東華大學
DONGHUA UNIVERSITY

第12章 统一过程RUP

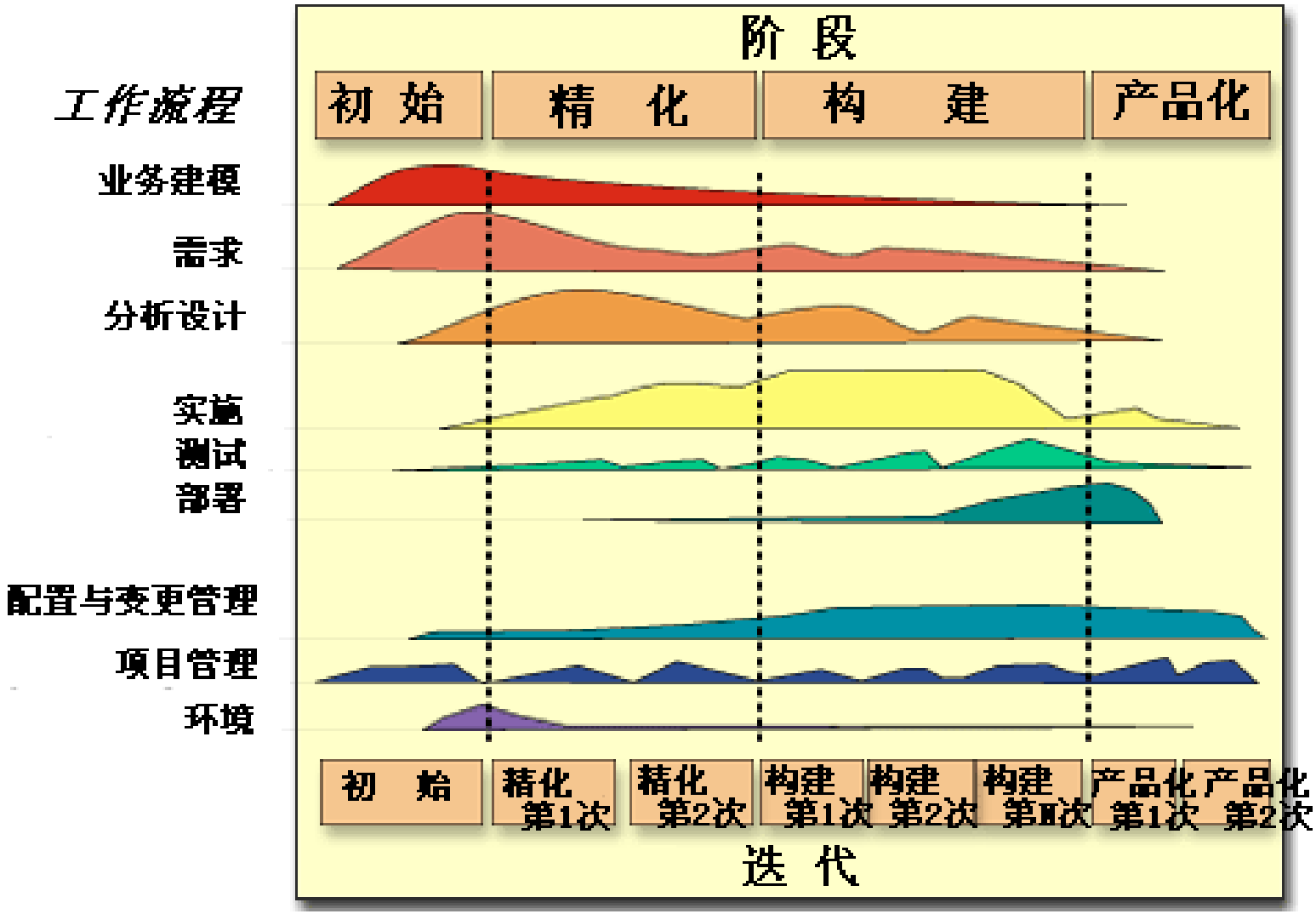
东华大学计算机学院

石秀金

12.1 简介

- **Rational Unified Process（简称RUP）是一套软件工程过程，主要由Ivar Jacobson的 The Objectory Approch 和 The Rational Approch 发展而来。**
- **是文档化的软件工程产品，所有RUP 的实施细节及方法导引均以Web文档的方式集成在一张光盘上。**
- **RUP又是一套软件工程方法的框架，各个组织可根据自身的实际情况，以及项目规模对RUP进行裁剪和修改，以制定出合乎需要的软件工程过程。**

12.1.1 生命周期

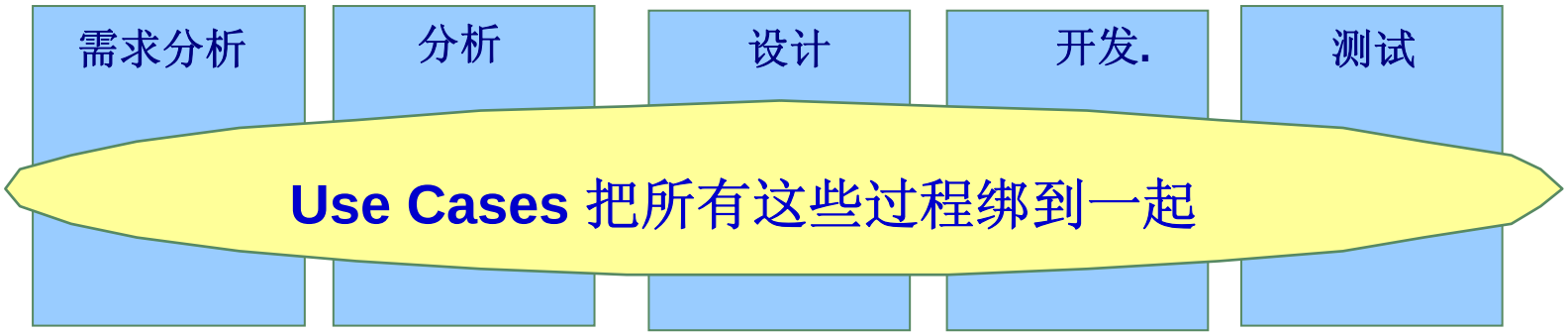

阶 段
工作流程
初 始
精 化
构 建
产品化
业务建模
需求
分析设计
实施
测试
部署
配置与变更管理
项目管理
环境
初 始
精化 第1次
精化 第2次
构建 第1次
构建 第2次
构建 第n次
产品化 第1次
产品化 第2次
迭 代

12.1.2 最佳开发经验

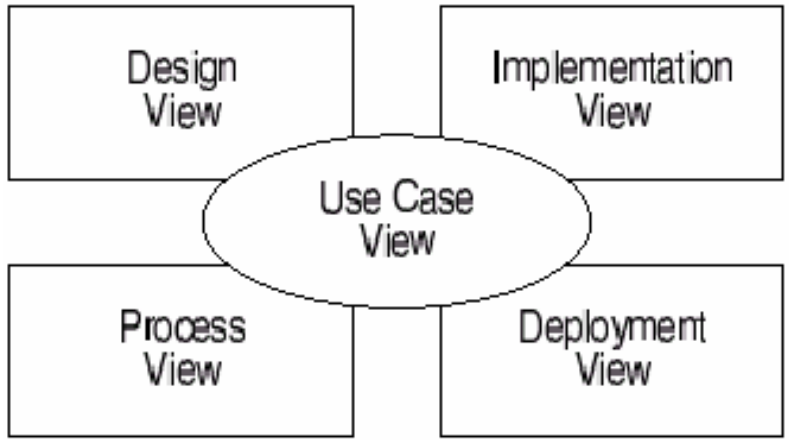
- **RUP是最佳软件开发经验的总结：**
  - 迭代式开发（**develop software iteratively**）
  - 管理需求（**manage requirements**）
  - 使用基于构件的体系结构（**use component-based architectures**）
  - 可视化软件建模（**visually model software**）
  - 验证软件质量（**verify model quality**）
  - 控制软件变更（**control changes to software**）

12.1.3 RUP的特点

- 用例驱动

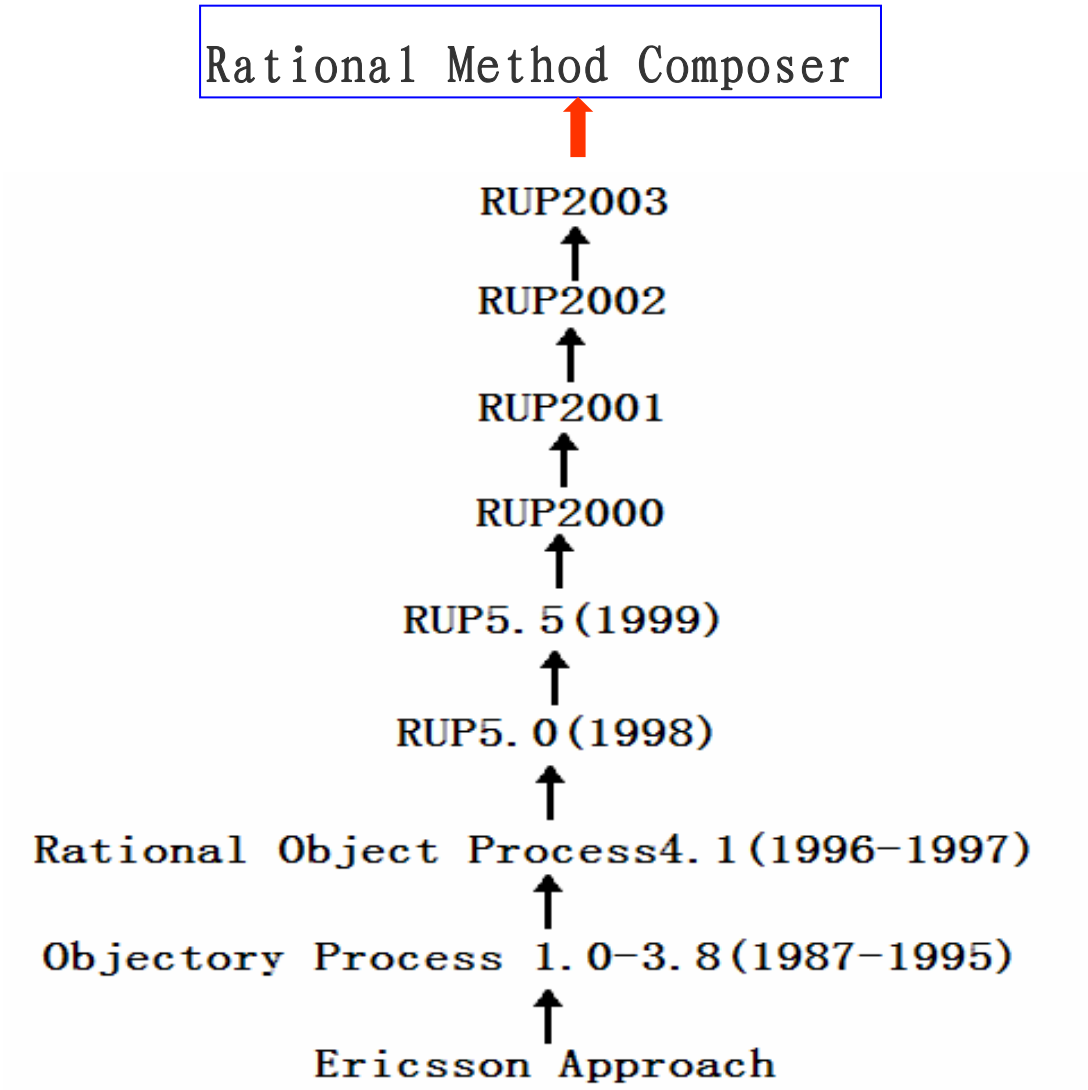


- 以体系结构为中心



- 迭代和增量

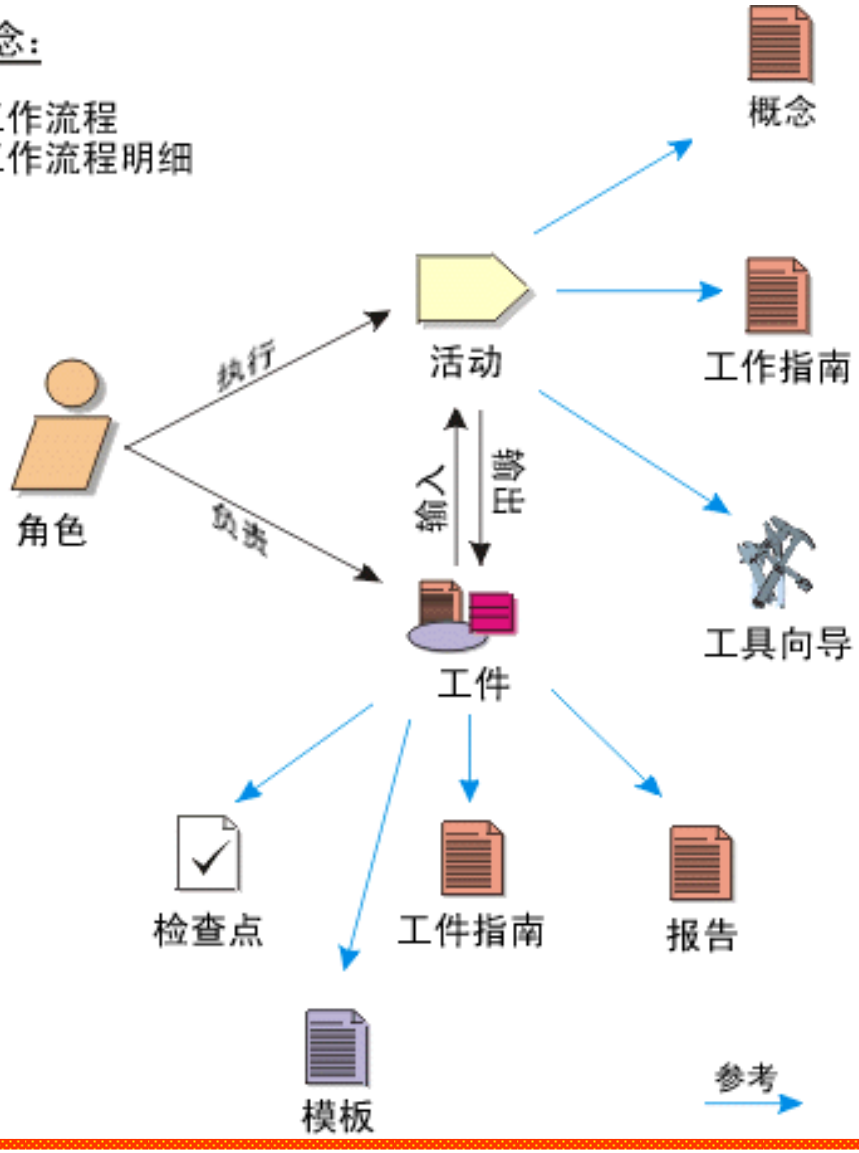
12.1.4 历史


Rational Method Composer
RUP2003
RUP2002
RUP2001
RUP2000
RUP5.5(1999)
RUP5.0(1998)
Rational Object Process4.1(1996-1997)
Objectory Process 1.0-3.8(1987-1995)
Ericsson Approach

12.2 主要概念

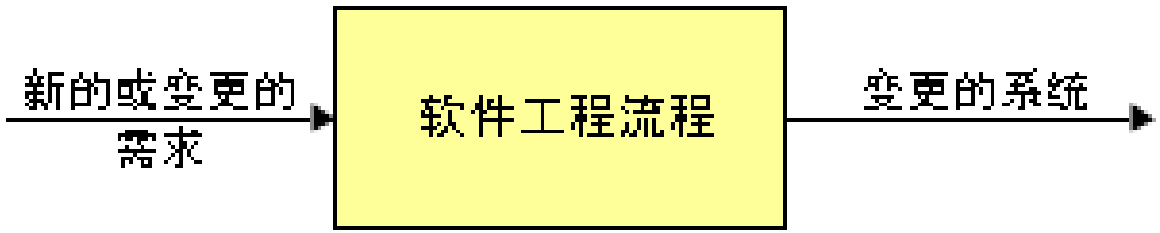
其他概念：

工作流程
工作流程明细



12.2.1 软件工程流程

- 流程
  - 是为实现某个目标而设定的一系列次序相对固定的步骤；
- **Rational Unified Process**
  - 是一个面向对象软件工程的通用业务流程。它描述了一系列相关的软件工程流程。**RUP** 为在开发组织中分配任务和职责提供了一种规范方法。其目标是确保在可预计的时间安排和预算内开发出满足最终用户需求的高品质的软件。
- 软件工程流程是按需求开发系统的过程，可以是新需求（最初开发阶段），也可以是变更需求（演进阶段）。



12.2.2 角色

- **Role**
  - 角色定义了在软件工程组织的环境中，个人或协同工作的多人小组的行为和职责。角色代表项目中个人承担的任务，并定义其如何完成工作。
- **Rup预定义的角色：**
  - 分析员角色
    - （业务流程分析员 、业务设计员 、业务模型复审员、需求复审员 、系统分析员 、用例阐释者 、用户界面设计员 ）
  - 开发人员角色
    - （构架设计师 、构架复审员 、封装体设计员、代码复审员 、数据库设计员 、设计复审员 、设计员 、实施员 、集成员 ）
  - 测试专业人员角色
    - （测试设计员 、测试员 ）
  - 经理角色
    - （变更控制经理 、配置经理、部署经理、流程工程师、项目经理、项目复审员 ）
  - 其他角色
    - （任意角色 、课程开发员 、图形设计员 、涉众 、系统管理员、技术文档编写员 、工具专家）

12.2.3 活动

- 活动是参与项目的角色为提供符合要求的结果而进行的工作。
- 一项活动是一个工作单元，由参与项目的某一成员执行，其具体内容由角色进行说明。
- 活动有明确的目的，其内容通常表述为创建或更新某些工件，例如一个模型、一个类或一个计划。每个活动都被分配给具体的角色。
- 一个活动一般延续几个小时到几天，它通常涉及一个角色，只影响一个或少数几个工件。

12.2.3 活动

- 例：


构架设计师
构架分析
确定设计机制
确定设计元素
合并现有设计元素
说明运行时构架
说明分布
构架复审员
复审构架
设计员
用例分析
用例设计
子系统设计
类设计
数据库设计员
数据库设计
封装体设计员
封装体设计
设计复审员
复审设计

12.2.4 工件

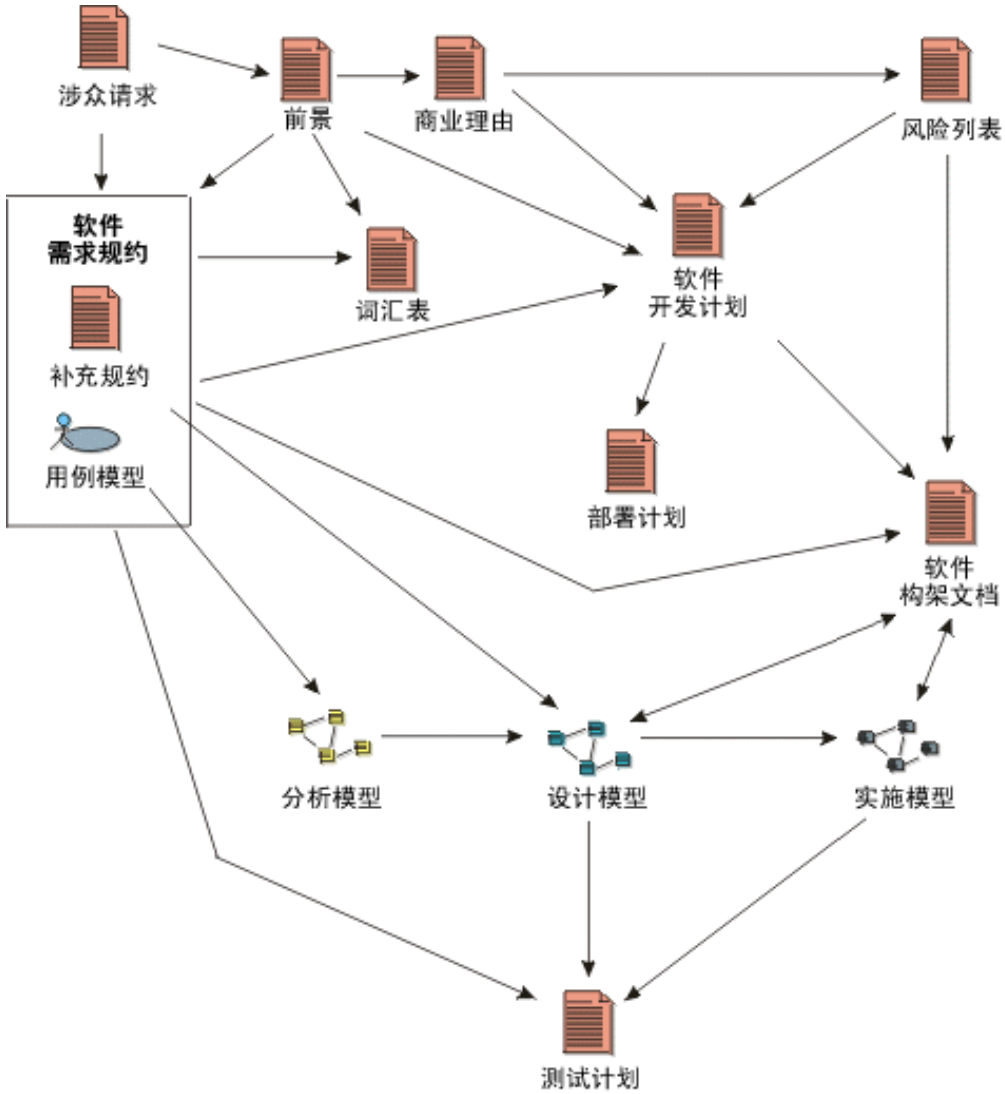
- 工件是流程的工作产品：
  - 角色使用工件执行活动，并在执行活动的过程中生成工件。
- 工件是单个角色的职责，它体现的是这样一种思想：
  - 流程中的每条信息都必须是一个具体的人的职责。即使一个人可能“拥有”某个工件，但其他人也可以使用该工件，如果授予权限，或许他们还可以更新这个工件。
- 工件分为输入工件和输出工件。

12.2.4 工件

- 工件有多种形式：
  - 模型，例如用例模型或设计模型，它包含其他工件。
  - 模型元素，即模型中的元素，例如设计类、用例或设计子系统
  - 文档，例如商业理由或软件构架文档
  - 源代码和可执行程序（某种构件）
  - 可执行程序
- 工件示例：
  - 存储在 **Rational Rose** 中的设计模型。
  - 存储在 **Microsoft Project** 中的项目计划。
  - 存储在 **ClearQuest** 中的缺陷。
  - **RequisitePro** 中的项目需求数据库。

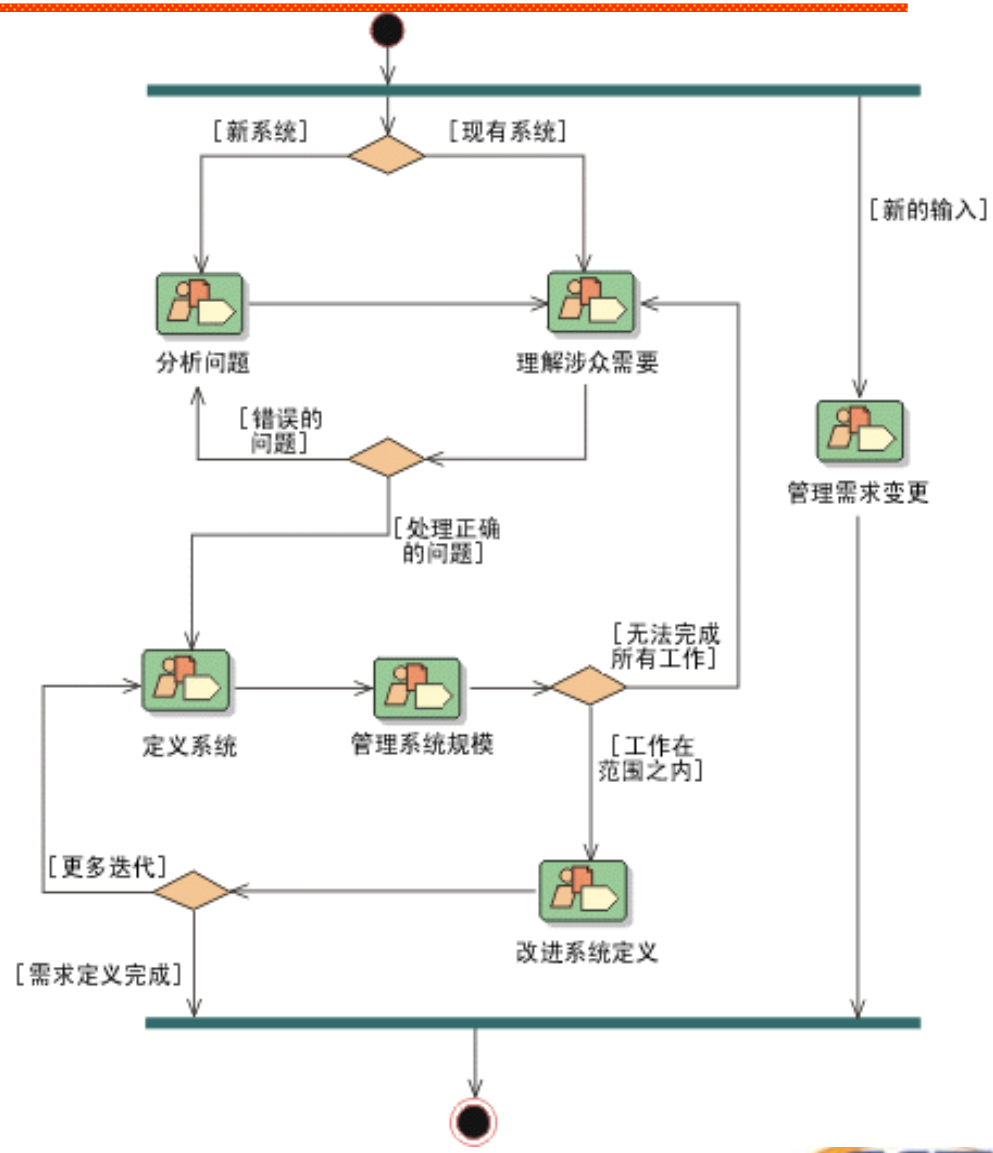
12.2.4 工件

- 主要工件


涉众请求
前景
商业理由
风险列表
软件
需求规约
词汇表
软件
开发计划
补充规约
用例模型
部署计划
软件
构架文档
分析模型
设计模型
实施模型
测试计划

12.2.5 工作流程

- 一个*工作流程*就是一系列活动
- 按 **UML** 术语，工作流程可以表现为序列图、协作图或活动图。在 **RUP** 中，使用活动图。

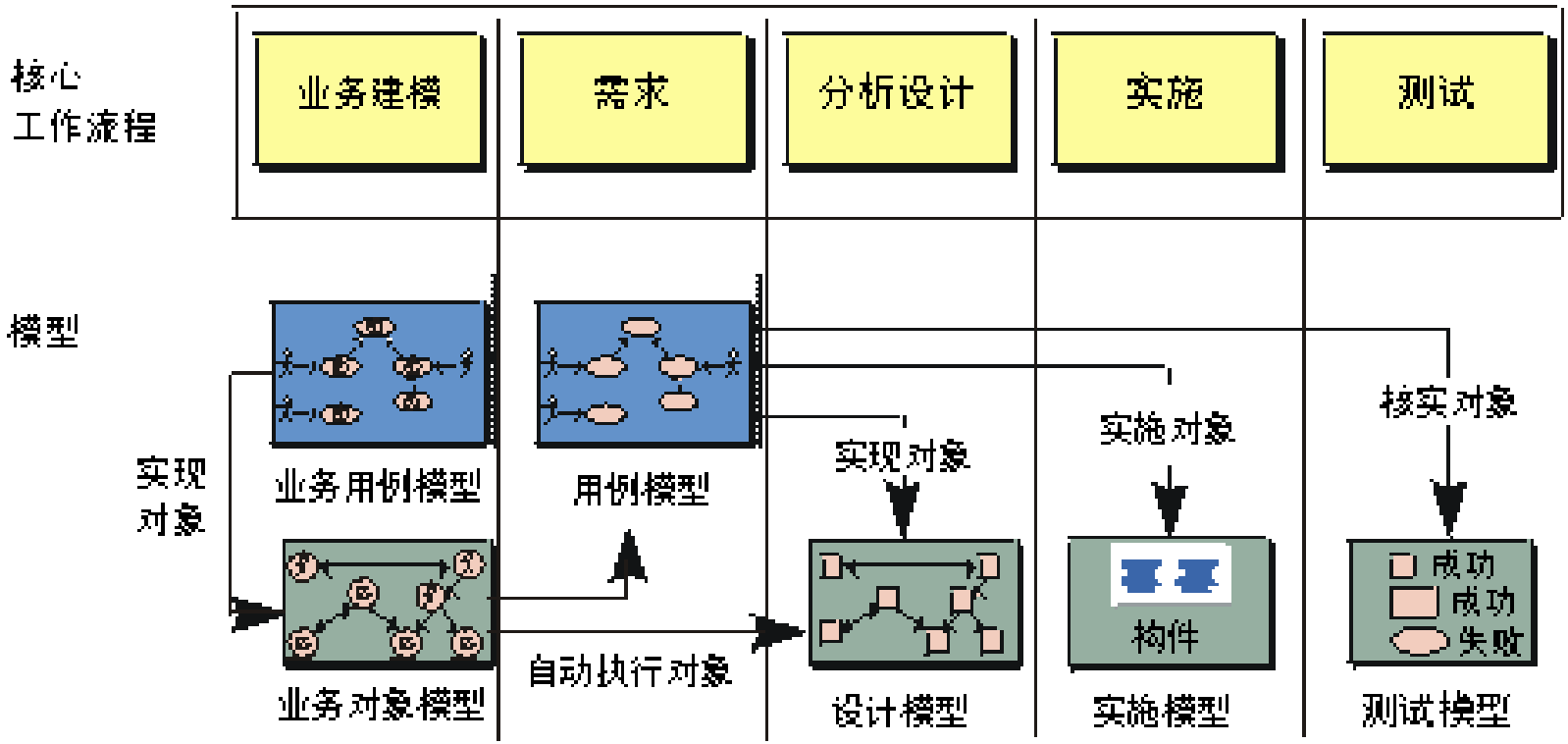


12.3 核心工作流

- 核心过程工作流（在项目中的流程）
  - 业务建模（**Business Modeling**）
    - 对开发系统所在的机构及其商业规则进行建模；
  - 需求（**Requirement**）
    - 定义系统功能及用户界面；
  - 分析设计（**Analysis & Design**）
    - 建立分析模型和设计模型；
  - 实现 （**Implementation**）
    - 编码实现；
  - 测试 （**Test**）
    - 软件测试；
  - 部署 （**Deployment**）
    - 打包、分发、安装软件

12.3 核心工作流

- 每个核心工作流程都与一个特定的模型集相关联



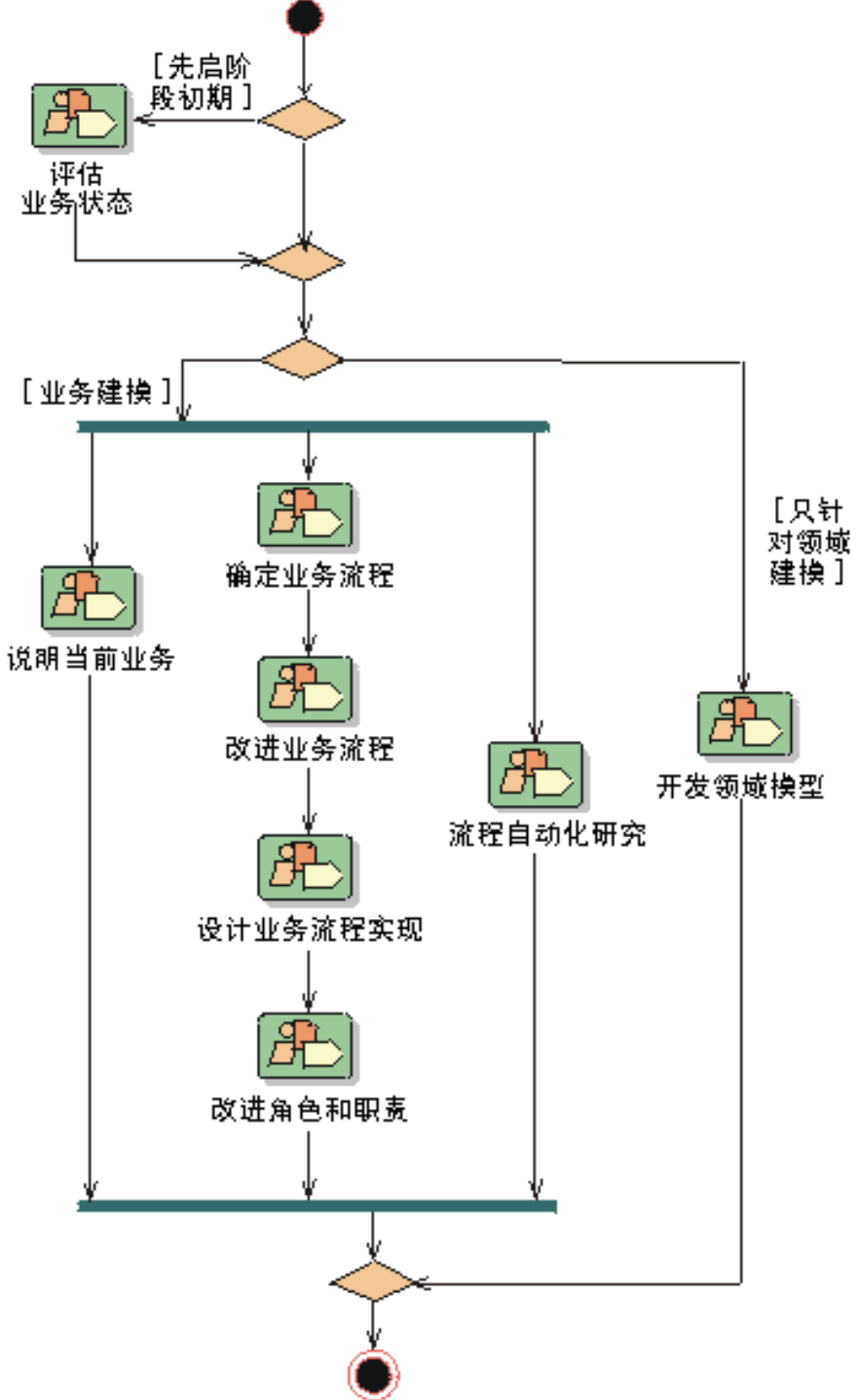
12.3 核心工作流

- 核心支持工作流（在组织中的流程）
  - 配置与变更管理（**Configuration & Change Management**）
    - 跟踪并维护系统开发过程中所产生的所有制品的完整和一致性；
  - 项目管理 （**Project Management**）
    - 为软件项目提供计划、人员分配、执行、监控等方面的管理；
  - 环境 （**Environment**）
    - 为软件开发机构提供软件开发环境的支持。

12.3.1 业务需求建模

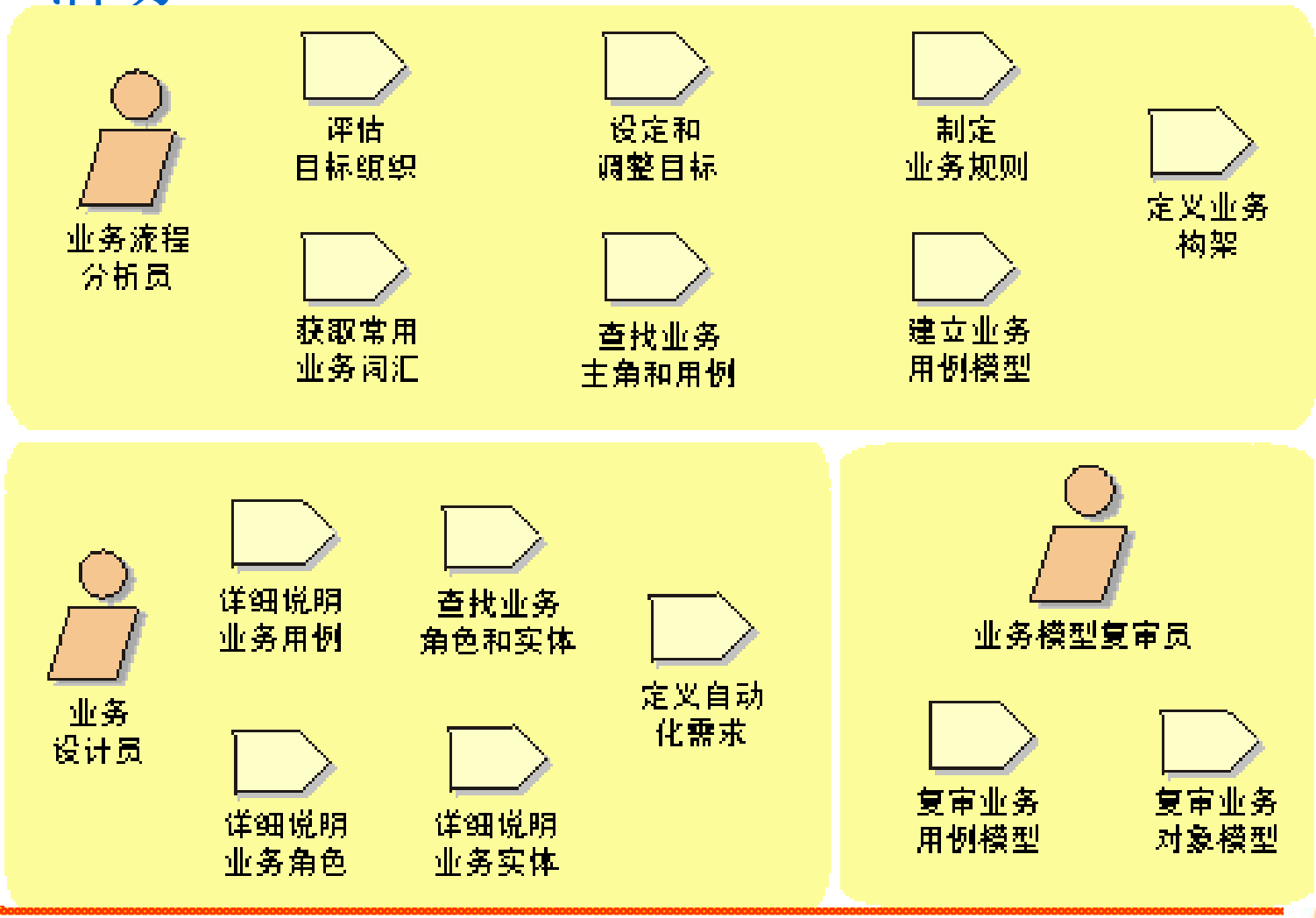
- 目的：
  - 了解目标组织（将要在其中部署系统的组织）的结构及机制。
  - 了解目标组织中当前存在的问题并确定改进的可能性。
  - 确保客户、最终用户和开发人员就目标组织达成共识。
  - 导出支持目标组织所需的系统需求。
- 为实现这些目标，业务建模工作流程说明了如何拟定新目标组织的前景，并基于该前景来确定该组织在业务用例模型和业务对象模型中的流程、角色以及职责。
- 作为对这些模型的补充，还编写了以下文档：
  - 补充业务规约
  - 词汇表

工作流程


[先启阶段初期]
评估业务状态
[业务建模]
确定业务流程
说明当前业务
改进业务流程
设计业务流程实现
改进角色和职责
流程自动化研究
[只针对领域建模]
开发领域模型

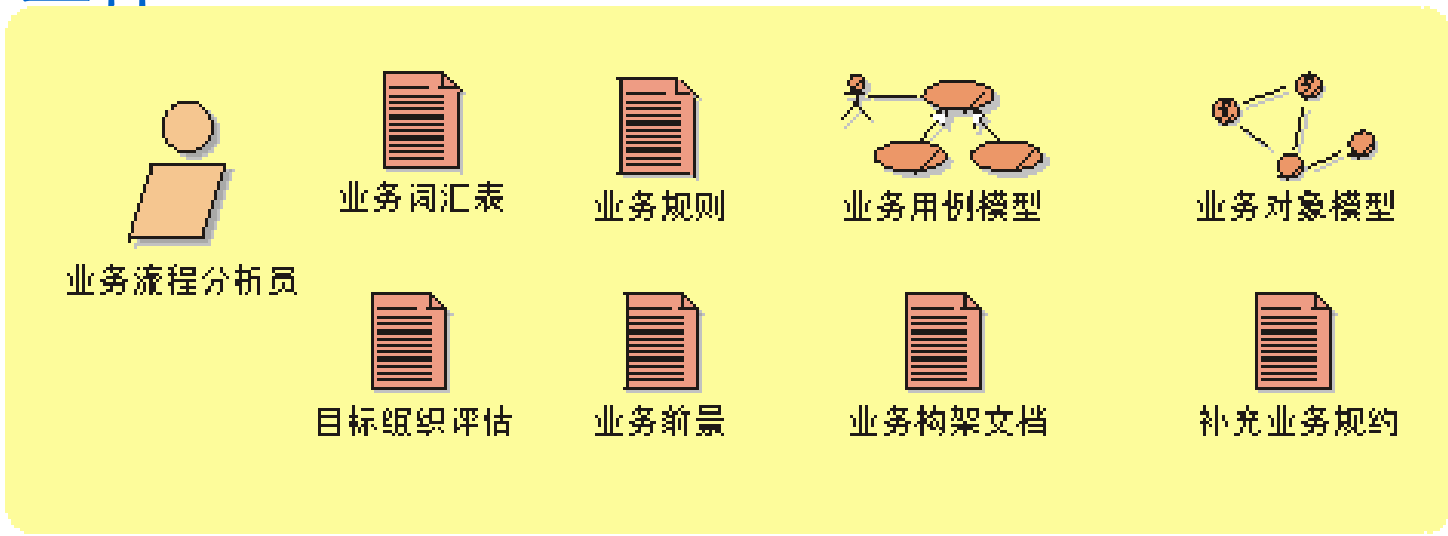
12.3.1 业务需求建模

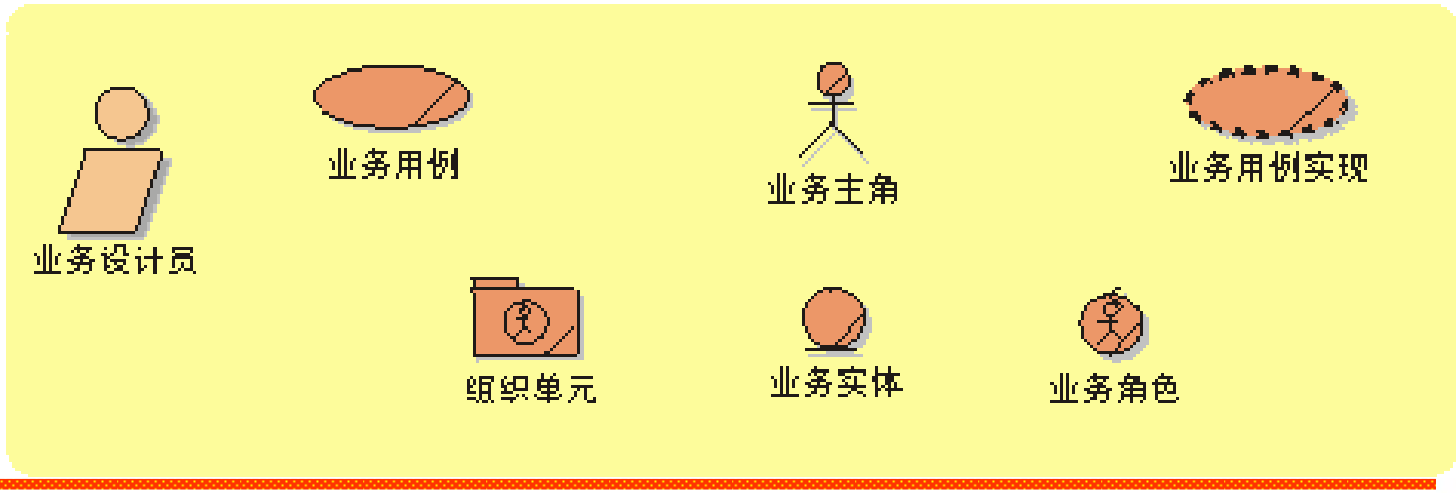
- 活动



12.3.1 业务需求建模

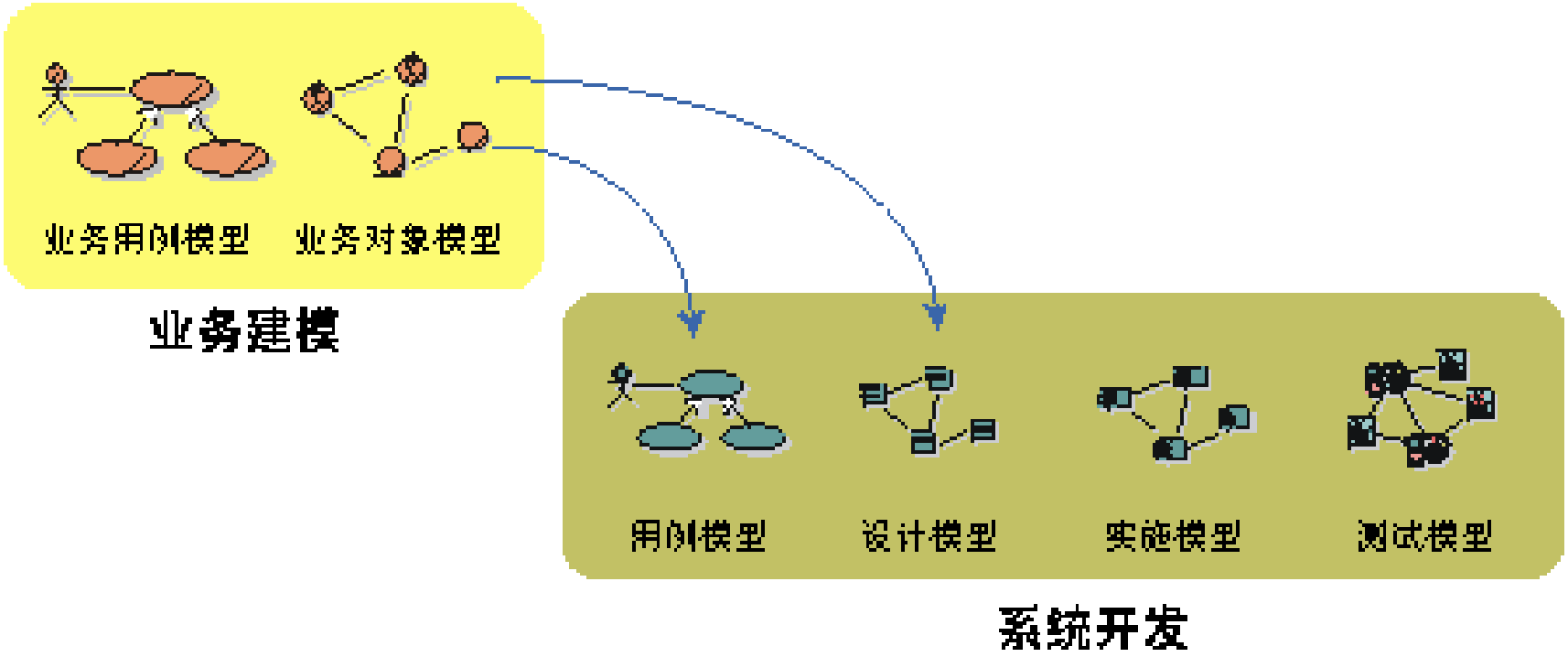
- 工件





12.3.1 业务需求建模

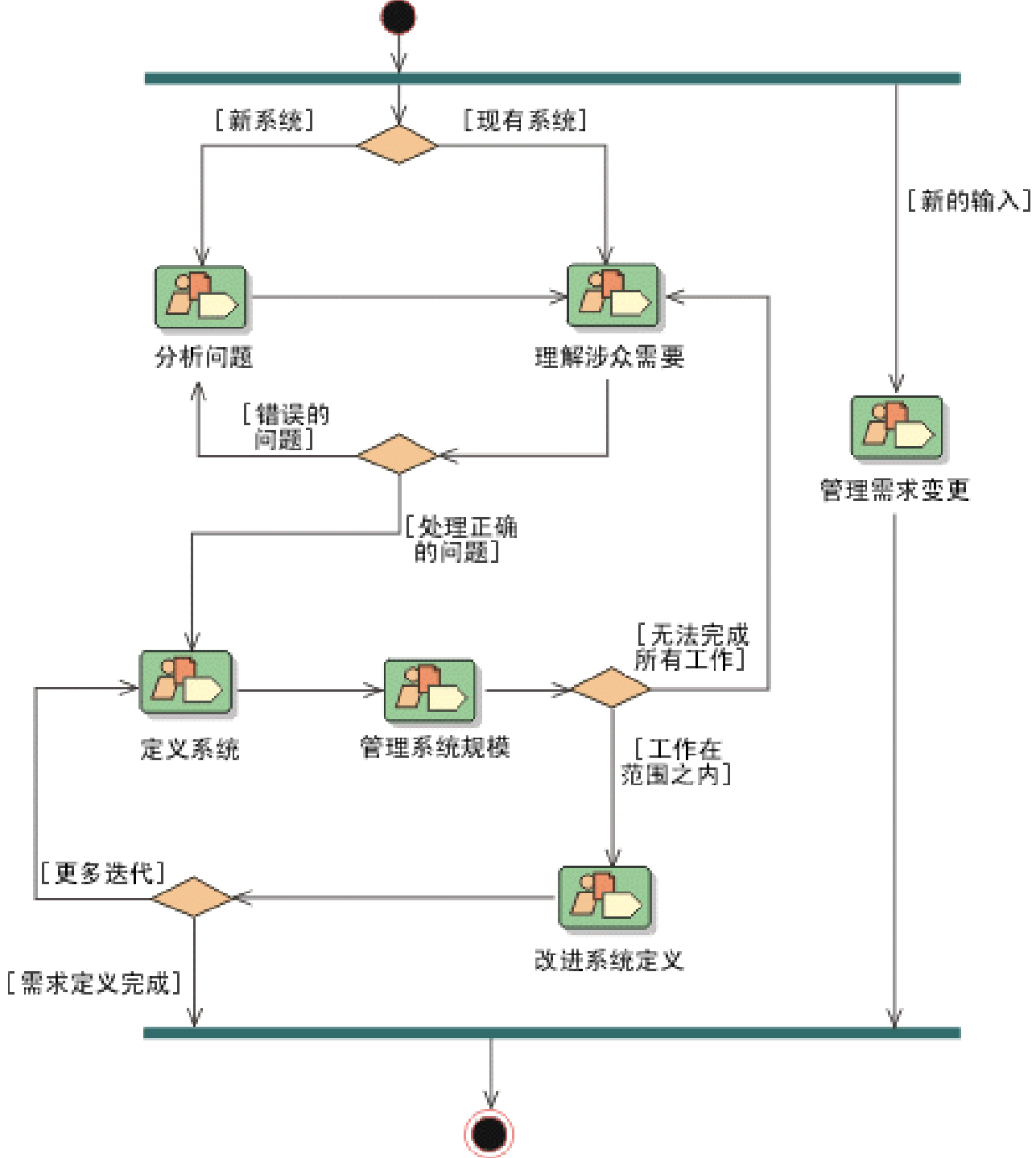
- 从业务模型到系统



12.3.2 需求

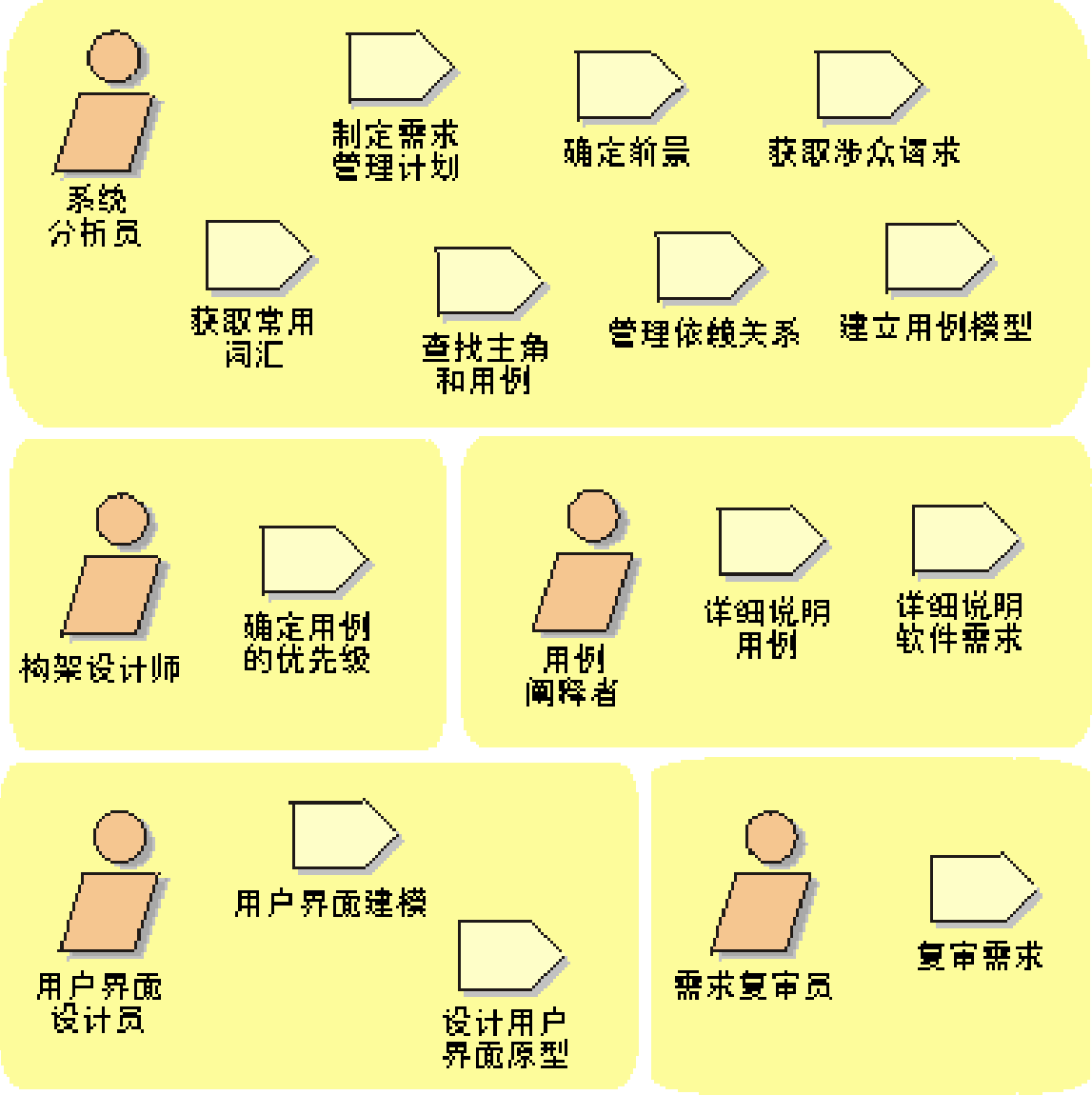
- 目的：
  - 与客户和其他涉众在系统的工作内容方面达成并保持一致。
  - 使系统开发人员能够更清楚地了解系统需求。
  - 定义系统边界（限定）。
  - 为计划迭代的技术内容提供基础。
  - 为估算开发系统所需成本和时间提供基础。
  - 定义系统的用户界面，重点是用户的需要和目标。
- 开发前景文档、用例模型、用例和补充规约。

工作流程


[新系统]
[现有系统]
[新的输入]
分析问题
理解涉众需要
[错误的问题]
管理需求变更
[处理正确的问题]
[无法完成所有工作]
定义系统
管理系统规模
[工作在范围之内]
[更多迭代]
改进系统定义
[需求定义完成]

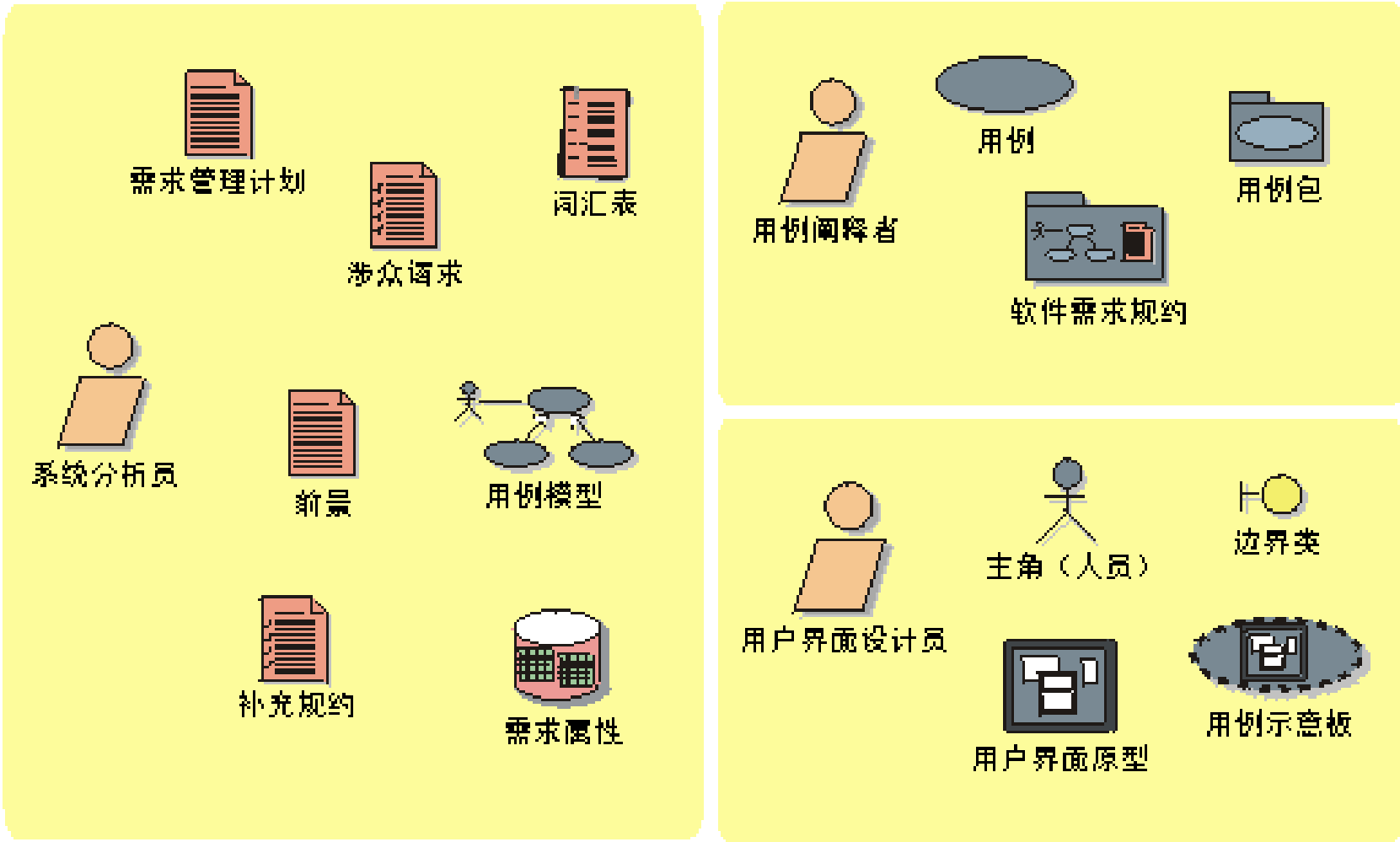
12.3.2 需求

- 活动


系统分析员
制定需求管理计划
确定前景
获取涉众请求
获取常用词汇
查找主角和用例
管理依赖关系
建立用例模型
构架设计师
确定用例的优先级
用例阐释者
详细说明用例
详细说明软件需求
用户界面设计员
用户界面建模
设计用户界面原型
需求复审员
复审需求

12.3.2 需求

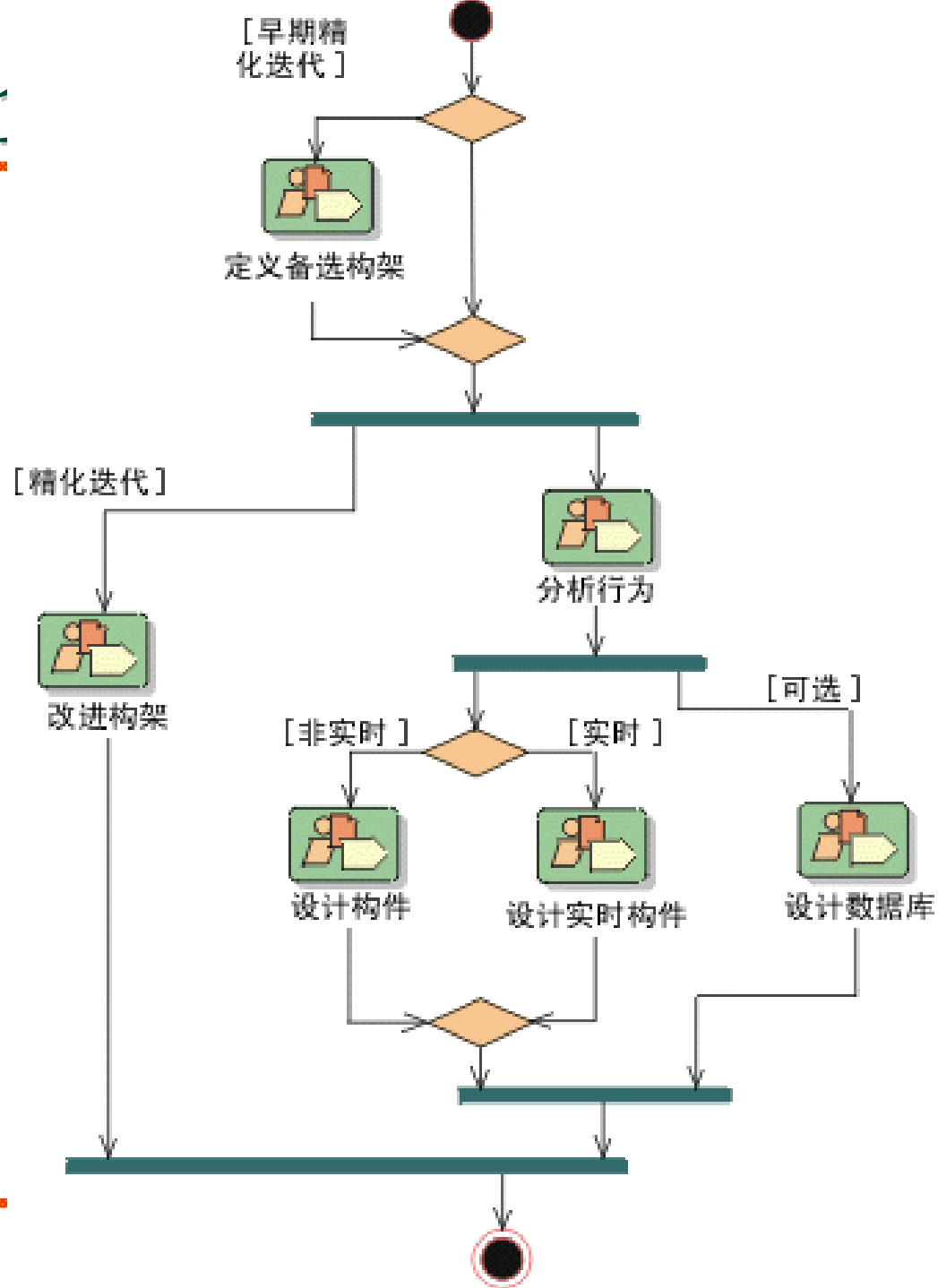
工件


需求管理计划
词汇表
涉众请求
系统分析员
新景
用例模型
补充规约
需求属性
用例阐释者
用例
用例包
软件需求规约
用户界面设计员
主角（人员）
边界类
用户界面原型
用例示意板

12.3.3 分析与设计

- 分析设计的目的在于：
  - 将业务需求转换为未来系统的设计。
  - 逐步开发强壮的系统构架。
  - 使设计适合于实施环境，为提高性能而进行设计。

工作流程

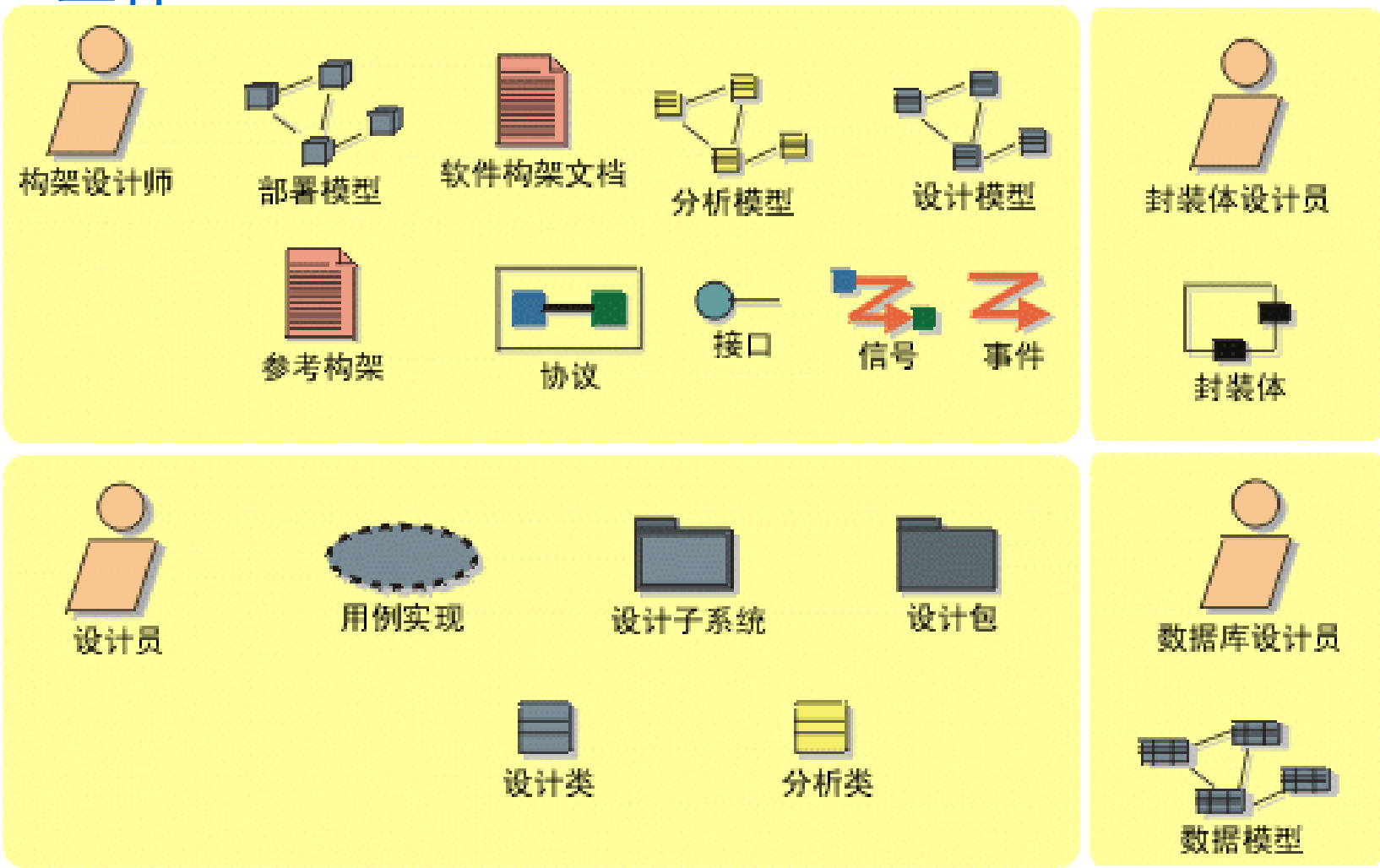


12.3.3 分析与设计

活动

12.3.3 分析与设计

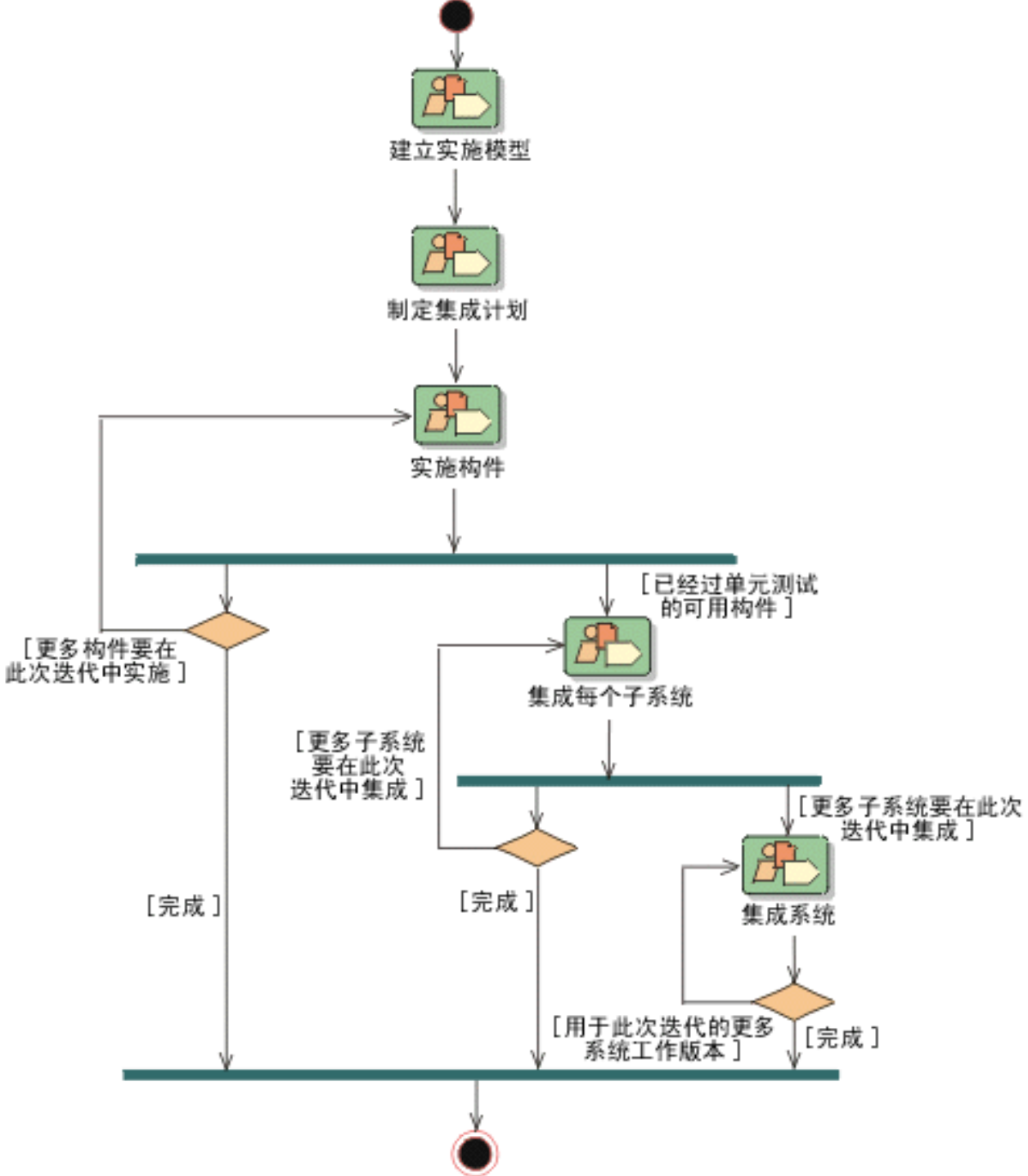
- 工件



12.3.4 实施

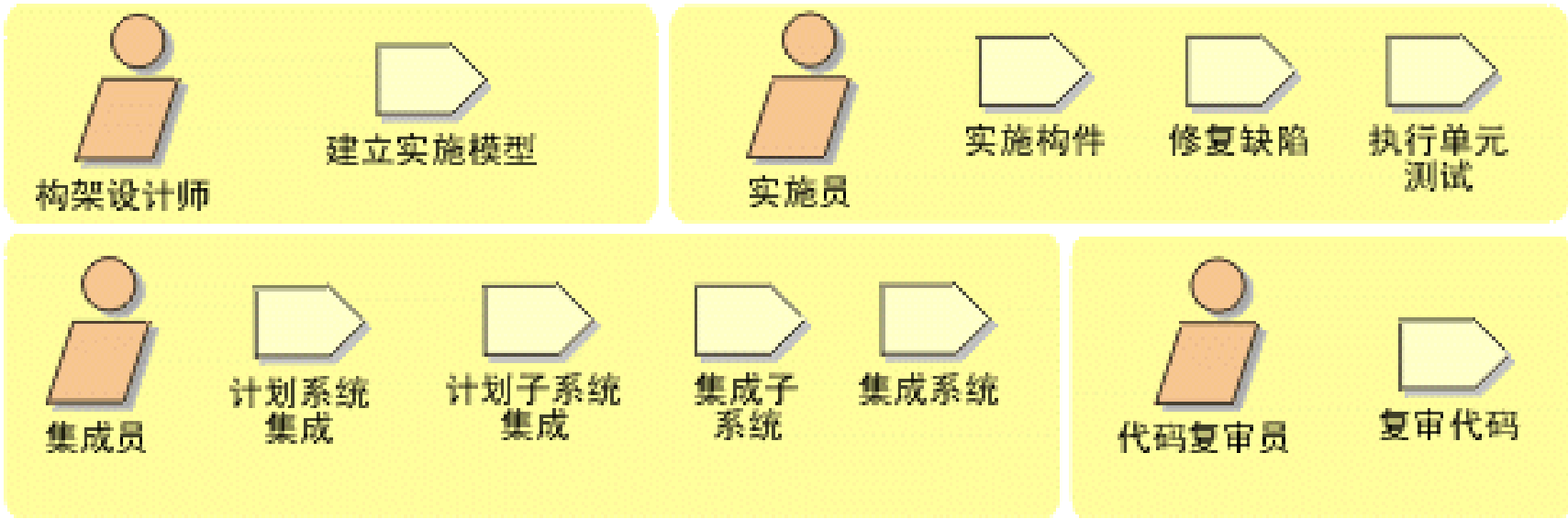
- 实施的目的包括：
  - 对照实施子系统的分层结构定义代码结构；
  - 以构件（源文件、二进制文件、可执行文件以及其他文件等）的方式实施类和对象；
  - 对已开发的构件按单元来测试；
  - 将各实施员（或团队）完成的结果集成到可执行系统中。

工作流程


建立实施模型
制定集成计划
实施构件
[已经过单元测试
的可用构件]
[更多构件要在
此次迭代中实施]
集成每个子系统
[更多子系统
要在此次
迭代中集成]
[更多子系统要在此次
迭代中集成]
[完成]
[完成]
集成系统
[用于此次迭代的更多
系统工作版本]
[完成]

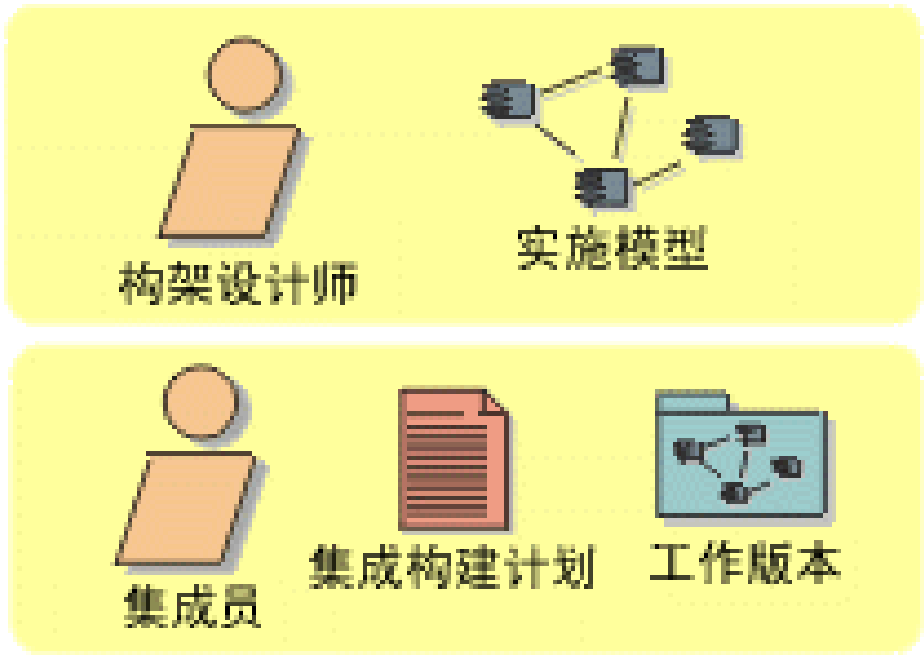
12.3.4 实施

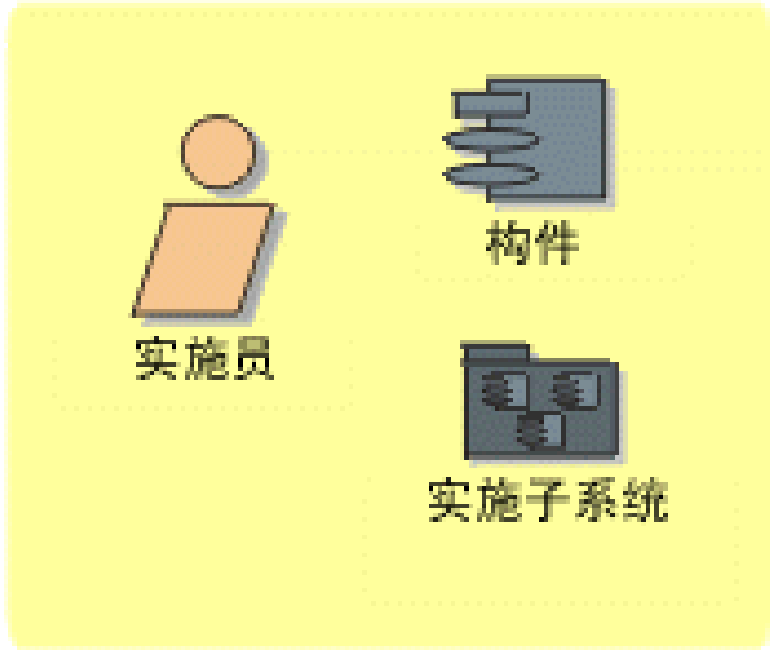
- 活动


构架设计师
建立实施模型
实施员
实施构件
修复缺陷
执行单元测试
集成员
计划系统集成
计划子系统集成
集成子系统
集成系统
代码复审员
复审代码

12.3.4 实施

- 工件

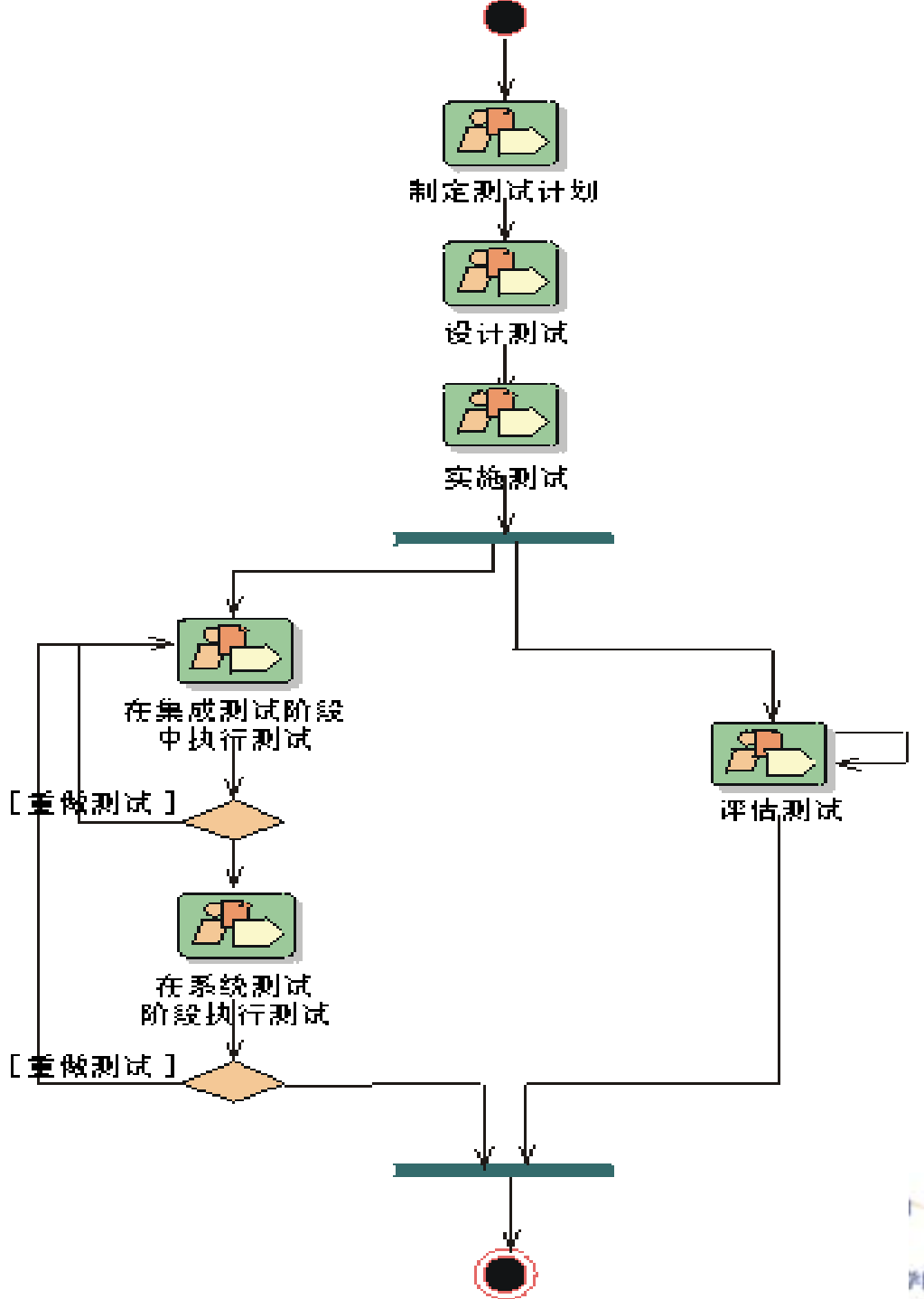




12.3.5 测试

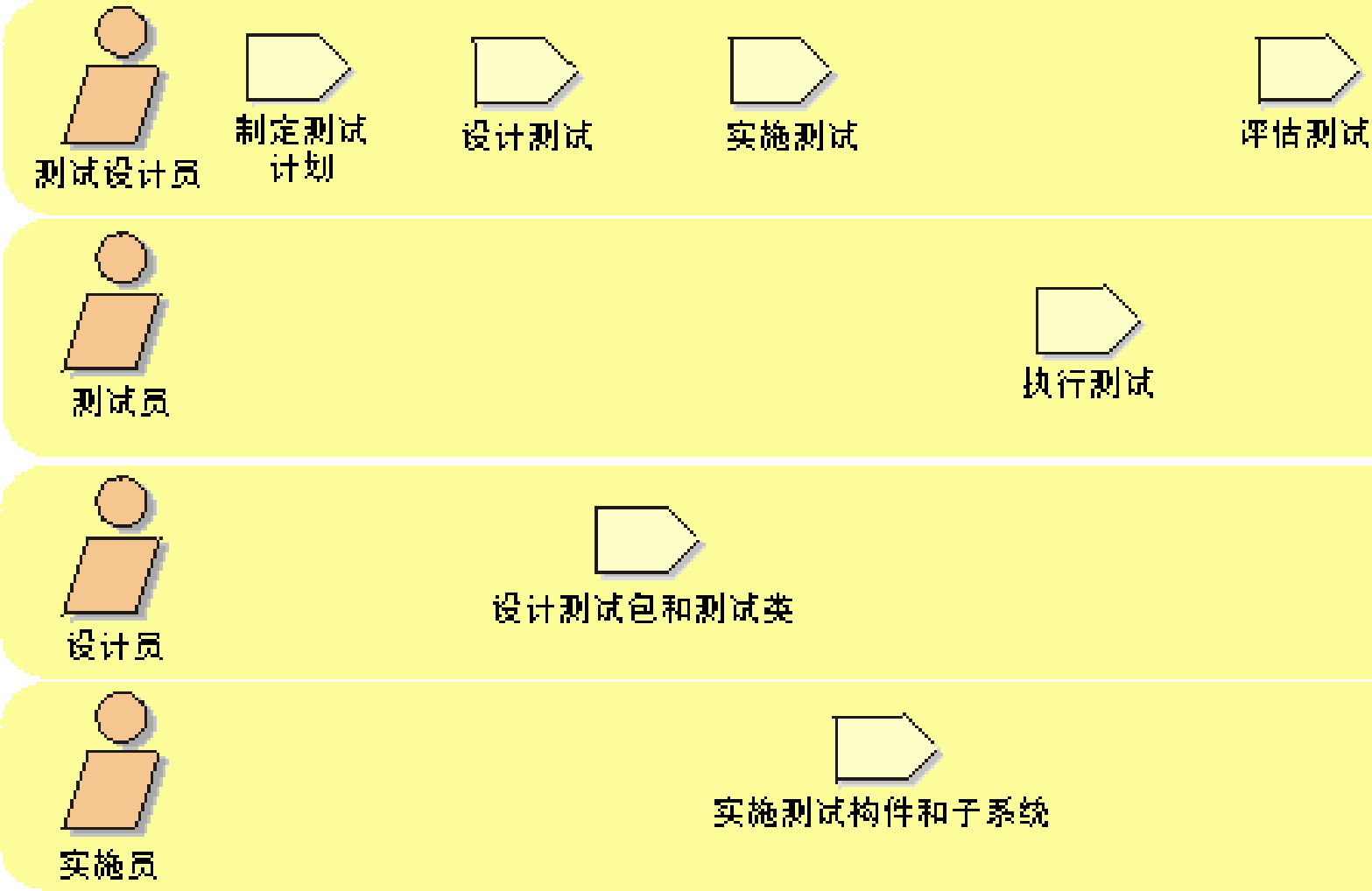
- 测试的目的在于：
  - 核实对象之间的交互。
  - 核实软件的所有构件是否正确集成。
  - 核实所有需求是否已经正确实施。
  - 确定缺陷并确保在部署软件之前将缺陷解决。

工作流程



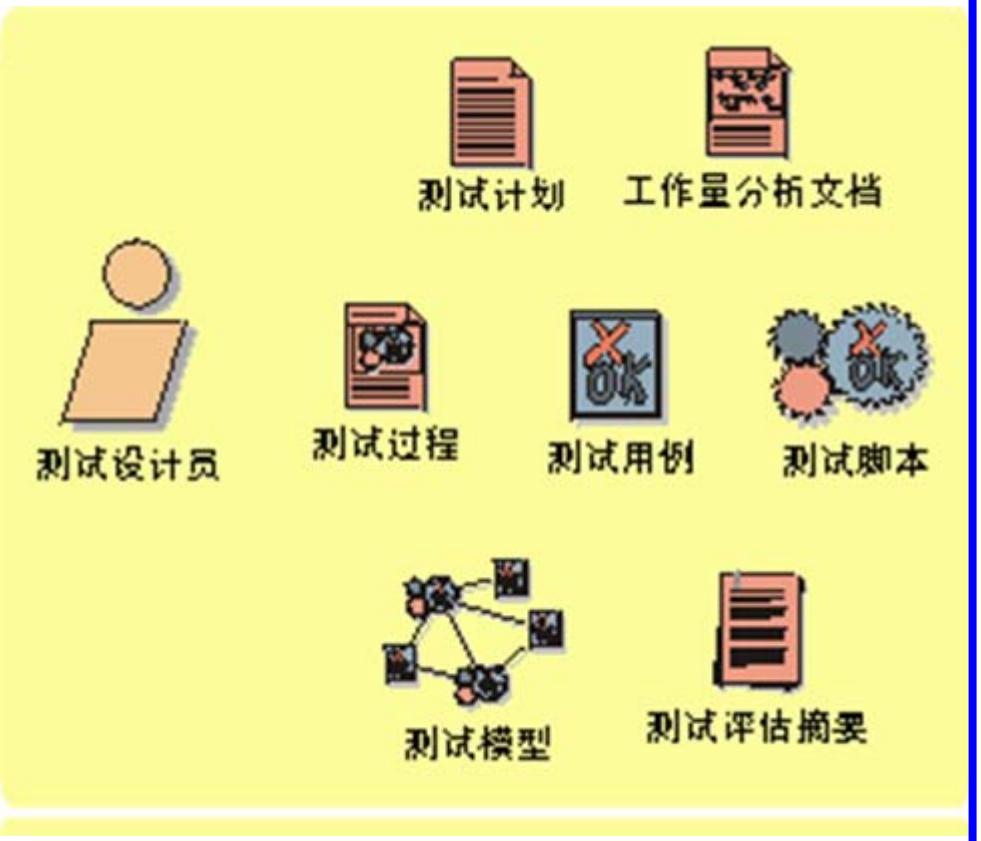
12.3.5 测试

活动


测试设计员
制定测试计划
设计测试
实施测试
评估测试
测试员
执行测试
设计员
设计测试包和测试类
实施员
实施测试构件和子系统

12.3.5 测试

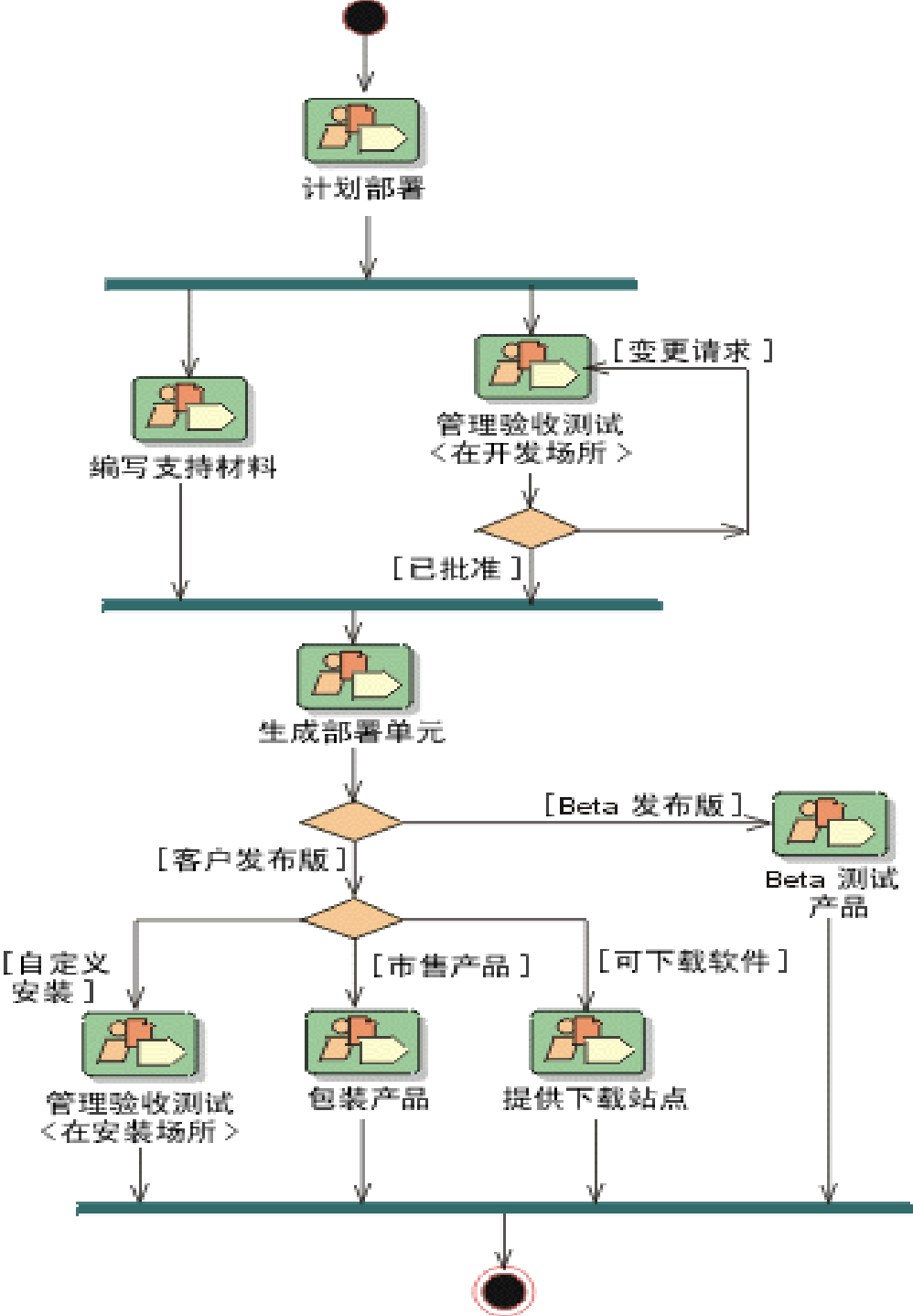
- 工件


测试计划
工作量分析文档
测试设计员
测试过程
测试用例
测试脚本
测试模型
测试评估摘要
测试员
测试结果
设计员
测试类
测试包
实施员
测试构件
测试子系统

12.3.6 部署

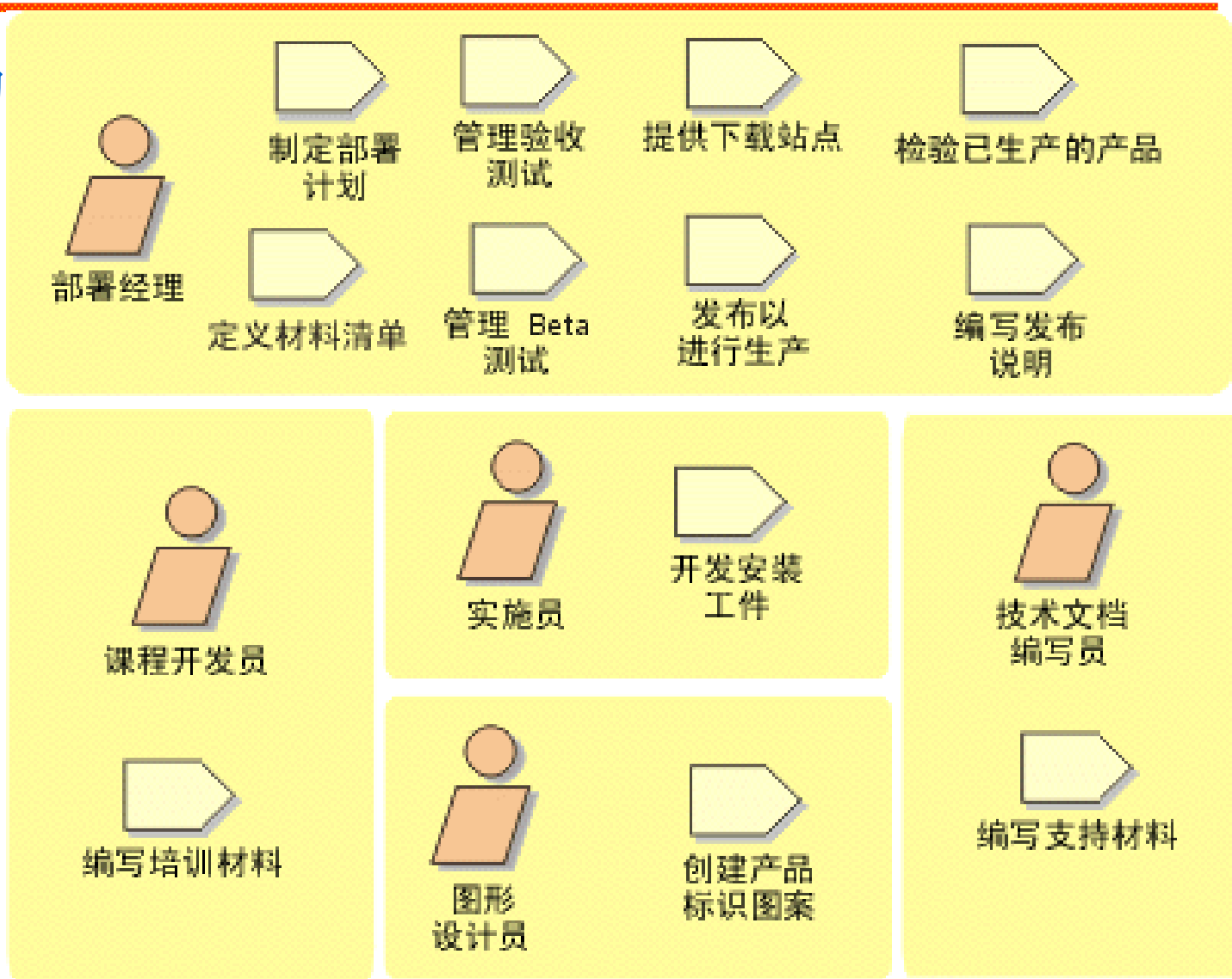
- 目的
  - 部署工作流程用来描述那些为确保最终用户可以正常使用软件产品而进行的活动。
- 三种产品部署的模式
  - 自定义安装
  - “市售”
  - 通过 **Internet** 使用软件

工作流程


计划部署
[变更请求]
管理验收测试
<在开发场所 >
编写支持材料
[已批准]
生成部署单元
[Beta 发布版]
Beta 测试
产品
[客户发布版]
[自定义
安装]
[市售产品]
[可下载软件]
管理验收测试
<在安装场所 >
包装产品
提供下载站点

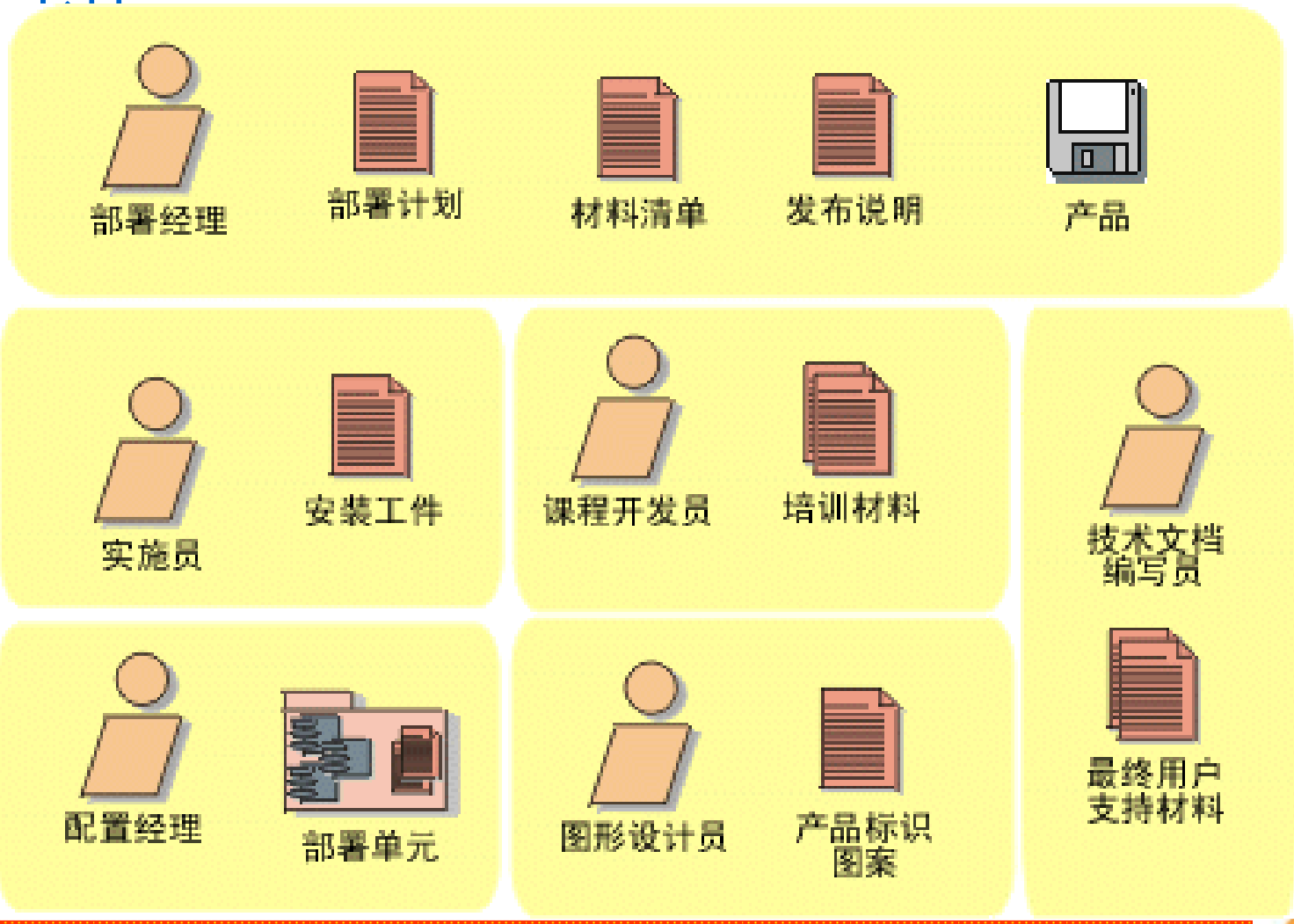
12.3.6 部署

- 活动


部署经理
制定部署计划
管理验收测试
提供下载站点
检验已生产的产品
定义材料清单
管理 Beta 测试
发布以进行生产
编写发布说明
课程开发员
编写培训材料
实施员
开发安装工件
图形设计员
创建产品标识图案
技术文档编写员
编写支持材料

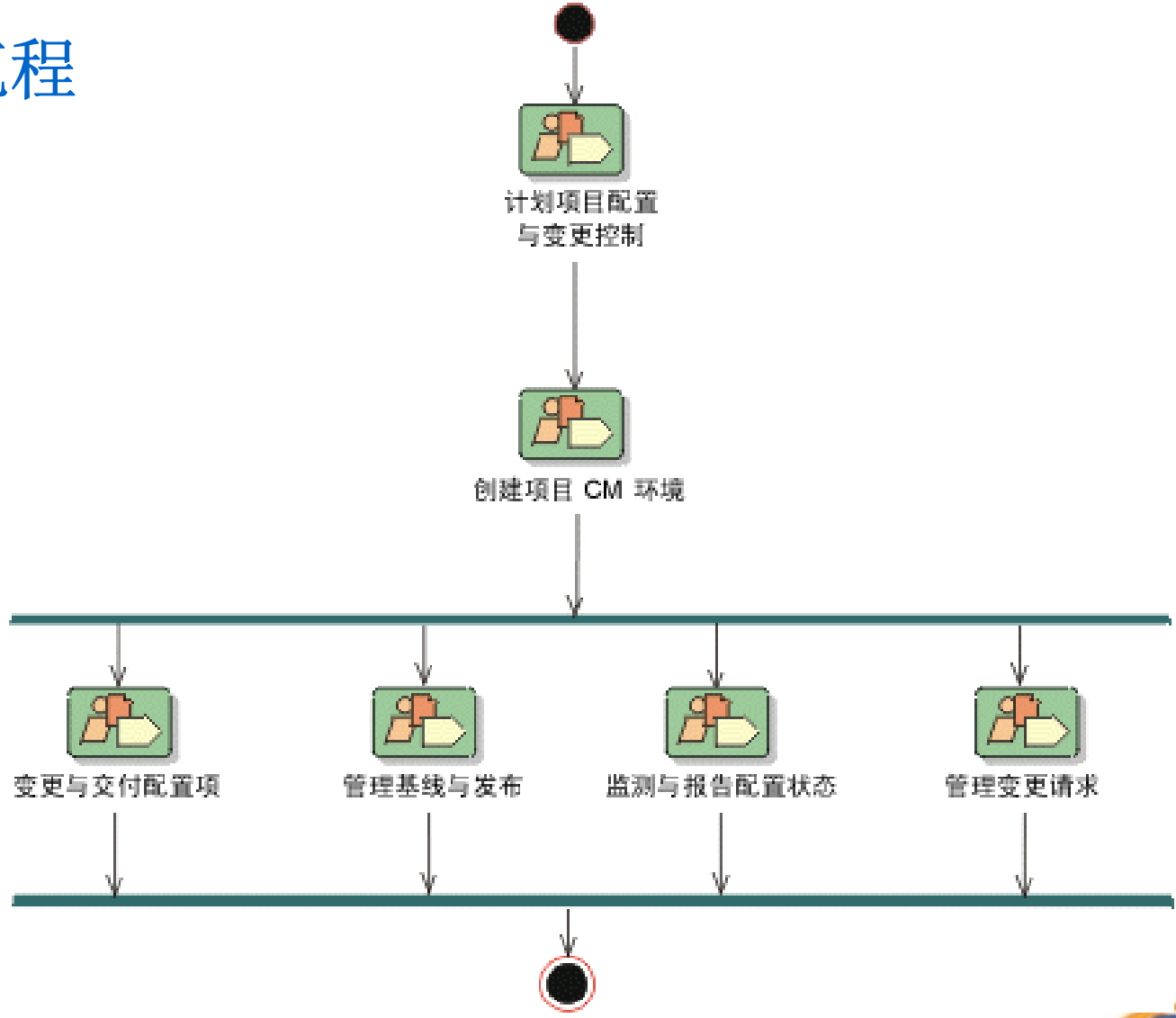
12.3.6 部署

- 工件


部署经理
部署计划
材料清单
发布说明
产品
实施员
安装工件
课程开发员
培训材料
技术文档
编写员
配置经理
部署单元
图形设计员
产品标识
图案
最终用户
支持材料

12.3.7 配置与变更管理

- 工作流程

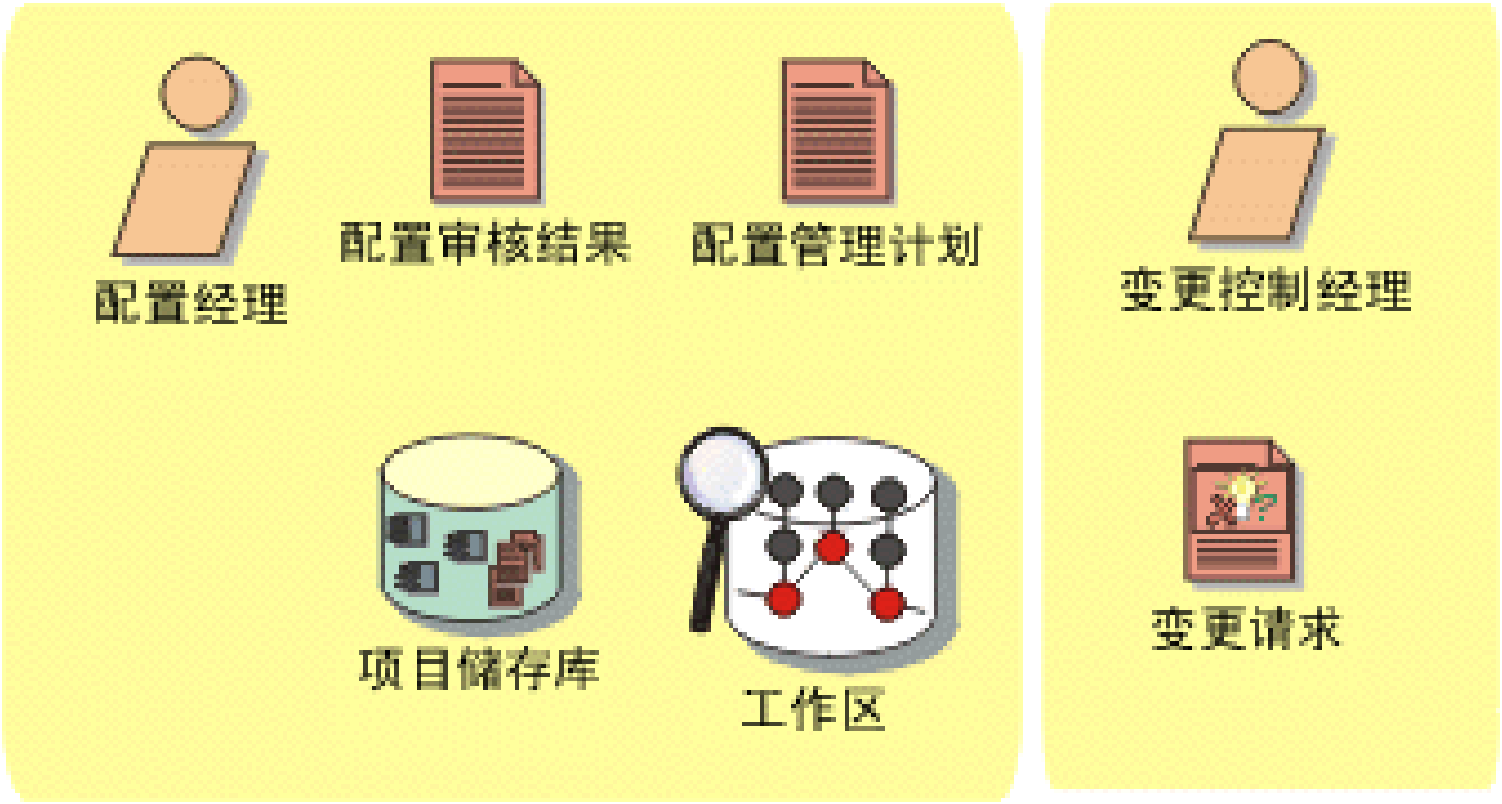


12.3.7 配置与变更管理

活动

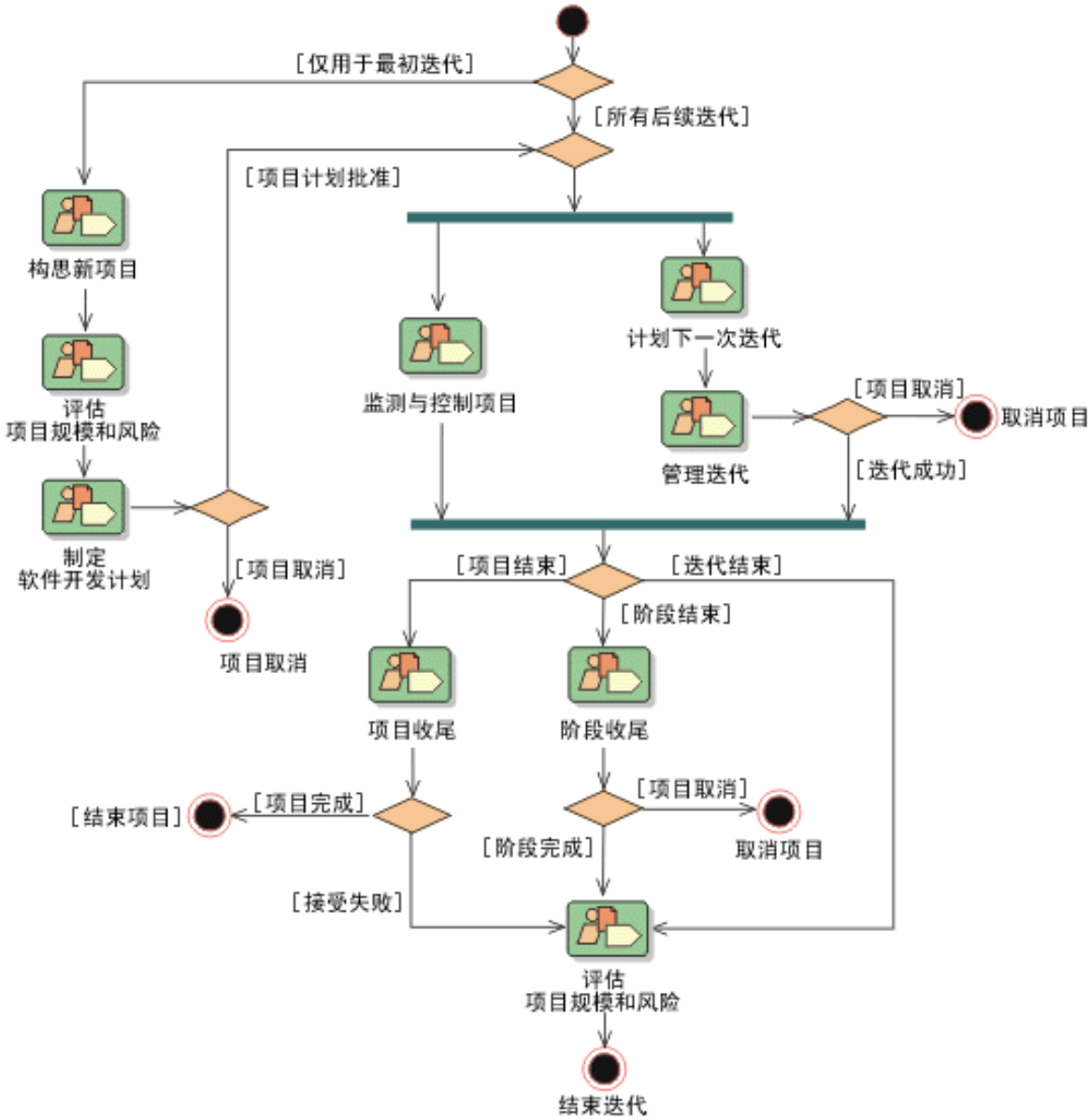
12.3.7 配置与变更管理

- 工件



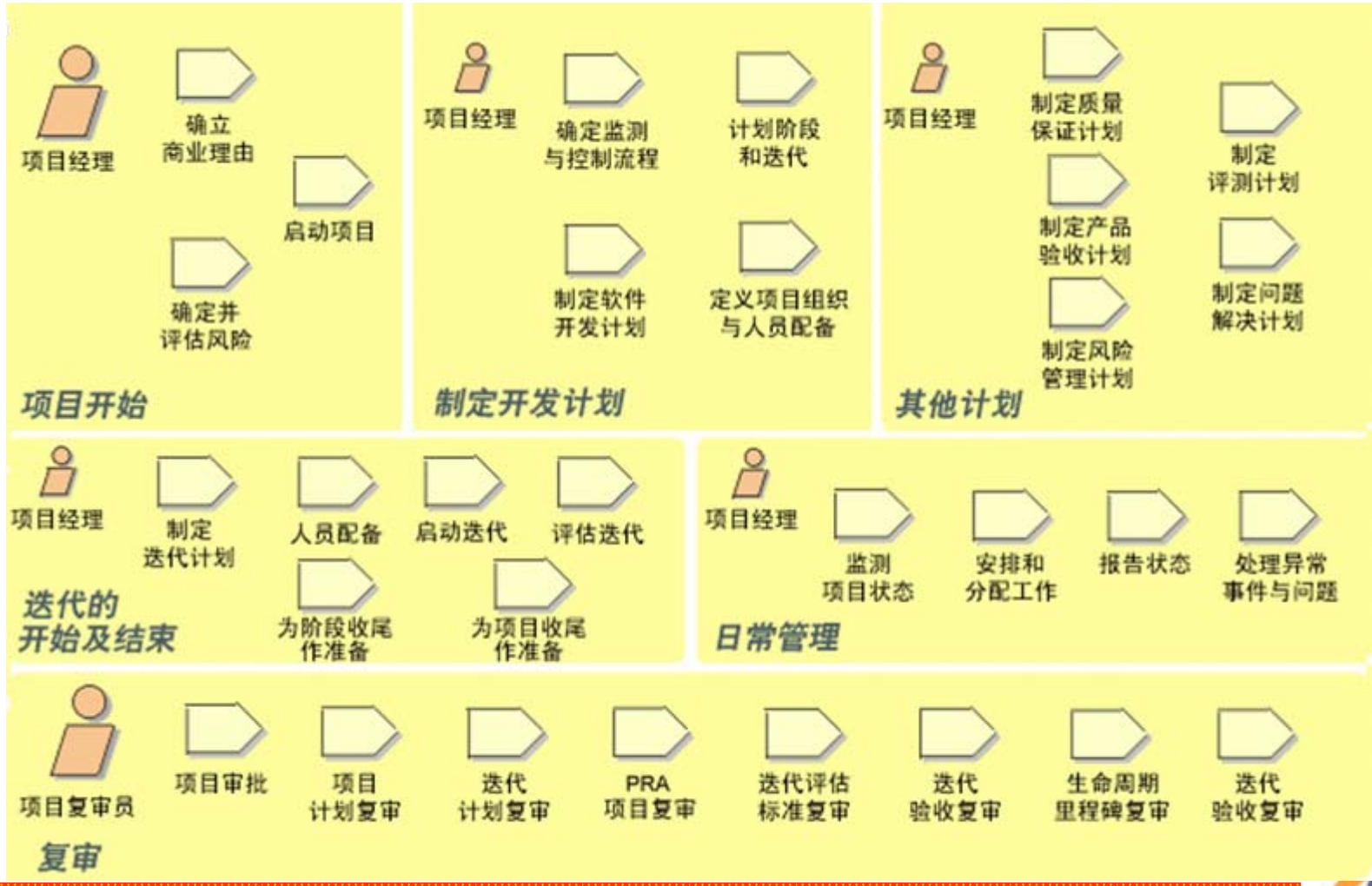
12.3.8 项目管理

- 工作流程



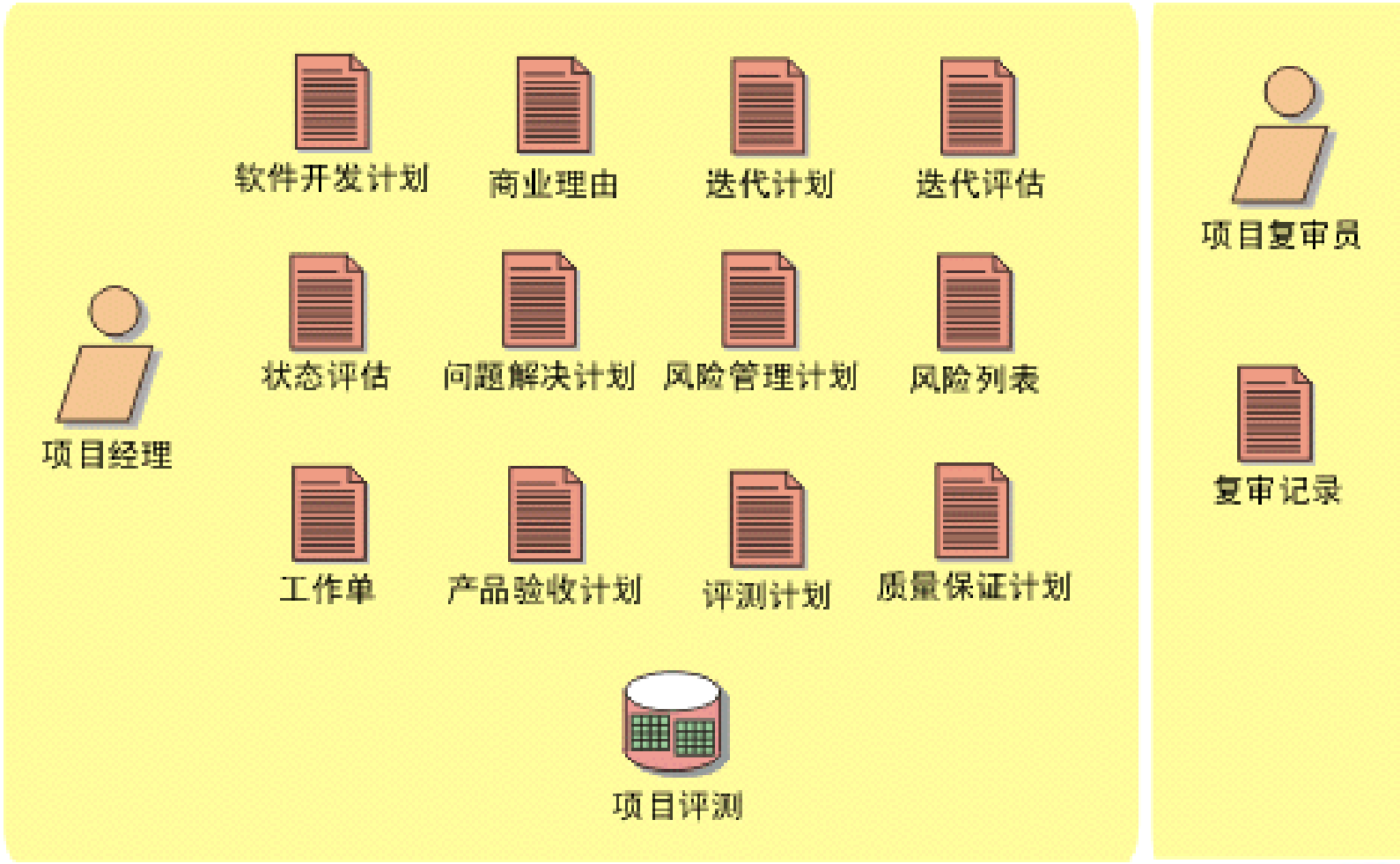
12.3.8 项目管理

活动


项目经理
确立
商业理由
启动项目
确定并
评估风险
项目开始
项目经理
确定监测
与控制流程
计划阶段
和迭代
制定软件
开发计划
定义项目组织
与人员配备
制定开发计划
项目经理
制定质量
保证计划
制定
评测计划
制定产品
验收计划
制定问题
解决计划
制定风险
管理计划
其他计划
项目经理
制定
迭代计划
人员配备
启动迭代
评估迭代
为阶段收尾
作准备
为项目收尾
作准备
迭代的
开始及结束
项目经理
监测
项目状态
安排和
分配工作
报告状态
处理异常
事件与问题
日常管理
项目复审员
项目审批
项目
计划复审
迭代
计划复审
PRA
项目复审
迭代评估
标准复审
迭代
验收复审
生命周期
里程碑复审
迭代
验收复审
复审

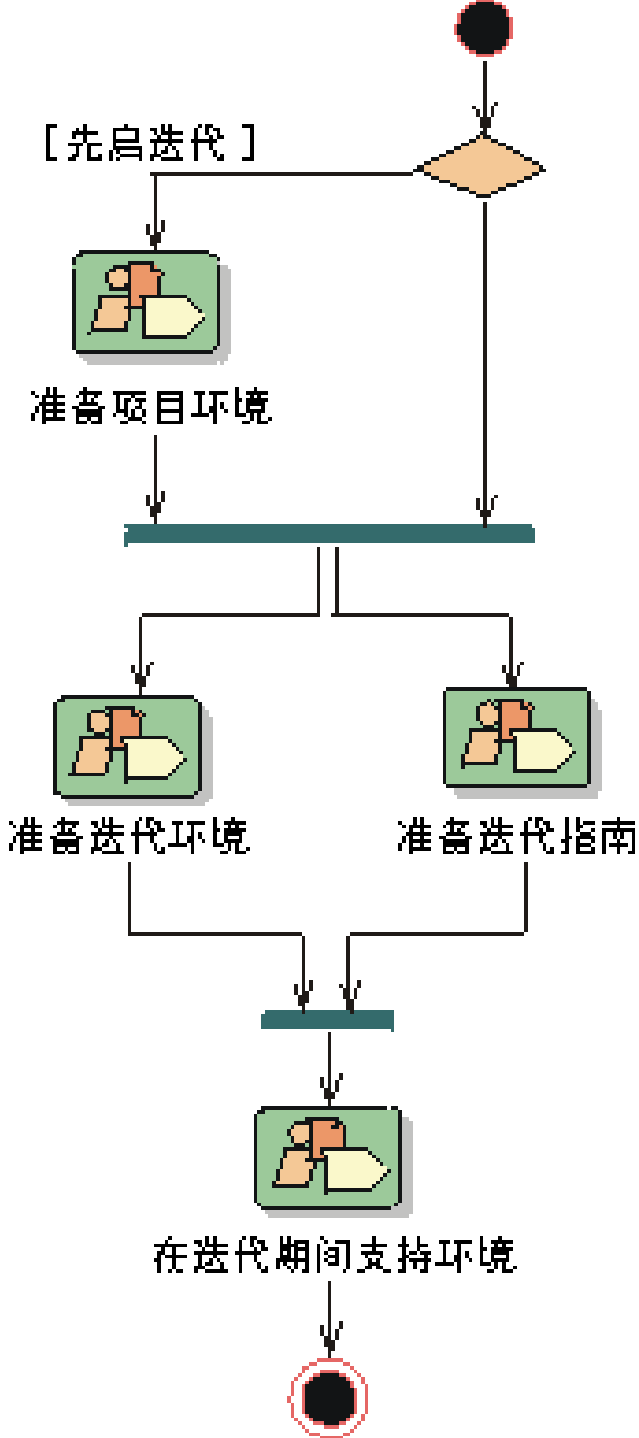
12.3.8 项目管理

工件


软件开发计划
商业理由
迭代计划
迭代评估
项目复审员
状态评估
问题解决计划
风险管理计划
风险列表
项目经理
复审记录
工作单
产品验收计划
评测计划
质量保证计划
项目评测

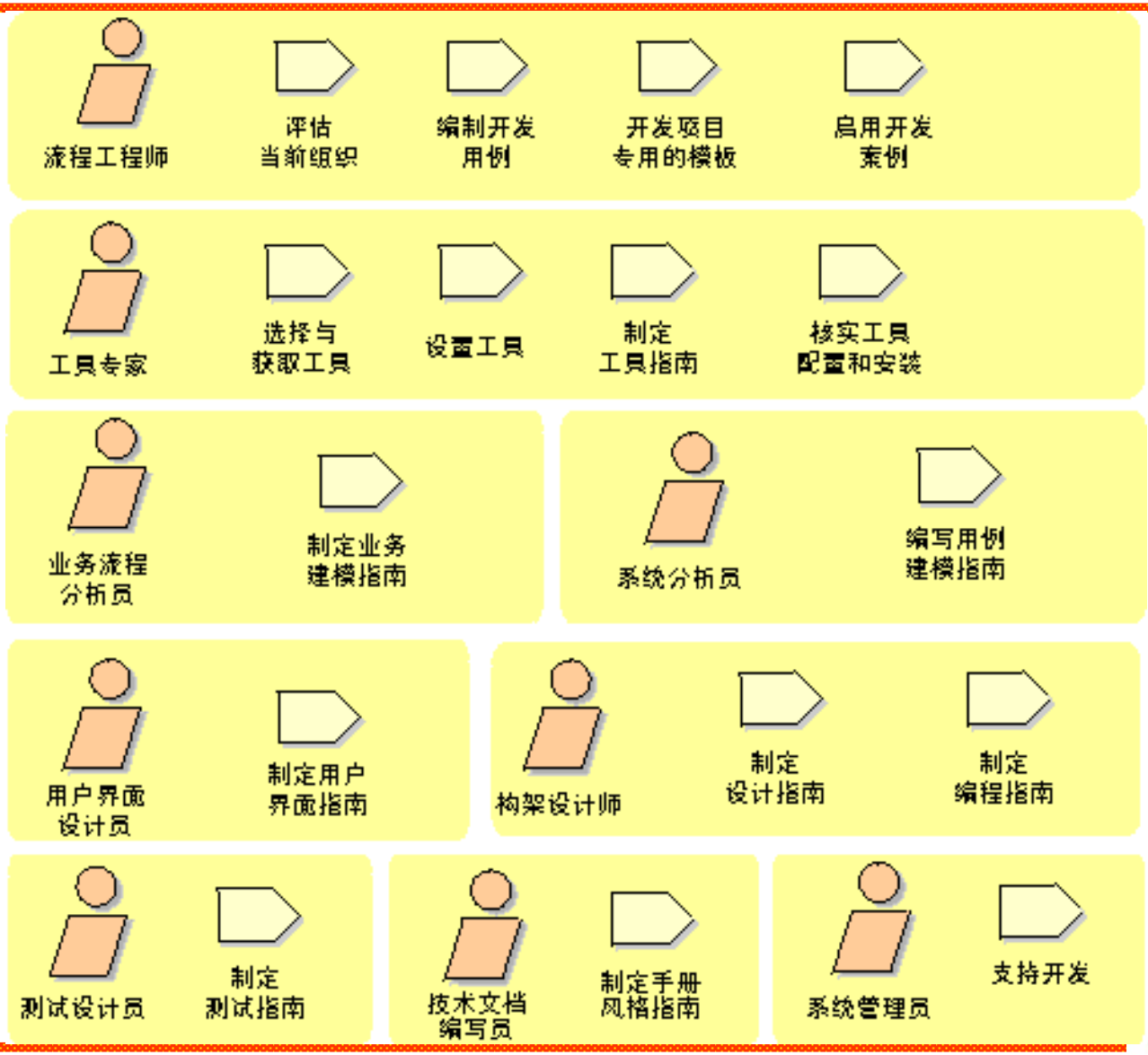
12.3.9 环境设置

- 工作流程



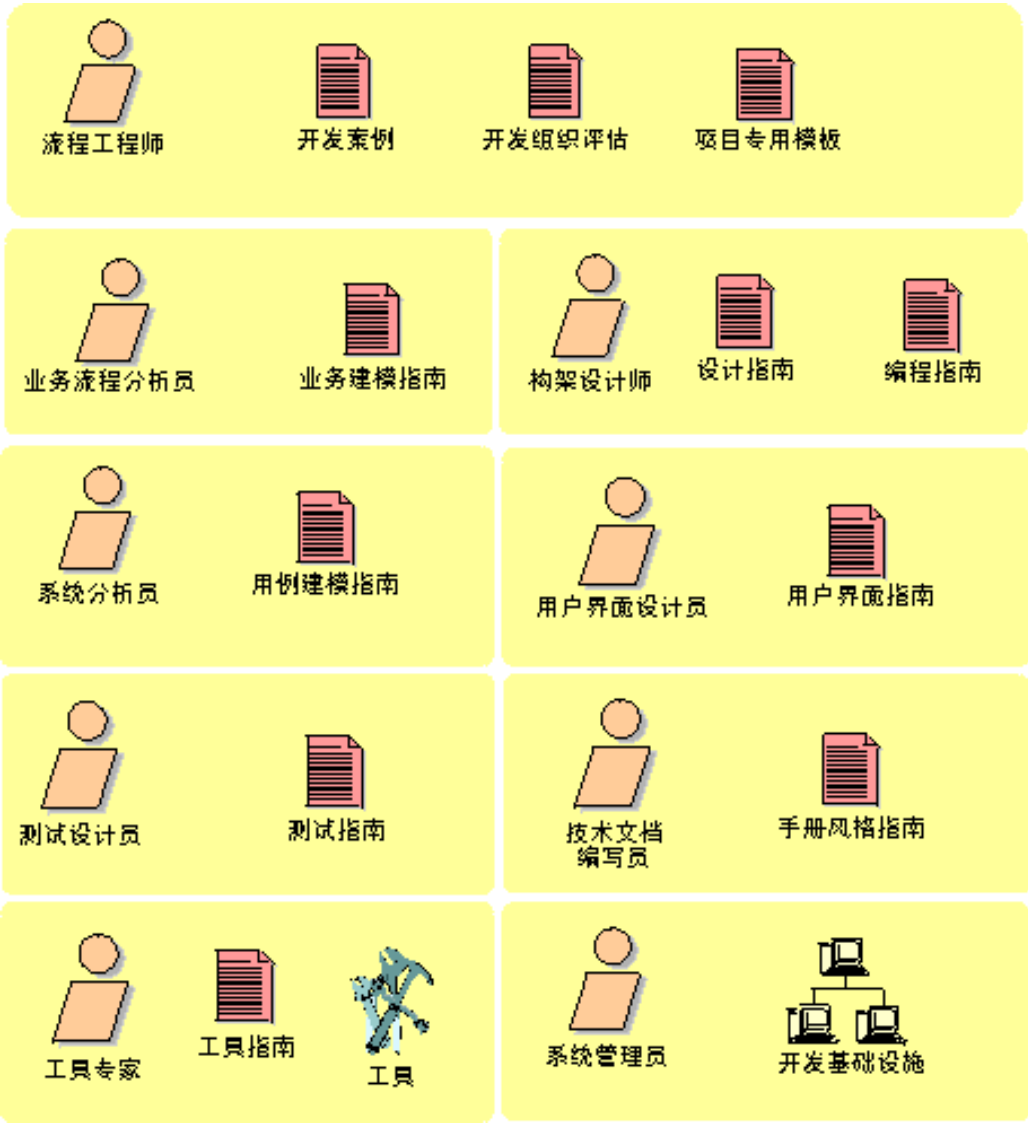
12.3.9 环境设置

- 活动


流程工程师
评估
当前组织
编制开发
用例
开发项目
专用的模板
启用开发
案例
工具专家
选择与
获取工具
设置工具
制定
工具指南
核实工具
配置和安装
业务流程
分析员
制定业务
建模指南
系统分析员
编写用例
建模指南
用户界面
设计员
制定用户
界面指南
构架设计师
制定
设计指南
制定
编程指南
测试设计员
制定
测试指南
技术文档
编写员
制定手册
风格指南
系统管理员
支持开发

12.3.9 环境设置

- 工件


流程工程师
开发案例
开发组织评估
项目专用模板
业务流程分析员
业务建模指南
构架设计师
设计指南
编程指南
系统分析员
用例建模指南
用户界面设计员
用户界面指南
测试设计员
测试指南
技术文档
编写员
手册风格指南
工具专家
工具指南
工具
系统管理员
开发基础设施

12.4 阶段

- RUP把生命周期分为若干个循环（Cycle），每个循环有4个阶段组成，并生成产品的一个新版本。
- 四个阶段：
  - 初始（Inception）
    - 定义最终产品视图和业务模型，确定系统范围；
  - 细化（Elaboration）
    - 设计系统的体系结构、制定工作计划和资源要求；
  - 构造（Construction）
    - 构造并完成产品；
  - 移交（Transition）
    - 产品交付给用户使用；

12.4.1 初始阶段

- 主要目标包括：
  - 建立项目的软件规模和边界条件，包括运作前景、验收标准以及希望软件中包括和不包括的内容。
  - 识别系统的关键用例（也就是将造成重要设计折衷操作的主要部分）。
  - 评估整个项目的总体成本和进度（以及对即将进行的精化阶段进行更详细的评估）
  - 评估潜在风险（不可预测性的来源）
  - 准备项目的支持环境。

12.4.1 初始阶段

核心活动

- 明确地说明项目规模。
- 计划和准备商业理由。
  - 评估风险管理、人员配备、项目计划和成本/进度/收益率折衷的备选方案。
- 综合考虑备选构架，评估设计和自制/外购/复用方面的折衷，从而估算出成本、进度和资源。
- 准备项目的环境，评估项目和组织，选择工具，决定流程中要改进的部分。

12.4.1 初始阶段

- 里程碑：生命周期目标
  - 生命周期目标里程碑评估项目的基本可行性。
  - 先启阶段末是第一个重要的项目里程碑，即生命周期目标里程碑。此时，检查项目的生命周期目标，并决定继续进行项目还是取消项目。
- 评估标准
  - 规模定义和成本／进度估算中，所有相关因素（如客户等）可并行
  - 对是否已经获得正确的需求集达成一致意见，并且对这些需求的理解是共同的。
  - 对成本／进度估算、优先级、风险和开发流程是否合适达成一致意见。
  - 已经确定所有风险并且有针对每个风险的减轻风险策略。
- 如果项目无法达到该里程碑，则它可能中途失败或需要进行相当多的重新考虑。

12.4.2 细化阶段

- 主要目标包括：
  - 确保构架、需求和计划足够稳定，充分减少风险，从而能够有预见性地确定完成开发所需的成本和进度。对大多数项目来说，通过此里程碑也就相当于从简单快速的低风险运作转移到高成本、高风险的运作，并且在组织结构方面面临许多不利因素。
  - 处理在构架方面具有重要意义的所有项目风险
  - 建立一个已确定基线的构架，它是通过处理构架方面重要的场景得到的，这些场景通常可以显示项目的最大技术风险。
  - 制作产品质量构件的演进式原型，也可能同时制作一个或多个可放弃的探索性原型，以减小特定风险，例如：
    - 设计/需求折衷
    - 构件复用
    - 产品可行性或向客户和最终用户进行演示。
  - 证明已建立基线的构架将在适当时间、以合理的成本支持系统需求。
  - 建立支持环境。

12.4.2 细化阶段

核心活动

- 快速确定构架、确认构架并为构架建立基线。
- 根据此阶段获得的新信息改进前景，对推动构架和计划决策的最关键用例建立可靠的了解。
- 为构建阶段创建详细的迭代计划并为其建立基线。
- 改进开发案例，定位开发环境，包括流程和支持构建团队所需的工具和自动化支持。
- 改进构架并选择构件。 评估潜在构件，充分了解自制/外购/复用决策，以便有把握地确定构建阶段的成本和进度。集成了所选构架构件，并按主要场景进行了评估。通过这些活动得到的经验有可能导致重新设计构架、考虑替代设计或重新考虑需求。

12.4.2 细化阶段

- 里程碑：生命周期构架
  - 生命周期构架里程碑为系统构架建立管理基线，并使项目团队能够在构建阶段调整规模。
  - 精化阶段末是第二个重要的项目里程碑，即生命周期构架里程碑。此时，您检查详细的系统目标和规模、选择的构架以及主要风险的解决方案。

12.4.2 细化阶段

- 评估标准
  - 产品前景和需求是稳定的。
  - 构架是稳定的。
  - 可执行原型表明已经找到了主要的风险元素，并且得到妥善解决。
  - 构建阶段的迭代计划足够详细和真实，可以保证工作继续进行。
  - 构建阶段的迭代计划由可靠的估算支持。
  - 所有客户方人员一致认为，如果在当前构架环境中执行当前计划来开发完整的系统，则当前的前景可以实现。
  - 实际的资源耗费与计划的耗费相比是可以接受的。
- 如果项目无法达到该里程碑，则它可能中途失败或需要进行相当多的重新考虑。

12.4.3 构造阶段

- 主要目标包括：
  - 通过优化资源和避免不必要的报废和返工，使开发成本降到最低。
  - 快速达到足够好的质量 。
  - 快速完成有用的版本（**Alpha** 版、**Beta** 版和其他测试发布版）。
  - 完成所有所需功能的分析、开发和测试。
  - 迭代式、递增式地开发随时可以发布到用户群的完整产品。这意味着描述剩余的用例和其他需求，充实设计，完成实施，并测试软件。
  - 确定软件、场地和用户是否已经为部署应用程序作好准备。
  - 开发团队的工作实现某种程度的并行。

12.4.3 构造阶段

- 核心活动
  - 资源管理，控制和流程优化；
  - 完成构件开发并根据已定义的评估标准进行测试；
  - 根据前景的验收标准对产品发布版进行评估。

12.4.3 构造阶段

- 里程碑：最初操作性能
  - 最初操作性能里程碑确定产品是否已经可以部署到 **Beta** 测试环境。
  - 在最初操作性能里程碑，产品随时可以移交给产品化团队。此时，已开发了所有功能，并完成了所有 **Alpha** 测试（如果有测试）。除了软件之外，用户手册也已经完成，而且有对当前发布版的说明。
- 评估标准
  - 该产品发布版是否足够稳定和成熟，可部署在用户群中？
  - 是否已准备好将产品发布到用户群？
  - 实际的资源耗费与计划的相比是否仍可以接受？
- 如果项目无法达到该里程碑，产品化可能要推迟一个发布版。

12.4.4 移交阶段

- 主要目标
  - 进行 **Beta** 测试，按用户的期望确认新系统
  - **Beta** 测试和相对于正在替换的遗留系统的并行操作
  - 转换操作数据库
  - 培训用户和维护人员
  - 市场营销、进行分发和向销售人员进行新产品介绍
  - 与部署相关的工程，例如接入、商业包装和生产、销售介绍、现场人员培训
  - 调整活动，如进行调试、性能或可用性的增强
  - 根据产品的完整前景和验收标准，对部署基线进行的评估
  - 实现用户的自我支持能力
  - 在用户之间达成共识，即部署基线已完成
  - 在用户之间达成共识，即部署基线与前景的评估标准一致

12.4.4 移交阶段

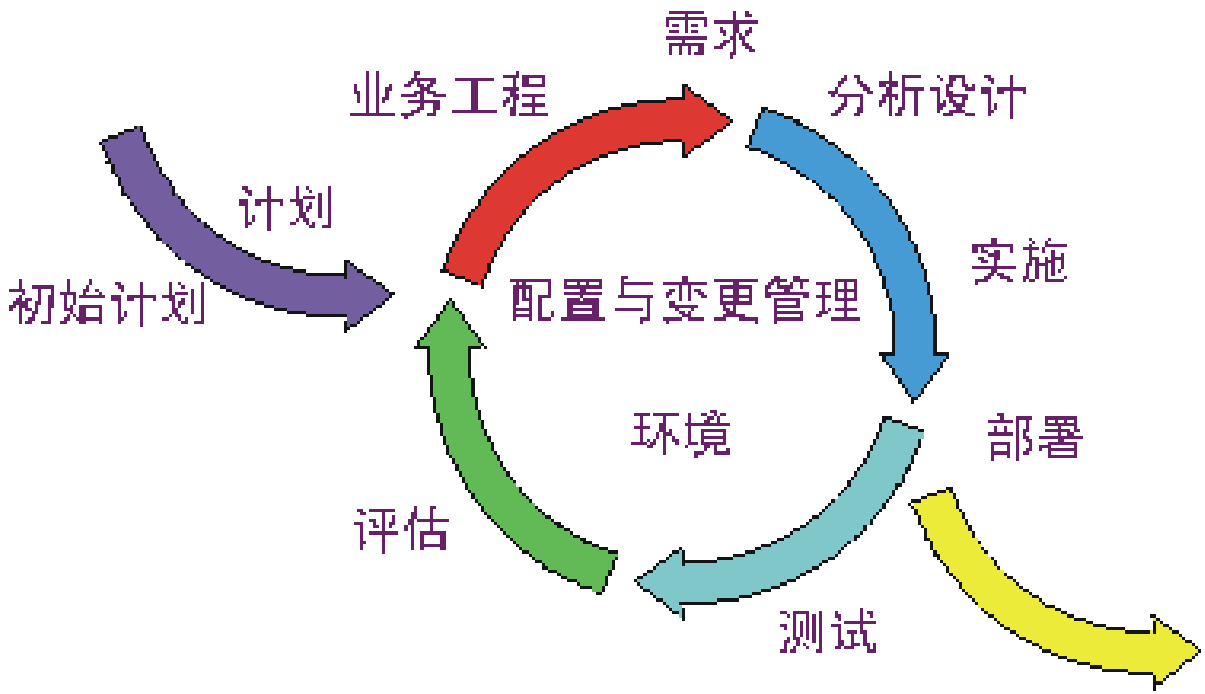
- 核心活动
  - 执行部署计划
  - 对最终用户支持材料定稿
  - 在开发现场测试可交付产品
  - 制作产品发布版
  - 获得用户反馈
  - 基于反馈调整产品
  - 使最终用户可以使用产品

12.4.4 移交阶段

- 里程碑：产品发布
  - 产品化阶段末是第四个重要的项目里程碑，即产品发布里程碑。此时，您确定是否达到目标，以及是否应该开始另一个开发周期。有时候，该里程碑可能与下一周期的先启阶段末重合。产品发布里程碑是项目验收复审成功完成的结果。
- 评估标准
  - 用户是否满意？
  - 实际的资源耗费与计划的耗费相比是否可以接受？
- 在产品发布里程碑处，发布后的维护周期同时开始。这涉及开始一个新的周期，或某个其他的维护发布版。

12.5 迭代

- 每个阶段有若干次迭代（**Iteration**），每次迭代是一个完整开发过程，每个阶段结束前有一个里程碑（**Milestone**）评估该阶段的工作。



12.5 迭代

阶段

核心工作流程

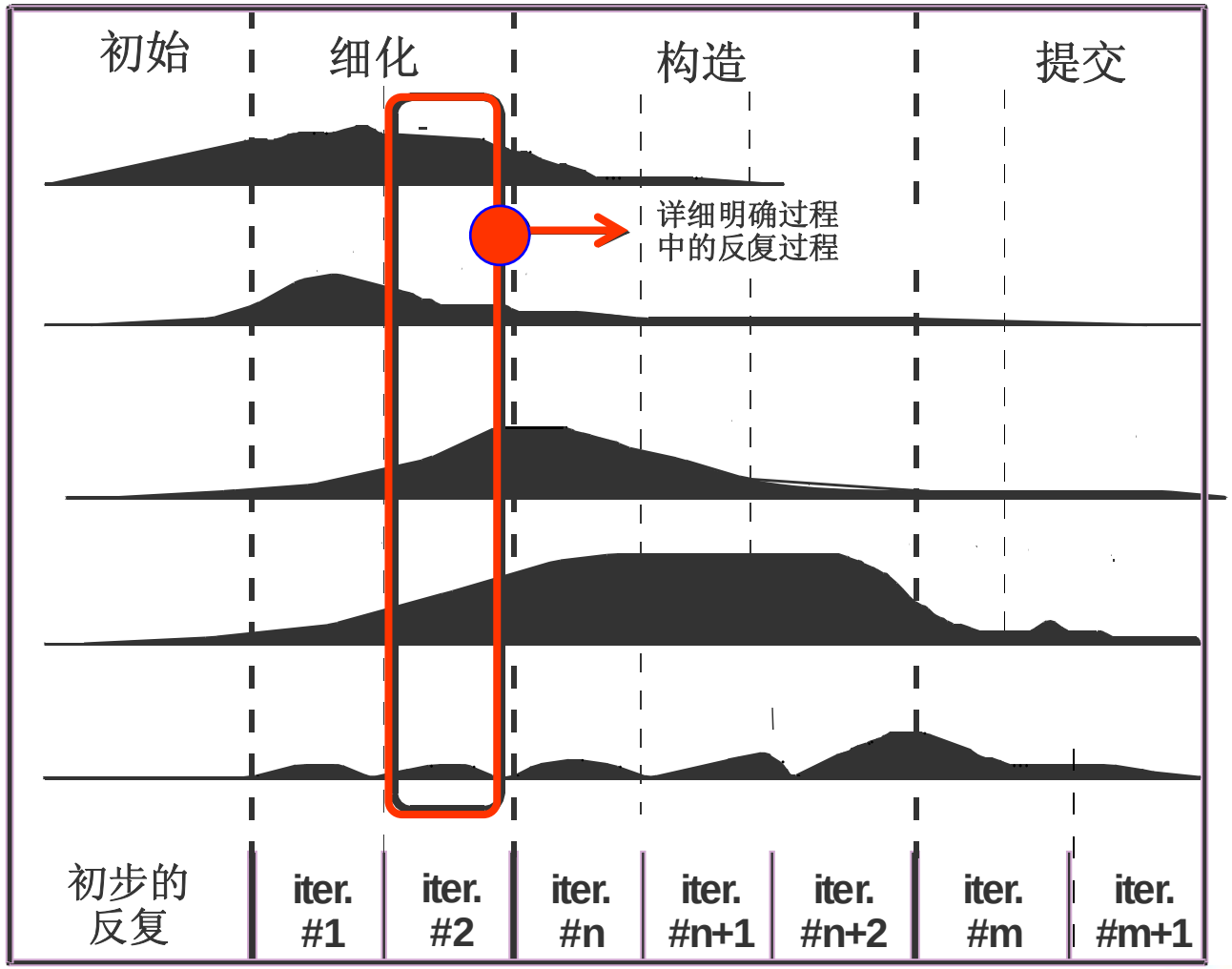
需求

分析

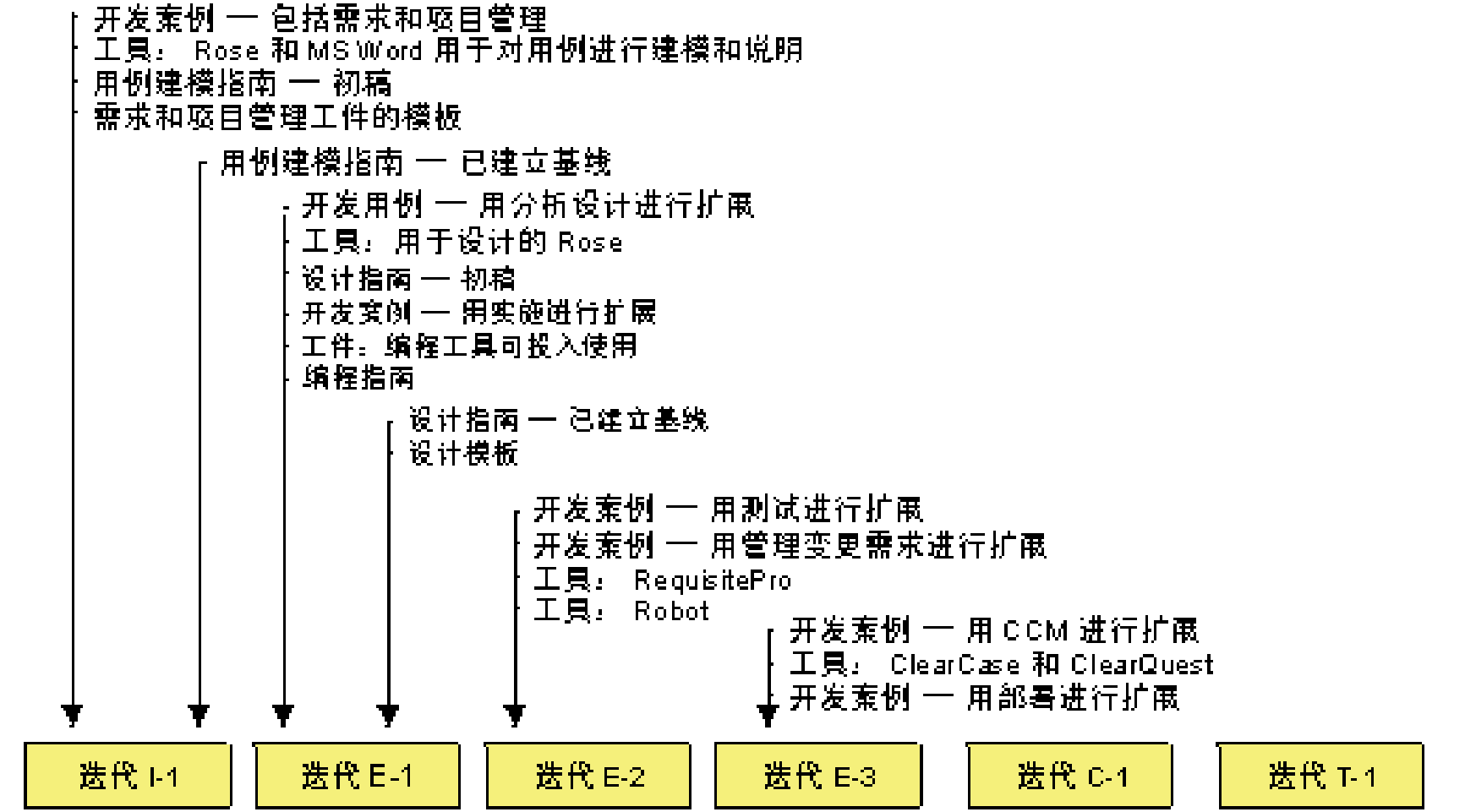
设计

开发

测试



12.5 迭代



12.6 工具

- **Rational Unified Process** 中的许多活动是由软件工程工具支持的。
- 工具向导详细说明了如何使用 **Rational** 软件工具来支持特定的步骤和活动。
- 提供以下向导：
  - **Rational AnalystStudio**　　**Rational ClearCase**
  - **Rational ClearQuest**　　**Rational RequisitePro**
  - **Rational Rose**　　**Rational PerformanceStudio**
  - **Rational Purify**　　**Rational PureCoverage**
  - **Rational Quantify**　　**Rational SoDA for Word**
  - **Rational Unified Process**
  - **Rational TeamTest/TestStudio** 的功能部件：
    - **Robot**　　**TestManager**
    - **TestFactory**　　**LogViewer**