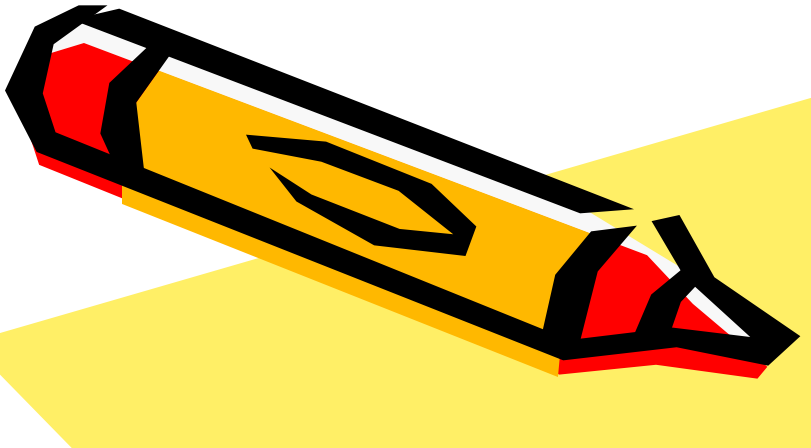
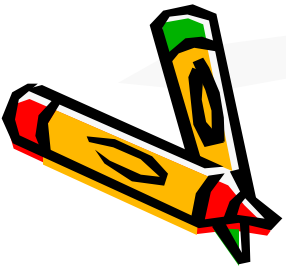


# Web应用开发过程和方法

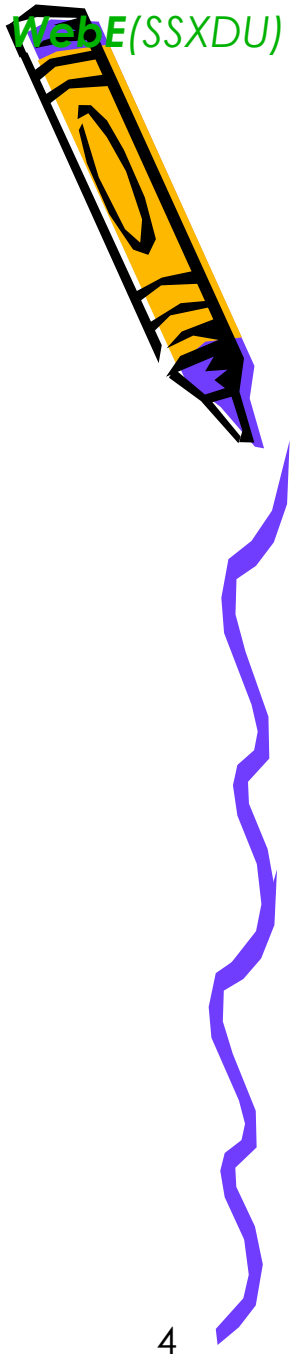
Qiuyan Huo 霍秋艳
qyhuo@mail.xidian.edu.cn
Software Engineering Institute

Is it possible to use traditional software development processes to develop Web application?

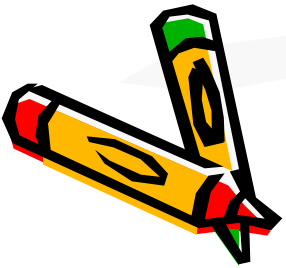


# Web应用开发过程和方法

- Web应用开发过程的特点
- 软件开发过程
  - RUP
  - XP
  - RUP与XP对Web应用的适应性
- 定制基于RUP和XP的Web应用过程
  - 基于RUP和XP的Web应用过程
  - 敏捷Web应用开发过程
- 总结与展望

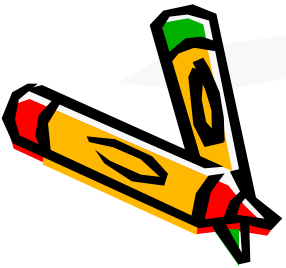


# WEB应用开发过程的特点



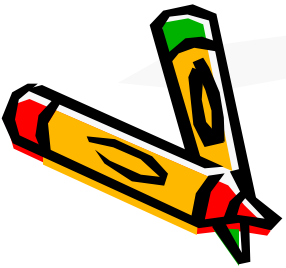
# Web应用开发过程的特点

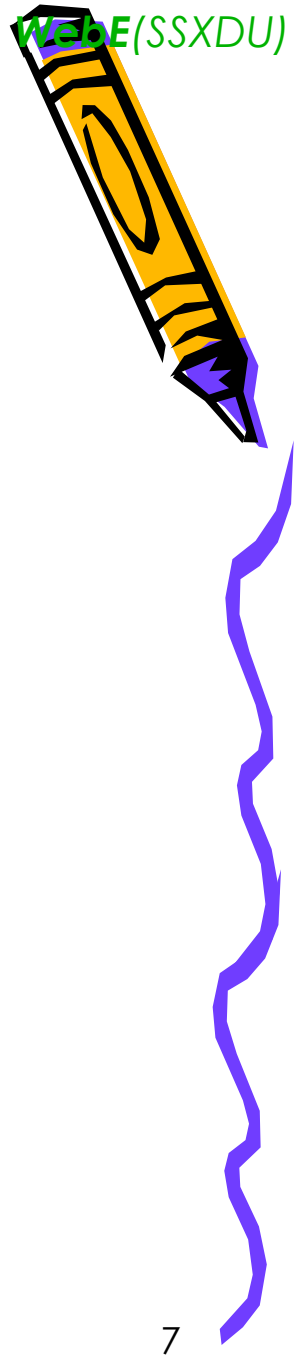
- 开发周期短
- 需求变更频繁
- 开发技术不断演化
- 并行开发不同版本
- 重用和集成
- 适应Web应用的复杂性程度



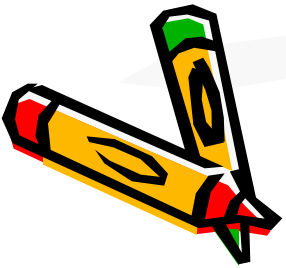
# Web应用开发过程的特性

- 基于迭代思想，重视系统的快速开发和不断演化，降低在一个增量上的开发风险
- 强调原型开发，并作为开发过程模型的重要组成部分
- 强调开发过程中各个阶段的追溯、调整和反馈





# 软件开发过程



# A Few Definitions

- *Process Model* – summarizes the development approach in the overall context.
  - Organizational aspects
  - Estimating resources; timing
  - Examples: RUP, XP

**The process model describes *when* something *should* be done *under organizational aspects*.**

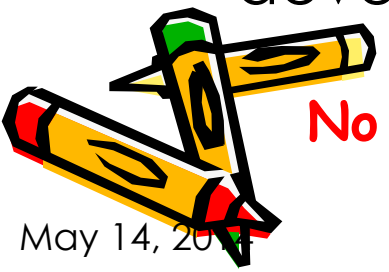
# Categories

- *Lightweight* processes
  - Agile processes
- *Heavyweight* processes

"light" or "heavy" refers to the degree of process formalization, i.e., how many documents and models are created.

lightweight and heavyweight **methods** because the creation of documents and models in specific form is defined by methods.

- Heavyweight are used particularly when large teams develop applications with high demands on the quality.
- Lightweight processes are suitable for smaller applications and accordingly smaller development teams.



**No single process model is equally suitable for all projects.**

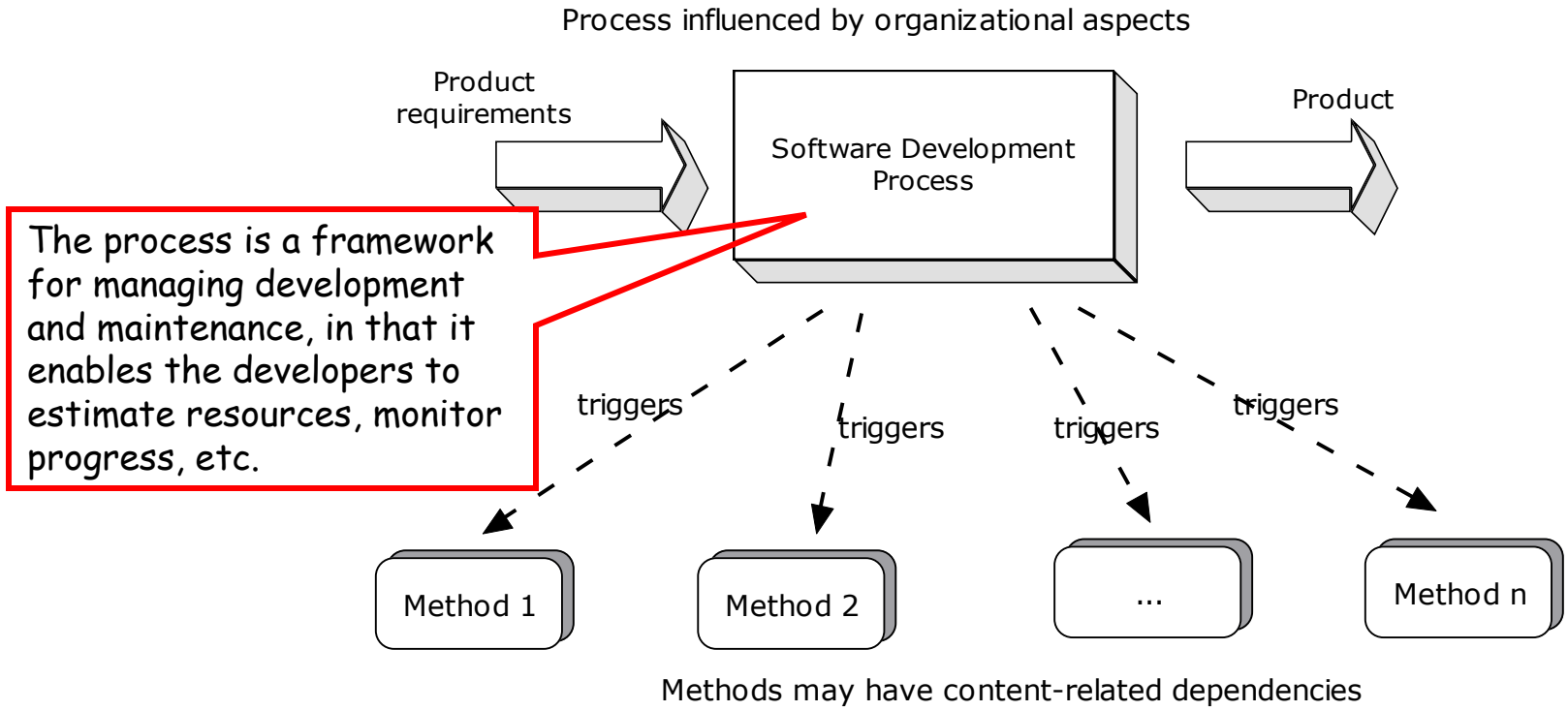
# A Few Definitions *(cont.)*

- *Method* – a detailed approach to solving a development problem.
  - Requirements and Technology-oriented
  - Content-specific aspects
  - Example: A UML diagram

***how* something should be done, and *when* it *can* be done**

- A ***process*** provides guidance to software engineers on the order in which the various methods should be carried out within a development projects.

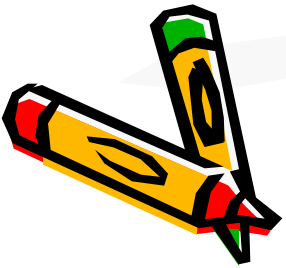
# Process vs. Methods




The degree of freedom for decisions in the process is limited by the underlying methods, since a process has to take methodological dependencies into account.
Organizational needs such as the need of people participating in a project to communicate define requirements on the methods.

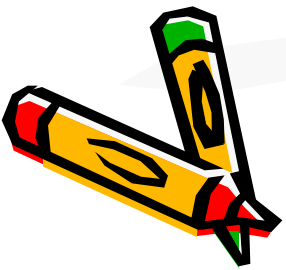
# 软件开发过程

- 常见的软件工程模型
  - 线性模型
  - 渐增式模型
  - 螺旋模型
  - 快速原型模型
  - 形式化描述模型
  - ……



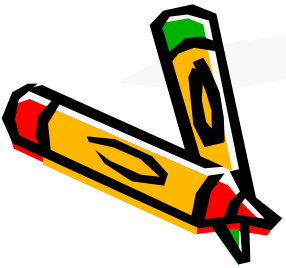
# Web应用开发方法

- 好的Web应用开发方法的主要特征
  - 易于掌握
  - 对复杂系统建模的能力
  - 展示层建模的能力
  - 系统定制的支持
  - 模型集成和连通的能力
  - 工具和文档化支持

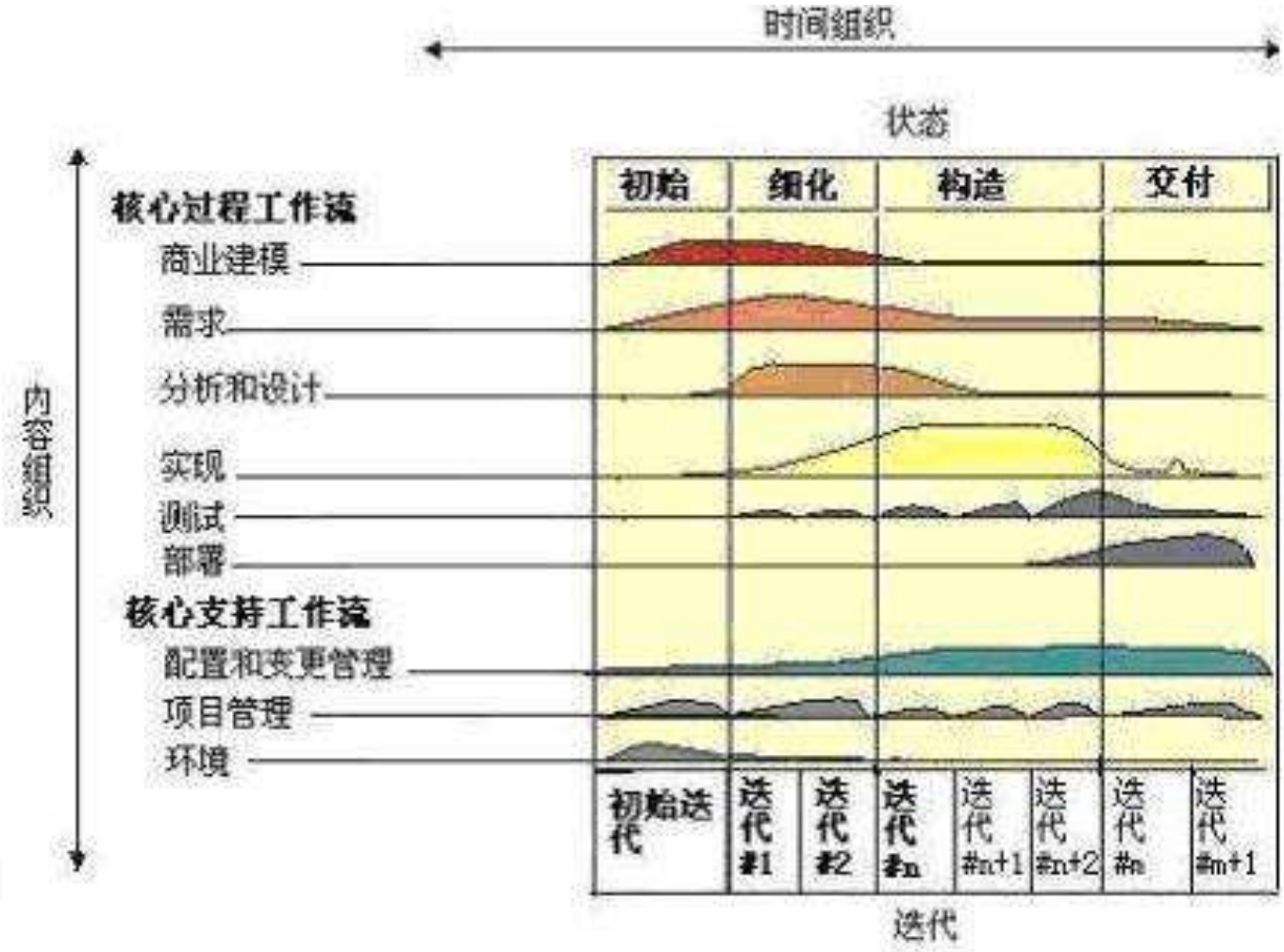


# RUP

- RUP（Rational Unified Process，统一软件开发过程）是一套软件工程方法，主要包含：用于成功开发软件的一组核心概念和做法；过程模型和相关联内容库；以及底层过程定义语言。



# RUP

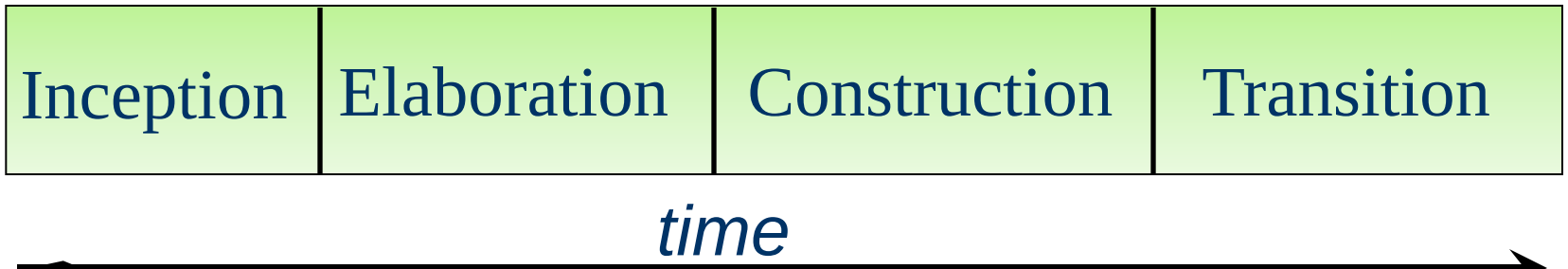

时间组织
状态
初始
细化
构造
交付
核心过程工作流
商业建模
需求
分析和设计
实现
测试
部署
核心支持工作流
配置和变更管理
项目管理
环境
内容组织
初始迭代
迭代#1
迭代#2
迭代#n
迭代#n+1
迭代#n+2
迭代#n
迭代#m+1
迭代

# RUP核心概念

- RUP开发过程的二维结构
  - 横轴代表了制订开发过程时的时间，体现了过程的动态结构
    - 周期(Cycles)：每一个周期工作在产品新的一代上
    - 阶段(Phases)：初始、精化、构建、提交
    - 迭代(Iterations)：每个阶段进行若干次
    - 里程碑(Milestones)：迭代正式结束的时间点
  - 纵轴表现了过程的静态结构
    - 工作者(Workers)：行为和责任
    - 活动(Activities)：工作者要执行的工作单元
    - 工件(Artifacts)：活动的结果
    - 工作流(Workflow)：对应于特定的迭代的连续活动

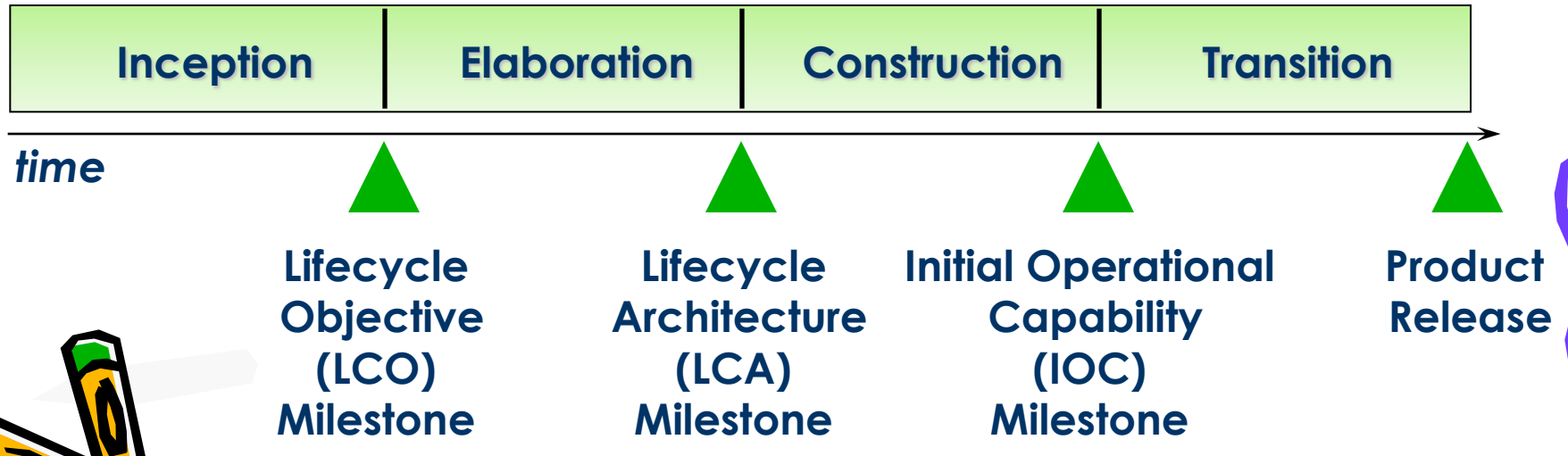
# RUP生命周期

- RUP中的软件生命周期
  - RUP在时间上被分解为四个连续的阶段
    - 初始：定义项目的范围
    - 细化：计划项目，说明特性和构架基线
    - 构建：建立产品
    - 交付(产品化)：交付产品到最终用户团体



# 阶段边界——主要里程碑

- 生命周期目标(LCO)
- 生命周期构架(LCA)
- 初始功能(IOC)
- 产品发布



# RUP阶段——初始阶段

- 目标是为系统建立商业案例并确定项目的边界
- 结束时是第一个重要的里程碑：生命周期目标（Lifecycle Objective）里程碑
- 初始阶段的主要目标：
  - 建立项目的软件规模和边界条件
  - 识别系统的关键用例
  - 评估整个项目的总体成本和进度
  - 评估潜在风险
  - 准备项目的支持环境

# RUP阶段——细化阶段

- 目标是分析问题领域，建立健全的架构基础，编制项目计划，淘汰项目中最高风险的元素
- 结束时第二个重要的里程碑：生命周期架构（Lifecycle Architecture）里程碑
- 细化阶段的主要目标：
  - 确保架构、需求和计划足够稳定
  - 处理在架构方面具有重要意义的所有项目风险
  - 建议一个已确定基线的架构
  - 制作产品质量构件的演进式原型
  - 证明已建立基线的架构支持系统需求
  - 建立支持环境

# RUP阶段——构造阶段

- 所有剩余的构件和应用程序功能被开发并集成为产品，所有的功能被详细测试
- 结束时是第三个重要的里程碑：初始功能（Initial Operational）里程碑
- 构造阶段的主要目标：
  - 优化资源，使开发成本降到最低
  - 尽快达到质量要求
  - 快速完成有用的版本
  - 完成所有功能的分析、开发和测试
  - 迭代式、递增地开发随时可以发布的产品
  - 确定准备好软件系统的外部环境

# RUP阶段——交付阶段

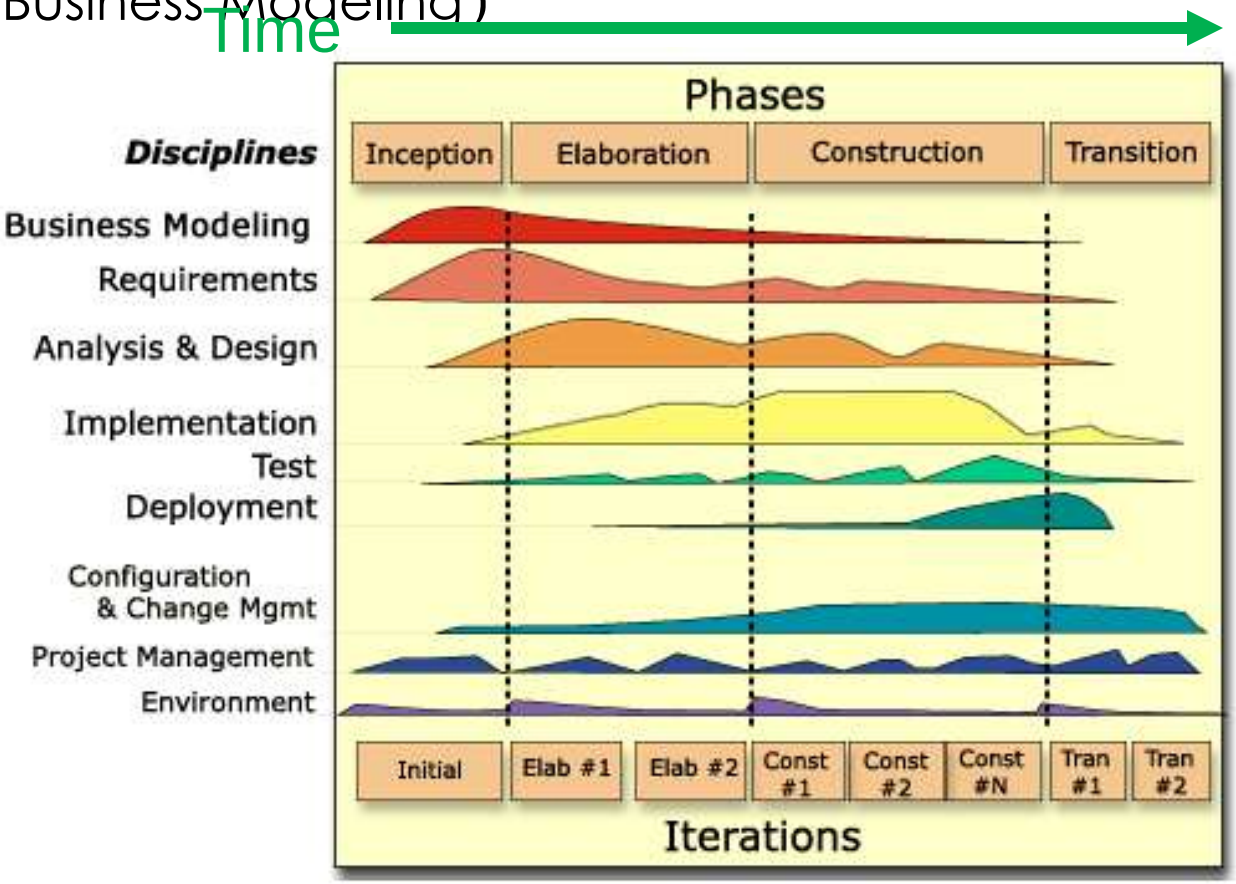
- 重点是确保软件对最终用户是可用的
- 终点是第四个里程碑：产品发布（Product Release）里程碑
- 交付阶段的主要目标：
  - 进行Beta测试，按用户的期望确认新系统
  - Beta测试和相对于正在替换的遗留系统的并行操作
  - 转换操作数据库，培训用户和维护人员
  - 市场营销、进行分发和向销售人员进行新产品介绍
  - 进行与部署相关的工程
  - 根据产品的完整前景和验收标准，对部署基线进行评估

# RUP:核心工作流

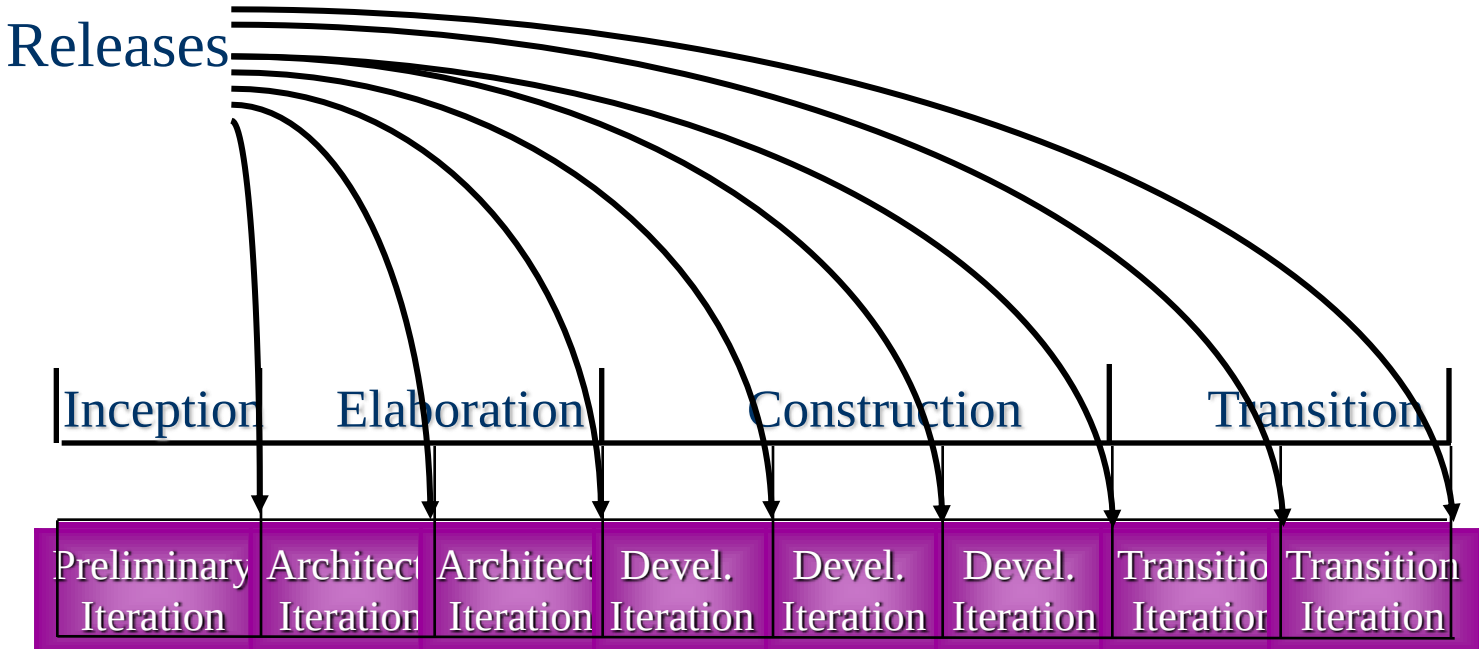
- 6个核心过程工作流
  - 商业建模（Business Modeling）
  - 需求（Requirements）
  - 分析和设计（Analysis & Design）
  - 实现（Implementation）
  - 测试（Test）
  - 部署（Deployment）
- 3个核心支持工作流
  - 配置和变更管理（Configuration & Change Management）
  - 项目管理（Project Management）
  - 环境（Environment）

# RUP:核心工作流

- 6个核心过程工作流
  - 商业建模（Business Modeling）
  - 需求（Req
  - 分析和设计
  - 实现（Imp
  - 测试（Test
  - 部署（Dep
- 3个核心支持
  - 配置和变更 Managem
  - 项目管理（
  - 环境（Envi

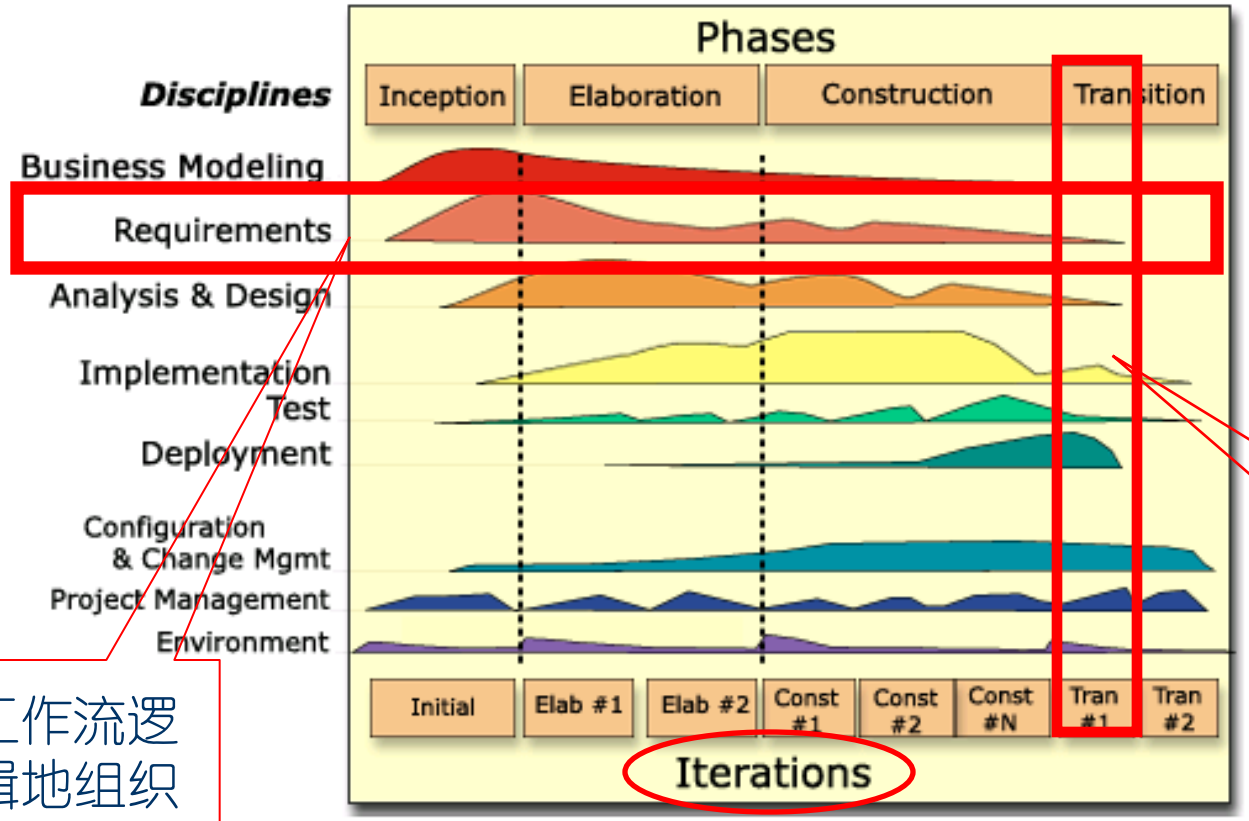


# 迭代和阶段




- 迭代：一个基于建立的计划(baselined plan)和评定标准的一个清晰活动的顺序，产生一个可执行的版本发布(内部或者外部)

# 迭代方法




工作流逻辑地组织活动

In an iteration, you walk through all disciplines. *在一个迭代中，实施各种工作流*

# RUP对Web应用的适应性

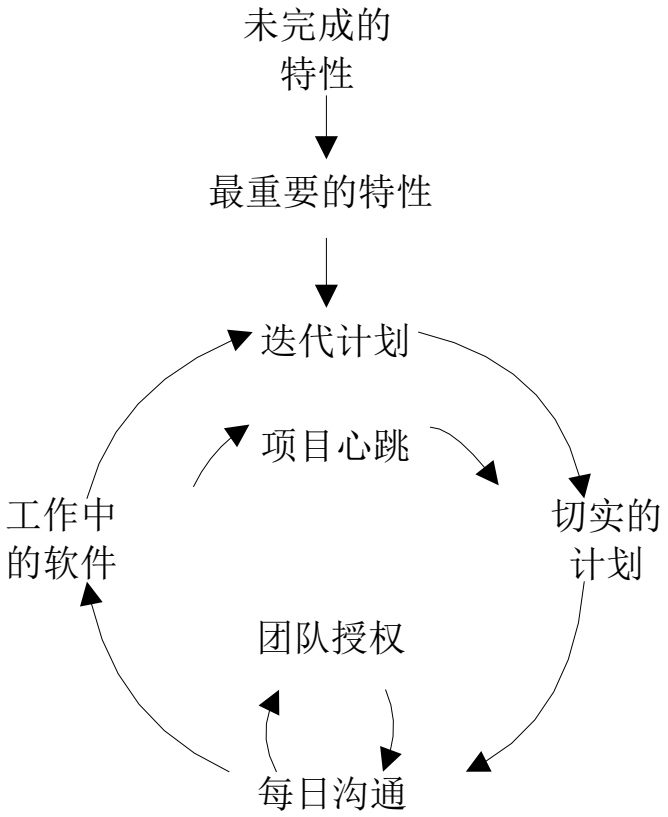
- 迭代式开发
- 管理需求
- 使用构件构架
- 可视化建模
- 检验质量
- 控制变更

# XP

- XP（Extreme Programming，极限编程）源于快速响应问题域频繁变化的需求，是敏捷过程的一种具体形式，提供敏捷方法（Agile Method）最一般的原则的指导方针。
- XP从沟通、简单、反馈、尊重和勇气五个方面改善任何一个软件项目。

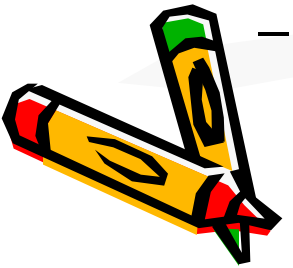
# 一个简单的XP过程

- “XP是一种轻量、高效、低风险、柔性、可预测、科学而充满乐趣的软件开发方法。”
  - ——2000年，美国软件工程专家Kent Beck



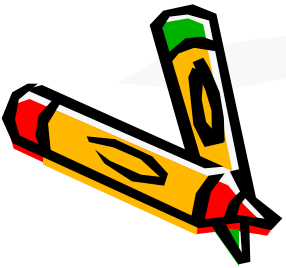
# XP特性——简单规则

- 计划
  - 编写用户故事
  - 制定发布版本计划
  - 不断创建小的发布版本
  - 项目分为多个迭代
  - 迭代计划
- 管理
  - 营造开放的工作场所
  - 设置可持续的速度
  - 每天第一件事是举行简短的站立会议
  - 度量项目速度
  - 让开发人员动起来
  - XP无效时进行修复



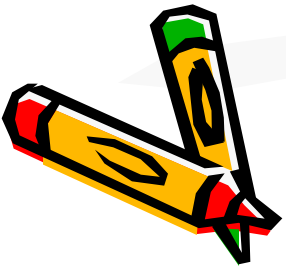
# XP特性——简单规则

- 设计
  - 保持简单而简洁的设计
  - 选择系统隐喻
  - 使用CRC卡片进行设计
  - 创建微小系统以降低风险
  - 尽早添加功能
  - 一有可能就进行重构



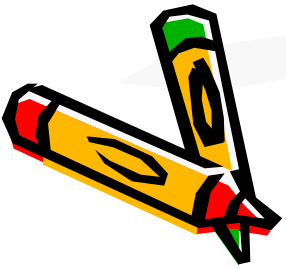
# XP特性——简单规则

- 编码
  - 客户始终在场
  - 编码要遵循标准
  - 代码要先进行单元测试
  - 所有的编码工作都结对完成
  - 每次只有一对进行集成
  - 持续集成
  - 进行控制版本
  - 代码集体拥有权

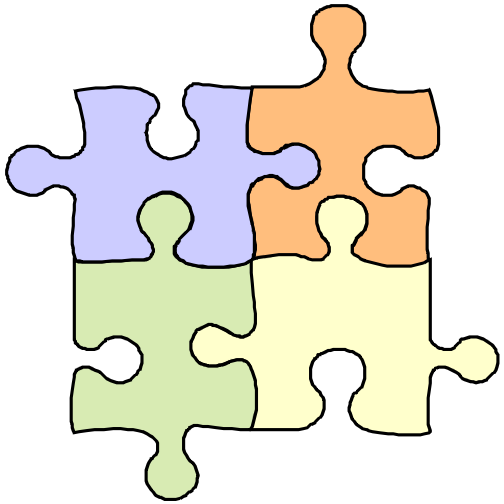


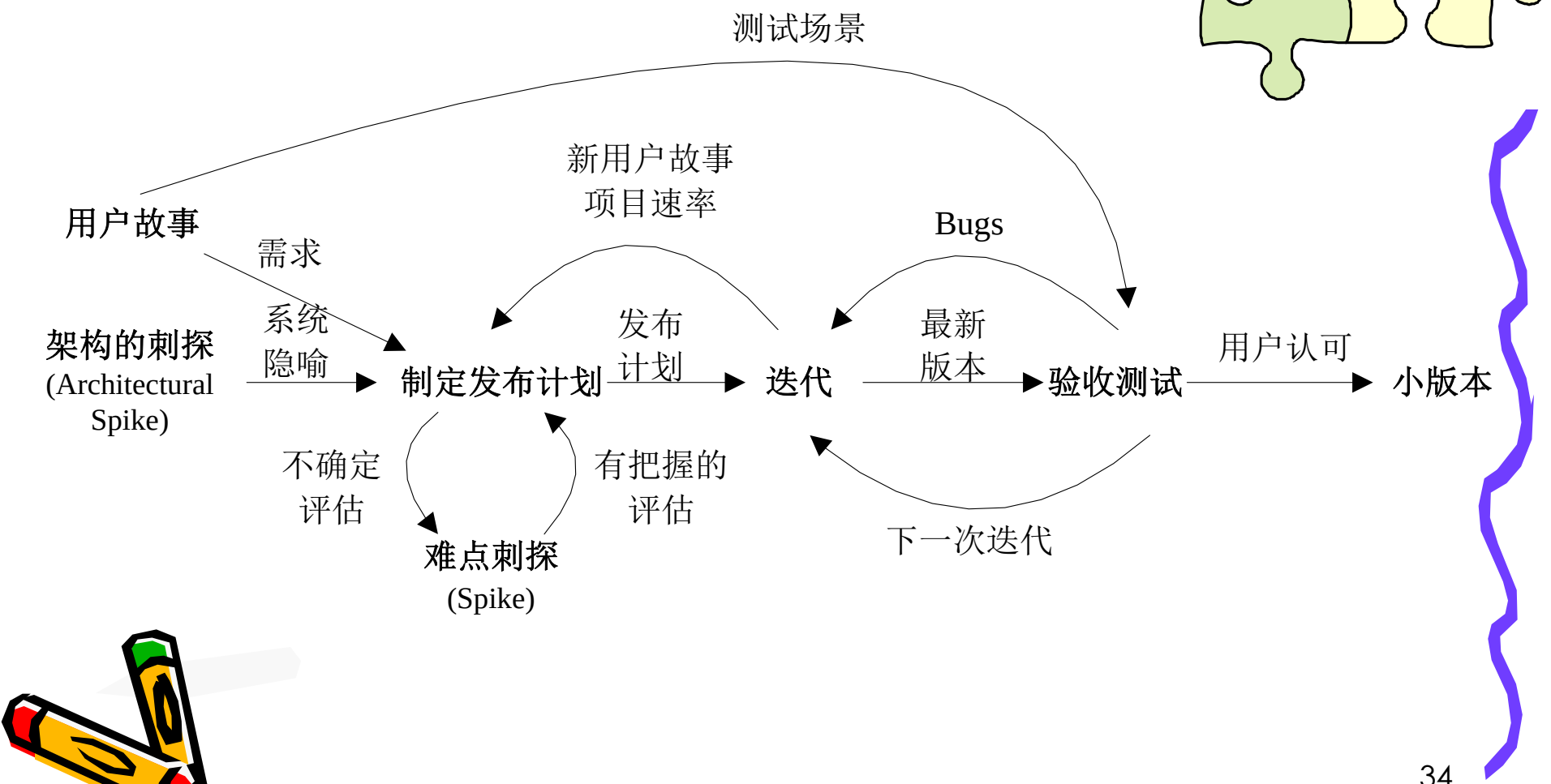
# XP特性——简单规则

- 测试
  - 所有代码必须进行单元测试
  - 所有代码必须在发布之前完全通过单元测试
  - 当发现问题时编写测试代码
  - 经常运行接受测试并公布得分



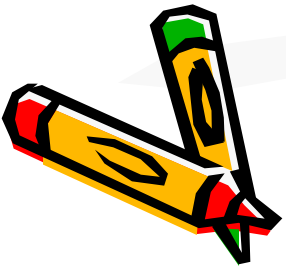
# XP项目流程图




测试场景
新用户故事
项目速率
用户故事
需求
Bugs
系统
隐喻
发布
计划
最新
版本
用户认可
架构的刺探
(Architectural
Spike)
制定发布计划
迭代
验收测试
小版本
不确定
评估
有把握的
评估
下一次迭代
难点刺探
(Spike)

# XP的核心价值

- XP的五个核心价值
  - 沟通（Communication）
  - 简单（Simplicity）
  - 反馈（Feedback）
  - 勇气（Courage）
  - 尊重（Respect）

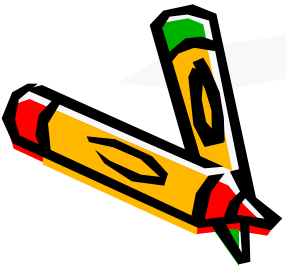


# XP的三个重点

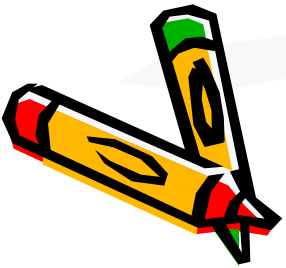
- 角色定位
  - 不仅让客户参与设计讨论，而且让客户负责编写用户故事（User Story），也就是功能需求
- 敏捷开发
  - 敏捷开发追求合作与响应变化
  - 迭代就是缩短版本的发布周期，缩短到周、日，完成一个小的功能模块，可以快速测试、并及时展现给客户，以便及时反馈
- 追求价值
  - XP把软件开发变成自我管理的挑战，追求沟通、简单、反馈、勇气和尊重，体现开发团队的人员价值，激发参与人员的情绪，调动开发者的积极性
  - 结对编程就是激发队员才智的一种方式

# XP对Web应用的适应性

- 处理短开发周期
- 处理需求变更
- 固定期限和灵活内容的发布
- 不同版本的并行开发
- *重用和集成*
- 适应Web应用的复杂性水平



# 基于RUP和XP的WEB应用过程



# 基于RUP和XP的Web应用过程

- 迭代开发


Documentation should be
"Just Barely Good Enough" (**JBGE**).

# 基于RUP和XP的Web应用过程

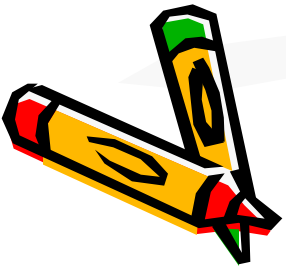
- Web应用需求捕获
  - 主要任务：Web应用需求捕获和用户故事以及基于用例的Web需求描述
  - 描述视觉外观
    - 必须与要解决的问题相适应
    - 能明确定义系统的边界
    - 描述出系统最重要的特征
  - 描述用例
    - 捕获系统潜在的使用者
    - 捕获不同角色与系统交互的过程
    - 书写的用例文档建立用例图和活动图

# 基于RUP和XP的Web应用过程

- Web应用设计
  - 构建创意设计大纲
  - 设计导航图
  - 设计用户创意设计方案和界面原型
  - 设计Web设计元素
  - 初始Web页面原型
  - 编写 Web页面指南
  - 架构分析
  - 用例分析
  - 确定设计元素

# 基于RUP和XP的Web应用过程

- Web应用的构建与部署
  - 构建和获取所有Web应用的内容，并将其集成到Web应用的架构之中
  - 选择合适的产生Web页面的工具集，实现每个页面的布局、功能、表单和导航功能
  - 在实现代码阶段要大量地采用重构的方法，将现有的模型改变成更优秀的模型

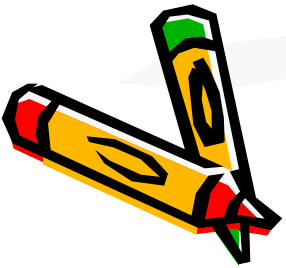


# 基于RUP和XP的Web应用过程

- Web应用的测试
  - 在Web应用开发与实现中，采用了测试先行的方法，由程序员实现对象，然后将构件交由集成人员将其集成到系统中
  - Web应用的测试很大程度上注重于性能测试，以确保 Web 应用程序可支持并发用户数量的激增
  - 必须测试用户交互来验证 Web 应用程序的结构适合其用户
  - 进行浏览器测试，因为浏览器和浏览器版本之间的兼容性经常会限制用户界面中的设计选项

# 基于RUP和XP的Web应用过程

- Web应用的发布
  - 将构建好的Web应用部署在用户的环境中，并从最终用户那里得到反馈，建立修改的基础
    - Web应用的产品发布往往是递增式和连续的，而较少注重于传统的介质发布
    - Web环境中的用户培训往往集成到 Web 站点自身的设计中，使站点的使用直观
    - 必须注重于在不可预测的负载情况下维持高可用性
    - 研究用户如何使用应用程序

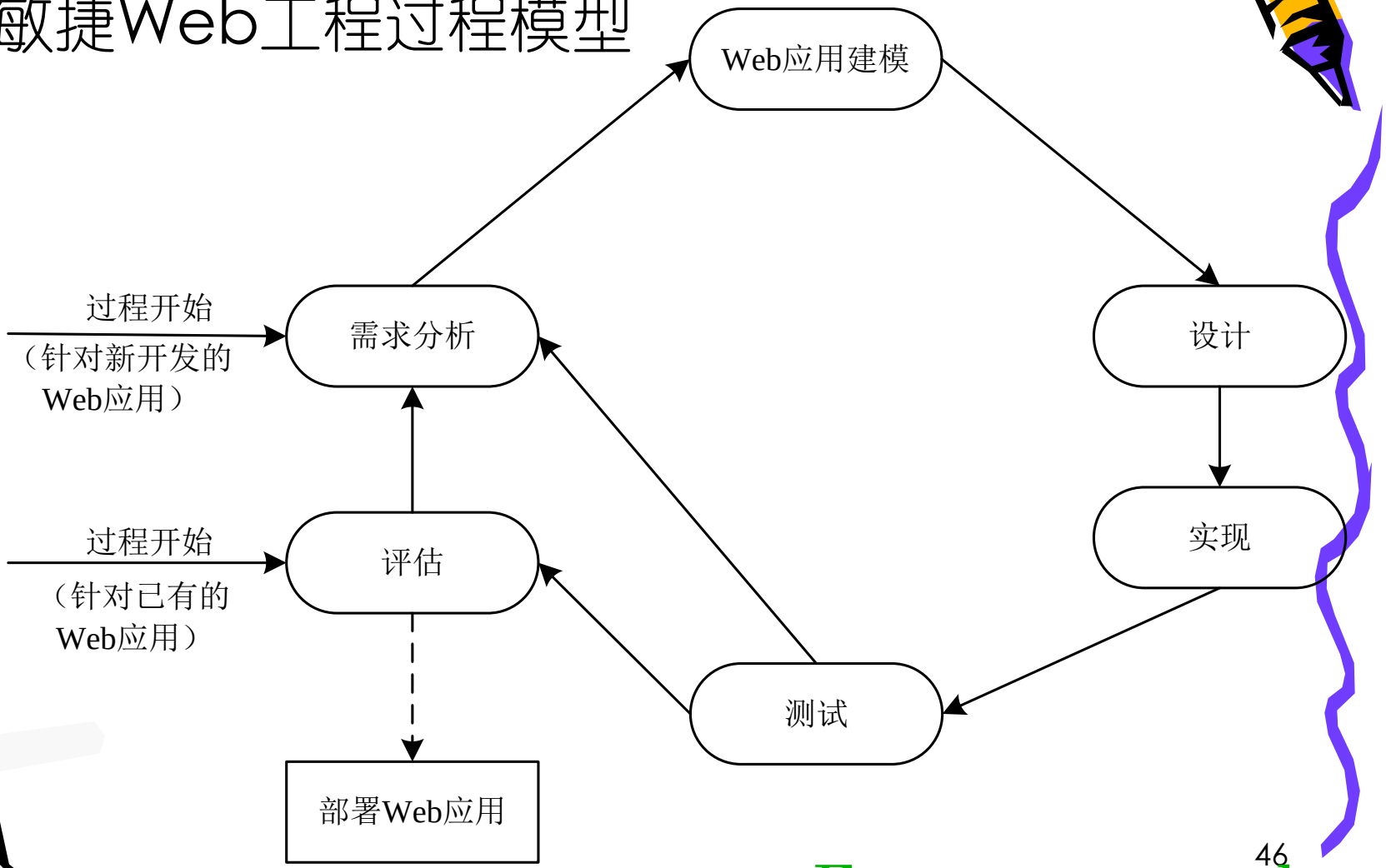


# 基于RUP和XP的Web应用过程

- Web重用与集成
  - Web应用开发的巨大时间压力的一个直接后果是开发人员应该尽可能的去重用已有的组件
  - 基于Web Services实现对遗留系统的快速改造、集成和重用
  - 将遗留系统进化为Web Services的再工程方法
    - 评估遗留系统
    - 解耦遗留系统
    - 业务规则抽取
    - 业务规则确认
    - 服务包装和集成

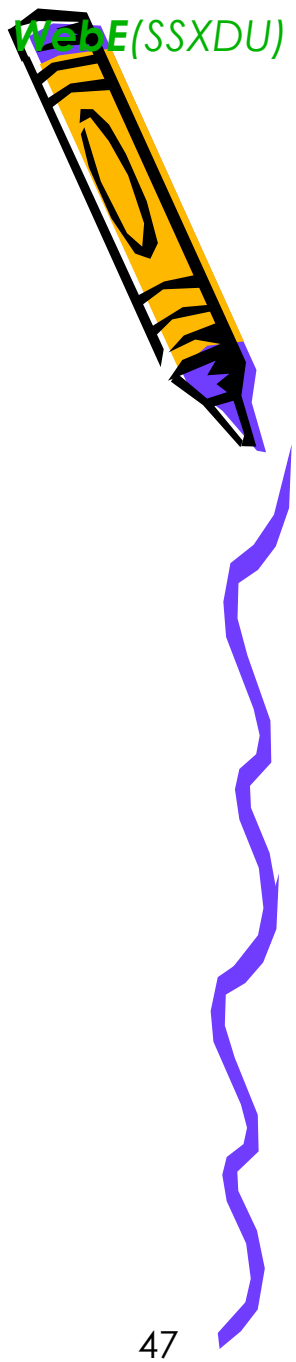
# 敏捷Web应用开发过程

- 敏捷Web工程过程模型

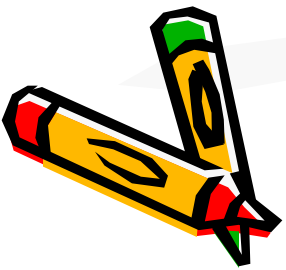





Features example



# 总结与展望



# 总结

- Web应用的开发不同于传统应用软件的开发，有着自身的特性，需要根据Web应用的特性定义一个完整的Web应用开发过程
- Web应用开发过程是一个迭代的过程
- 在Web应用中用户界面受到极大的关注
- 基于Web Services重用和集成遗留系统的再工程方法

# 展望

- 针对Web应用的高可扩展性和高复杂性的要求，定义软件开发的元过程(Meta-process)
- 元过程主要监控Web应用的特点，标识Web应用的复杂性水平，并根据复杂性水平，实例化为具体的轻量级过程或重量级过程，以及两种过程之间的过渡阶段
- Web应用的开发也会逐步演化成一个产品生产的过程，这时重量级和迭代过程的优点将更为凸显

# Meta-process

- We have to set up a higher-order process – meta-process – serves to ensure strategic planning beyond project boundaries.