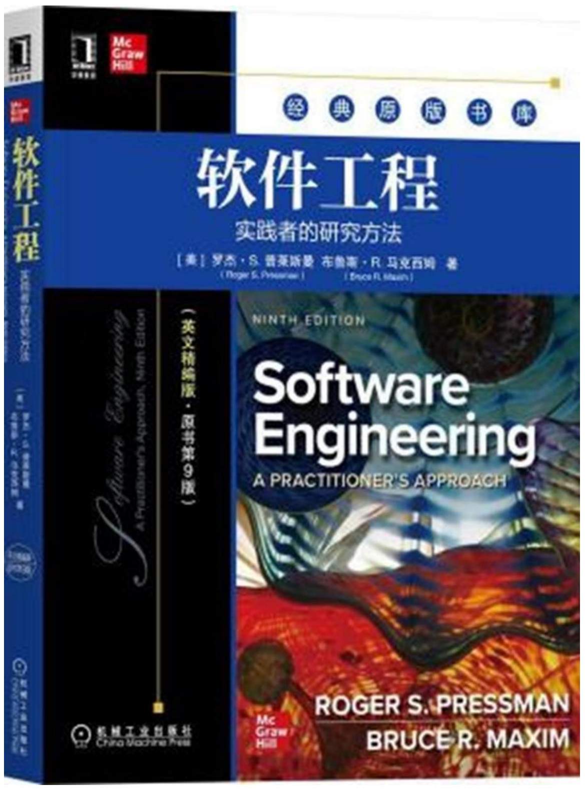
UNIT 8

过程 & 过程模型

Software Process & Process Model

王传栋

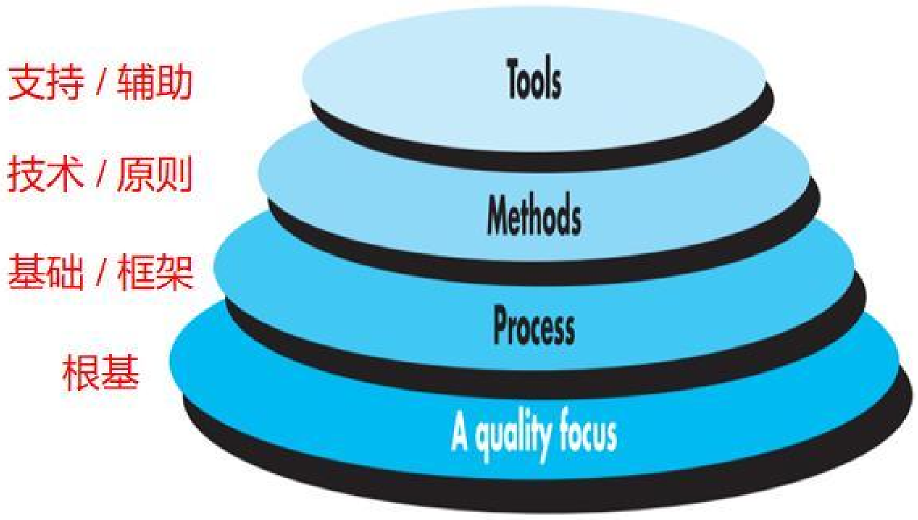


"traditional" Process Models

SE is a layered technology

The **bedrock** that supports **SE** is a **quality focus**.

- **process** ： framework & basis, to hold the technology
- **methods** ： the technical (principles) to build software
- **tools** ： support (aided) for the process and the methods



software process

A process is defined as a set of activities, actions, and tasks

An **activity** achieve a broad objective.

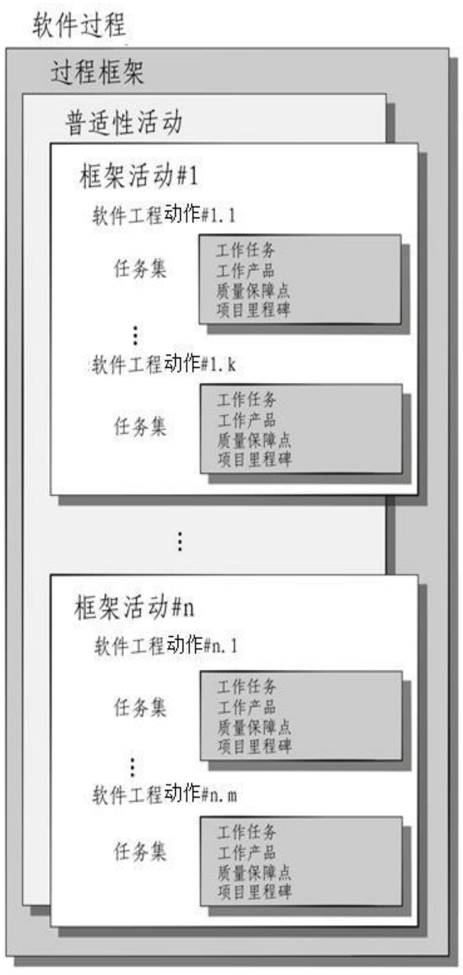
An **action** produce a major work product.

A **task** focuses on a small, but well-defined objective that produces a tangible outcome.

A process framework includes five **framework activities** and a set of **umbrella activities**.

A **framework activity** contains some **actions**, each **action** is defined as a set of **task sets**.

each **task set** encompasses a series of “the work tasks, the work products, the quality assurance points and the milestones”



Framework Activities

Communication.

Planning.

Modeling.

- Analysis of requirements.
- Design.

Construction:

- Code generation.
- Testing.

Deployment.

不同案例，过程细节差别很大，但框架活动是一致的

- ✧ 无论是简单小程序，还是 WebApp 及大型复杂系统工程
- ✧ 项目开展中，框架活动可以多次迭代应用，5个框架活动不断重复
- ✧ 每次迭代产生一个实现部分特性和功能的增量（software increment）
- ✧ 随着每一次增量的产生，软件逐渐趋于完善

Umbrella Activities

- Software project tracking and control.
- Risk management.
- Software quality assurance.
- Technical reviews.
- Measurement.
- Software configuration management.
- Reusability management.
- Work product preparation and production.

Process Flow

① A **linear process flow**

② An **iterative process flow**

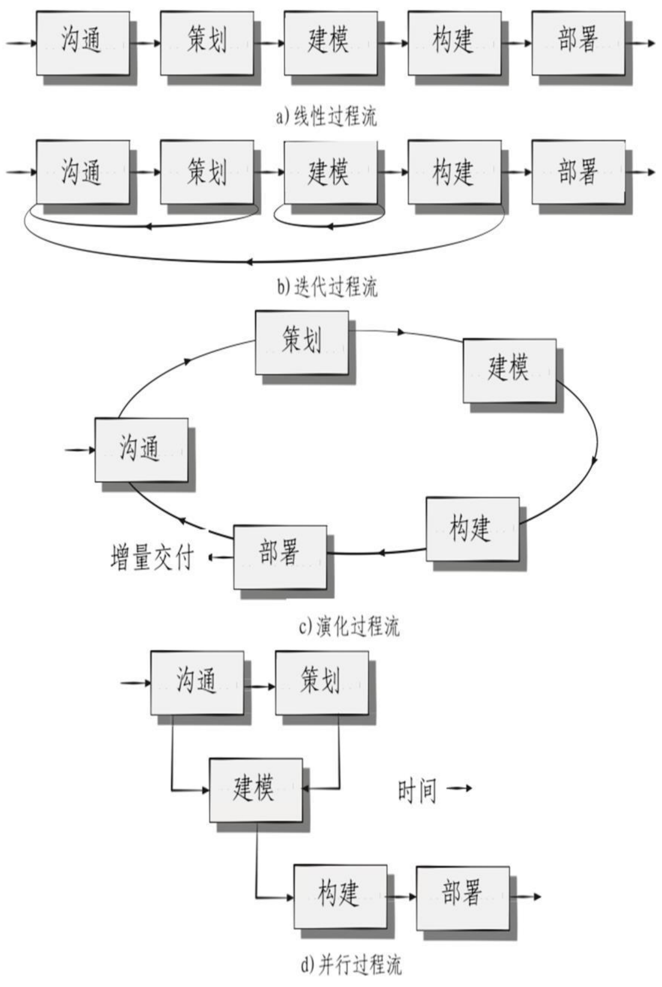
repeats one or more of the activities before proceeding to the next.

③ An **evolutionary process flow**

executes the activities in a “circular” manner.

④ A **parallel process flow**

executes one or more activities in parallel with other activities



Defining A framework Activity

What actions **are appropriate for** a framework activity, given the **nature** of the problem to be solved, the **characteristics** of the people doing the work, and the **stakeholders** who are sponsoring the project.

示例：个人负责的小型软件项目	示例：稍复杂的项目
• 需求简单明确，沟通活动仅与合适利益相关者的一个电话或一封邮件 • 主要动作是电话交流，主要任务集： ① 与利益相关者取得联系 ② 讨论需求并做记录 ③ 将笔记整理成简单的书面需求 ④ email 利益相关者审阅并认可	• 利益相关者众多，需求各不相同，有时需求甚至相互冲突 • 沟通活动可能包含 6 个不同的动作：起始、需术获取、需求细化、协商、规格说明和确认 • 每个动作都可能有很多工作任务和一些不同的工作产品

identifying A Task Set

How does a team choose a task set for a particular project ?

a task set is a collection of **SE** work tasks, related work products, quality assurance points, and project milestones.

chooses the task set based on the characteristics of the team, the project, and the problem to be solved.

活动	沟通	
动作	需求获取	
	小型项目	大型项目
任务集	① 制定项目的利益相关者列表 ② 邀请所有的利益相关者参加一个非正式会议 ③ 征询每个人对于软件特性和功能的需求 ④ 讨论需求，并确定最终的需求列表 ⑤ 划定需求优先级 ⑥ 标出不确定域	① 制定项目的利益相关者列表 ② 与利益相关者的每个成员单独讨论，获取所有要求 ③ 基于利益相关者的输入，建立初步的功能和特性列表 ④ 安排一系列促进需求获取的会议 ⑤ 组织会议 ⑥ 在每次会议上建立非正式的用户场景 ⑦ 根据利益相关者的反馈，细化用户场景 ⑧ 建立一个修正的利益相关者需求列表 ⑨ 使用质量功能部署技术，划分需求优先级 ⑩ 将需求打包以便于软件可以实施增量交付 ⑪ 标注系统的约束和限制 ⑫ 讨论系统验证方法

过程评估和改进

1. 过程不能保证软件按期交付，也不能保证软件满足客户要求，不能保证软件具备长期质量保证的技术特点
2. 评估软件过程，确保过程满足成功软件所必需的基本过程标准要求
3. 使用数字测度或度量方式，分析与评估软件过程和活动

为什么软件过程的敏捷性（灵活）很重要?

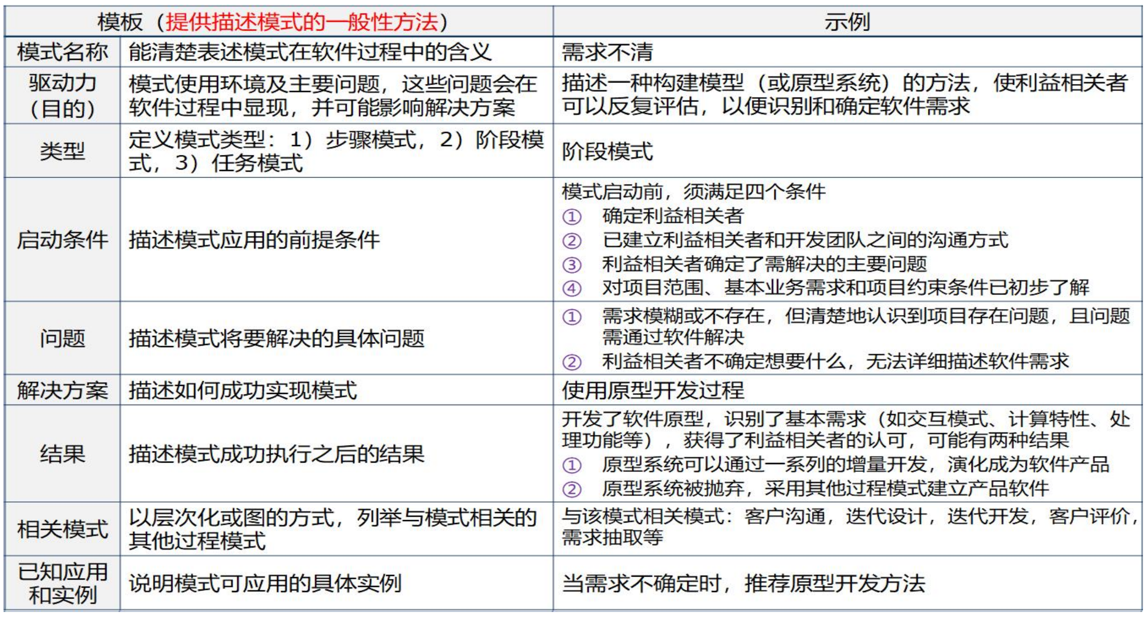
- 过程为活动提供稳定性、控制性和组织性，防止活动变得混乱
- 现代软件工程方法必须是“灵活”的，要求活动、控制和创建的工作产品，必须适于项目团队和待开发系统，确保适应变化并提供高产品的质量

过程模式：过程中遵循的做事套路

过程模式如何帮助团队高效地构建软件产品？

- 流程模式是解决常见开发问题的行之有效的解决方案
- 如果开发人员能够使用已知方法解决问题，就无需花时间发明新的解决方案

记录模板



模板（提供描述模式的一般性方法）		示例
模式名称	能清楚表述模式在软件过程中的含义	需求不清
驱动力（目的）	模式使用环境及主要问题，这些问题会在软件过程中显现，并可能影响解决方案	描述一种构建模型（或原型系统）的方法，使利益相关者可以反复评估，以便识别和确定软件需求
类型	定义模式类型：1）步骤模式，2）阶段模式，3）任务模式	阶段模式
启动条件	描述模式应用的前提条件	模式启动前，须满足四个条件 ① 确定利益相关者 ② 已建立利益相关者和开发团队之间的沟通方式 ③ 利益相关者确定了需解决的主要问题 ④ 对项目范围、基本业务需求和项目约束条件已初步了解
问题	描述模式将要解决的具体问题	① 需求模糊或不存在，但清楚地认识到项目存在问题，且问题需通过软件解决 ② 利益相关者不确定想要什么，无法详细描述软件需求
解决方案	描述如何成功实现模式	使用原型开发过程
结果	描述模式成功执行之后的结果	开发了软件原型，识别了基本需求（如交互模式、计算特性、处理功能等），获得了利益相关者的认可，可能有两种结果 ① 原型系统可以通过一系列的增量开发，演化成为软件产品 ② 原型系统被抛弃，采用其他过程模式建立产品软件
相关模式	以层次化或图的方式，列举与模式相关的其他过程模式	与该模式相关模式：客户沟通，迭代设计，迭代开发，客户评价，需求抽取等
已知应用和实例	说明模式可应用的具体实例	当需求不确定时，推荐原型开发方法

traditional process models

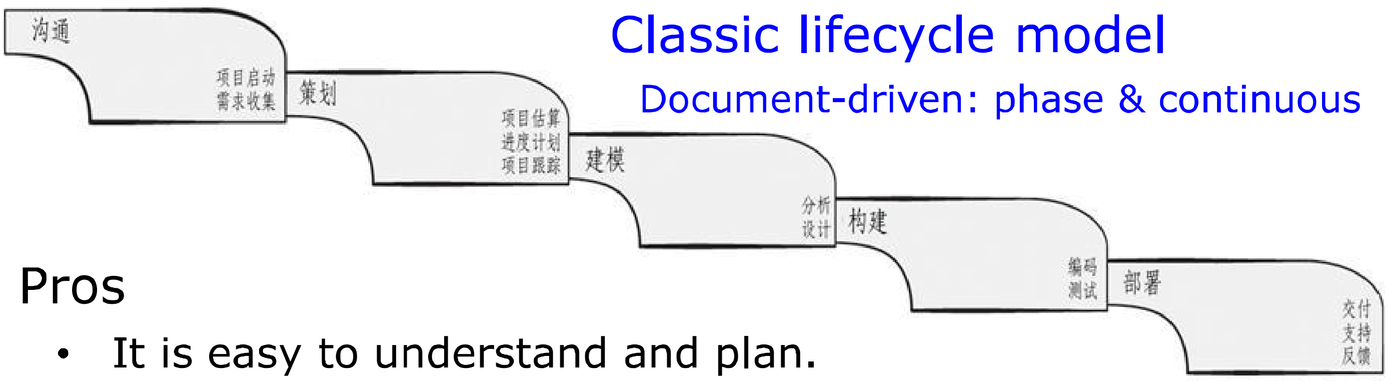
define a predefined set of process elements and a predictable process work flow.

- prescribe **a set of process elements** — framework activities, SE actions, tasks, work products, quality assurance, and change control mechanisms for each project.
- each process model also prescribes **a process work flow** — that is, the manner in which the process elements are interrelated to one another.

strive for structure and order in software development.

activities and tasks occur sequentially with defined guidelines for progress.

Waterfall Process Model



Classic lifecycle model

Document-driven: phase & continuous

Pros

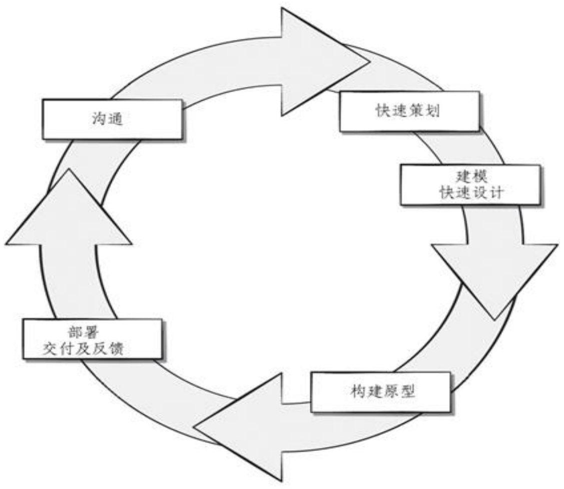
- It is easy to understand and plan.
- It works for well-understood small projects.
- Analysis and testing are straightforward.

Cons

- It does not accommodate change well.
- Testing occurs late in the process.
- Customer approval is at the end.

Prototyping Process Model

What are the phases of the prototyping model for software development?



gathering requirements begins with **communication** by having the stakeholders and developer meet and identify whatever the overall objectives and requirements they can.

quick plan and design modeling focus on representation of the software that will be visible to the customer.

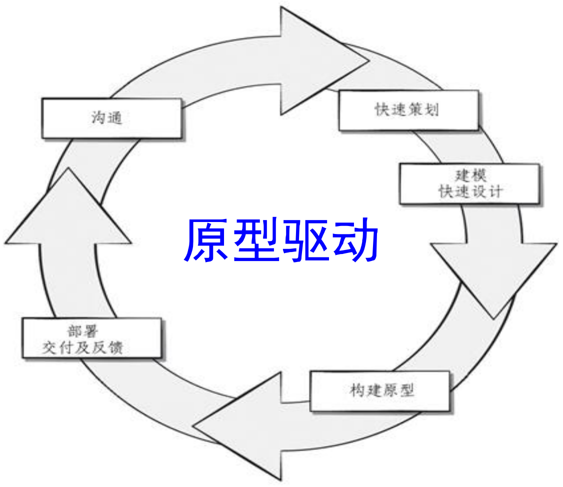
construct and deployed a prototype by the developer and evaluated by the customer and used to refine the requirements.

Iteration occurs and the **prototype is tuned** to satisfy the customer's needs.

Prototyping Process Model

Pros

- Reduced impact of requirement changes.
- Customer is involved early and often.
- Works well for small projects.
- Reduced likelihood of product rejection.



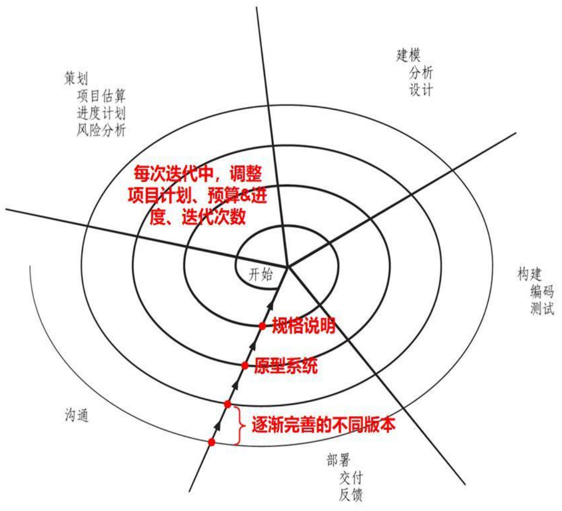
Cons

- Customer involvement may cause delays.
- Temptation to “ship” a prototype.
- Work lost in a throwaway prototype.
- Hard to plan and manage.

Spiral Process Model

Why the spiral model is the best approach to software development in a modern context?

- timelines for the development of modern software are getting shorter and shorter
- customers are becoming more diverse and making the understanding of requirements even harder
- changes to requirements are becoming even more common before delivery.



- need a way to provide incremental or evolutionary delivery to better adapt to uncertainty, allows the delivery of partial solutions in an orderly and planned manner, most importantly, reflects what really happens when complex systems are built.

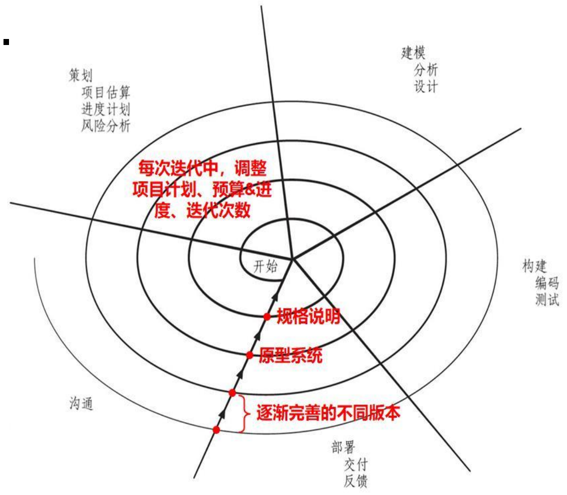
Spiral Process Model

演化模型

风险驱动：瀑布与原型的结合

Pros

- Continuous customer involvement.
- Development risks are managed.
- Suitable for large, complex projects.
- It works well for extensible products.



Cons

- Risk analysis failures can doom the project.
- Project may be hard to manage.
- Requires an expert development team.

Unified Process Model

用例驱动，阶段并发

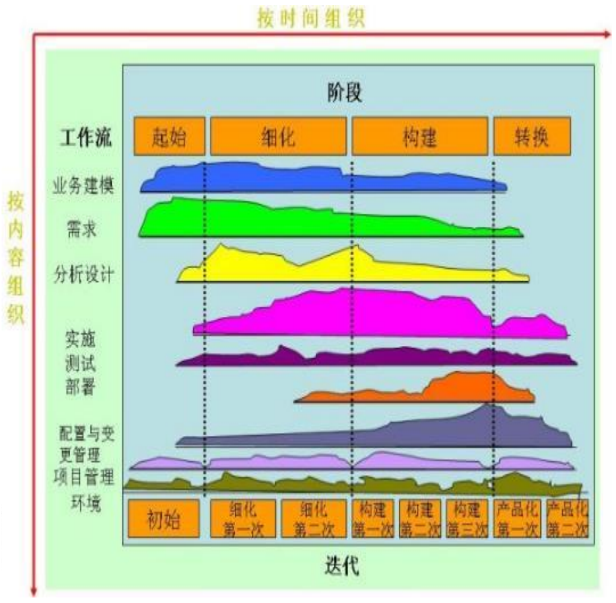
Pros

Quality documentation emphasized.

Continuous customer involvement.

Accommodates requirements changes.

Works well for maintenance projects.



Cons

Use cases are not always precise.

Tricky software increment integration.

Overlapping phases can cause problems.

Requires expert development team.

"Agility"

Process Models

major failing for traditional process

two issues:

- If process models **strive for** structure and order, are they suitable for a software world that thrives on change ?
- If we **reject** traditional process models and **replace** them with something less structured, do we make it impossible to achieve coordination and coherence in software work ?

failing for traditional process model: *forget the **frailties** of* software engineers.

- They are not robots, and exhibit great variation in working styles and significant differences in skill level, creativity, orderliness, consistency, and spontaneity.
- Some communicate well in written form, others do not.

↓

Agility Process Models

Agile Manifesto

A process model is working, it is necessary to adopt a realistic mechanism to ensure discipline while treating software engineers with tolerance.

What are the tradeoffs proposed by the "Agile Manifesto" ?

1. individuals and interactions **over** processes and tools,
2. working software **over** comprehensive documentation,
3. customer collaboration **over** contract negotiation, and
4. responding to change **over** following a plan.

Agile is not **antithetical** to **solid SE practice**, and can be applied as an **overriding philosophy** for all software work.

What is Agility ?

- Effective (rapid and adaptive) response to change.
- Effective communication among all stakeholders.
- Drawing the customer onto the team.
- Organizing a team so that it is in control of the work performed.
- Rapid, incremental delivery of software.

What roles do customers and end-users play in an agile team?

- Customers and end-users, participating in agile process teams as full **collaborators**.
- Customers and end-users, **creating use cases** and **providing the business information** for software features and functionalities to be developed.
- Customers and end-users, also **provide feedback** on operational prototypes during software incremental delivery.

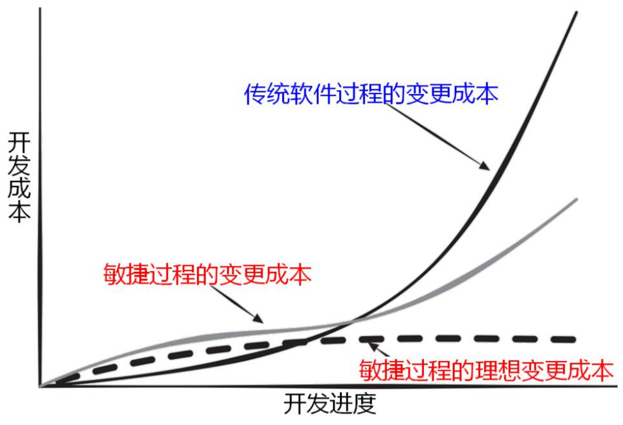
Agility and Cost of Change

X-axis: schedule progress

Y-axis: cost

3 curves:

solid black curve: cost of change using conventional process shows a slow beginning and then displays a peak towards the end.



dashed black curve: idealized cost of change using agile process displays a slow beginning and a constant rate after that.

solid gray curve: cost of change using agile process shows a slow and steady increase.

What is an Agile Process?

adaptive agile **incremental** development strategy

1. Driven by customer requirement description (scenarios).
2. Customer **feedback** is frequent and acted on.
3. Recognizes that plans are short-lived.
4. Develops iteratively with a heavy emphasis on construction.
5. Delivers multiple ‘software increments’ as executable prototypes.
6. Adapts as project or technical changes occur.

Project’ unpredictability

What are the key assumptions regarding projects that each agile process must address ?

- It is difficult to predict "which requirements will persist, which will change, and how customer priorities will change as the project proceeds".
- It is difficult to predict "how much design is necessary" before construction is used to prove the design, so design and construction are need be interleaved.
- Analysis, design, construction, and testing are not always predictable processes and this makes planning difficult.

Agility Principles

不同项目、不同过程模型，原则的权重不同

	敏捷原则
1	最优先做的是通过**尽早、持续地交付有价值的软件**使客户满意
2	即使**开发后期也欢迎需求变更**，利用变更为客户创造竞争优势
3	经常交付可运行软件，**交付时间间隔越短越好**（几个月\星期）
4	项目开发期间，业务人员和开发人员必须**天天都在一起**工作
5	围绕有**积极性的个人**构建项目，提供环境和支持，信任他们能够完成工作
6	团队内部，最富有效果和效率的信息传递方法是**面对面交谈**
7	**可运行软件**是进度的首要度量标准
8	敏捷过程提倡**可持续的开发速度**，责任人（sponsor）、开发者和用户应该能够长期保持稳定的开发速度
9	不断关注**优秀的技能**和**好的设计**会增强敏捷能力
10	**简单**（使不必做的工作最大化的艺术）是必要的
11	最好的架构、需求和设计出自于**自组织**团队
12	每隔一定时间，团队会反省如何才能更有效地工作，并相应**调整自己的行为**

agile philosophy

What are the key issues stressed by an agile philosophy?

1. Emphasis on rapid delivery of software that satisfies the customer
2. Recognition that change represents opportunity
3. Communication and collaboration between team members and customers
4. The importance of self-organizing teams

Scrum Framework

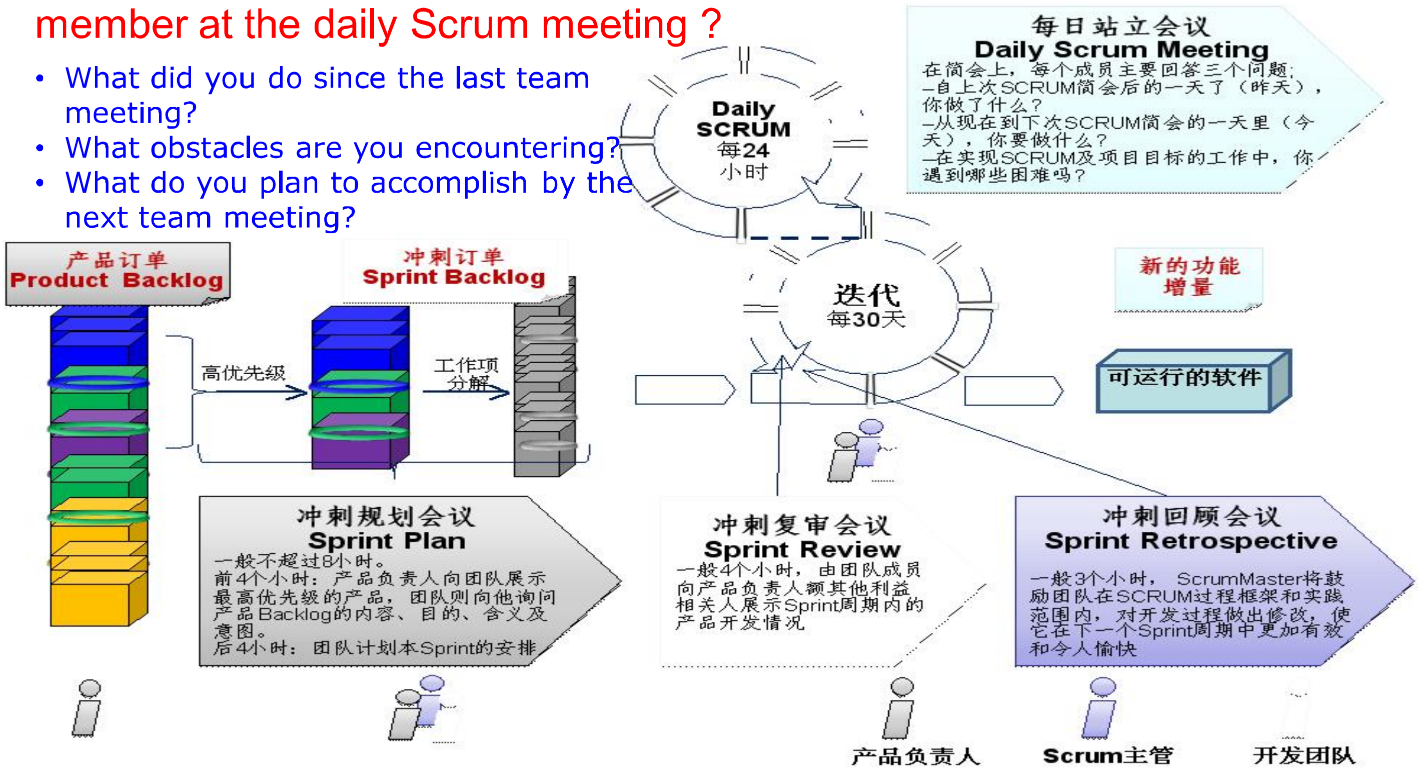
Pros	Cons
Product owner sets priorities. Team owns decision making. Documentation is lightweight. Supports frequent updating.	Difficult to control the cost of changes. May not be suitable for large teams. Requires expert team members.

- Backlog Refinement Meeting — Developers work with stakeholders to create product backlog.
- Sprint Planning Meeting — Backlog partitioned into "sprints" derived from backlog and next sprint defined.
- Daily Scrum Meeting — Team members synchronize their activities and plan work day (15 minutes max).
- Sprint Review — Prototype "demos" are delivered to the stakeholders for approval or rejection.
- Sprint Retrospective — After sprint is complete, team considers what went well and what needs improvement.

Scrum Framework

What are the three questions that should be answered by each team member at the daily Scrum meeting ?

- What did you do since the last team meeting?
- What obstacles are you encountering?
- What do you plan to accomplish by the next team meeting?

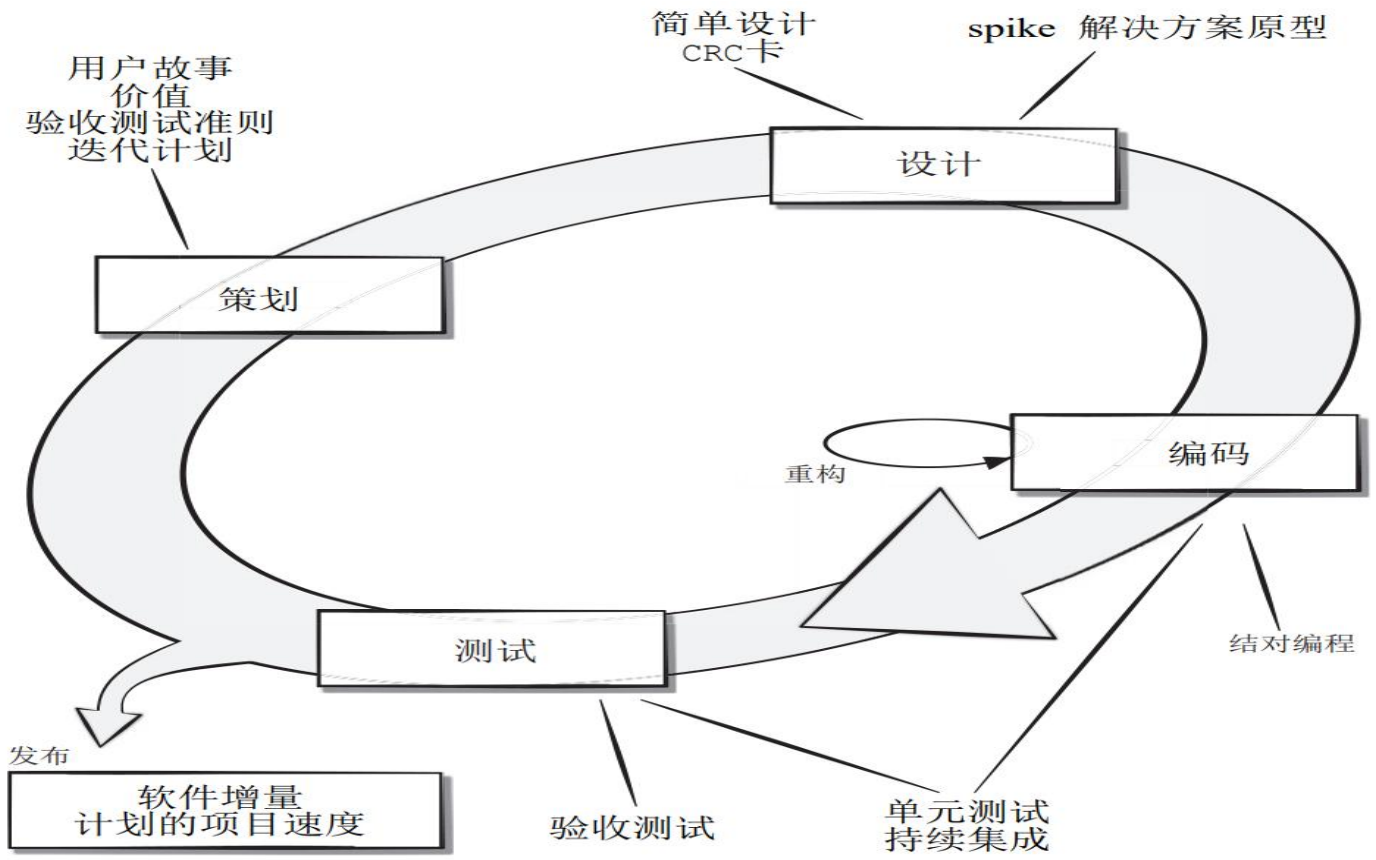

每日站立会议
Daily Scrum Meeting
在简会上，每个成员主要回答三个问题:
–自上次SCRUM简会后的一天了（昨天），你做了什么？
–从现在到下次SCRUM简会的一天里（今天），你要做什么？
–在实现SCRUM及项目目标的工作中，你遇到哪些困难吗？
Daily SCRUM
每24小时
迭代
每30天
新的功能增量
可运行的软件
产品订单
Product Backlog
冲刺订单
Sprint Backlog
高优先级
工作项分解
冲刺规划会议
Sprint Plan
一般不超过8小时。
前4个小时：产品负责人向团队展示最高优先级的产品，团队则向他询问产品Backlog的内容、目的、含义及意图。
后4小时：团队计划本Sprint的安排
冲刺复审会议
Sprint Review
一般4个小时，由团队成员向产品负责人额其他利益相关人展示Sprint周期内的产品开发情况
冲刺回顾会议
Sprint Retrospective
一般3个小时，ScrumMaster将鼓励团队在SCRUM过程框架和实践范围内，对开发过程做出修改，使它在下一个Sprint周期中更加有效和令人愉快
产品负责人
Scrum主管
开发团队

Extreme Programming (XP)

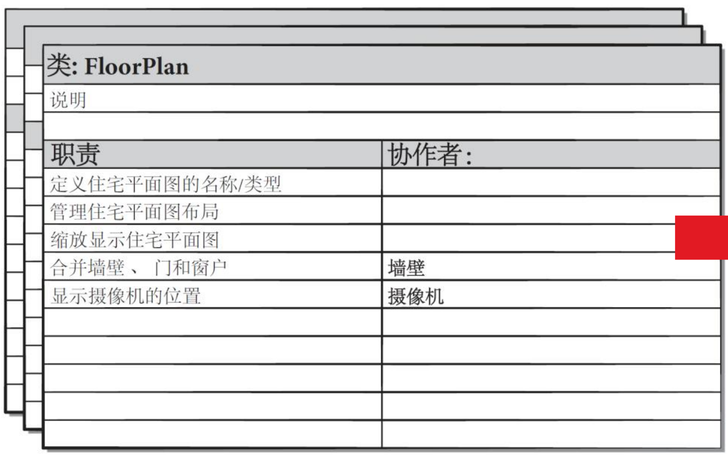
Pros	**Cons**
customer involvement.	Temptation to "ship" a prototype.
rational plans and schedules.	frequent meeting, increasing cost.
High developer commitment.	Allows for excessive changes.
Reduced likelihood of rejection.	Depends on highly skilled team.

- XP Planning – Begins with user stories, team estimates cost, stories grouped into increments, commitment made on delivery date, computer project velocity.
- XP Design – Follows KIS principle, encourages use of CRC cards, design prototypes, and refactoring.
- XP Coding – construct unit tests before coding, uses pair.
- XP Testing – unit tests executed daily, acceptance tests define by customer.

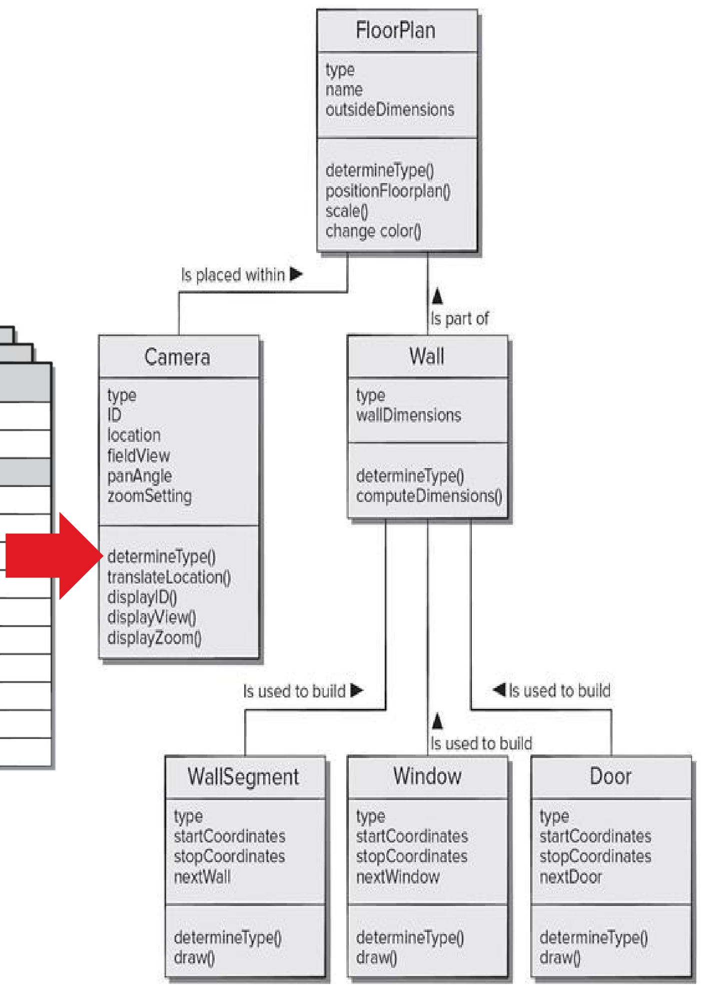
XP 框架



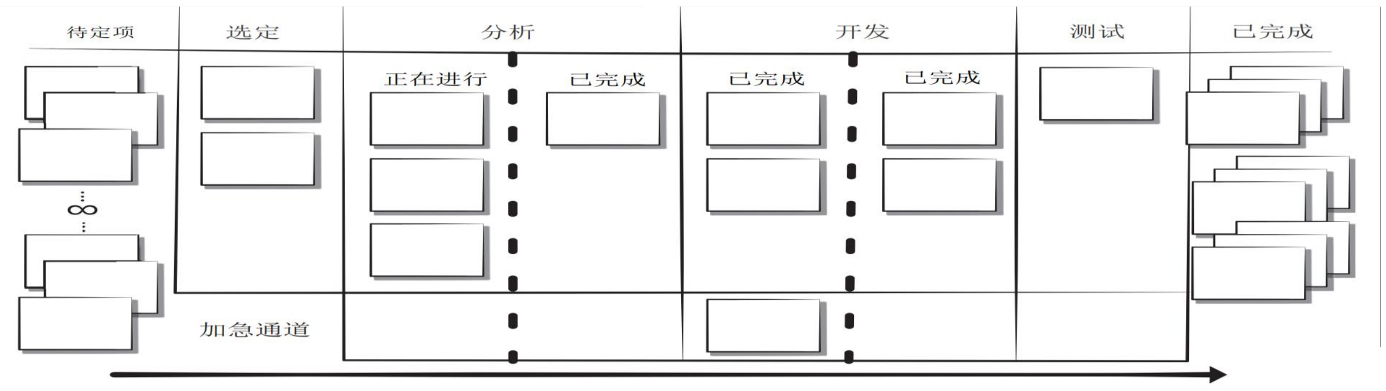
CRC卡



类: FloorPlan	
说明	
职责	协作者:
定义住宅平面图的名称/类型	
管理住宅平面图布局	
缩放显示住宅平面图	
合并墙壁 、 门和窗户	墙壁
显示摄像机的位置	摄像机

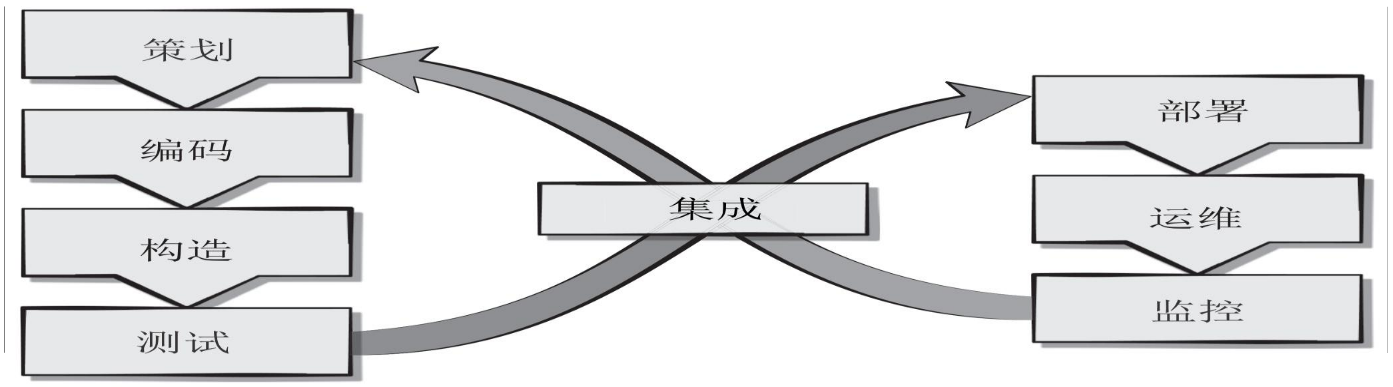


Kanban: Visualizing monitor



- Visualizing workflow using a Kanban board. Limiting the amount of work in progress at any given time.
- Managing workflow to reduce waste by understanding the current value flow.
- Making process policies explicit and the criteria used to define “done”.
- Focusing on continuous improvement by creating feedback loops where changes are introduced.
- Make process changes collaboratively and involve all stakeholders as needed.

DevOps



- **Continuous development.** Software delivered in multiple sprints.
- **Continuous testing.** Automated testing tools used prior to integration.
- **Continuous integration.** Code pieces with new functionality added to existing code running code.
- **Continuous deployment.** Integrated code is deployed to the production environment.
- **Continuous monitoring.** Team operations staff members proactively monitor software performance in the production environment.

“Recommended” Process Models

“推荐的”过程模型

Process Adaptation

What are the possible differences in process models for different projects? (What factors need to be considered when selecting a process model for a project?)

- Overall flow of activities, actions, and tasks and the interdependencies among them.
- Degree to which actions and tasks are defined within each framework activity.
- Degree to which work products are identified and required.
- Manner which quality assurance activities are applied.
- Manner in which project tracking and control activities are applied.
- Overall degree of detail and rigor with which the process is described.
- Degree to which the customer and other stakeholders are involved with the project.
- Level of autonomy given to the software team.
- Degree to which team organization and roles are prescribed.

过程适应性调整：项目成功的关键

项目性质、规模、团队成员素养不同，项目过程需调整

- 不同项目之间的过程模型，可能存在的差异是什么？（为项目选择过程，需考虑什么因素 ？）

过程的适应性调整	
1	活动、 动作和任务的总体流程以及相互依赖关系
2	在每一个框架活动中， 动作和任务细化的程度
3	工作产品的定义和要求的程度
4	质量保证活动应用的方式
5	项目跟踪和控制活动应用的方式
6	过程描述的详细程度和严谨程度
7	客户和利益相关者对项目的参与程度
8	软件团队所赋予的自主权
9	团队组织和角色的明确程度

Adapting process model

Every project needs a “**road map**” or “**generic process**” of some kind.

- Every project is different, and every team is different.
- No single **SE** framework is appropriate for every project product.

Any **road map** or **generic process** should be based on best industry practices.

Developers and stakeholders **adapt** generic process models and tailor them to fit the current project, the skills of the team members, and the user needs.

调整过程模型

每个软件项目，都需要某种“**路线图**”或“通用过程”

- 每个项目都是独特的，每个开发团队也是独特的
- 没有哪种过程框架，适合所有的软件产品

任何路线图或通用过程，都应以**最佳行业实践**作为基础

开发人员和利益相关者**调整通用流程模型**，使其适合当前项目、团队成员的技能和用户的需求

Principles for Organizing Projects

1. It is risky to use a linear process model without ample feedback.
2. It is never possible nor desirable to plan big up-front requirement gathering.
3. Up-front requirements gathering may not reduce costs or prevent time slippage.
4. Appropriate project management is integral to software development.
5. Documents should evolve with the software and should not delay the start of construction.
6. Involve stakeholders early and frequently in the development process.
7. Testers need to become involved in the process prior to software construction.

Some Ideas of Adapting process model

瀑布模型： 不适于应对需求变更，用户反馈仅限于项目开始和结束时，模型建议在编码或测试前完成所有分析和设计工作

增量模型： 是创建可适应性过程的基础：**鼓励利益相关者尽早且经常参与项目，降低开发出不被客户接受产品的风险**

- 模型善于处理**需求不确定**问题，允许**有控制地将需求扩展到演化的增量中**，但项目团队规模小，不适于大项目（除非项目可分解为子系统实施），尤其是Scrum，通过每日例会，保证项目不会偏离用户目标
- 螺旋模型：风险评估的**演化原型模型**，依赖于适量利益相关者参与项目，每次迭代中创建可扩展原型，文档与每个原型同步，强调早期测试，适于大项目，缺点是项目起始难以确定项目范围

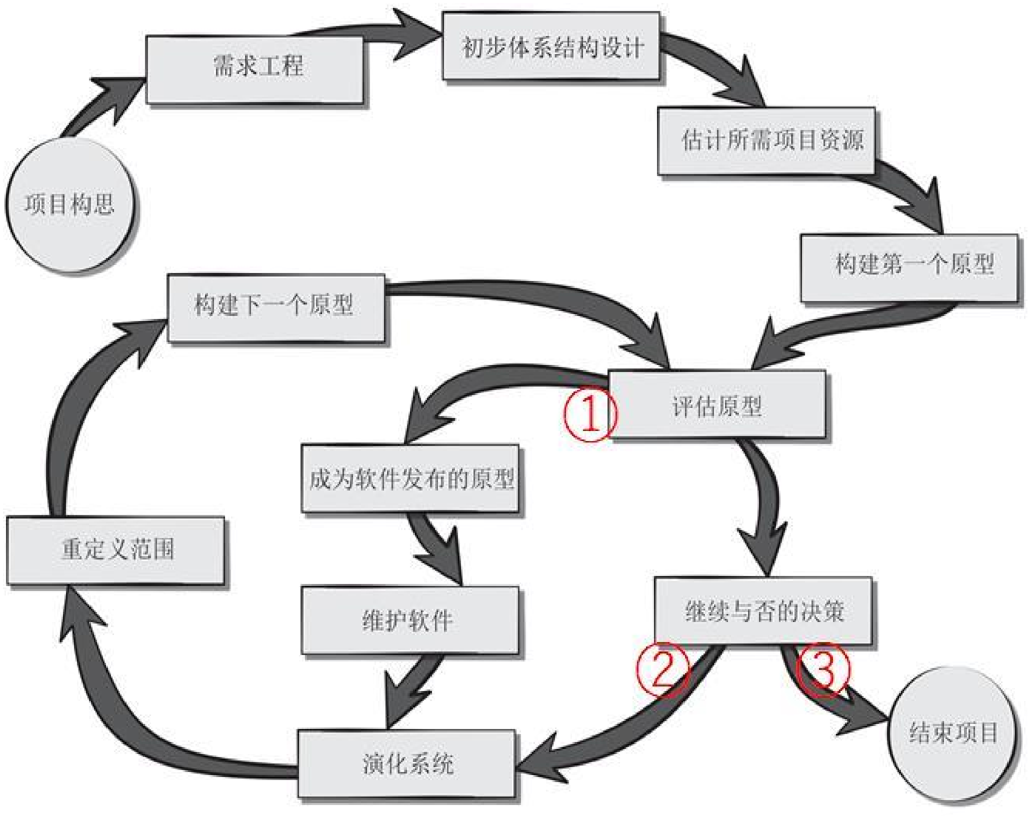
实际项目过程，吸收**增量模型**特性：构造一个适应性强的模型，**在每个周期中都内置灵活螺旋式的原型演化过程**

Incremental Prototype Design

推荐的软件过程步骤	
1 需求工程 ✧ 获取所有利益相关者的用户故事。 ✧ 让利益相关者给出用户故事的接受标准。	**5 评价原型** ✧ 在设计原型时创建测试用例。 ✧ 使用适当的用户测试原型。 ✧ 获取利益相关者的反馈，以便在修订过程中使用。
2 初步体系结构设计 ✧ 利用纸质原型和模型。 ✧ 评估使用非功能性需求的备选方案。 ✧ 记录体系结构设计决策。	**6 继续与否的决策** ✧ 确定当前原型的质量。 ✧ 修改完成开发的时间和成本估算。 ✧ 确定不能满足利益相关者期望的风险。 ✧ 获得继续开发的承诺。
3 估计所需项目资源 ✧ 使用历史数据估算完成每个用户故事的时间。 ✧ 将用户故事组织成冲刺。 ✧ 确定完成产品所需的冲刺数。 ✧ 在添加或删除用户故事时修改时间估算。	**7 演化系统** ✧ 定义新原型范围。 ✧ 构建新原型。 ✧ 评价新原型，包括回归测试。 ✧ 评估与持续演化相关的风险。
4 构建第一个原型 ✧ 选择对利益相关者最重要的用户故事子集。 ✧ 把创建纸质原型作为设计过时程的一部分。 ✧ 设计具有输入和输出的用户界面原型。 ✧ 实现第一个原型所需的算法。 ✧ 考虑部署方案。	**8 发布原型** ✧ 进行验收测试。 ✧ 记录发现的缺陷。 ✧ 与管理层互通质量风险。
	9 维护软件 ✧ 变更前理解代码。 ✧ 变更后测试软件。 ✧ 记录变更。 ✧ 告知利益相关者已知的缺陷和风险。

Incremental Prototype Design

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的决策
7	原型演化
8	发布原型
9	软件维护



Requirements Engineering

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的决策
7	原型演化
8	发布原型
9	软件维护

Collect user stories from all stakeholders.

- Prototype: Focus on visible user behavior and short-term goals.

develop acceptance criteria for user stories.

- Prototyping improves communication among stakeholders.
- Prototype is a tangible implementation of the project plan.
- Stakeholders review the prototype and refer to it to propose changes in more specific terms.

Requirements Engineering

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的决策
7	原型演化
8	发布原型
9	软件维护

Why should requirements engineering be an iterative process?

- Two facts of requirement definition: ① It is impossible for stakeholders to describe an entire system before seeing the working software; ② it is difficult for stakeholders to describe quality requirements needed for the software before seeing it in action as a prototype.
- Developers must recognize that requirements will be added and refined as the increments are created.
- Using prototypes may increase the volatility of the requirements if stakeholders are not focused on getting things right the first time.

Requirements Engineering

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的决策
7	原型演化
8	发布原型
9	软件维护

Agile Requirements Definition

1	Encourage active stakeholder participation by matching their availability and valuing their input.
2	Use simple models (e.g., Post-it notes, fast sketches, user stories) to reduce barriers to participation.
3	Take time to explain your requirement representation techniques before using them.
4	Adopt stakeholder terminology, and avoid technical jargon whenever possible.
5	Use a breadth-first approach to get the big picture of the project done before getting bogged down in details.
6	Allow the development team to refine (with stakeholder input) requirement details "just in time" as user stories are scheduled to be implemented.
7	Treat the list of features to be implemented like a prioritized list, and implement the most important user stories first.
8	Collaborate closely with your stakeholders and only document requirements at a level that is useful to all when creating the next prototype.
9	Question the need to maintain models and documents that will not be referred to in the future.
10	Make sure you have management support to ensure stakeholder and resource availability during requirements definition.

Protype Architectural Design

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的决策
7	原型演化
8	发布原型
9	软件维护

Use paper prototypes and models

- Evaluate alternative solutions for using NFRs
- Record architecture design decisions

Why architectural decision is making critical to the success of software system?

- architectural decision during prototype design can help to understand the requirements.
- architecture determines software qualities and impacts the system throughout its life cycle.
- architectural decisions can reduce design uncertainty when defining requirements, facilitate architects to use architectural knowledge and reduce the probability of design errors.
- evaluating prototype, documenting early decisions, reasons, and lessons learned, helps to improve project processes before starting the next increment.

Protype Architectural Design

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的 决策
7	原型演化
8	发布原型
9	软件维护

Elements of Agile Architectural Design

- Focus on key quality attributes and incorporate them into prototypes as they are constructed.
- Keep in mind that successful products combine customer-visible features and the infrastructure needed to enable them.
- Agile architectures enable code maintainability and evolvability if attention is paid to architectural decisions and quality issues.
- Managing and synchronizing dependencies among functional and architectural requirements is needed to ensure evolving architecture will be ready for future increments.

Resource Estimation

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的 决策
7	原型演化
8	发布原型
9	软件维护

How to estimate resource ?

1	Team should use historic data to develop an estimate of number of days needed to complete each of user stories known at the start of the project.
2	Loosely organize the user stories into sets that will make up each sprint planned to complete a prototype.
3	Sum the number of days to complete each sprint to provide an estimate for the duration of the total project.
4	Revise the estimate as requirements are added to the project or prototypes are delivered and accepted by the stakeholders.

注意： 可由开发人员数量与同时完成的用户故事数量，调整过程模型，但：开发人员数量增加一倍不会减半开发时间

- **项目范围没有完全确定时，无法估计完成成本**
- 定义项目范围，须了解初始需求、项目类型和团队经验

First Prototype Guidelines

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的 决策
7	原型演化
8	发布原型
9	软件维护

Select subset of user stories most important to stakeholders (Define prototype scope).

1. Transition from paper prototype to software design.
2. Prototype a user interface.
3. Create a virtual prototype.
4. Add input and output to your prototype.
5. Engineer your algorithms.
6. Test your prototype.
7. Prototype with deployment in mind.

First Prototype Guidelines

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的 决策
7	原型演化
8	发布原型
9	软件维护

Goal of prototype: transform system objectives and user storys into detailed functional statements.

Why should you create prototypes with deployment in mind?

- first prototype can to prove initial architectural is a feasible to delivering the required functionality while satisfying the performance constraints.
- prototype with deployment in mind is very important because can avoid taking shortcuts that lead to creating software that will be hard to maintain.
- As iterative prototype development occurs, should carefully consider the software architectural choices you make.
- It is very expensive to change the architecture of a software application once it has been released to end users around the globe.

Prototype Evaluation

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的决策
7	原型演化
8	发布原型
9	软件维护

- Create test cases as prototype is being designed.
- Test prototype using appropriate users.
- Capture stakeholder feedback for use in revision process.

the best practice tips for gathering feedback on your prototype.

1. Provide scaffolding when asking for prototype feedback.
2. Test your prototype on the right people.
3. Ask the right questions.
4. Be neutral when presenting alternatives to users.
5. Adapt while testing.
6. Allow the user to contribute ideas.

原型评价

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的决策
7	原型演化
8	发布原型
9	软件维护

Why is it important to test a protype using the stakeholders ?

- getting feedback through testing the prototype, and risk assessment based on "user satisfaction" and "whether the budget is exceeded and delivery is delayed", making a decision on whether to create the next prototype.
- Using stakeholders is essential to reduce the risk of developing the wrong product.
- Having developers do all the testing is not wise because they are not likely to be the representative of the intended user population.
- It’s important to have the right mix of users (e.g., novice, typical, and advanced) to give you meaningful feedback on the prototype.

Go, No-Go Decision

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的决策
7	原型演化
8	发布原型
9	软件维护

1. Determine the quality of prototype follows the evaluation process.
2. Revise cost estimates and schedule based on the changes for completing prototype.
3. Determine the risk of failing to meet stakeholder expectations.
4. Get commitment to continue development.

Prototype Evolution

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的决策
7	原型演化
8	发布原型
9	软件维护

After review, the prototype may be 3 possible trends in the project:

1. Prototype becomes a system candidate
2. Evolve the prototype to the next version:
   ① Define new prototype scope
   ② Constructing new prototype
   ③ Evaluate new prototype and include regression testing.
   ④ Assess risks associated with continuing evolution.
3. Risk assessment failed, project terminated

Prototype Evolution

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的决策
7	🟩 原型演化
8	发布原型
9	软件维护

How to test a new prototype ?

- Testing should be performed using test cases created during design process before programming was completed.
- Each user story has an acceptance criteria attached to it and it should guide the creation of the test cases to ensure the prototype meets customer needs.
- Prototypes need to be tested for defects and performance issues.
- Regression testing ensures that adding new features to evolutionary prototypes does not accidentally break features working correctly in the previous prototype.

Release Candidates

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的决策
7	原型演化
8	🟩 发布原型
9	软件维护

- A prototype considered as a release candidate is subjected to **user acceptance testing**.
- Functional testing and performance testing, and **document defects identified**.
- Organize user feedback according to the user-visible functions via the user interface, and **share quality risks with management**.

基于验收标准，只能证明测试用例运行正确，不能保证产品没有错误

Software Maintenance

1	需求工程
2	基础 体系结构设计
3	资源估计
4	首个原型构建
5	原型评价
6	继续与否的 决策
7	原型演化
8	发布原型
9	软件维护

What are the types of maintenance activities?

- Corrective maintenance - reactive modification to **repair problems** discovered after delivery.
- Adaptive maintenance - reactive modification to **keep** software **usable in a changing context** after delivery.
- Perfective maintenance - proactive modification to **provide new** user features, **better** program code structure, or **improved** documentation after delivery.
- Preventive maintenance – proactive modification to **correct product faults** before discovery by users after delivery.

类型		功能	对比	工作量分布
改正性维护	**反应性（被动）修改**	✧ 修复发现的问题	**不产生**新功能	21%
适应性维护		✧ 保持软件在不断变化的用户环境中可用		25%
完善性维护	**主动性修改**	✧ 提供新功能、更好代码结构或改进文档	**产生**新功能	4%
预防性维护		✧ 在用户发现故障之前，检测并纠正产品故障	**可能产生**新功能	50%