College of Computer Science, Chongqing University

Software Engineering

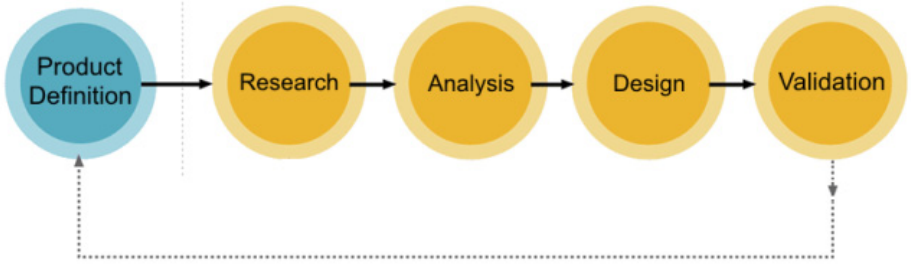
A Practitioner's Approach Seventh Edition

3 软件过程结构

zmqmail@cqu.edu.cn 13708390417

重庆大学 计算机学院 COLLEGE OF COMPUTER SCIENCE

产品(一般)设计流程概览



定义→调研→分析→设计→验证

Think 软件产品是不是也可以分步骤?

设计软件的过程是否可以归纳出几个步骤或者过程出来?

Think 什么是软件过程(过程模型)?

做什么?

框架(Framework)

为建造高质量软件所需要完成的活动、动作和任务的框架

软件过程 Software Process(Process Model)?

- I(this book) define a **software process** as a **framework** for the activities, actions, and tasks that are required to build high-quality software.

  为建造高质量软件所需要完成的活动、动作和任务的**框架**

- But software engineering also encompasses technologies that populate the process—technical methods and automated tools.

  软件工程还包含该过程中应用的技术——技术方法和自动化工具

- More important, software engineering is performed by creative, knowledgeable.

Think 框架活动是哪几个?

框架活动的5个?

软件生命周期 7个?

2.2 软件过程——框架活动

Framework Activities

框架活动

- **Communication**(沟通)
- **Planning**(策划)
- **Modeling**(建模)
  - Analysis of requirements
  - Design
- **Construction**(构建)
  - Code generation
  - Testing
- **Deployment**(部署)

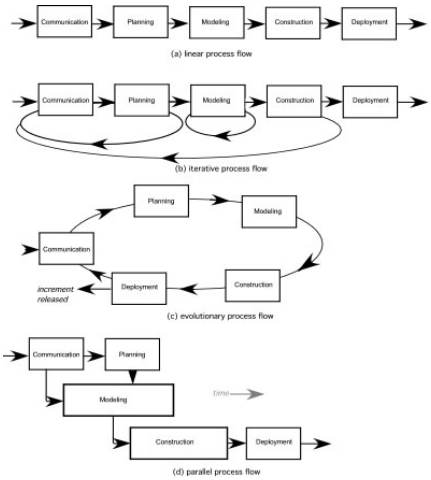
3.1 通用过程模型

Linear process flow
线性过程流

Iterative process flow
迭代过程流

Evolutionary process flow
演化过程流

Parallel process flow
并行过程流



Think 什么是过程流?

描述了在执行顺序和执行时间上如何组织框架中的活动、动作和任务。

3.2 定义框架的活动

- 个人负责的小型软件项目
  - 其需求简单明确，沟通也许仅仅是与合适的利益相关者的一个电话。因此，主要的动作是电话交流，这个动作所包括的主要工作任务集有:
    - 1. 通过电话与利益相关者取得联系。
    - 2. 讨论需求并做记录。
    - 3. 将笔记整理成一份简单的书面需求。
    - 4. 通过E-mail ， 请利益相关者审阅并批准。

大型的项目呢?

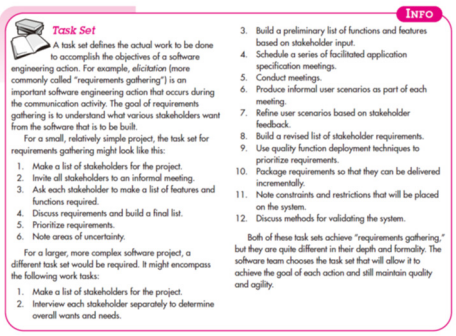
3.3 明确任务集

- 每一个软件工程动作由若干个任务集(task set)构成。
- 任务集:
  - 软件工程工作任务
  - 相关工作产品
  - 质量保证点
  - 项目里程碑组成

有模板不?

Task Set

工作任务、相关工作产品、质量保证点和项目里程碑等组成



INFO

Task Set

A task set defines the actual work to be done to accomplish the objectives of a software engineering action. For example, elicitation (more commonly called "requirements gathering") is an important software engineering action that occurs during the communication activity. The goal of requirements gathering is to understand what various stakeholders want from the software that is to be built.

For a small, relatively simple project, the task set for requirements gathering might look like this:

1. Make a list of stakeholders for the project.
2. Invite all stakeholders to an informal meeting.
3. Ask each stakeholder to make a list of features and functions required.
4. Discuss requirements and build a final list.
5. Prioritize requirements.
6. Note areas of uncertainty.

For a larger, more complex software project, a different task set would be required. It might encompass the following work tasks:

1. Make a list of stakeholders for the project.
2. Interview each stakeholder separately to determine overall wants and needs.
3. Build a preliminary list of functions and features based on stakeholder input.
4. Schedule a series of facilitated application specification meetings.
5. Conduct meetings.
6. Produce informal user scenarios as part of each meeting.
7. Refine user scenarios based on stakeholder feedback.
8. Build a revised list of stakeholder requirements.
9. Use quality function deployment techniques to prioritize requirements.
10. Package requirements so that they can be delivered incrementally.
11. Note constraints and restrictions that will be placed on the system.
12. Discuss methods for validating the system.

Both of these task sets achieve "requirements gathering," but they are quite different in their depth and formality. The software team chooses the task set that will allow it to achieve the goal of each action and still maintain quality and agility.

2.1.2 Identifying a Task Set

- A **Task Set** defines the actual work to be done to accomplish the objectives of a software engineering action.
  - A list of the task to be accomplished
    软件工程工作任务
  - A list of the work products to be produced
    相关工作产品
  - A list of the quality assurance filters to be applied
    质量保证点
  - A list of milestone to be planned
    项目里程碑

没有"模板"！

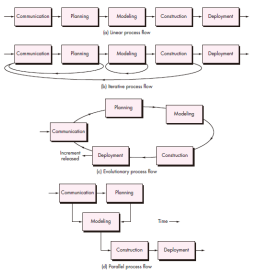
软件工程动作可以根据软件项目的特定需要和开发队伍的特点作适当的调整。

A Generic Process Model

通用过程模型

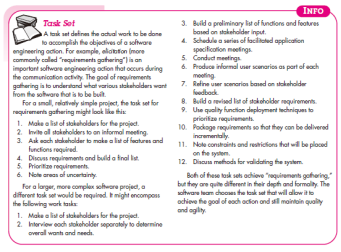
定义活动框架

What actions are appropriate for a framework activity.



明确任务集

A number of different task sets.



INFO

Task Set

A task set defines the actual work to be done to accomplish the objectives of a software engineering action. For example, elicitation (more commonly called "requirements gathering") is an important software engineering action that occurs during the communication activity. The goal of requirements gathering is to understand what various stakeholders want from the software that is to be built.

For a small, relatively simple project, the task set for requirements gathering might look like this:

1. Make a list of stakeholders for the project.
2. Invite all stakeholders to an informal meeting.
3. Ask each stakeholder to make a list of features and functions required.
4. Discuss requirements and build a final list.
5. Prioritize requirements.
6. Note areas of uncertainty.

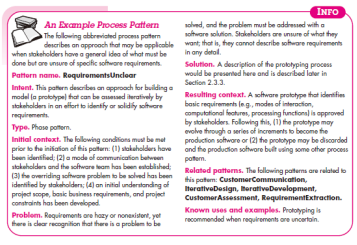
For a larger, more complex software project, a different task set would be required. It might encompass the following work tasks:

1. Make a list of stakeholders for the project.
2. Interview each stakeholder separately to determine overall wants and needs.
3. Build a preliminary list of functions and features based on stakeholder input.
4. Schedule a series of facilitated application specification meetings.
5. Conduct meetings.
6. Produce informal user scenarios as part of each meeting.
7. Refine user scenarios based on stakeholder feedback.
8. Build a revised list of stakeholder requirements.
9. Use quality function deployment techniques to prioritize requirements.
10. Package requirements so that they can be delivered incrementally.
11. Note constraints and restrictions that will be placed on the system.
12. Discuss methods for validating the system.

Both of these task sets achieve "requirements gathering," but they are quite different in their depth and formality. The software team chooses the task set that will allow it to achieve the goal of each action and still maintain quality and agility.

过程模式

Describes a process-related problem and suggests one or more proven solutions to the problem.



INFO

An Example Process Pattern

The following abbreviated process pattern describes an approach that may be applicable when stakeholders have a general idea of what must be done but are unsure of specific software requirements.

Pattern name. RequirementsUnclear

Intent. This pattern describes an approach for building a model (a prototype) that can be assessed iteratively by stakeholders in an effort to identify or solidify software requirements.

Type. Phase pattern.

Initial context. The following conditions must be met prior to the initiation of this pattern: (1) stakeholders have been identified; (2) a mode of communication between stakeholders and the software team has been established; (3) the overriding software problem to be solved has been identified by stakeholders; (4) an initial understanding of project scope, basic business requirements, and project constraints has been developed.

Problem. Requirements are hazy or nonexistent, yet there is clear recognition that there is a problem to be solved, and the problem must be addressed with a software solution. Stakeholders are unsure of what they want; that is, they cannot describe software requirements in any detail.

Solution. A description of the prototyping process would be presented here and is described later in Section 2.3.3.

Resulting context. A software prototype that identifies basic requirements (e.g., modes of interaction, computational features, processing functions) is approved by stakeholders. Following this, (1) the prototype may evolve through a series of increments to become the production software or (2) the prototype may be discarded and the production software built using some other process pattern.

Related patterns. The following patterns are related to this pattern: **CustomerCommunication, IterativeDesign, IterativeDevelopment, CustomerAssessment, RequirementExtraction.**

Known uses and examples. Prototyping is recommended when requirements are uncertain.

3.5 过程评估与改进

- 软件过程并不能保证软件按期交付，也不能保证软件满足客户要求，或是软件具备了长期滞留保证的技术特点。
- 软件过程本身也要进行评估：
  - Standard **CMMI** Assessment Method for Process Improvement (SCAMPI)
  - **CMM**-Based Appraisal for Internal Process Improvement (CBA IPI)
  - SPICE (**ISO**/IEC15504)
  - ISO 9001:2000 for Software

College of Computer Science, Chongqing University

Software Engineering

A Practitioner's Approach Seventh Edition

4 过程模型

zmqmail@cqu.edu.cn 13708390417

重庆大学 计算机学院 COLLEGE OF COMPUTER SCIENCE

惯用(传统)过程模型

惯用过程模型有时候被称为“传统”过程模型

称为“传统” → **过程流**

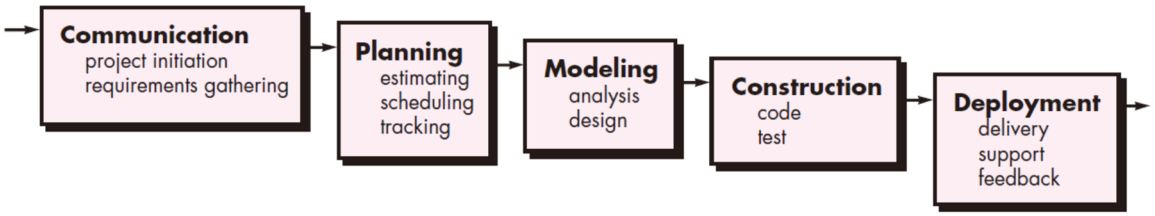
按照特定指引顺序进行。

规定了一套过程元素：
框架活动、软件工程动作、任务、工作产品、质量保证以及每个项目的变更控制机制。

过程流(工作流)：
过程元素相互之间关联的方式。

是否适应富于变化?

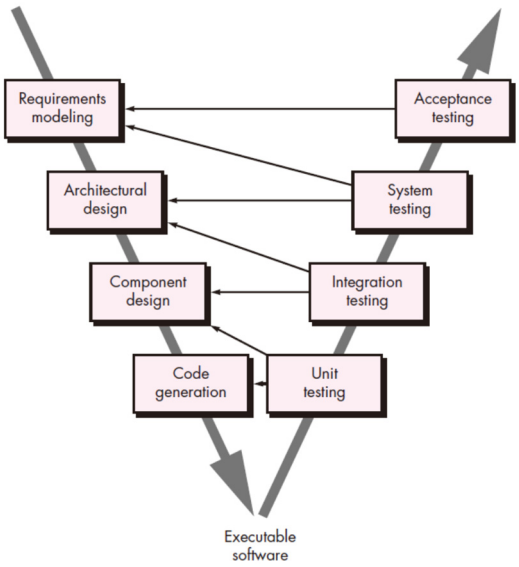
4.1.1 The Waterfall Model 瀑布模型



The **Waterfall Model**, sometimes called the **Classic Life Cycle**.

瀑布模型，又被称为经典生命周期，系统的、顺序的软件开发方法。

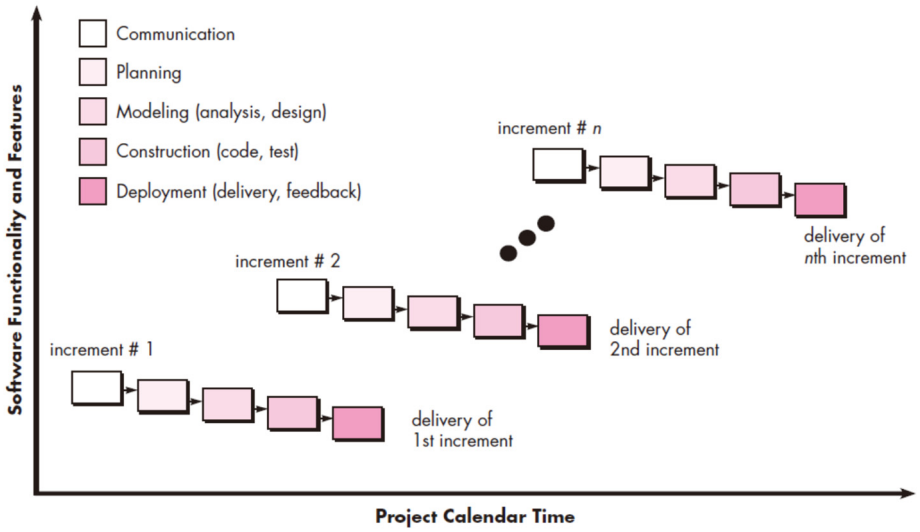
The V-Model V模型



瀑布模型的一个变体称为V模型(V-model)

Think 有什么问题吗?

4.1.2 增量过程模型 The Incremental Process Model

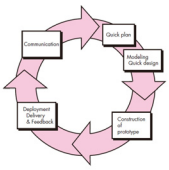


一套功能有限的软件产品，后续版本不断演化。

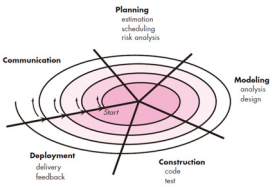
Think 增量过程模型与瀑布 or 其它模型的关系?

演化模型

Prototyping 原型法



The Spiral 螺旋模型

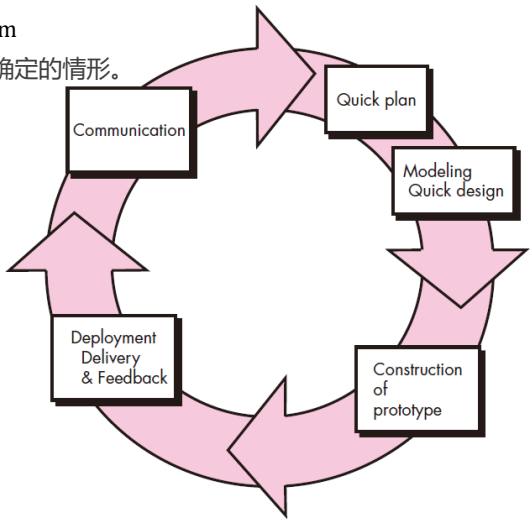


Concurrent 协同开发模型

4.1.3 原型法 Evolutionary Models: Prototyping

The prototyping paradigm

适合算法、交互等都不确定的情形。

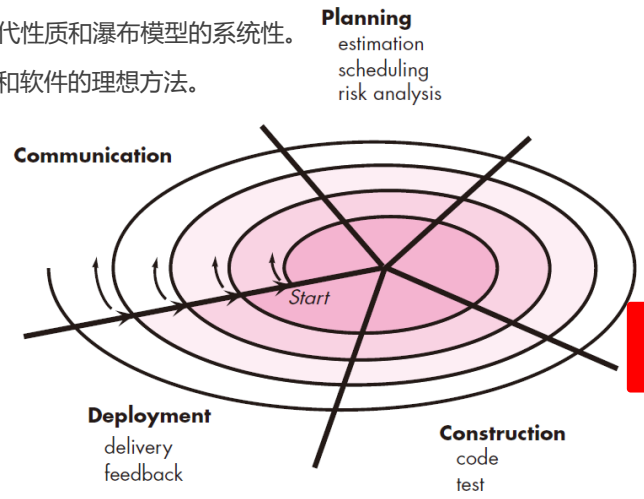


4.1.3 螺旋模型 Evolutionary Models: The Spiral

A typical spiral model

结合原型的迭代性质和瀑布模型的系统性。

开发大型系统和软件的理想方法。



风险驱动！

4.1.5 演化模型的最终评估 A Final Word on Evolutionary Processes

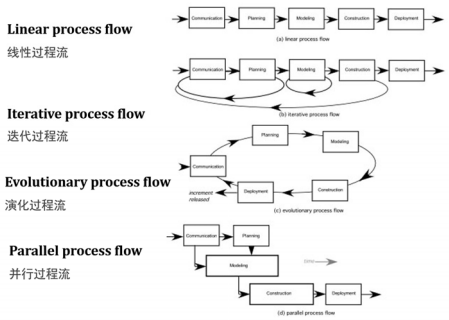
- **问题(不足)**Weaknesses
  - the uncertain number of cycles required to construct the product.
    周期数目不确定导致项目计划困难
  - do not establish the maximum speed of the evolution
    缺乏确定演化的最快速度：太快混乱，太慢影响效率
  - focused on flexibility and extensibility rather than on high quality
    软件过程应该侧重于灵活性和可延展性，而不是高质量。
    片面追求高质量而延长开始时间势必造成产品推迟交付，失去市场先机

4.2 其它模型 Still Other Process Models

- Component based development(基于构建的开发)
  - the process to apply when reuse is a development objective
- Formal methods(形式化方法模型)
  - emphasizes the mathematical specification of requirements
- AOSD(面向方面的软件开发)
  - provides a process and methodological approach for defining, specifying, designing, and constructing aspects
- **Unified Process(统一过程)**
  - a "use-case driven, architecture-centric, iterative and incremental" software process closely aligned with the Unified Modeling Language (UML)

Think 和3.1 通用过程模型的关系?

3.1 通用过程模型



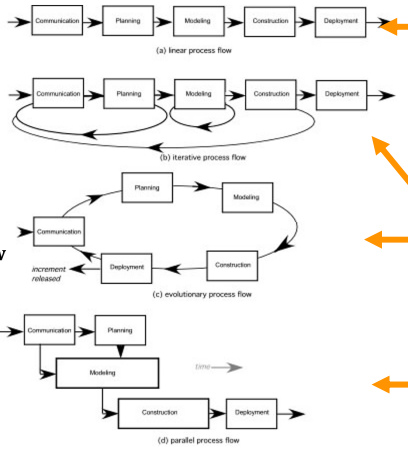
和3.1 通用过程模型的关系?

Linear process flow
线性过程流

Iterative process flow
迭代过程流

Evolutionary process flow
演化过程流

Parallel process flow
并行过程流



瀑布模型/V模型

原型开发
螺旋模型

增量过程模型

4.3 The Unified Process (UP) 统一过程

UML

Unified Modeling Language (UML)

- 不同的方法进行建模不利于开发者之间的交流。

- 1997 年，UML 被国际对象组织OMG采纳为**面向对象的建模语言**的国际标准，溶入了软件工程领域的新思想、新方法和新技术。

- UML不限于支持面向对象的分析与设计，还支持从需求分析开始的软件开发的全过程。

用例驱动，以架构为核心，迭代并且增量

Use Case Driven, Architecture-centric, Iterative and Incremental

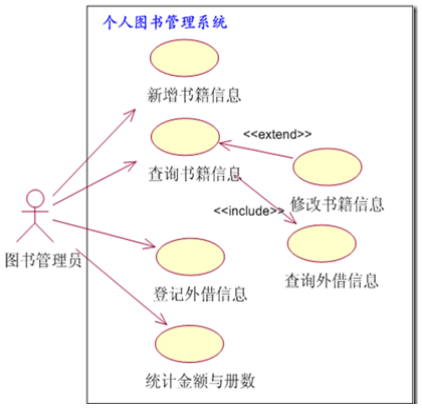
In their seminal book on the Unified Process, Ivar Jacobson, Grady Booch, and James Rumbaugh [Jac99]

4.3 The Unified Process (UP)

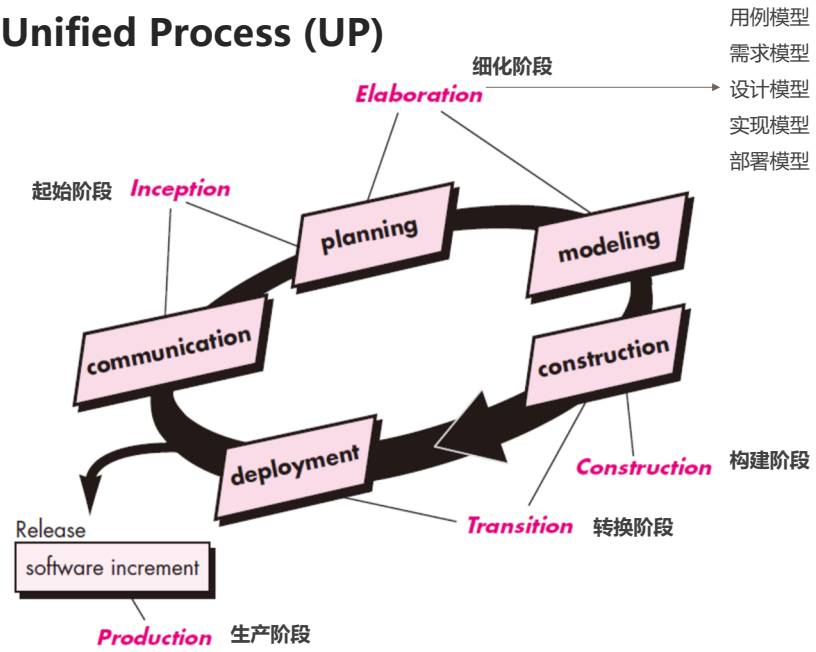
自学,考试重点+工作技能

Use Case Diagrams

用例图等



4.3 The Unified Process (UP)



Summary

describe

Framework

为建造高质量软件所需要完成的活动、动作和任务的框架。

Process Models

软件过程。

Summary

include

Framework

为建造高质量软件所需要完成的活动、动作和任务的框架

Process Models

过程模型

Process Flow

过程流

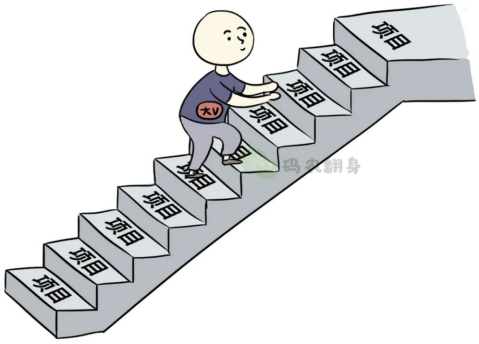
Task Set

任务集

各种模式的适合场景！

学习要求！

TDD/DDD/BDD/UDD... （统称xDD）



原创作者

若喜欢本漫画，记得扫码关注哟~

多个项目的历练让他们对某个领域的规律认识得非常深刻，他们又非常擅长总结和抽象，终于迎来了“啊哈”的时刻。

Think “我”能不成也创造一种模式?

灵魂拷问?

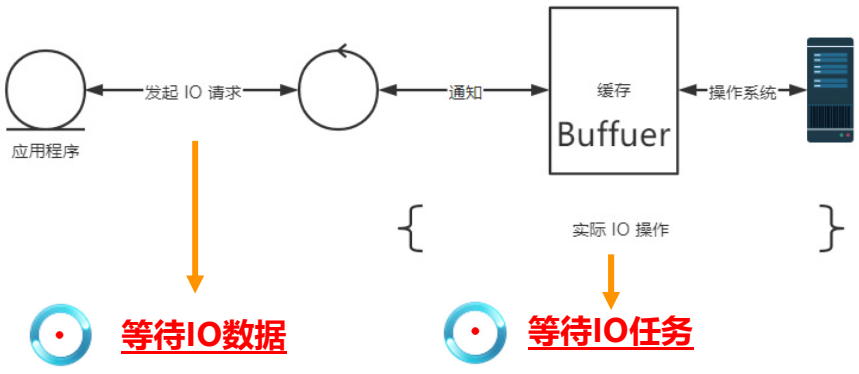
软件工程实践的精髓——解决问题

解决问题的本质，也是软件工程实践的精髓！

策略的重要性!

Think **BIO、NIO和AIO的区别?**

策略的重要性。



阻塞 **同步**

非阻塞 **异步**

同步非阻塞、异步阻塞、异步非阻塞

Question?

Q&A？

College of Computer Science, Chongqing University

Software Engineering

A Practitioner's Approach Seventh Edition

3 软件过程结构

zmqmail@cqu.edu.cn 13708390417

重庆大学 计算机学院 COLLEGE OF COMPUTER SCIENCE

College of Computer Science, Chongqing University

Software Engineering

A Practitioner's Approach Seventh Edition

4 过程模型

zmqmail@cqu.edu.cn 13708390417

重庆大学 计算机学院 COLLEGE OF COMPUTER SCIENCE