Software Project Management (4)

Du Yugen

ygdu@sei.ecnu.edu.cn

Chapter 4
Selection of An Appropriate Project Approach

OBJECTIVES

When you have completed this chapter you will be able to:

Take account of the characteristics of the system to be developed when planning a project;考虑待开发系统的特征

Select an appropriate process model;选择合适的过程模型

Make best use of the waterfall process model where appropriate;最佳利用瀑布过程模型

Reduce risks by the creation of appropriate prototypes;创建合适的原型来降低风险

Reduce other risks by implementing the project in increments.用增量方式推行项目以降低其他风险

把通过敏捷开发(Agile Development)可以消除的那些不必要的组织级障碍情形加以标识

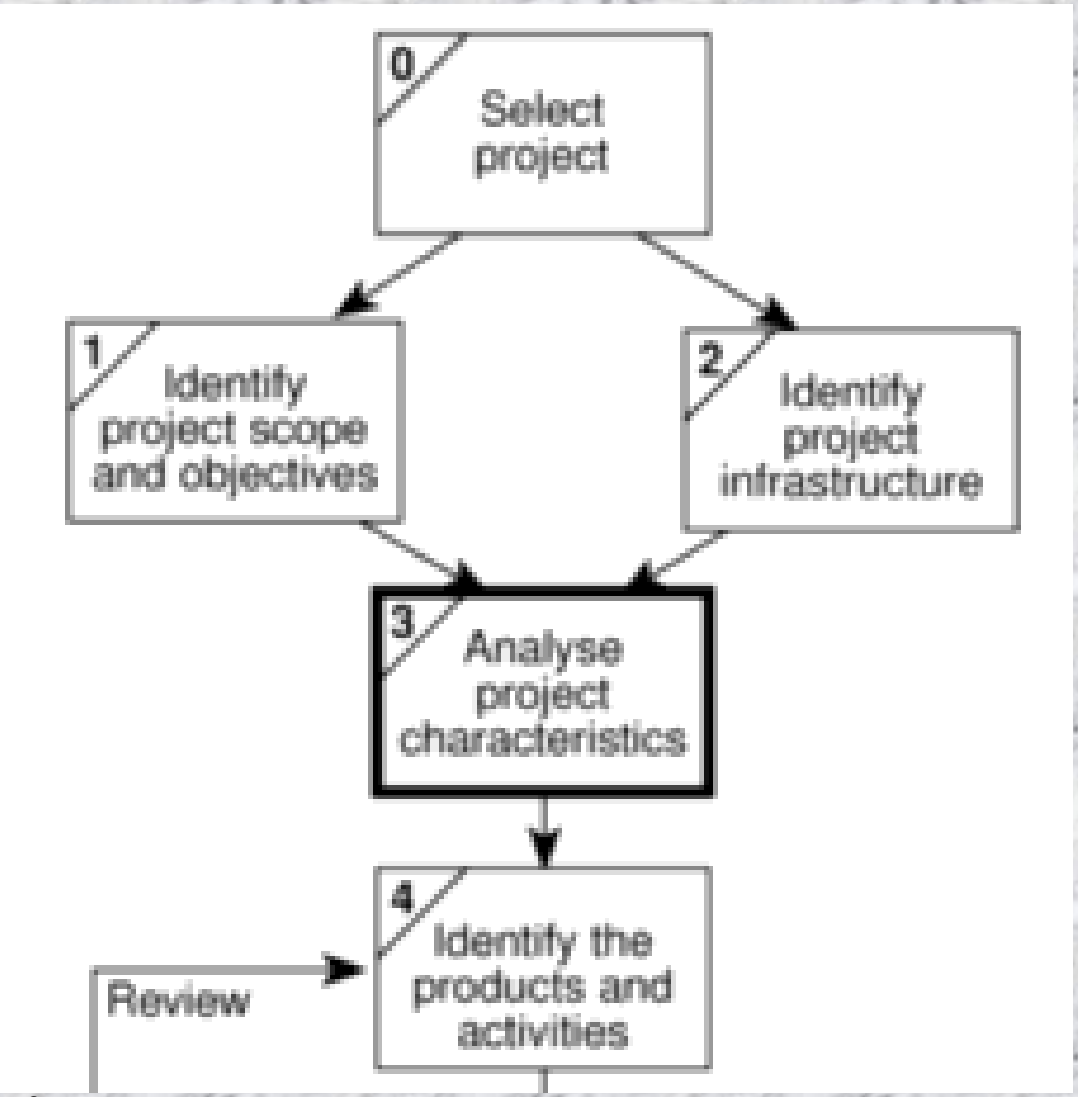
4.1 introduction

内部开发的软件项目的特征

- 项目组和用户属于同一个组织
- 在考虑做的项目嵌入(slot)到现存计算机系统的组合中
- 要使用的方法和技术不是由项目经理来选择的,而是由局部标准(Local standard)规定的

但是软件公司为外部客户开发的项目,必须根据每个项目来决定使用的方法和技术。这种决策过程有人已经称为“技术策划”（technical planning），尽管这里使用术语“项目分析”（project analysis）

The relevant part of the step wise approach is Step 3



4.2 choosing technologies选择技术

- **与产品和活动一样,选择的技术将影响一个项目以下方面:**
  - **开发人员的培训需求**
  - **要招聘的人员的类型**
  - **开发环境——硬件和软件**
  - **系统维护安排**

步骤：

- 分析项目是目标驱动的还是产品驱动的
- 分析项目其他特征
  - 面向数据(data-oriented),如IS,还是面向过程控制(process-oriented),如ES
  - 通用工具，如EXCEL，还是专用工具，航空公司座位预定系统
  - 是否涉及需要专用工具支持的专门技术
    - 如并发处理
    - 基于知识的系统,如专家系统
  - 是否有特殊的安全性要求
    - 如系统故障危及人身安全的
  - 系统所需软硬件环境特性是什么

练习

- **对下列系统进行分类**
  1. **工资支付系统**
  2. **饮料灌装企业的控制系统**
  3. **供水企业管理对企业供水计划的系统**
  4. **支持项目管理的软件**
  5. **供律师查询法律条文的系统**

参考答案

1. 面向数据或特定领域的应用系统
2. 包含嵌入式软件的过程控制或者工业系统
3. 使用图形的信息系统
4. 通用信息系统软件包
5. 信息收集的通用软件包

识别高层的项目风险

- **产品不确定性：系统需求理解的准确性。用户在开始时有可能对系统应该什么样都无法确定。在某些环境中，精确而有效的需求描述可能迅速变得过时。**
- **过程不确定性：在项目开始时需要选择方法或过程模型，或者一种新的工具，任何对原先采用的开发方法的变化都将引入不确定性**
- **资源不确定性：项目进行中资源的数量可能发生变化**

练习

- **识别学院工资系统中的风险**
  - **财务和人事职员之间的矛盾**
  - **人事职员对系统不接受**
  - **缺少运行该系统的经验**
  - **缺少管理系统的计算机专业人员**
  - **需求变化**

选择方法

- **考虑用户关于实现的需求**
  - **用户可能在合同中限定了有关实现方面的方法。例如，规定了企业必须具有相应的CMM等级，或者通过了ISO9000方法**
- **选择通用的生命周期方法**
  - **控制系统：一般为实时系统，比如实时系统的并发处理会用到象Petri网这样一来的技术**
  - **信息系统：**
  - **通用工具**
  - **专用工具**
  - **硬件环境**
  - **安全相关的系统**
  - **不准确的需求（imprecise requirement)**

技术计划内容清单

技术计划可能包括：

1.约束条件的介绍和总结

a）待开发系统的特征

b）项目的风险和不确定性

c）关系到实行的用户需求

2.推荐的方法

a）选择方法或过程模型

b）开发的方法

c）需要的软件工具

d）用到的软硬件环境

3.实现

a）要求的开发环境

b）要求的维护环境

c）要求的培训

4.牵涉的问题

a）项目的产品和活动-这些会影响项目进度和总的项目效果

b）财务的-这个报告将用来得到成本

Too often, software work follows the first law of bicycling: No matter where you're going, it's uphill and against the wind

通常地，软件工作遵守骑车第一法则：无论你往哪里，都是上坡且逆风而行。

4.4过程模型的选择process models

- **过程process：相关联的一项或多项活动开发一个软件需要选择开发策略（包括过程，方法和工具）以及通用阶段，这些策略和阶段被称为过程模型**

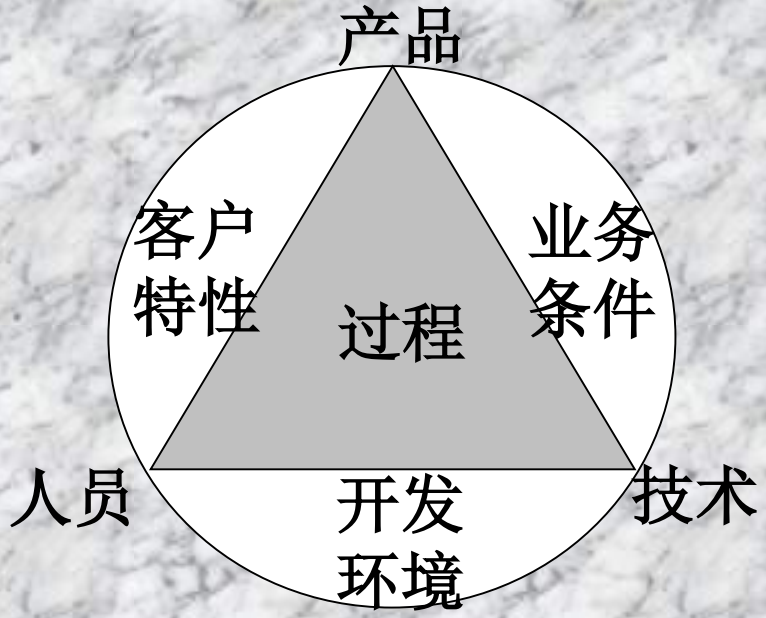
- **过程模型的选择基于项目和应用的特性，使用的工具和方法，所需要的控制方法和交付物。**

软件过程的概念

- 软件过程由关于项目的阶段、状态、方法、技术和开发、维护软件的人员以及相关对象（计划、文档、模型、编码、测试、手册等）组成。
- 一个过程定义了为达到某个确定的目标，需要什么人在什么时间以何种方式做何种工作
- 软件需要一个可用于指导顾客、用户、开发人员和项目经理等参与者的过程

软件开发过程

- 软件工程的核心是过程。产品、人员、技术通过过程关联起来。软件开发过程能够将技术集成在一起从而使软件的开发能够以一种合理而及时的方式完成。

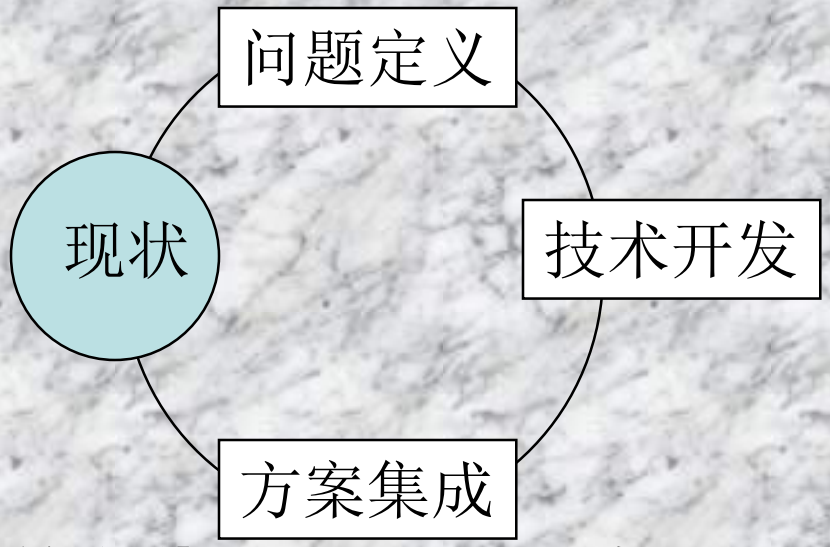


有效的软件过程

- 有效的软件过程可以提高组织的生产能力
  - 保证软件开发的基本原则的实现，辅助软件项目管理者作出明智的决定；
  - 使软件开发活动标准化，提高软件的可重用性和Team间的协作；
- 有效的软件过程也可以改善软件组织对软件的维护能力
  - 通过有效地定义如何管理需求变更，使得变更部分能够在未来的版本中恰当分配，实现平滑过渡；
  - 在具体操作和相关支持中定义如何平滑地改造软件，并且这种具体操作和支持是可实施的；不可实施的软件过程被剔除。

问题求解的一般过程

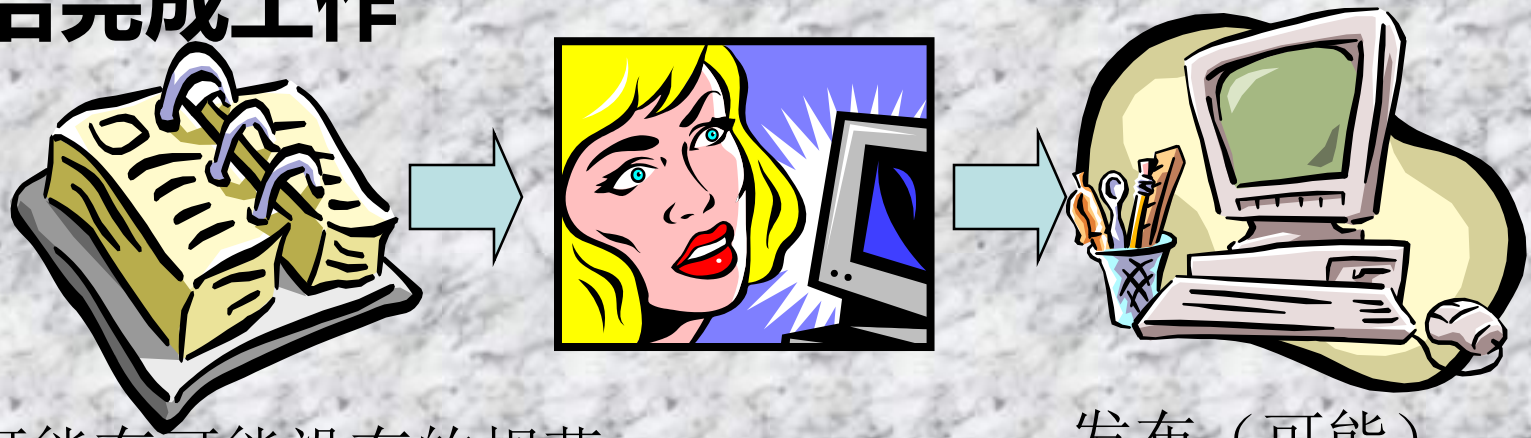
- **问题求解的一般过程**



  - **实际问题并不能简单划为四个阶段，各个阶段会在问题的不同层次上同时并存**
  - **软件开发实际上是一个“混沌”（chaos）过程（Raccoon）**

4.5编码修正模型

- **Code and Fix**
- **Code like Hell(鲁莽编码)**
- **从一个大致的想法开始工作，然后经过非正规的设计、编码、调试和测试方法，最后完成工作**



可能有可能没有的规范

发布（可能）

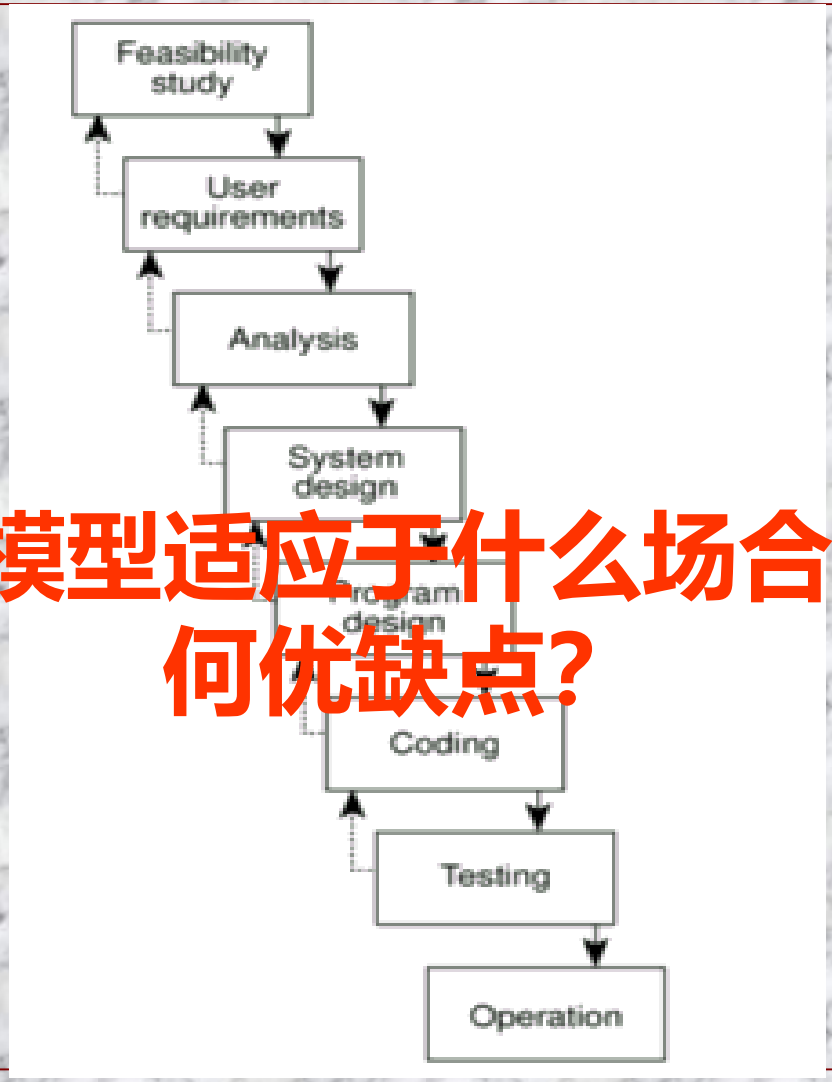
编码修正模型

- **好处：**
  - **成本可能很低**
  - **只需要很少的专业知识，任何写过程序的人都可以**
  - **对于一些非常小的、开发完后就会很快丢弃的软件可以采用**
- **对于规模稍大的项目，采用这种模型是很危险的**

4.6瀑布模型（Waterfall Model)

- 所有过程模型的祖宗
- 项目从开始到结束按照一定的顺序执行
- 瀑布模型是文档驱动的，各个阶段不连续也不交叉

瀑布模型



瀑布模型适应于什么场合？有何优缺点?

瀑布模型适合的场合

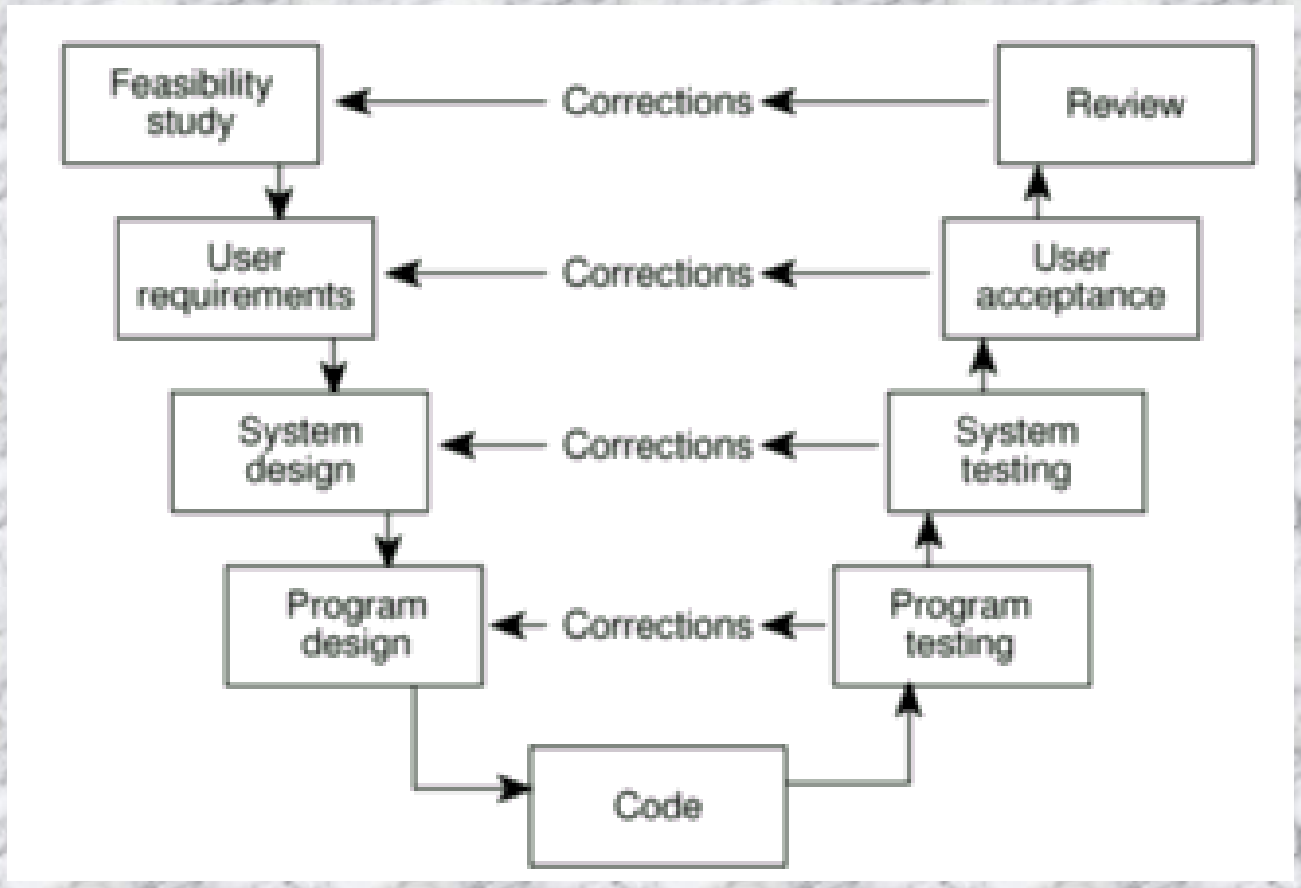
- 当有一个稳定的产品定义和很容易被理解的技术解决方案时，纯瀑布模型特别合适
- 当你对一个定义得很好的版本进行维护或将一个产品移植到一个新的平台上，瀑布模型也特别合适。
- 纯瀑布模型能够降低管理费用，因为你可以预先完成所有计划。
- 对于那些容易理解但很复杂的项目，采用纯瀑布模型比较合适，因为可以用顺序方法处理问题。
- 在质量需求高于成本需求和进度需求的时候，它尤为出色。
- 当开发队伍的技术力量比较弱或者缺乏经验时，瀑布模型更为适合。

瀑布模型的缺点

- **纯瀑布模型的缺点是在项目开始的时候，在设计工作完成前和代码写出来前，很难充分描述需求。**
- **瀑布模型最主要的问题是缺乏灵活性。必须在项目开始前说明全部需求。但这恰恰是非常困难的。**

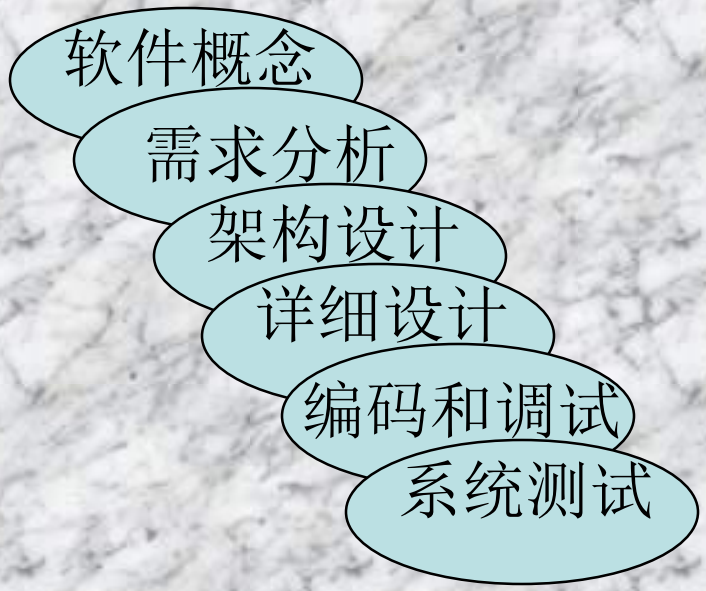
4.7瀑布模型变种：V型模型

- **该方法是对瀑布模型的修正，强调了验证活动**



4.8瀑布模型变种：生鱼片模型

- **把阶段重叠起来的瀑布模型**
- **起源于日本硬件开发模型（富士通—施乐）**



瀑布模型变种：生鱼片模型

- 传统的瀑布模型强调阶段之间最小的重叠，而生鱼片模型强调大幅度的重叠，即在需求分析完成之前就可以进行架构设计和部分详细设计
- 纯瀑布模型强调在任意两个阶段交接时，文档从一个团队交给另一个完全隔离的团队，但是如果一个团队完成各个阶段任务时，可以没有那么多文档。
- 问题：缺点是什么?
- 生鱼片模型因为阶段重叠，因而里程碑不明确，很难有效地进行过程跟踪和控制。

瀑布模型变种：具有子项目的瀑布模型

- **纯瀑布模型的一个问题是必须完成全部的架构设计后才能进行详细设计，但是，整个系统中有些部分可能有些特殊性，可以有自己的步骤，即将这些部分划分为为子项目。**
- **问题：该模型有何问题?**
- **这种方法的主要风险是相关性无法预料。**

瀑布模型变种：能够降低风险的瀑布模型

- **纯瀑布模型要求在开始架构设计前，必须将用户的所有需求都搞清楚，但是实际中是很困难的。**
- **可降低风险的瀑布模型是在顶端，即需求分析和架构设计阶段引入螺旋以便降低风险。**
- **在该螺旋中，先开发一个用户界面原型，采用系统情节串联图版（system storyboarding）引导用户提出需求，记录用户与系统的交互操作方式，或者采用其它需求获取方法。**

4.9螺旋模型(Spiral model)

- Spiral 模型（Boehm提出）
- 以风险为导向的生命期模型
- 从一个小范围的关键中心地带开始寻找风险因素，制定风险控制计划，并交付给下一步骤，如此迭代，每次迭代将项目扩展到一个更大的规模

Spiral model to SSADM

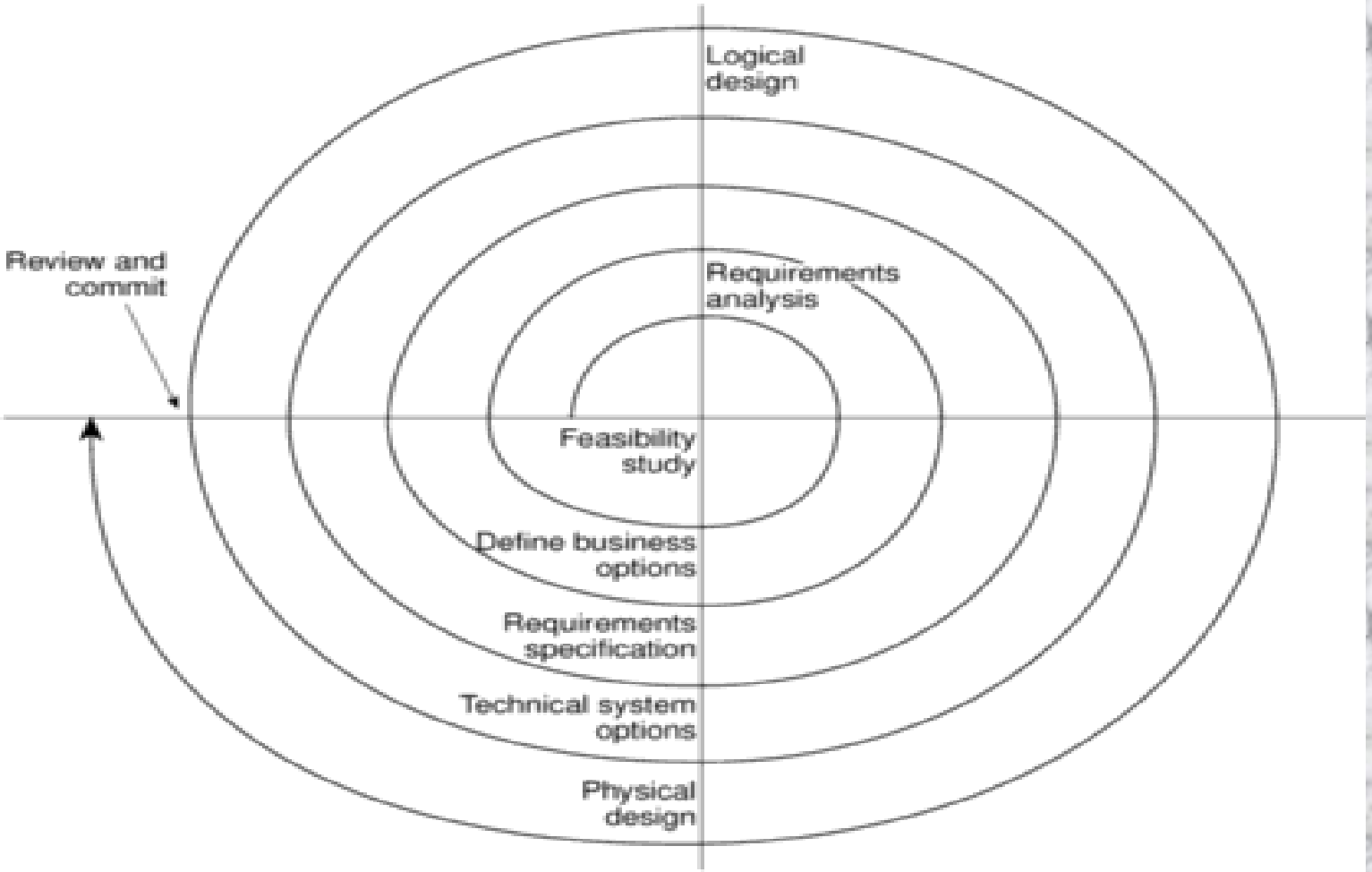
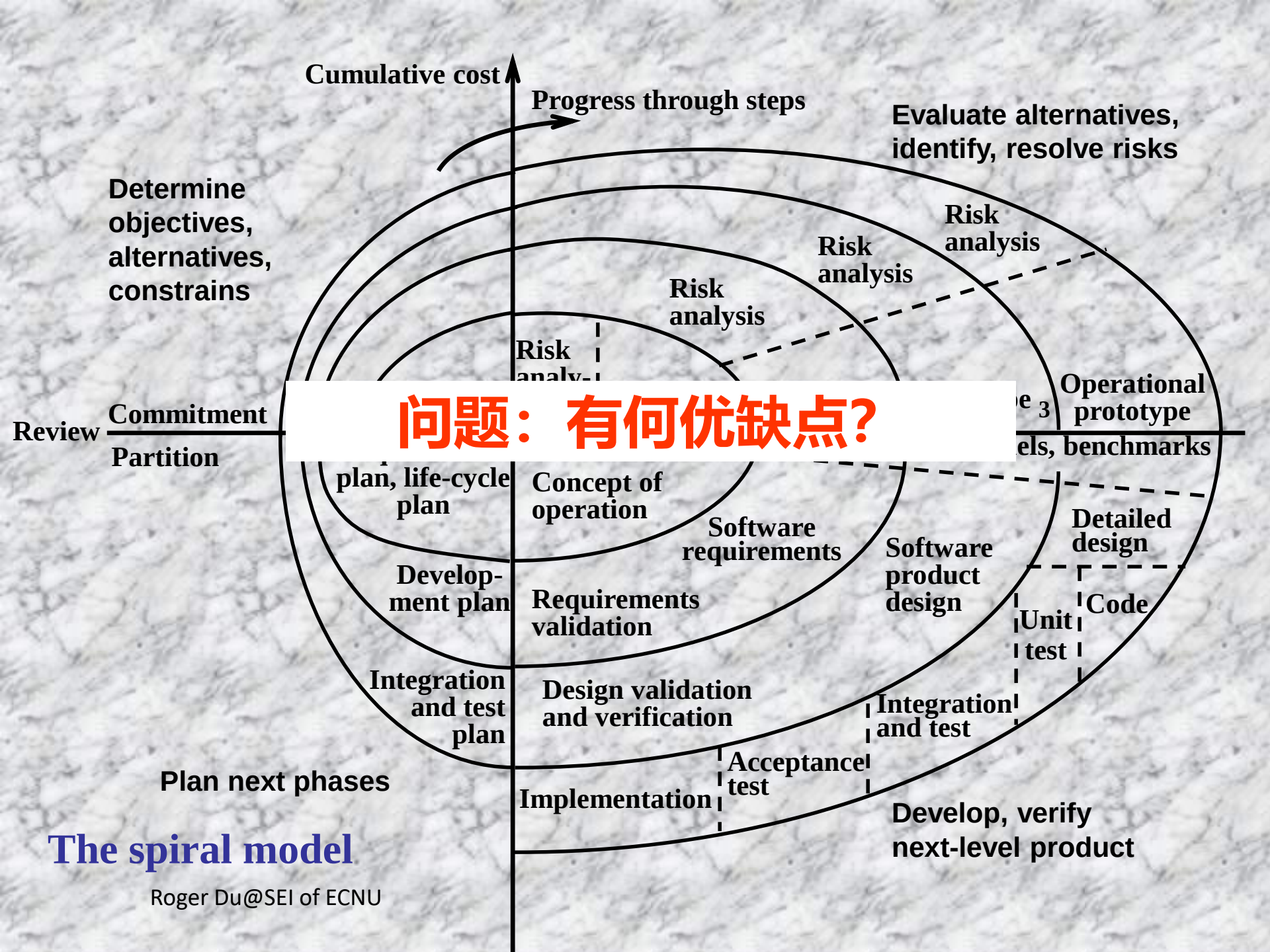


Figure 4.4 *The application of the spiral model to SSADM version 4.*



问题：有何优缺点?

The spiral model

Roger Du@SEI of ECNU

螺旋模型优缺点

- **优势：随着迭代的增加（成本的增加），风险程度随之降低**
- **缺陷：比较复杂，需要责任心，专注和管理方面的知识。**

WINWIN螺旋模型

- 在螺旋模型中，通过与客户的通信获取客户的需求，实际上，客户的需求与最终确定的需求是不一致的，客户与开发者之间需要进行协商，最终达到双赢的局面。
- Boehm提出的WINWIN螺旋模型中，客户与开发者之间需要
  - 识别系统或子系统的干系人（stakeholders）
  - 确定干系人的“win conditions”
  - 对这些条件进行协商获得互赢条件

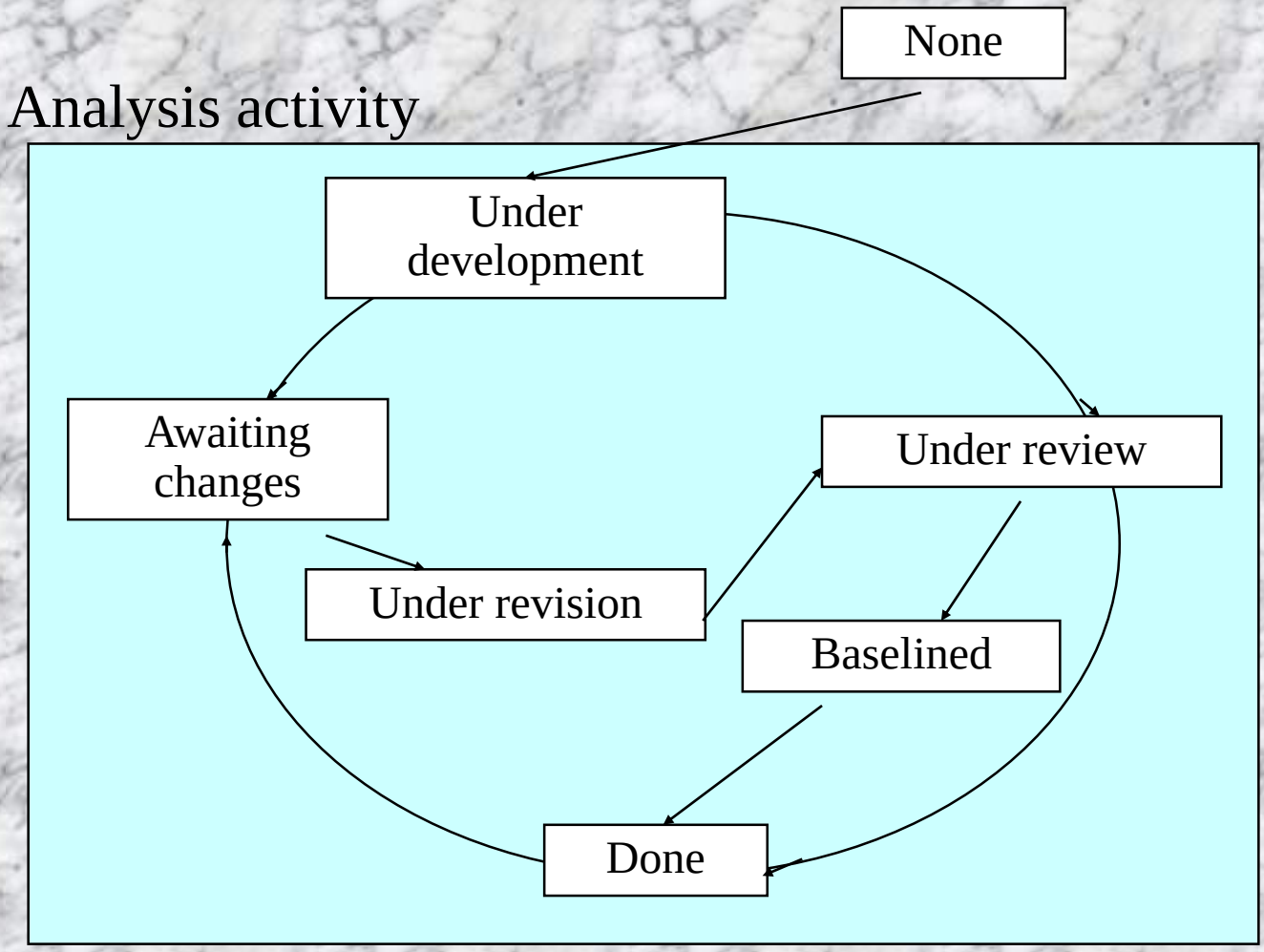
WINWIN螺旋模型

- WINWIN螺旋模型还引入了三个过程的里程碑，被称为定位点（Anchor points）
  - 生命周期目标（life cycle objectives）定义了每个主要活动的目标
  - 生命周期体系结构（life cycle architecture）定义了系统和软件的体系结构目标
  - 初步操作能力（initial operational capability）定义了软件安装，发布的目标。

4.10并行开发模型concurrent development model

- 并行开发模型（concurrent development model）又被称为并行工程（concurrent engineering）（By Davis and Sitaram）
- 软件开发中的所有活动可能同时并存，但是都处于不同的状态中
- 并行开发模型定义了活动从一个状态转化为另一个状态的事件

并行开发模型


None
Analysis activity
Under development
Awaiting changes
Under review
Under revision
Baselined
Done

并行开发模型

- **并行过程模型经常被用于开发C/S系统。该系统的活动可以被分为系统维和部件维。系统维包含了设计，装配和使用三个活动，而部件维包含了设计和实现两个活动。并发性表现在两个方面：**
  - **系统和部件的活动同时发生**
  - **各个部件可以并行设计和开发**

4.11原型法prototype

- 原型法
- 原型是项目系统中的一个方面或者多个方面的工作模型。
  - 抛弃型原型：用于试验某些概念，试验完系统将无用处
  - 进化型原型：原型系统不断被开发和被修正，最终它变为一个真正的系统。

原型法的好处

- 从实践中学习(Learning by doing)
- 改善通信
- 改善用户参与
- 使部分已知的需求清晰化
- 验证描述的一致性和完整性
- 可能可以减少文档
- 减少了维护成本
- 特征约束（利用工具构造原型可以将某些特性落到实处，而非在纸上写的那样容易失误）
- 试验是否能产生期待的结果

原型法的缺点

- 用户有时误解了原型的角色，例如他们可能误解原形应该和真实系统一样可靠
- 缺少项目标准，进化原型法有点像编码修正
- 缺少控制，由于用户可能不断提出新要求，因而原型迭代的周期很难控制
- 额外的花费：研究结果表明构造一个原型可能需要10%额外花费
- 运行效率可能会受影响
- 原型法要求开发者与用户密切接触，有时这是不可能的。例如外包软件。

从另外的角度看待原型

- 从中学到什么?
  - 学生经常会做一些软件作业，这些作业被称为原型，
  - 问题：这些原型和软件系统原型是否相同?
  - 但是作为一个原型必须：描述他们希望从中学到的东西，规划原型评价的方法，报告从原型中真正学到的内容。
  - 在不同的阶段，原型具有不同的作用。
- 原型起到作用的程度
  - 实物模型（Mock-ups）
  - 仿真交互
  - 部分模型：水平，垂直（某些特性构造详细的原型）

哪些要进行原型化

- **人机界面**
- **系统的功能**

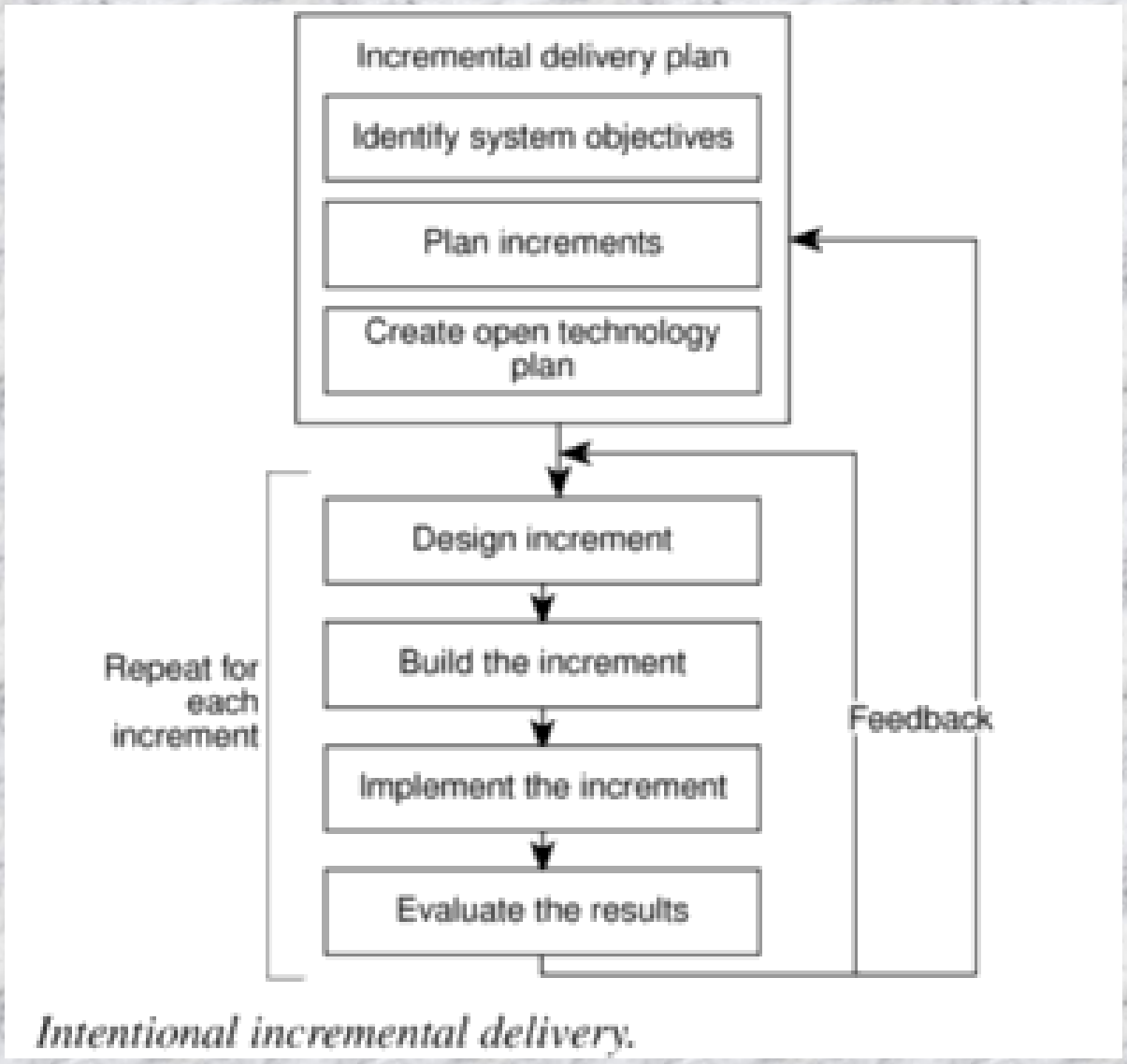
练习：何时引入原型系统

- 保险公司的经理需要通过个人计算机上的一个系统来访问管理信息。该系统价格必须合适。很多人怀疑是否经理真需要使用该系统。
  - 可行性研究阶段，采用实物模型的方法
- 支持客户销售人员通过电话回答有关客户询问汽车保险价格的系统
  - 设计用户对话界面时
- 保险公司考虑实施一个基于MS Access的电话销售系统，他们不知道Access是否能够开发出相应界面的系统并具备足够快的相应时间。
  - 方案设计阶段

4.12 增量式交付incremental delivery

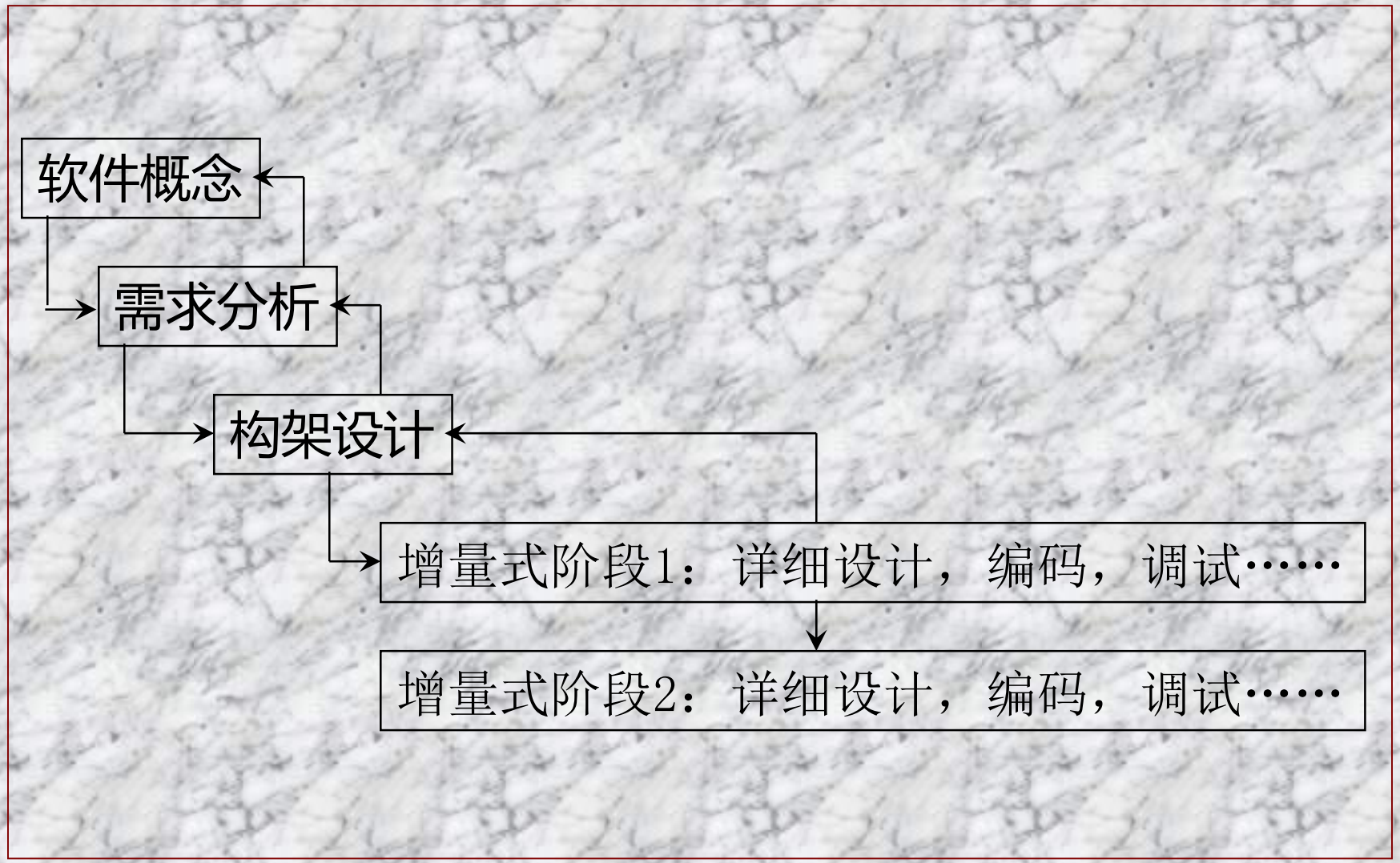
- **增量式交付持续地在确定的阶段向用户展示软件。**
- **和渐进原型不同，在增量式交付的时候，你明确地知道下一步要完成什么工作。增量式交付的特点是不会在项目结束的时候一下交付全部软件，而是在项目整个开发过程中持续不断地交付阶段性成果。**

Intentional incremental delivery



Intentional incremental delivery.

增量式交付



增量式交付的优点

- 增量式交付的优点是项目结束交付全部成果前，分阶段将有用的功能交付给用户。
- 增量式交付的主要缺点是，如果管理层面和技术层面上缺乏仔细的规划，工作就无法进行。
- 使用增量式交付的注意点是：
  - 必须确定每一增量式的交付是对用户有用的
  - 必须确保考虑了不同产品组成部分的技术依赖关系

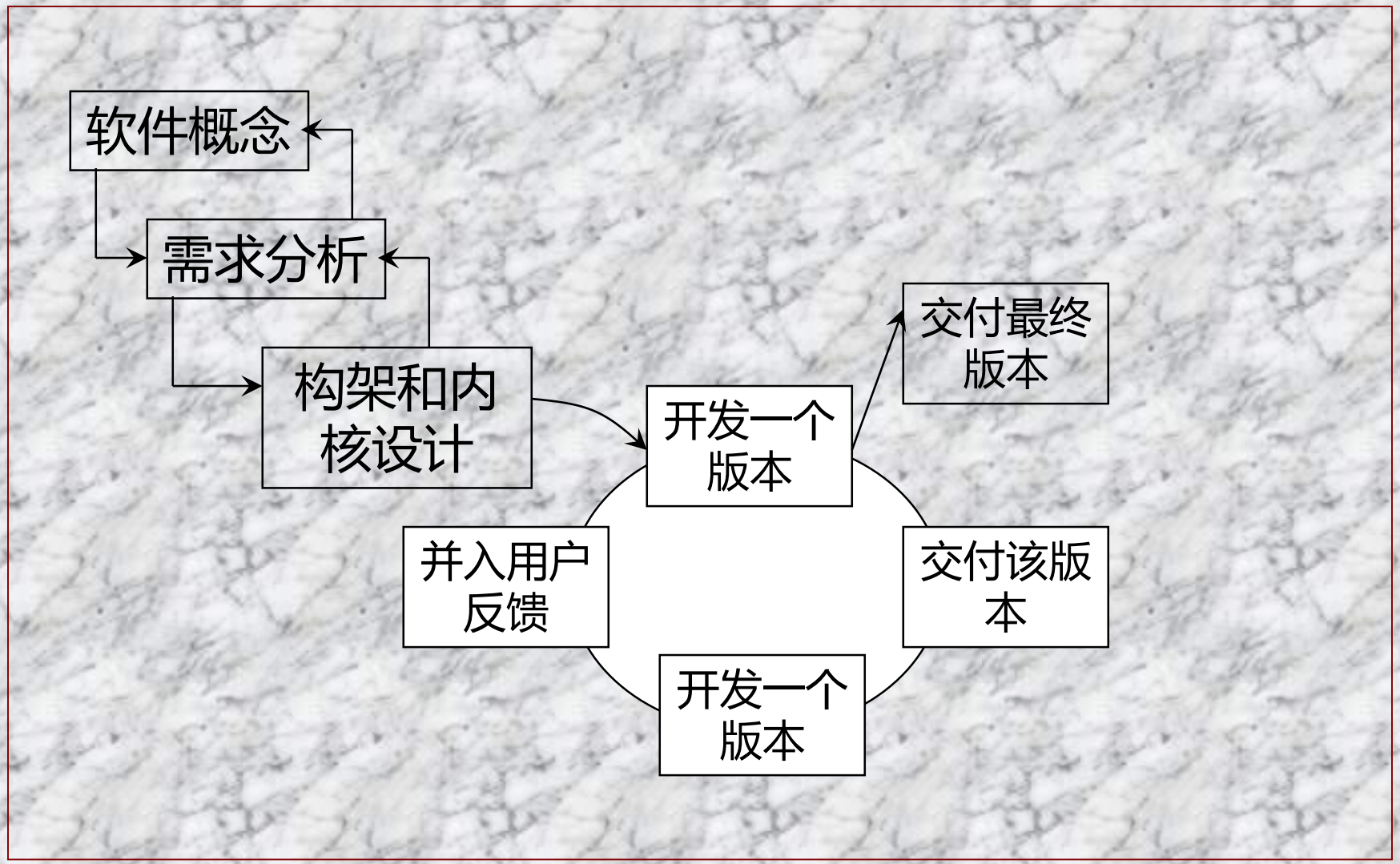
面向进度的设计

- 类似于增量式交付，但是面向进度的设计生命周期模型在开始的时候不必知道究竟能达到何目标，但是要确保最后的期限。
- 该模型的关键是要按优先级别划分系统特性并规划开发阶段，保证前面的阶段具有高优先级的特性，低特性具有低优先级别。
- 是否采用这种方法决定于你是否对系统目标具有足够的信心，如果有信心，则没必要采用这种方法。

渐进交付

- 渐进交付是一种跨越了渐进原型和增量式交付两种模型的过程模型。
- 基本过程：开发一个产品的版本，展示给用户，根据反馈改善产品。
- 如果计划满足用户的绝大部分需求，渐进交付与渐进原型差不多，如果计划满足少量的需求，渐进交付就和增量式交付差不多。
- 渐进原型，强调的是系统看得见的样子，再回来堵漏洞，渐进交付中，最初的重点是系统核心和底层系统功能。

渐进交付



确定渐进交付目标的一种方法

- 价值成本比

Table 4.1 *Ranking by value to cost ratio*

Step	*Value*	*Cost*	*Ratio*	*Rank*
Profit reports	9	1	9	(2nd)
Online database	1	9	0.11	(6th)
Ad hoc enquiry	5	5	1	(4th)
Production sequence plans	2	8	0.25	(5th)
Purchasing profit factors	9	4	2.25	(3rd)
Clerical procedures	0	7	0	(7th)
Profit based pay for managers	9	0	∞	(1st)

面向开发工具的设计

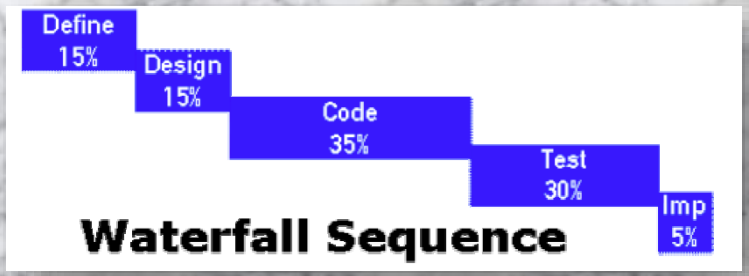
- **只在现有软件工具直接支持的情况下增强产品的功能，如果它不支持，就放弃这些功能。**
- **当时间成为主要约束时，采用该模型能够比其他模型能够更完整地实现功能。**
- **该方法的缺点是你失去了很多对产品的控制能力。**

敏捷方法（Agile Methodologies）

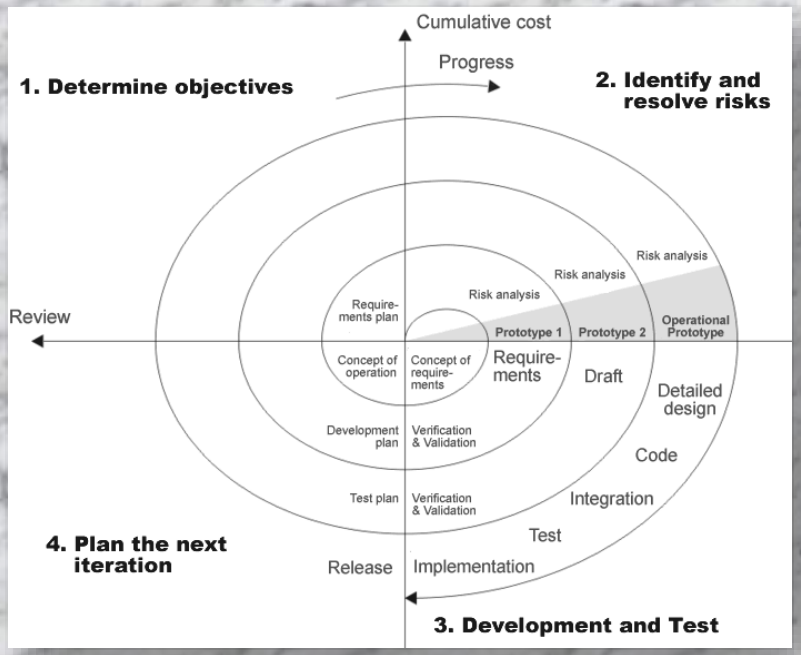
- Extreme Programming (XP)
- Scrum
- Crystal
- Dynamic Software Development Method (DSDM)
- Feature Driven Development (FDD)
- Agile Unified Process (AUP)

重量级（Heavyweight Methodologies）

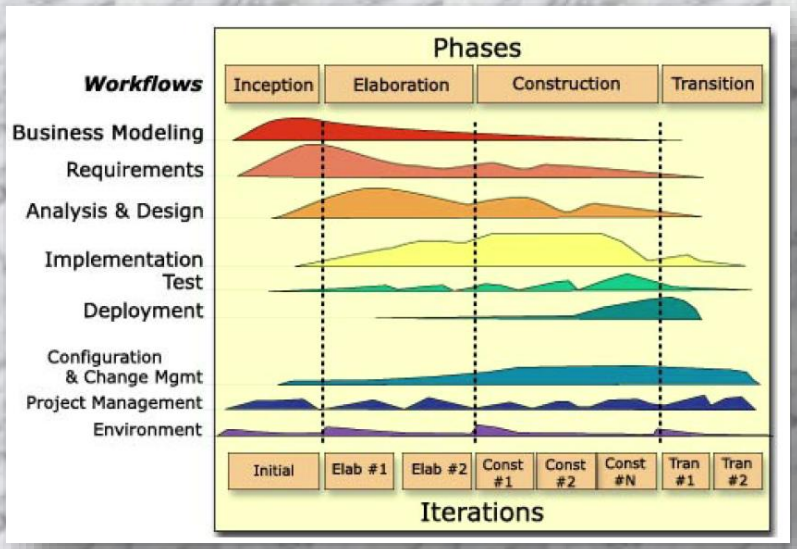
- Waterfall Process



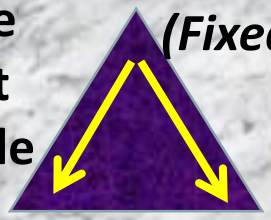
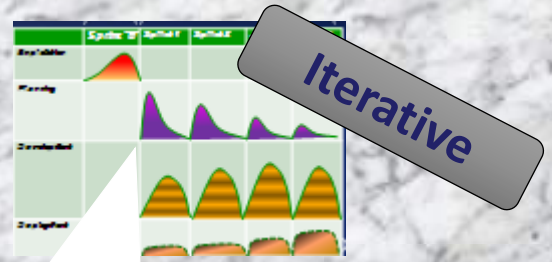
- Spiral Model



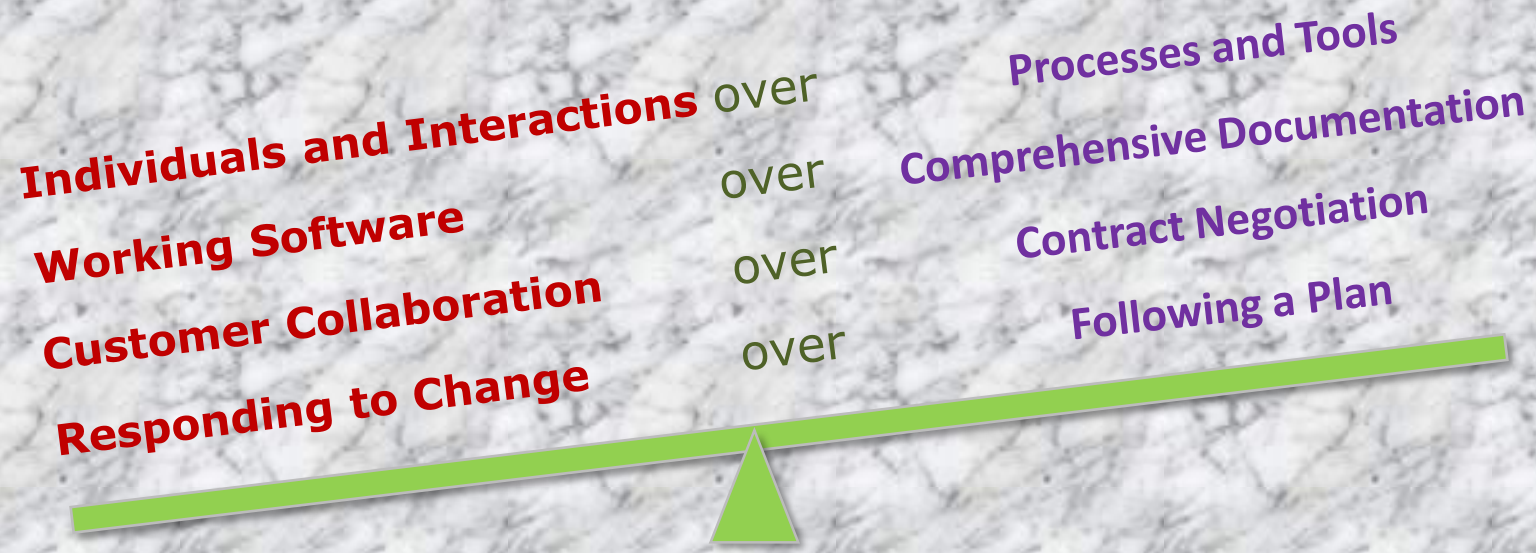
- Unified Process



敏捷与重量级（Agile vs. Heavyweight）

	Heavyweight(CMM)	Agile
Project Triple Constrain	Scope drive the budget and schedule  	Scope is driven by the budget and schedule 
Lifecycle	 	
Process	“Theoretical” Fully defined processes Software Engineering	“Empirical” Adaptive black box Software Craftsmanship

核心价值和原则（Core Value and Principle）

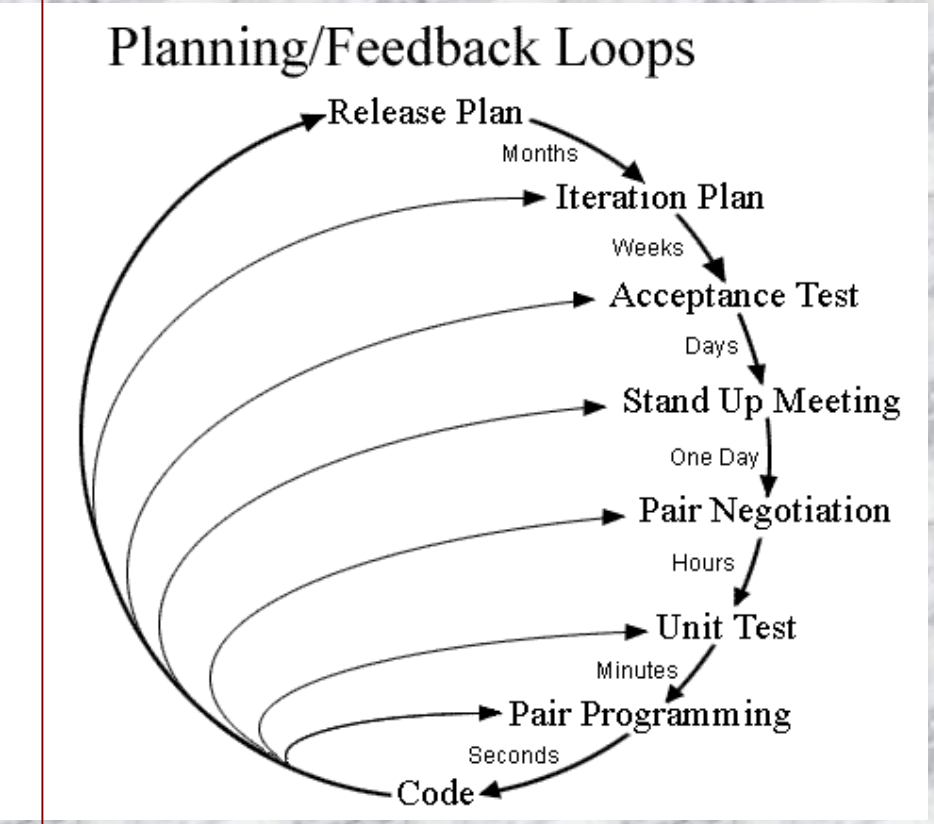


➢ *Agile Adoption Scope*

Recommend	*Not Recommend*
– *Requirement is volatile* – *Devs are responsible and positive* – *User can participate and easily engaged*	– *Team member are more than 50* – *Scope is fixed*

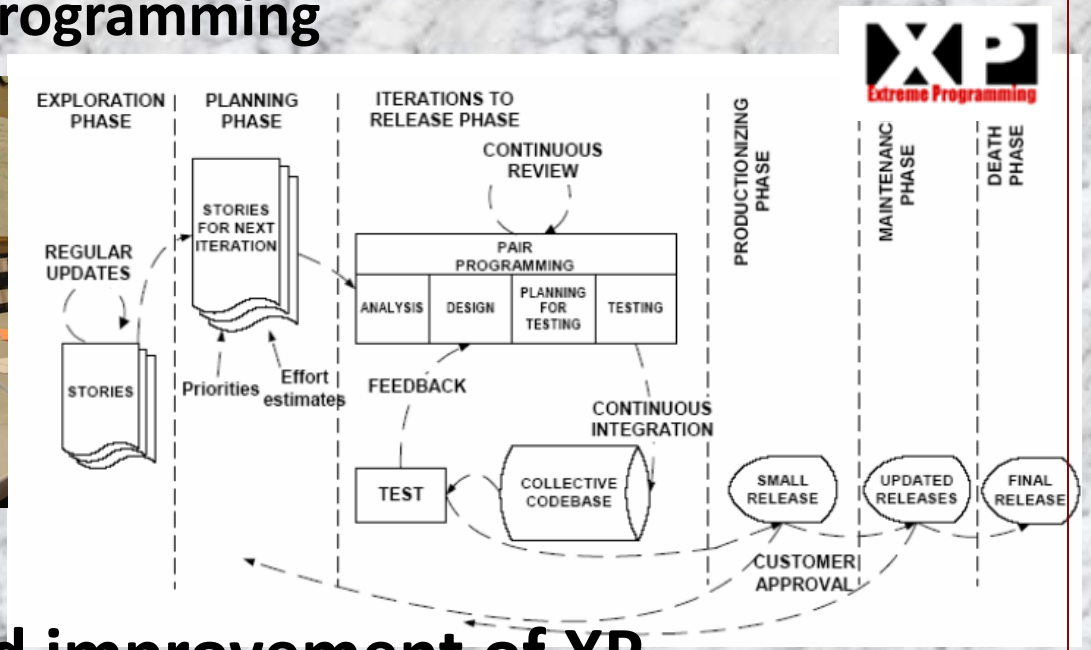
XP (Extreme Programming)

- 4 Simple Values:
  - Simplicity
  - Communication
  - feedback
  - courage
- 12 Core Practices
  - Planning Game
  - Small Releases
  - Customer Acceptance Tests
  - Simple Design
  - Pair Programming
  - Test Driven Development
  - Refactoring
  - Continuous Integration
  - Collective Code Ownership
  - Coding Standards
  - Metaphor
  - Sustainable Pace可持续的步调



XP (Extreme Programming)

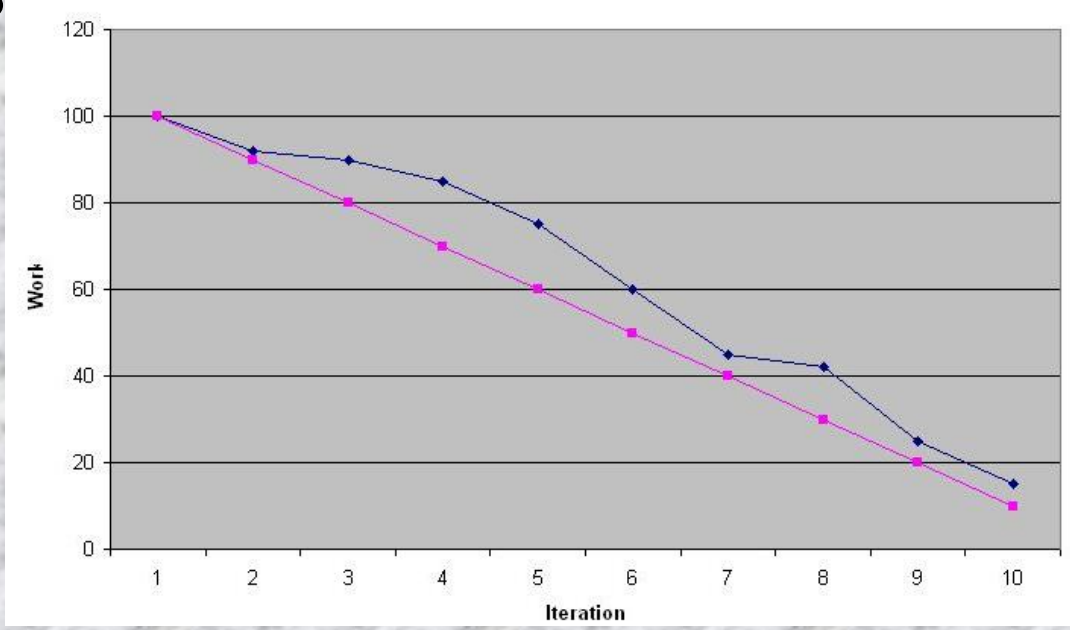
- XP development Process
  - The whole process
  - Iteration Process
  - Code Sharing Programming



  - The practice and improvement of XP

Scrum

- **Not SCRUM!**–Scrum in Rugby
- **3 Roles**
  - **Product Owner**
  - **Scrum Master**
  - **Team Member(5~9 members)**
- **3 Artifacts**
  - **Product Backlog**
  - **Sprint Backlog**
  - **Burndown Chart**



Scrum

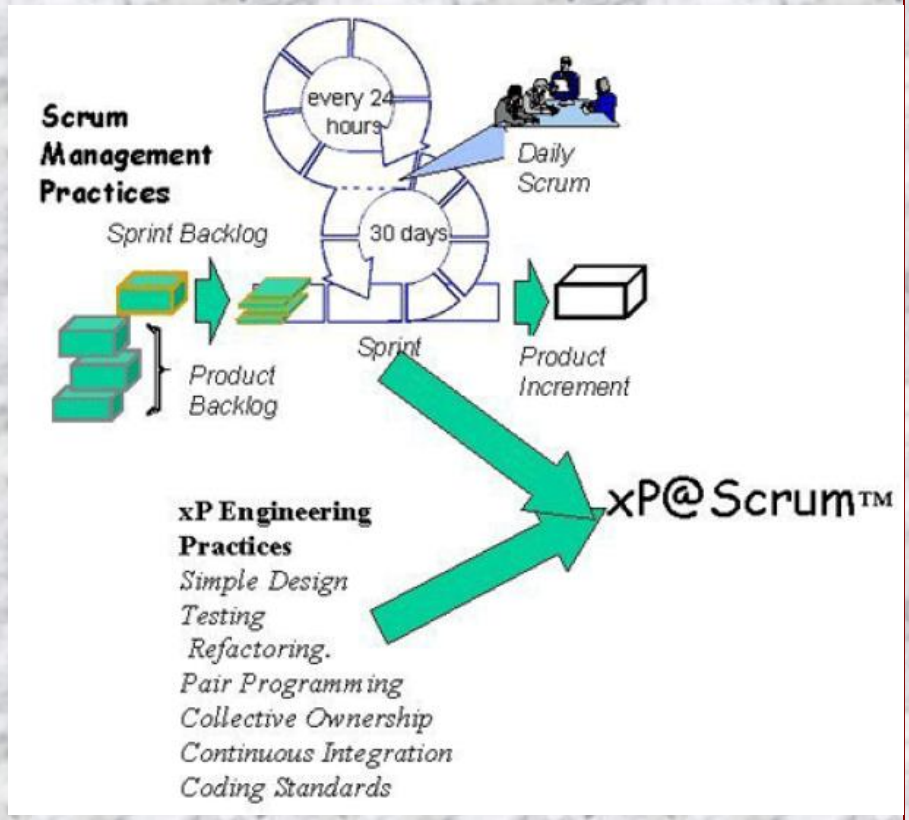
- **5 Key Ceremonies（仪式）**
  - **Sprint**
  - **Sprint Planning**
  - **Daily Scrum Meeting**
  - **Sprint Review**
  - **Sprint Retrospective（回顾）**

1. What did I do since the last Scrum meeting?
2. What do I plan on doing between now and next Scrum Meeting?
3. Do I have any roadblocks?

Scrum

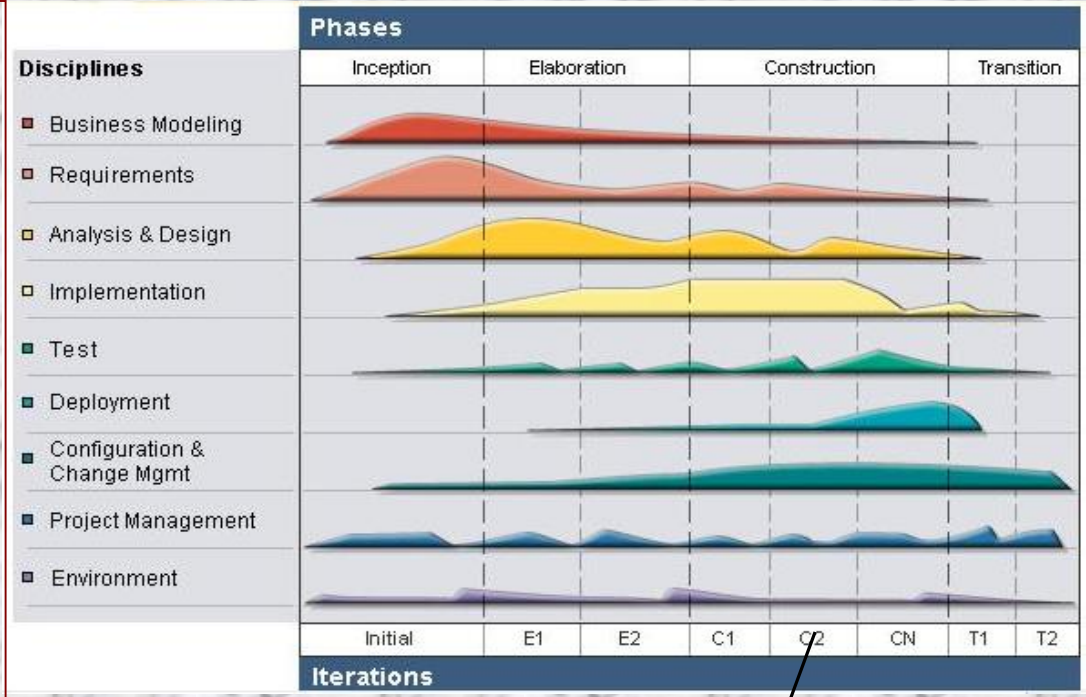
➢ **The practice and the improvement of Scrum**

- **Improve the efficiency**
  - »***6 times compare with tradition one –Capers Jones***
- **Restriction:**
  - »**not best approach for large, complex team structures**
- **Latest practice: learn from XP**
  - » **XP@ Scrum**



AUP (Agile Unified Process)

- **Unified Process (UP) & RUP**
- **4 major phase of UP project:**
  - **Inception**
  - **Elaboration**
  - **Construction**
  - **Transition**



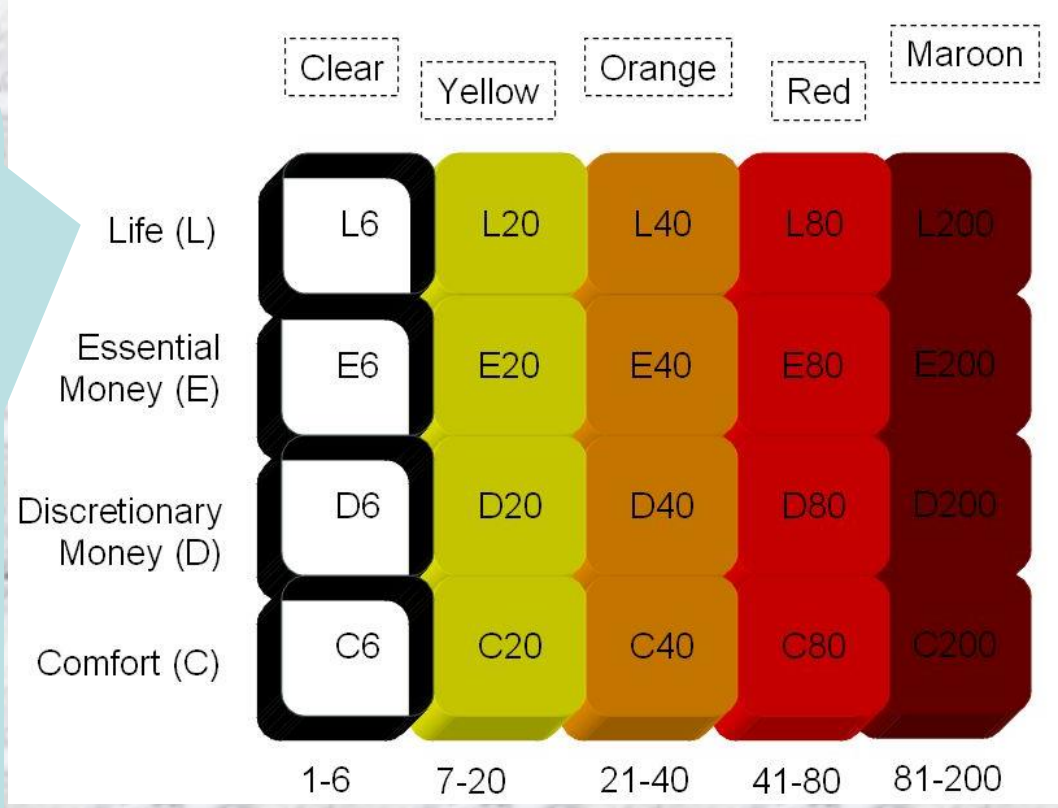
AUP is an implementation of the Unified Process which tailors it by selecting only seven disciplines (model, implementation, test, deployment, configuration management, project management, and environments).

Crystal

- Comprised of a family of methodologies:

Crystal Clear, Crystal Yellow, Crystal Orange, etc. - *Alistair Cockburn*

- As the project size increases (moves to the right of the diagram), the harder the project, and hence a more comprehensive (darker color) of Crystal is necessitated.
- As the criticality of a project increases (moves from the bottom to the top of the diagram), aspects of the methodology need to be put in place to accommodate the additional requirements, including the artifacts generated by the team; however criticality does not affect the color of Crystal used.

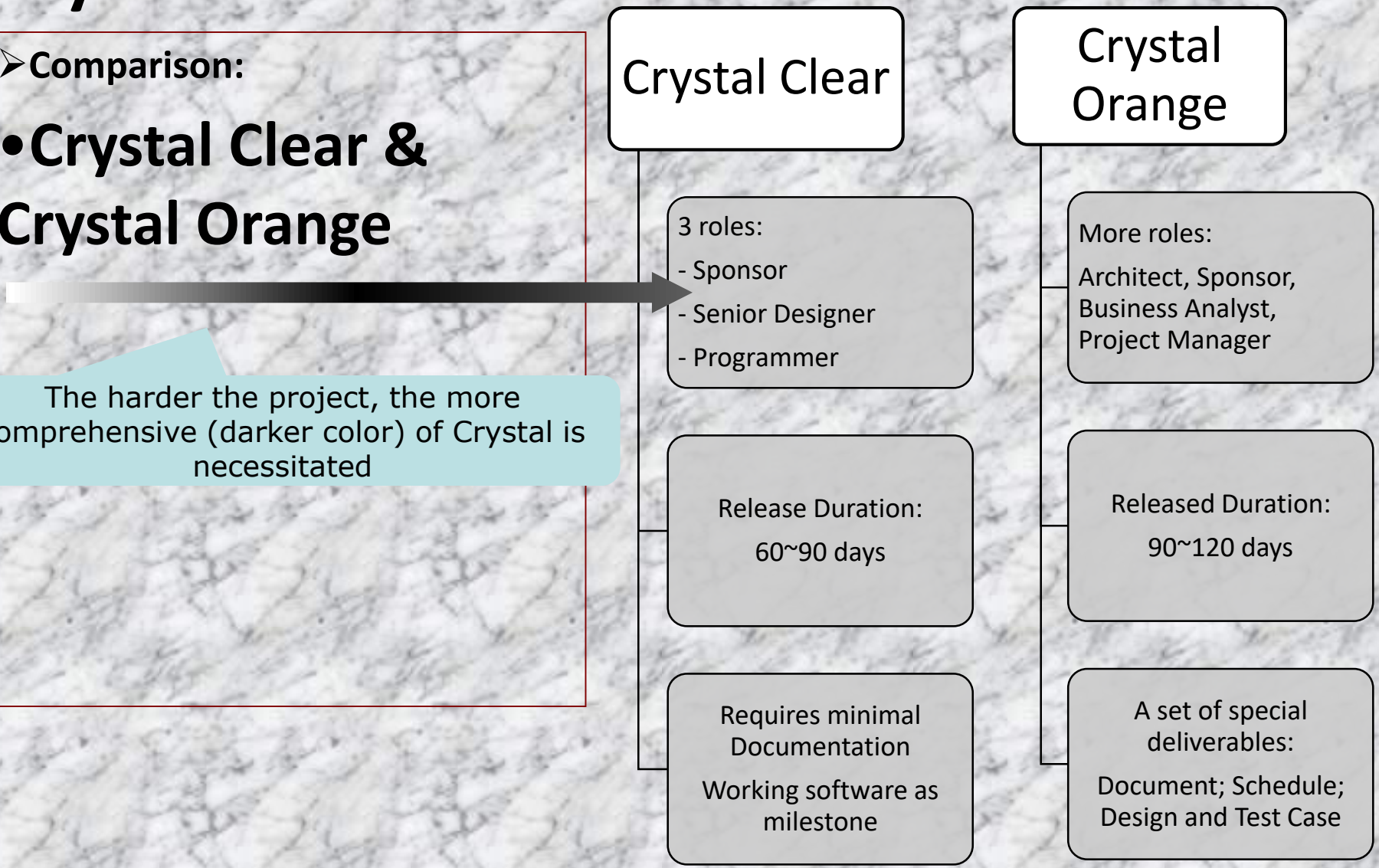


Crystal

➢**Comparison:**

- **Crystal Clear & Crystal Orange**

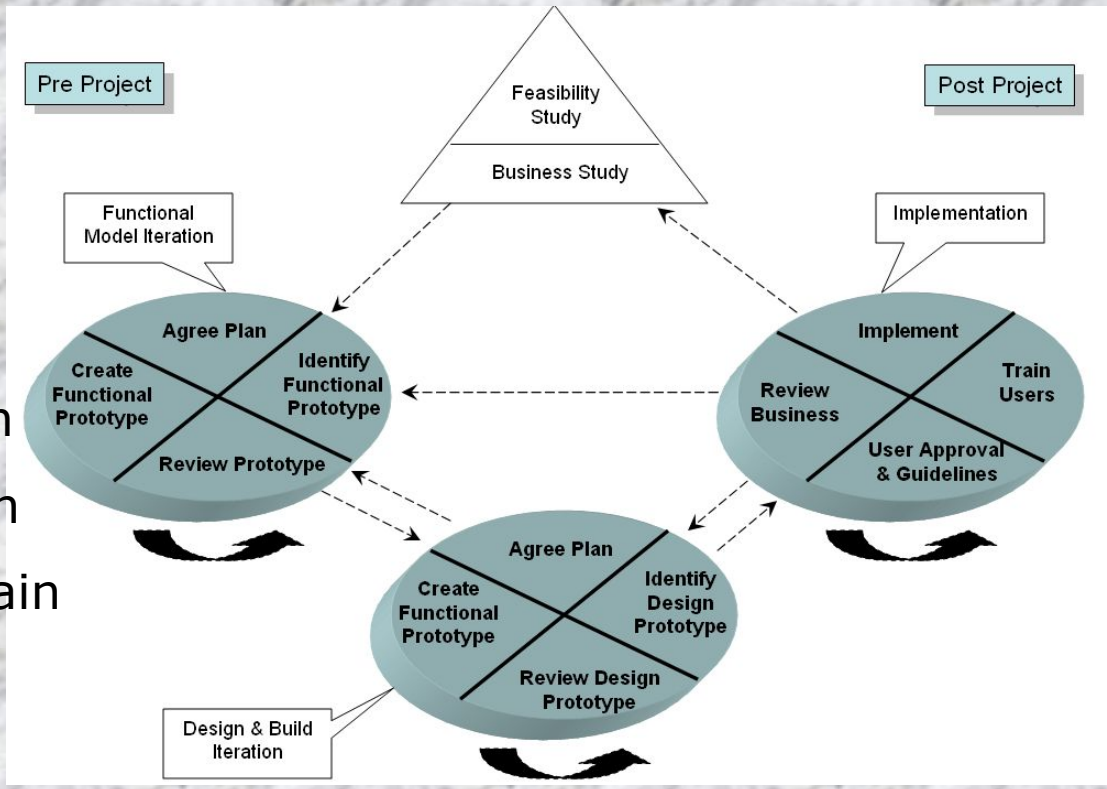
The harder the project, the more comprehensive (darker color) of Crystal is necessitated



Dynamic Systems Development Method (DSDM Atern)

➢ The DSDM Development Process

- Pre-Project
- Project Lifecycle:
  - Feasibility Study
  - Business Study
  - Functional Model Iteration
  - Design And Build Iteration
  - Implement/Deploy/Maintain
- Post-Project - Maintenance

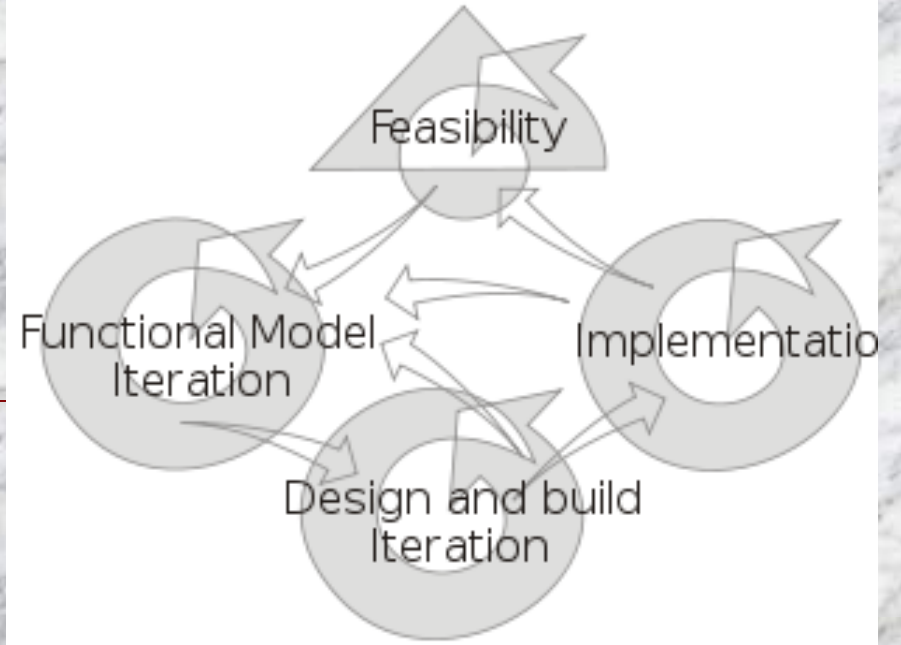


DSDM Lifecycle: A lot of Steps

Dynamic Systems Development Method (DSDM Atern)

- 5 techniques in DSDM:

  - Iterative development
  - Timeboxing
  - MoSCoW Rules
  - Facilitated workshops
  - Modeling



- **MoSCoW Rules:**

M – Must have requirements
S – Should have if at all possible
C – Could have but not critical
W - Won't have this time, but potentially later

Feature-Driven Development (FDD)

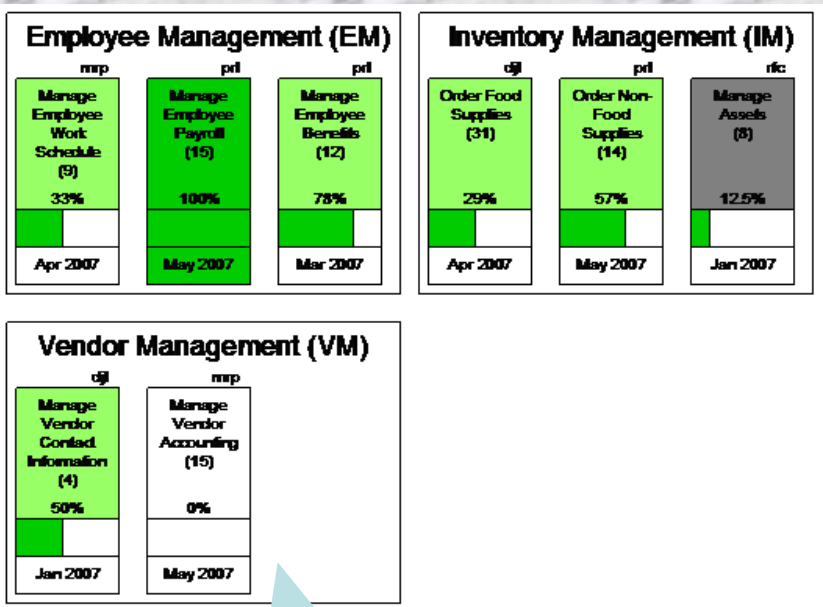
- **Domain Model**
- **Unified Model & Feature Set**
- **Unit feature in FDD**

  <action><result>[of|to|for|from]<object>

  --" Calculate the monthly interest rate for an adjustable rate mortgage"

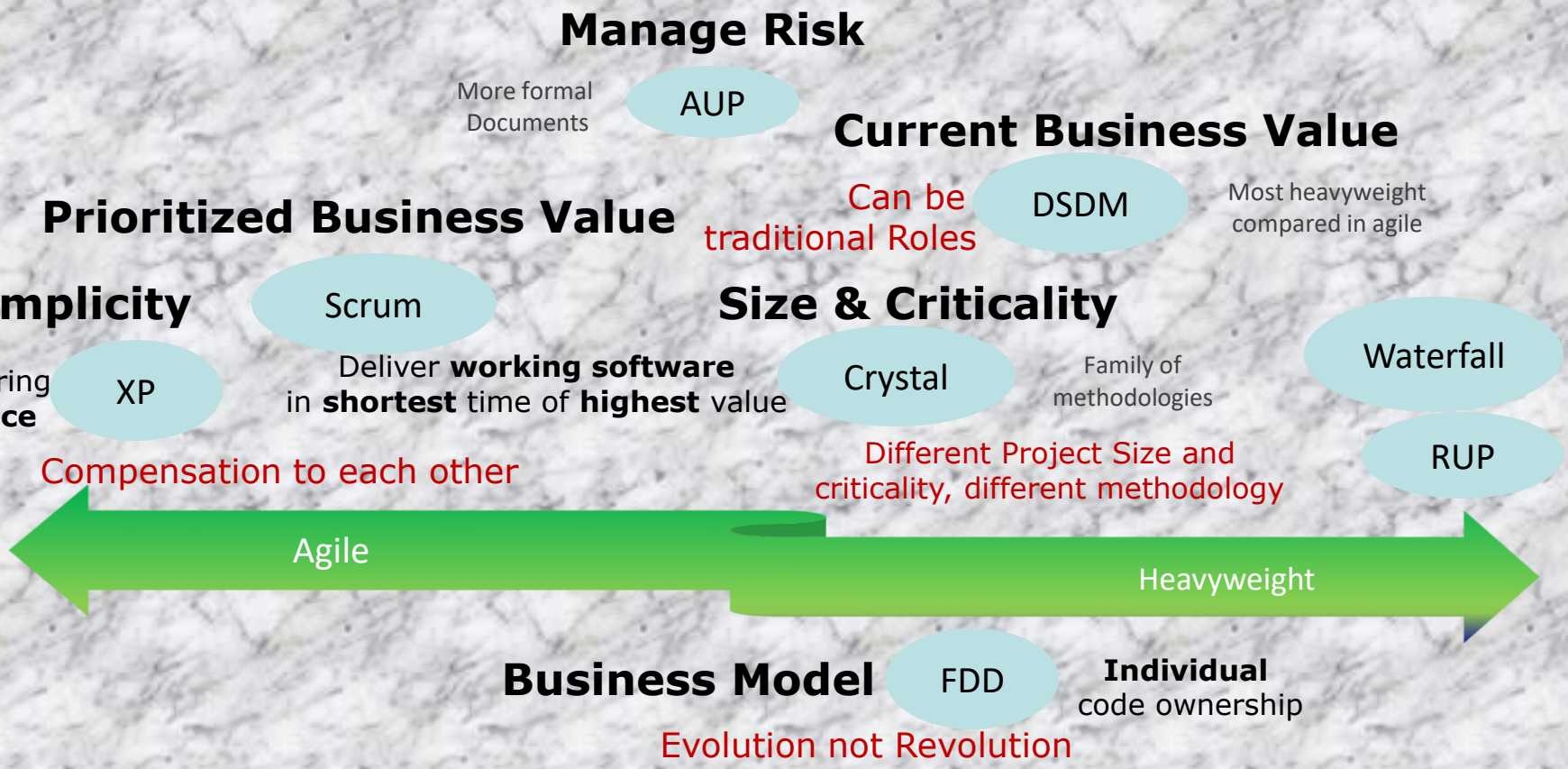
- **Specific in FDD:**

  FDD describes specific, very short phases of work which are to be accomplished separately per feature. These include Domain Walkthrough, Design, Design Inspection, Code, Code Inspection, and Promote to Build.



Feature Set Progress Report: Light green Feature Sets are in progress and on schedule, dark green Feature Sets are completed, and grey Feature Sets are behind schedule.

Comparison & Overview



Comparison & Overview

	Strengths	Weaknesses
XP	• **Strong** technical practices. • Customer ownership of feature priority, developer ownership of estimates. • Frequent feedback opportunities. • Most widely known and adopted approach, at least in the U.S.	• Requires **onsite** customer. • Documentation primarily through verbal communication and code. For some teams these are the only artifacts created, others create minimal design and user documentation. • **Difficult** for **new** adopters to determine how to accommodate architectural and design concerns.
Scrum	• Complements existing practices. • **Self organizing** teams and feedback. • Customer participation and steering. • Priorities based on **business value**. • Only approach here that has a certification process.	• Only provides project management support, other disciplines are out of scope. • Does not specify technical practices. • Can **take some time** to get the business to provide unique priorities for each requirement.
FDD	• Supports **multiple** teams working in parallel. • All aspects of a project tracked by feature. • Design by feature and build by feature aspects are easy to understand and adopt. • Scales to **large** teams or projects well.	• Promotes **individual** code ownership as opposed to shared/team ownership. • Iterations are **not as well defined** by the process as other Agile methodologies. • The **model-centric** aspects can have huge impacts when working on existing systems that have no models.

	Strengths	Weaknesses
AUP	• **Robust** methodology with many artifacts and disciplines to choose from. • Scales up very well. • Documentation helps communicate in distributed environments. • Priorities set based on **highest risk**. Risk can be a business or technical risk.	• Higher levels of ceremony may be a hindrance in smaller projects. • Minimal attention to team dynamics. • Documentation is much more **formal** than most approaches mentioned here.
Crystal	• Family of methodologies designed to scale by **project size** and **criticality**. • Only methodology that specifically accounts for life critical projects. • As project size grows, cross-functional teams are utilized to ensure consistency. • The "**human**" component has been considered for every aspect of the project support structure. • An emphasis on testing is so strong that at least one **tester** is expected to be on each project team.	• Expects all team members to be co-located. May not work well for distributed teams. • Adjustments are required from one project size/structure to another in order to follow the prescribed flavor of Crystal for that project size/criticality. • Moving from one flavor of Crystal to another in mid project doesn't work, as Crystal was **not** designed to be **upward** or **downward compatible**.
DSDM	• An emphasis on testing is so strong that at least one **tester** is expected to be on each project team. • Designed from the ground up by business people, so **business value** is identified and expected to be the highest priority deliverable. • Has specific approach to determining how important each requirement is to an iteration. • Sets stakeholder expectations from the start of the project that not all requirements will make it into the final deliverable.	• Probably the **most heavyweight** project compared in this survey. • Expects **continuous** user involvement. • Defines several artifacts and work products for each phase of the project; heavier documentation. • Access to material is controlled by a Consortium, and fees may **be charged** just to access the reference material.

How to adopt these?

Condition	XP	Scrum	FDD	AUP	Crystal	DSDM
Small Team	√	√	X	X	-	√
Highly Volatile Requirements	√	√	√	-	-	X
Distributed Teams	X	√	√	√	X	X
High Ceremony Culture	X	X	-	√	-	√
High Criticality Systems	X	-	-	-	√	X
Multiple Customers / Stakeholders	X	√	-	-	-	X

favor (√), discourage (X), and neutral (-)

商品软件

- 商品软件也许未必满足你所有的要求，但是自己开发也需要一个周期，到那个时候，商品软件可能已经满足了你的要求。
- 商品软件可能存在不足，但是，你自己开发的产品也未必那么完美，当你补充了商品软件的不足时，也许带来了新的问题。
- 因而，商品软件始终是一个值得考虑的方案。

一个有趣的比喻

晚餐菜单

欢迎光临过程模型选择咖啡厅，希望您好胃口

正餐

螺旋模型

手制烤鸡，外配风险减少调味料

15.95美元

渐进交付模型

用水拌成的增量式交付及渐进原型

15.95美元

增量式交付模型
五门课程酒席，详情请问服务员
14.95美元

面向进度的设计
各种方法学，特别适合于快速运作
11.95美元

纯瀑布模型
按经典的原有菜谱制作
14.95美元

色拉
面向工具的设计
填鸭内填各种豆角
时价

商用软件
名厨手艺，每日变动
4.95美元

编码及查错
大碗面
整日供应
5.95美元