寻找程序中的错误或评估程序的代码的过程。可以手工进行、或借助软件静态分析工具自动进行。充分发挥人的逻辑思维优势。

（一）人工测试

1、桌前检查。程序员检查代码、静态分析；

2、交叉阅读代码。同行检查代码、静态分析；

3、代码走查。走查小组开会，集体扮演计算机角色，让测试用例沿程序逻辑一步一步的“走”，记录“走”的踪迹并分析；

4、代码会审。会审小组通过阅读、讨论和争议，开展程序静态分析。

（二）源程序静态分析

1、生成各种引用表。如程序结构、组件调用关系、接口调用关系、变量调用、方法调用；

2、静态错误分析。如数据类型和单位分析、引用分析、表达式分析、接口分析；

3、代码规范性检查。注释、嵌套缩进、程序风格、代码编写规范、企业财富库使用。

4.7 软件测试

——软件测试的方法

动态测试是在计算机、设备、相应软件系统的配置环境下，运行软件系统，进行的测试。从是否需要了解程序的内部结构和实现细节，可划分为白盒测试、黑盒测试。

白盒测试主要检查程序逻辑结构的合理性、实现的正确性；

黑盒测试是把程序组件看作一个黑盒子，应在所有可能的输入条件和输出条件中确定测试数据，来检查程序是否能产生正确的输出。

黑盒测试主要检查程序功能是否符合用户需求和系统设计。

4.7 软件测试

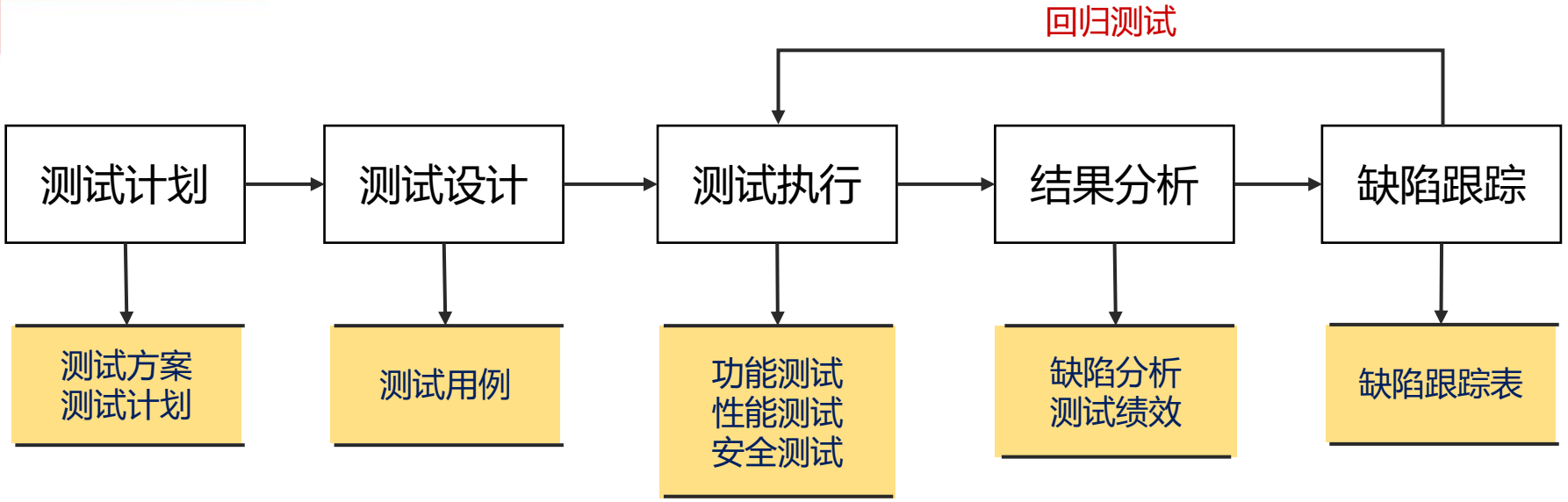
——软件测试的方法

白盒测试与黑盒测试两类方法的对比

		白盒测试	黑盒测试
测试规划		根据程序的内部结构，如语句的控制结构，模块间的控制结构以及内部数据结构等进行测试。	根据用户的规格说明，即针对命令、信息、报表等用户界面及体现它们的输入数据与输出数据之间的对应关系，特别是针对功能进行测试
特点	优点	能够对程序内部的特定部位进行覆盖测试	能站在用户的立场上进行测试
	缺点	无法检验程序的外部特性 无法对未实现规格说明的程序内部欠缺部分进行测试	不能测试程序内部特定部位 如果规格说明有误，则无法发现
方法举例		语句覆盖 判定覆盖 条件覆盖 判定－条件覆盖 基本路径覆盖 循环覆盖 模块接口测试	基于图的测试 等价类划分 边值分析 比较测试

4.7 软件测试

——软件测试的流程



测试工程师根据测试计划、需求文档、概要设计、产品经理提的进度、性能、安全等方面的需求编写测试用例。

测试工程师根据安装配置手册及基础数据搭建测试环境，进行功能测试、性能测试、安全测试等，编写测试总结报告，提交测试文档。

——软件测试的现实困惑

- 测试人员对领域知识、行业业务、软件需求的不重视
- 急需解决软件的功能可用性、系统稳定性、效率等问题
- 对接接口、业务协同、后台任务的正确性
- 数据采集、传输、交换、打包拆包的格式转换和代码变换的全面性

- 4.1 需求开发RD
- 4.2 需求管理RM
- 4.3 技术解决方案TS
- 4.4 产品集成PI
- 4.5 验证VER
- 4.6 确认VAL
- 4.7 软件测试
- 4.8 同行评审

4.8 同行评审

——什么是同行评审

同行评审是对项目文档、项目管理文档、代码、可运行程序包、可交付成果等产出物进行同行评审活动，其目的在于验证产出物满足设计规格、需求规格、合同要求和用户期望，是软件产品验证和确认的重要手段。

实践证明，同行评审贯穿了整个软件生命周期，能够及早发现并纠正工作产品中的问题，降低成本和返工，使软件产品达到用户要求、质量保证的一个重要手段，不是软件测试可以替代的。

4.8 同行评审

缺陷是指程序代码、文档等软件配置成分中存在各种不希望出现的问题，是隐藏在软件内部的“隐患”，是导致软件故障的原因，有人称为“看不见的定时炸弹”。

软件质量管理的实质是研究缺陷的产生、传播和消除规律，如何解决好软件缺陷。

三种典型缺陷：

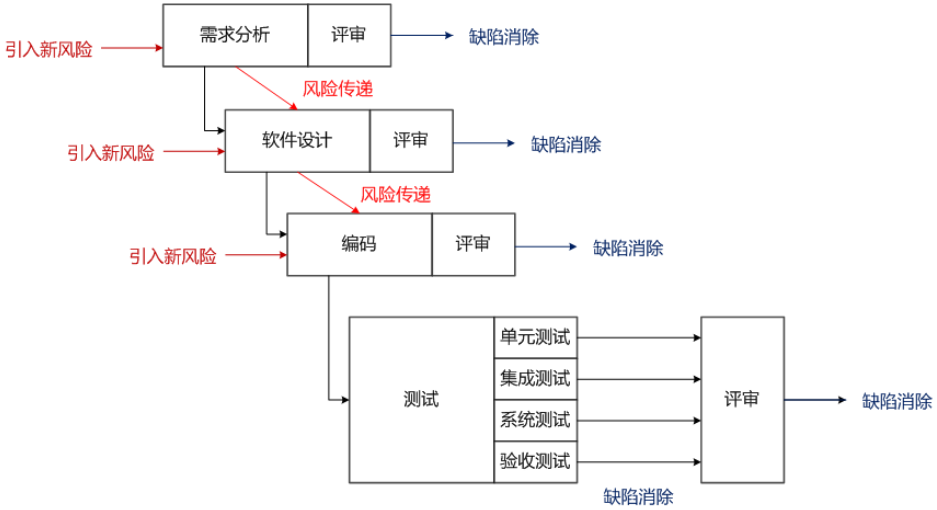
（1）错误。未能正确地实现需求和设计规格说明，发生了偏离，可能在设计阶段、开发阶段；

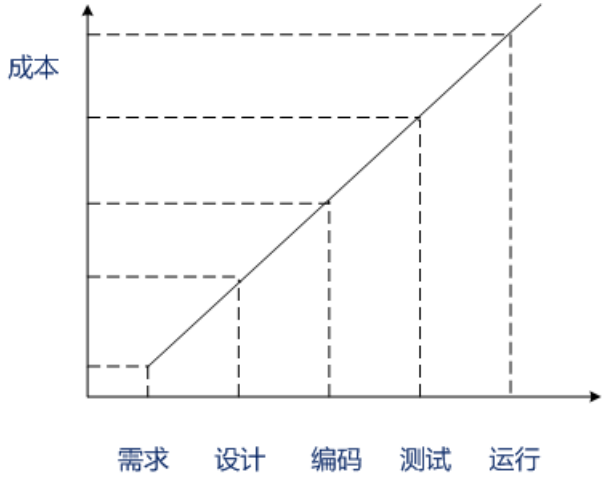
（2）遗漏。未能将规定的或预期的需求体现在软件产品中；

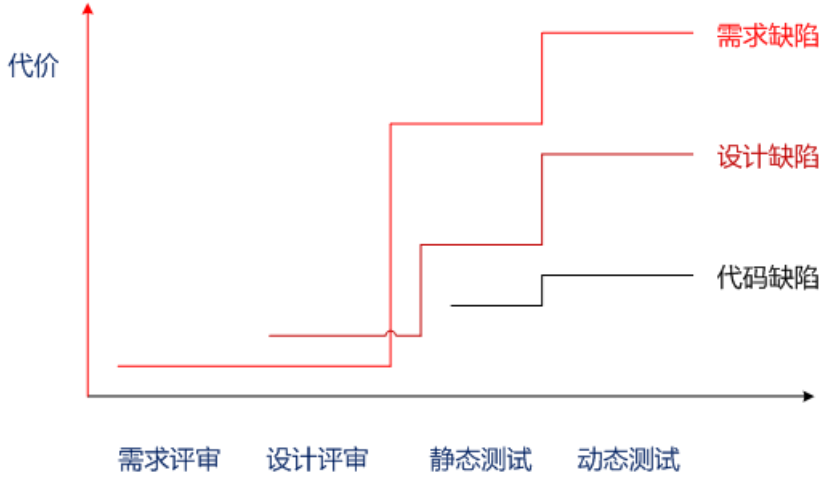
（3）额外的实现。超出需求和设计规格说明的多余实现。

4.8 同行评审

——同行评审与缺陷消除







缺陷消除的成本代价的发散特征

4.8 同行评审

Olsen等人研究总结的缺陷发现对比

方法	发现缺陷比例
系统设计审查	17.3%
部件设计审查	19.1%
代码审查	15.1%
集成测试	29.4%
系统测试	16.6%
运行后发现	0.1%

C.Jones研究总结的缺陷发现百分比

采用的验证方法	需求	设计	编码	文档
建立原型	40	35	35	15
需求评审	40	0	0	5
设计评审	15	55	0	15
代码审查	20	40	65	25
单元测试	1	5	20	0

4.8 同行评审

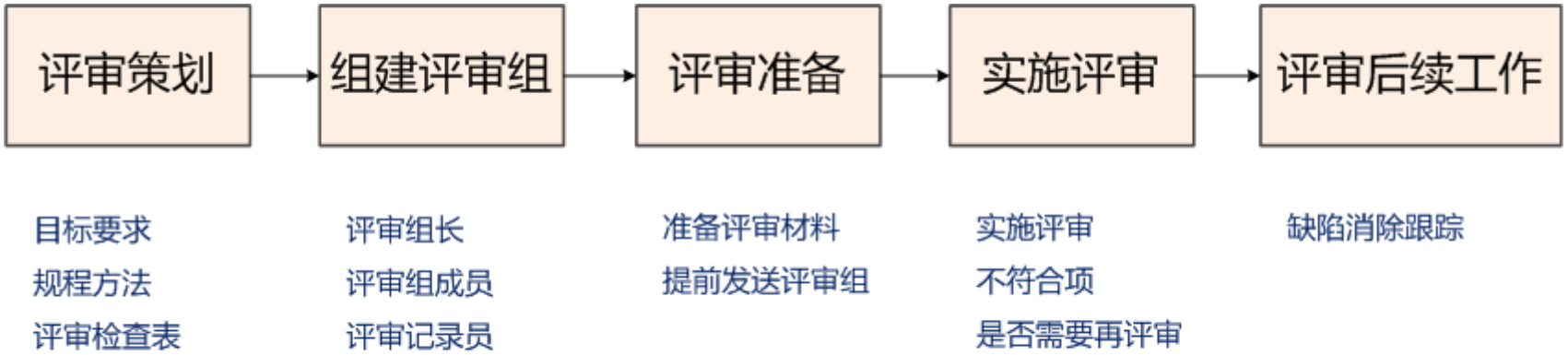
——同行评审与缺陷消除

IBM的机构Federal Systems Division总结的软件评审

1	采用软件评审方法及时消除缺陷的工作量小、难度低
2	及早发现缺陷的代价要低得多
3	实施评审的项目生产率比未实施评审的项目生产率要高出一倍以上
4	50%-85%的缺陷应在测试之前发现，减轻测试压力，降低测试成本
5	在测试之前对测试用例进行评审，可节省85%的测试工作量，提高测试有效性

4.8 同行评审

——同行评审的流程



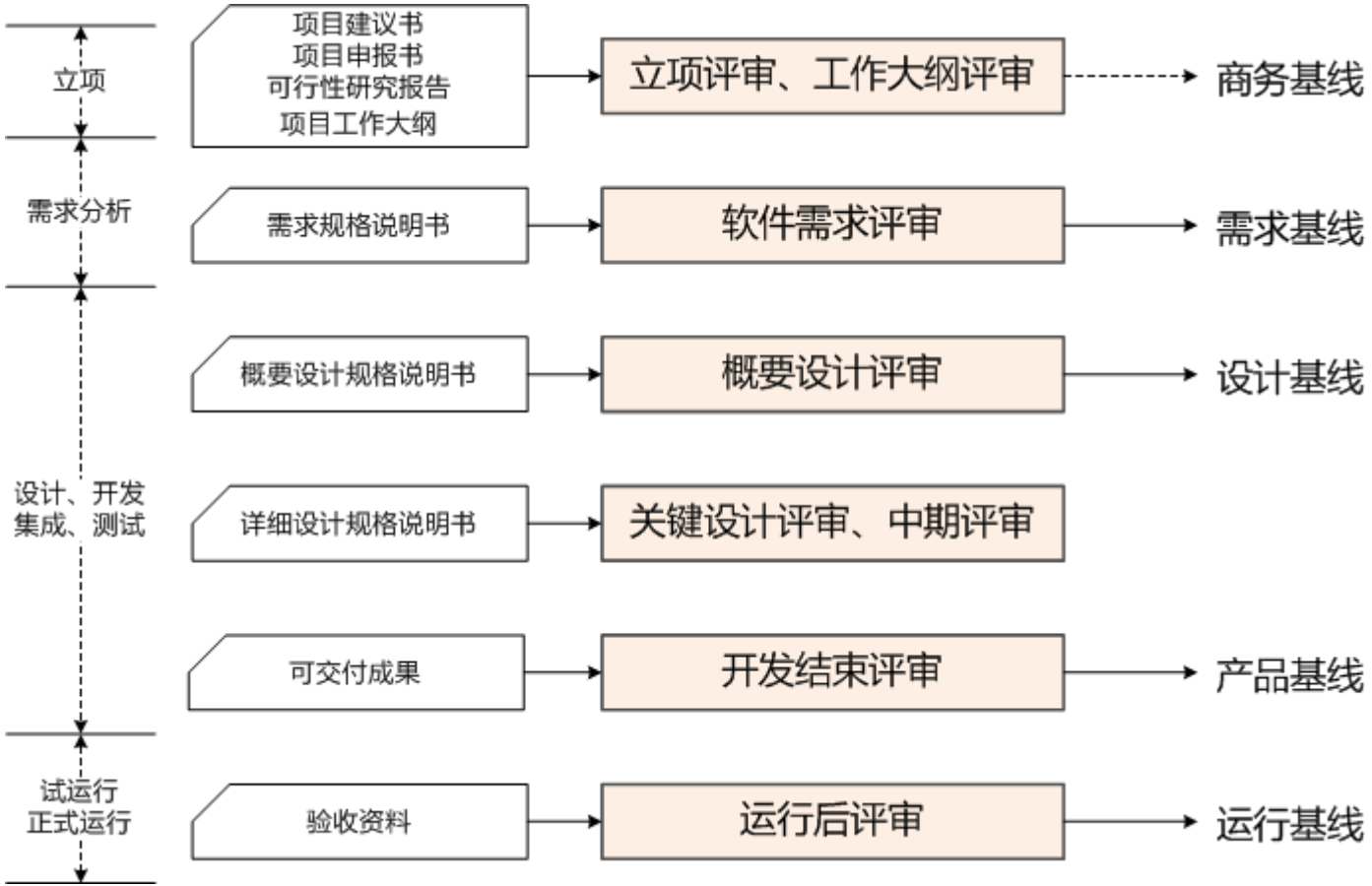
4.8 同行评审

——同行评审的方法

常规的同行评审包括：正式评审、技术审查和走查。“正式评审”是正式的，后两者是常用的非正式的同行评审方法。正式评审、技术审查和走查三种形式的同行评审的重要程度不同，目的、评审时间、准备、主持人、参与评审人员等不完全相同，应当严格遵循其流程、步骤和注意事项进行同行评审，以保证同行评审的有效性。

4.8 同行评审

——基线评审


立项
项目建议书
项目申报书
可行性研究报告
项目工作大纲
立项评审、工作大纲评审
商务基线
需求分析
需求规格说明书
软件需求评审
需求基线
概要设计规格说明书
概要设计评审
设计基线
设计、开发
集成、测试
详细设计规格说明书
关键设计评审、中期评审
可交付成果
开发结束评审
产品基线
试运行
正式运行
验收资料
运行后评审
运行基线

4.8 同行评审

——基线评审

基线评审	目的	内容
概念评审	验证计划开发的软件产品是满足用户需求的最佳选择； 验证项目的经济可行性和技术可行性； 保证用户和项目组对系统需求和系统目标有共同的理解； 保证初始系统需求对达到初始系统设想是充分的； 确保软件开发人员对产品关键特性的理解。	系统开发要求； 系统需求文件； 可行性研究报告、项目建议书、项目工作大纲； 项目开发初步计划； 设备规格说明。
需求评审	将软件开发组织对开发软件产品的承诺明确下来； 验证系统需求是否准确地反映了初始系统设想； 验证所有的文档已更新，包含了功能、数据、接口、性能、技术、工期、运行维护等完整需求； 保证所有新的配置项符合要求，并纳入产品开发控制。	用户需求报告； 需求规格说明； 项目计划（设计、开发、测试计划）； 数据需求文件； 接口控制文件。
概设评审	验证概要设计规格说明与功能基线、分配基线所建立的需求是一致的；验证概要设计规格说明对于详细设计和编码是足够明确和详尽的；验证技术体系、实现路径、关键技术是正确的；确保项目开发测试的正确策划能够得到采纳。	概要设计规格说明书 数据库设计说明书； 接口控制文件； 项目计划（详细设计计划、开发测试计划）。

4.8 同行评审

——基线评审

基线评审	目的	内容
关键设计评审	验证详细设计规格说明书与概要设计规格说明书是一致的； 保证主要的设计模块准确反映了需求、技术实现是正确的； 保证关键处理逻辑和算法是正确的、满足性能要求的； 确保项目开发测试的正确策划能够得到采纳。	详细设计规格说明书； 数据库设计说明书； 接口控制文件； 项目计划（开发测试计划）。
开发结束评审	确保所有需求规格均已在产品中实现； 确保所有的系统模块能正确地集成； 确保按计划完成测试，所有测试文档已齐备，测试中发现的所有错误和缺陷均以解决； 确保项目文档完整、规范、一致。	需求、设计、开发、集成、测试文档； 用户操作手册、管理维护手册； 项目计划（试运行、正式运行计划）； 项目研制报告，项目总结报告。
运行后评审	验证系统支持文档的适用性和完整性； 确保产品的可维护性。	用户操作手册、管理维护手册； 运行维护知识库； 版本描述文件； 运行台账、运行总结报告 项目总结报告。

4.8 同行评审

——需求评审

一个需求评审实例

评审目的	确定开发方提出的“软件开发计划”和“软件需求规格说明”是否符合用户方给出的“软件开发任务书”的要求； 确定需求分析与定义阶段开发的文档是否可作为概要设计的依据，是否能转入概要设计阶段	
评审对象	软件需求分析与定义阶段相关工作及其产出物	
评审的初始条件	已通过组织内部非正式评审； 已完成并于需求评审前规定时间提交下列文档：软件开发计划、软件需求规格说明书、软件质量保证计划、软件配置项测试计划	
评审内容及要求	对文档配置的审查	文档配置是否齐全； 文档编制是否符合规定的规范要求； 对有数据库应用的开发需提交“数据库需求规格说明”
	评审“软件需求规格说明”	（1）需求分析方法、工具使用是否合适 （2）软件需求内容是否符合系统要求 功能需求：包括每一项功能的输入、处理、输出 性能需求：包括容量要求、精度要求、时间要求 接口需求：软件配置项与内部外部的接口关系 数据需求：包括静态实体数据、动态输入输出数据和内部生成数据；数据的采集、逻辑结构、特性和取值范围 环境需求：开发环境、运行环境 （3）软件质量要求，每一项需求应满足正确性、完整性、一致性、无多义、可验证性和可维护性 （4）软件的安全性要求、可靠性要求 （5）软件的可维护性要求
	评审”软件开发计划“	阶段划分明确，计划安排明确、合理、可行，软件配置管理要求明确，项目组组成合理
	评审“软件质量保证计划”	软件质量保证计划完整、明确、合理、可行
	评审“配置项测试计划”	测试的范围和内容明确，测试结果评价准则明确、可行，测试组织、人员、进度安排合理
	评审数据库需求规格说明“	信息分类合理，信息处理要求完备，数据流图、ER图等规范准确
评审结论和处理要求	形成评审结论，对评审结论的处理要求，形成评审配置文档	

4.8 同行评审

——代码评审

代码评审是在编码以后，针对代码和相关文档的审查活动，是在测试之前对程序的实现细节的审查。

代码评审活动通常不需要客户或用户参加。事实上，他们也不会关心程序实现的具体细节。

（1）代码走查

是针对项目组提供的代码和相关文档的非正式评审活动。其做法是由评审员或代码编写人员针对被审查的代码，沿着控制流从头到尾遍历、发现软件缺陷的过程。

（2）代码检查

采用正式评审方式，评审组对照事先准备好的一份清单，检查代码或文档，形成检查报告单。

主要检查内容包括：数据类型和数据结构的定义和使用，程序中的算法和计算过程，考察其正确性和效率，程序中的注释是否准确、充分，部件间的接口正确性，代码的运行特性、内存使用、连接释放、处理效能等。

4.8 同行评审

——做好同行评审的建议

（1）管理者的足够重视，不要使得评审变成形式主义，这样反而得不偿失。

（2）评审开始之前评审者和参与者都要提前做好准备，例如提前发放材料，提前批注，提前准备问题，提前计划怎么阐述等，做到心中有数。

（3）要控制会议的时间，有的评审会一搞就是半天，搞得参与人员精神疲惫，评审会议尽量控制在1-2小时之间为佳。

（4）选择的参与人员要合适，是否有能力给出意见，不要太官僚，甚至因为某些领导的意见不敢发音，评审会议不能变成一言堂。

4.8 同行评审

——做好同行评审的建议

（5）评审形式也需要根据评审内容进行合适的选择，什么时候正式评审，什么时候走查等等，所有的缺陷最终都应追溯到需求，务必从根源上进行解决。

（6）软件开发人员应避免检查自己的程序，尽可能采用同行交叉评审，程序中的大部分错误往往是在一小部分模块中发现的，遵循普遍适用的"二八定理"。

（7）做好评审活动的度量工作，可以通过度量数据和结果对评审的相关流程进行优化，一般关注以下方面：每类评审会的工作量，每人投入的工作量，评审会议发现的问题，每个人发现的问题数，个人和整体评审的效率，以及评审文档或代码的规模等等。

课后作业2

1、什么是质量保证?

质量保证是为质量管理和质量控制活动、产品或项目的过程、产出物、交付成果是否符合规定标准、管理计划、质量目标和质量要求，提供恰当和足够的监督，并收集分析相关质量数据、确立产品或项目质量的置信度、定期发布质量报告的过程，也就是类似监理的角色和职责。

课后作业2

2、什么是基线，常见的软件配置基线有哪些?

基线（baseline）是一个软件配置管理的概念，通常反映一个阶段性工作结束后的、正式发布的产出物的集合。IEEE 对基线的定义：已经通过正式复审审核批准的某规约或产品，它因此可以作为进一步开发的基础，并且只能通过正式的变化/变更控制过程而改变。

常见的软件配置基线有功能基线，分配基线，产品基线和设计基线等。

3、度量的目标是什么?

课后作业2

3、度量的目标是什么?

度量的目标在于开发和维持度量能力，提供量化的数据，支持软件过程管理、项目管理的需要

课后作业3

1、如何做好软件需求开发和需求管理?

2、验证和确认二个关键过程域的区别?

邮箱：jgzhen4406@sina.com

文件命名：课后作业3（姓名，学号）