SA before DP

目录

Chapter 0 IT往事

1、 ACM是因为阿兰·麦席森·图灵做出了什么重要贡献而设立了图灵奖。

《论可计算数在判定问题中的应用》提出一种理想的计算器中的数学模型、提出图灵机模型、图灵测试

2、 从1966年到今天2020年，共有多少位图灵奖得主，他们大部分人的第一专业是什么？

颁发了55次、一共有74位获奖者、 数学

- 3、 在各大期刊会议中，我们经常能看到引文索引的概念，那么请问什么是引文索引。

  利用文献之间的引证关系，将文献的「参考文献表」编在一起，即「引文索引」比如cssci
- 4、 刊物和会议分类分为哪几类

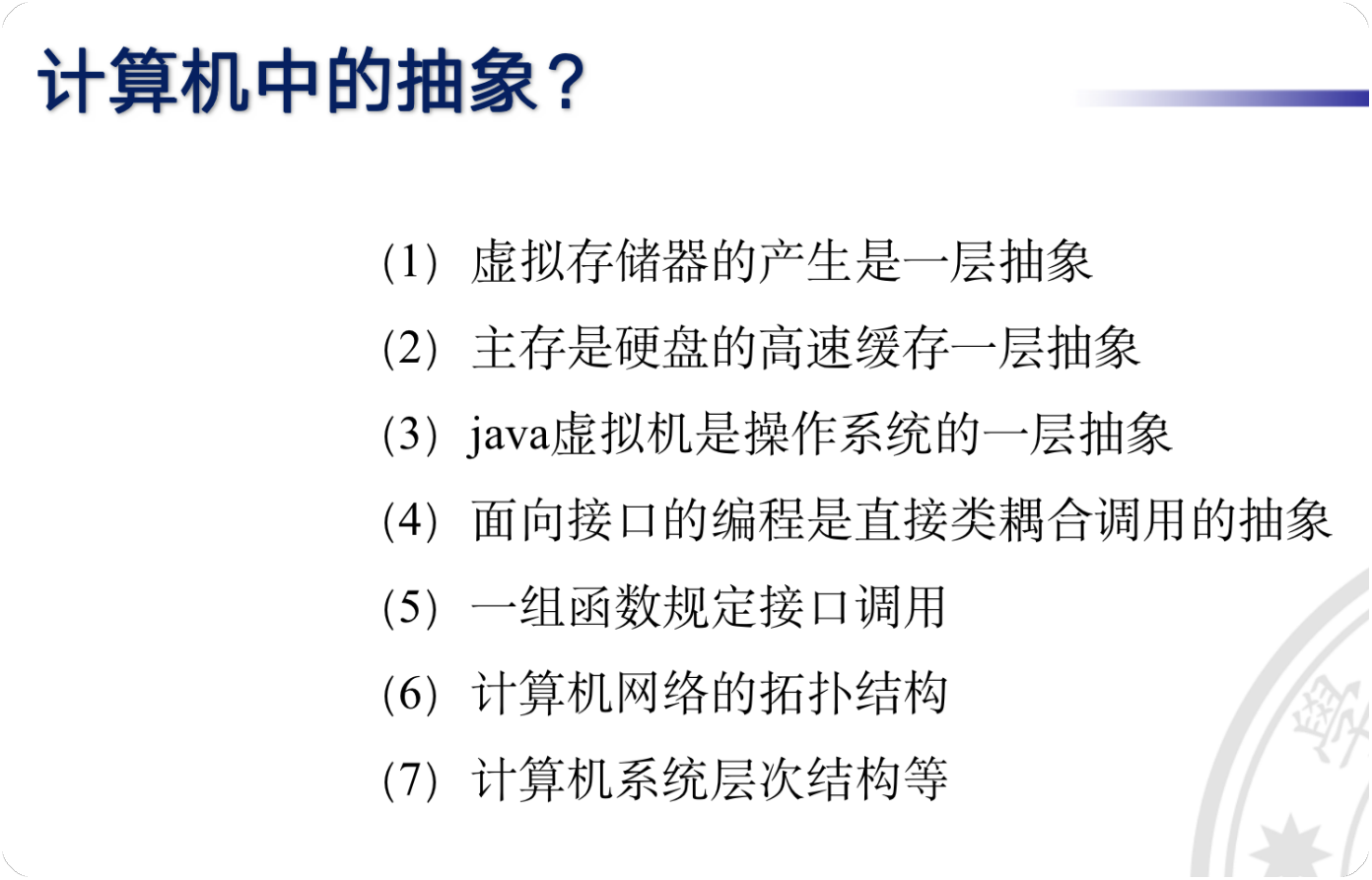
  A类表示国际上极少数的顶级刊物和会议，鼓励我国学者去突破；B类是指国际上著名和非常重要的会议、刊物，代表该领域的较高水平，鼓励国内同行投稿；C类指国际上重要、为国际学术界所认可的会议和刊物。

5、 三大科技文献检索系统是哪几个: SCI EI ISTP

抽象的意义：

- 抽象的意义在于忽略不是求解过程中必需的解，通过抽象能透过事物的表面现象抓住事物的本质

计算机中的抽象？



软件架构师应该知道的97件小事

- 取舍的艺术
- 关键问题可能不是出在技术上
- 天道酬勤
- 弃聪明，求质

软件工程中的角色：项目经理、开发人员、需求分析师、系统架构师、软件设计师、测试工程师、运维工程师、UI设计师

模型与算法：

- 模型是对某一个具体问题的 抽象描述 ，是一类问题的解题步骤。
- 算法是对具体问题的具体的解决方式

软件工程的定义：

IEEE：

1） 将系统化的、 规范的、 可量化的方法应用于软件的开发、 运行和维护， 即， 将工程应用于软件；

2） 在（1） 中所述方法的研究

软件工程发展：

1. 传统的软件工程
2. 面向对象的软件工程
3. 组件软件工程
4. SOA面向服务的软件工程：服务的请求者、服务的注册者、服务的提供者

Chapter 1 概述

1.1 软件体系结构

软件体系结构定义了**软件的局部和总体计算部件的构成**，以及这些**部件之间的相互作用关系**。

总的来看，体系结构是由结构和功能各异、相互作用的**部件集合**，按照**层次**构成的

--

体系结构是指整个系统构成的基本和主体形态。

软件体系结构是软件在设计构成上的基本、可供设计选择的形态和总体结构。

体系结构 = 组件 + 连接件 + 约束

软件设计的目标

> 软件设计的目标：在时间和各类环境资源的限制下，最大限度的满足用户的需求
>
> 1. 便于维护和升级，因而应该是模块化的
> 2. 设计应该是便于移植的（移植比重新设计花费要小的多）
> 3. 设计应该具有适应性
> 4. 设计过程应该受到理性的控制 Intellectual Control
> 5. 设计应该表现出概念的完整性（包括内在结构和外在表现）

1.2 当前的软件设计

1.3 软件设计的层次

每个层次的设计可单独进行

低层次级为了高层次级服务

每个层次都包括：

- 构成系统的部件
- 部件合成系统的规则

软件设计的层次

- 结构级
- 代码级
- 执行级

1.4 体系结构与软件的工程

实现体系结构指导下的软件工程设计，关键就是要建立软件结构的分析和构造方法。

软件复用技术是范例

1. 部件的抽取
2. 识别部件之工作环境

软件复用就是具有特定性能的模式或部件在不同应用中的多次频繁使用。

- 体现：部件、模式

体系结构对于软件具有宏观的视角，但体系结构设计的问题包含了软件结构的各个层次。

软件体系结构风格为大粒度的软件重用提供了**机会。**

1.5 软件体系结构的知识体系

Chapter 2 软件体系结构的研究与发展

2.1 软件工程设计和软件体系结构

软件设计的目标：在时间和各类环境资源的限制下，最大限度的满足用户的需求

1. 便于维护和升级，因而应该是模块化的
2. 设计应该是便于移植的（移植比重新设计花费要小的多）
3. 设计应该具有适应性
4. 设计过程应该受到理性的控制
5. 设计应该表现出概念的完整性（包括在结构和外在表现）

2.2 什么是软件体系结构

软件体系结构是在识别可重用构件和连接件的基础上,研究软件结构的表达和分析的理论和技术。

（1）现代定义

软件体系结构被定义为系统的基本组织结构，包括构件、构件之间的关系、环境以及管理系统设计和演化的原则。

（2）传统定义

软件体系结构是设计过程的一个层次，超越算法和数据结构的设计，研究整体结构的设计和描述的方法。

体系结构 = 组件 + 连接件 + 约束

2.3 软件体系结构的意义和目标

意义：

1. 软件体系结构是软件开发过程初期的产品，对于开发进度和软件质量的一切资金和劳务投入，可以获得最好的回报。
2. 体系结构设计是形成的投资高汇报的重要因素。
3. 正确有效的体系结构设计会给软件开发带来极大的便利。

软件体系结构的目标：

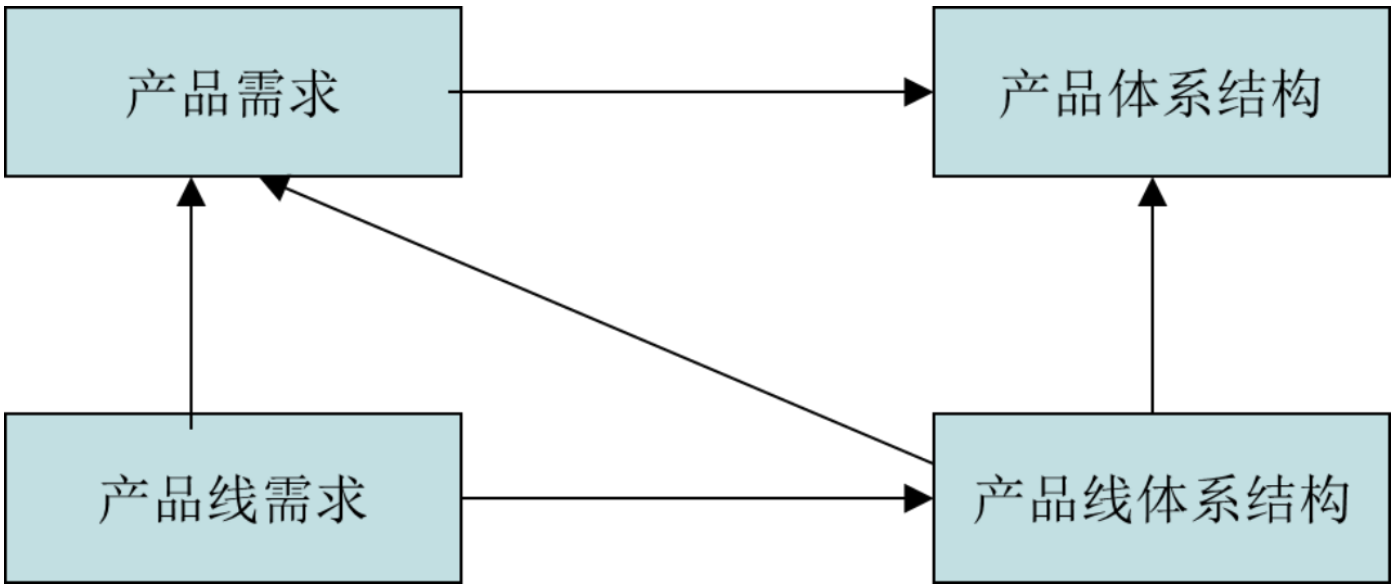
- 外向目标：建立满足终端用户要求的系统需求。
- 内向目标：建立满足后期设计者需要以及易于系统实现、维护和扩展的系统部件构成。

2.4 软件体系结构的研究范畴

（1）体系结构描述语言与工具：ADL（Architeture Description Language）、设计符号为体系结构建模

（2）产品线与标准：企业、需求、架构、系统

软件产品线是一个工程问题，它由一组软件密集型家族系统共享公共的、可管理的特征。



（3）软件体系结构风格及风格应用

软件体系结构风格是指一组设计词典、有关词典如何运用的限制条件、及词典语义的假设。

每个风格的应用仅适合于一定的目标，而不适用于其他目标。

（4）体系结构文档化

David Garlan体系结构文档化4个视图

1.Context-based Views 基于文本的视图

2.Code-based Views 基于代码的视图

3.Run-time Views 运行时视图

4.Hardware-based Views 基于硬件的视图

Chapter 3 软件体系结构的层次性

3.1 体系结构的基础和层次特性

来自建筑

3.2 软件的物质基础

计算机硬件体系结构

- **程序是对一组数据进行处理的一串指令的集合。**
- **根据处理指令流和数据流的数量，计算机分为：**
  - SISD
  - SIMD
  - MISD
  - MIMD 多处理机系统

3.3 软件的结构基础（vs3.2的物质基础）

软件结构基础的思想和概念， 包括四个方面： （越来越抽象）

1. 结构化控制流（if，while，continue）
2. 部件的连接模式
   - 何为部件?

     数据、外部设备、程序段
   - 部件连接器

     完成部件与部件之间的连接
   - 实现部件连接的方式
     - 过程调用
     - 远程过程调用
     - 事件触发
     - 服务连接
     - 循环连接
     - 查询连接
     - 中断/事件方式
     - 共享信息方式
3. 数据结构
   - 逻辑结构：集合结构、线性结构、树状结构、网络结构
   - 物理结构：顺序存储、链式存储、散列、索引
4. 抽象数据类型
   - ADT的概念

     抽象数据类型（Abstract Data Type) 一个数学模型以及定义在该模型上的一组操作。 ADT包括数据元素，数据关系以及相关的操作。
   - 问：关系模式是一个几元组?

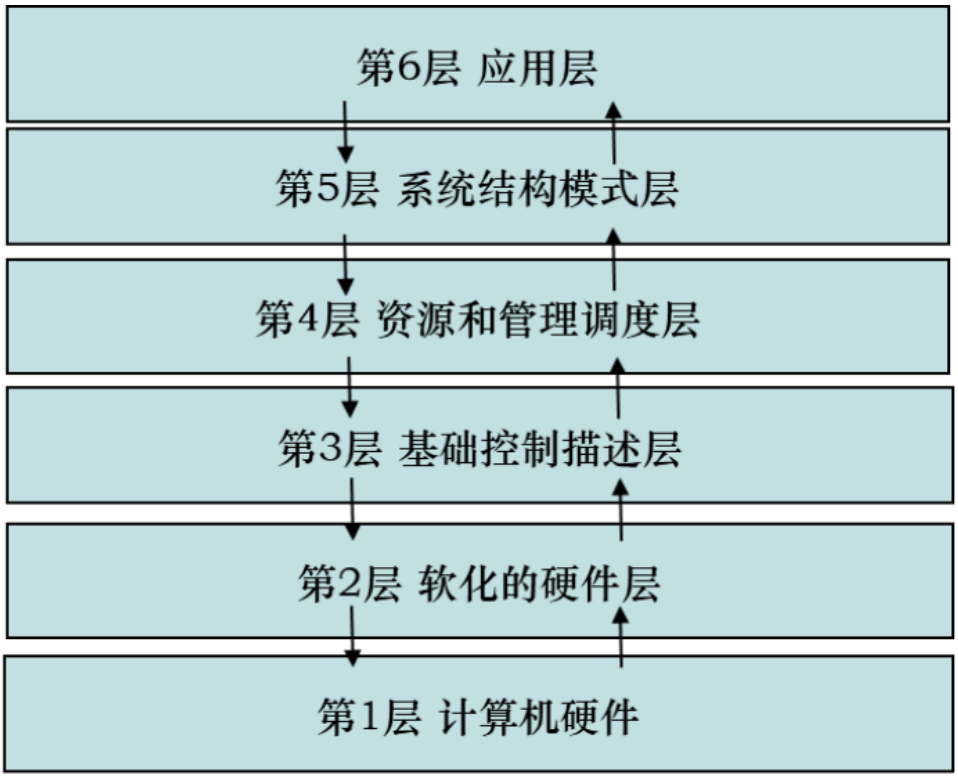
关系模式是一个**五元组**R（U，D，dom，F）R是关系名，U是关系属性名集合，D是属性组中属性所来自的域，dom是属性域，F是属性间数据的依赖关系集合。

3.4 软件的层次结构模型

课件上讲了：硬件到软件、程序、网络、操作系统的体系结构

3.5 软件体系结构的层次结构模型

6层



软件体系结构的层次模型

- 硬件基础层

  软件运行的物质基础

- 软化的硬件层

  处理器、中断

- 基础控制描述层
  - 程序控制概念：顺序、 条件、 选择、 循环、变量、 参数、 过程、 函数、 模块
  - 数据描述的概念：数组、 队列、 链表、 堆栈、 树、 图、 指针、 记录

- 资源和管理层

  共享资源、 同步、分时系统、 异常处理、 并行、 进程、 线程、消息、 远程调用

- 系统结构模式层（最高层次的软件结构概念）（属于体系结构风格或系统级别的设计模式）

  解释器、 编译器、 编辑器、 管道/过滤器、 黑板、 C/S 、 B/S、框架

- 应用层

3.6 软件体系结构的体系

Chapter 4 软件体系结构的设计原理

4.1体系结构设计中遵循的原理

1. 抽象
2. 封装
3. 信息的隐藏
4. 模块化
5. 注意点分离
6. 耦合和内聚
7. 原始性
8. 策略和实现的分离
    - 解释

      策略部件负责处理上下文相关的决策、信息的语义和解释的知识、把不相交计算组合形成结果、对参数值进行选择等问题

      实现部件负责全面规范算法的执行，执行中不需要上下文相关信息进行决策

      实现部件因为独立于上下文环境，因而更容易重用和维护

      策略部件因为与特定的应用相关，通常随时间的变化而改变
9. 接口与实现的分离
    - 解释

      接口定义了部件所提供的功能并规范了功能的使用方法

      实现部分包括了部件所提供功能的实际代码

      强调一个客户只应该知道他需要知道的东西

是一种信息隐藏技术

接口和实现的分离可以很好地支持可变性

10. 分而治之
11. 层次化

4.2 软件的非功能特性

虽然非功能性在软件体系结构中非常重要，但很难对他们的效果和作用进行衡量

- 1. 可变性

  为应对改变而设计的系统比没有进行这样的考虑而设计的系统更容易适应用户，因而该系统的适应性要好得多，生命周期要长得多

- 2. 互操作性

  互操作性是指不同的计算机系统、网络、操作系统和应用程序一起工作并共享信息的能力。

  将系统设计成具有互操作性的部件集合本身就自然地对系统的功能构成进行了分割。

- 3. 效率

  软件运行过程中对资源的使用情况、以及对系统的响应时间、存储消耗和I/O吞吐量的影响

  效率问题不仅仅是设计精良算法的问题，而且是部件操作责任合理的分配、部件之间的耦合关系等体系结构的问题

  良好结构、丰富功能和高效率等方面需要权衡利弊

- 4. 可靠性

  是软件系统在各种情况下维持其功能的能力，区分为2个方面的内容：

  容错性：当错误事件发生时确保正确的系统响应，必要时采取内部补救措施

  健壮性：健壮性不要求软件在发生错误之后还能继续执行，他只需要保证软件能以明确和可接收的方式终止

- 5. 可测试性

  测试变得越来越难、越来越昂贵

  支持可测试性的软件体系结构可以为错误探测和改正、以及代码调试和部件的临时集成给予支持

- 6. 可重用性

  “通过已经存在的来获得想要的”的软件设计和实现方法（有利于Cost、Time、Quality的平衡）

  用重用进行软件开发

  为重用进行软件开发

Chapter 5 部件和连接器

部件和连接器被公认为体系结构的两大类构成部分

部件是软件功能设计和实现的载体

连接器是专门承担连接作用的特殊部件

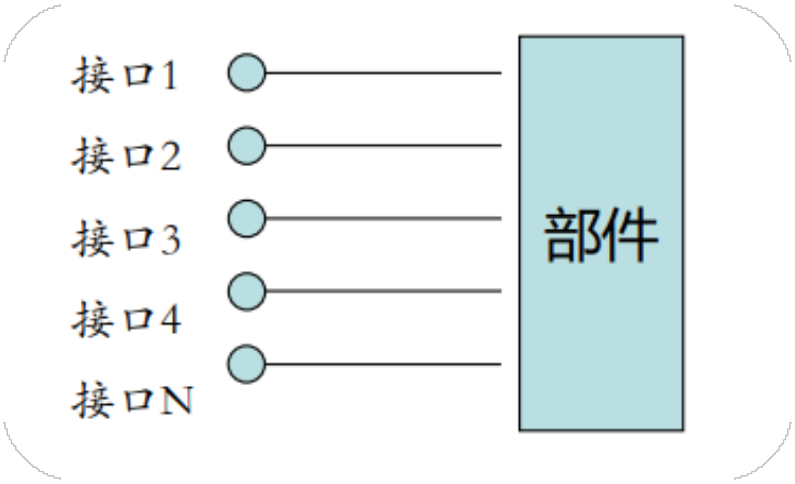
5.1 部件

- 部件是软件系统的结构块单元，是软件功能设计和实现的载体，任何具有独立结构和行为特性的软件体都可以称为部件。
- 系统是部件及其关联的集合。
- 使用的时候：一个部件至少有一个接口，每一个接口代表对外联系的一种角色，这是部件与外界发生联系的窗口。
- 设计系统时候：需要根据对部件的功能、与其他部件的关联、对部件的特殊性要求，建立内部处理和控制结构。

部件的表达形式

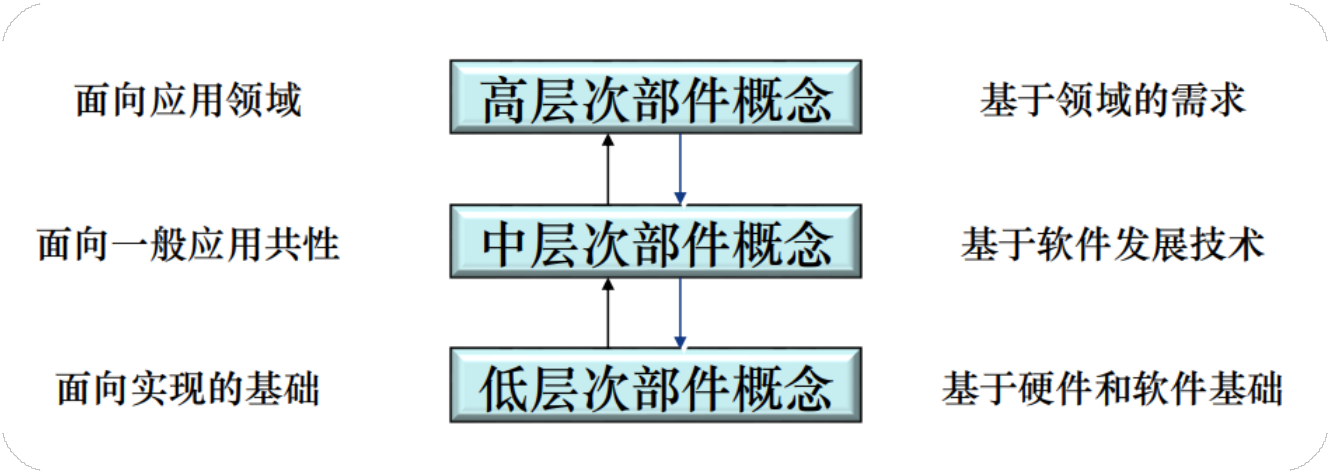
不同软件设计环境下服务于不同目的，部件具有不同的类型或名称

部件的一般表达形式



部件的特性

1. 部件的接口特性：
    1. 完备性：使用者可以用它来完成部件应该能够完成的一切工作
    2. 最小化：部件的接口或界面中任一操作，都不能由其他操作组合而实现
    3. 正交性
    4. 方便性
    5. 效率
2. 部件的概念等级或层次



5.2连接

连接是部件间建立和维持行为关联和信息传递的途径，是系统复杂性的来源。

机制和协议

机制：让连接得以发生和维持

协议：连接能够正确、无二义性、无冲突的进行

（1）连接的种类

过程调用、控制、事件、消息发送、数据传送 etc.

（2）连接的实现机制

无论多么复杂的连接关系，其实现都是基于以下基本连接机制：

过程调用、中断、I/O、DMA、进程、线程、共享、同步、并/串行、事件、并发 etc.

（3）连接的协议

协议是连接的规约，是实现有意义连接的保证。

连接的规约是建立在物理层之上的有意义信息形式的表达规定。

即使是简单的连接，也有协议在起作用。

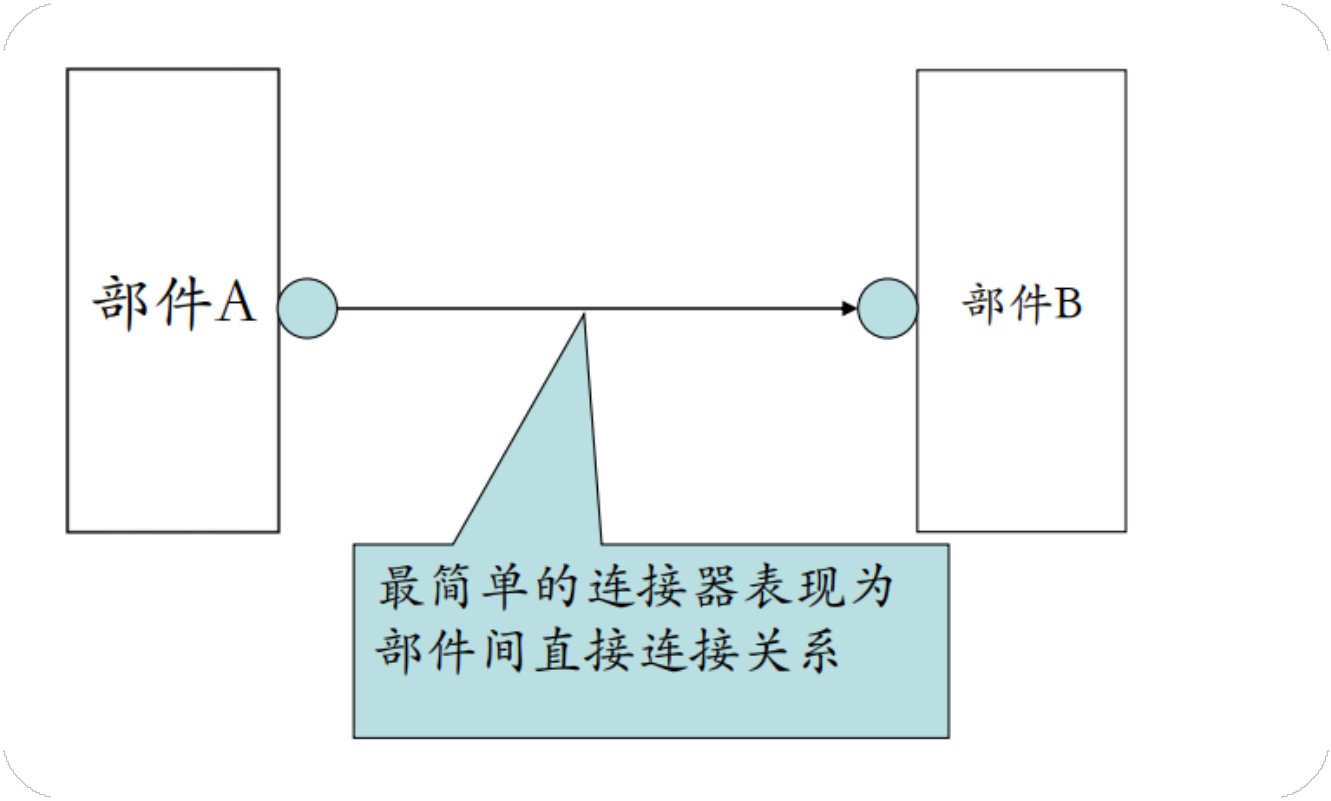
（4）连接的特性

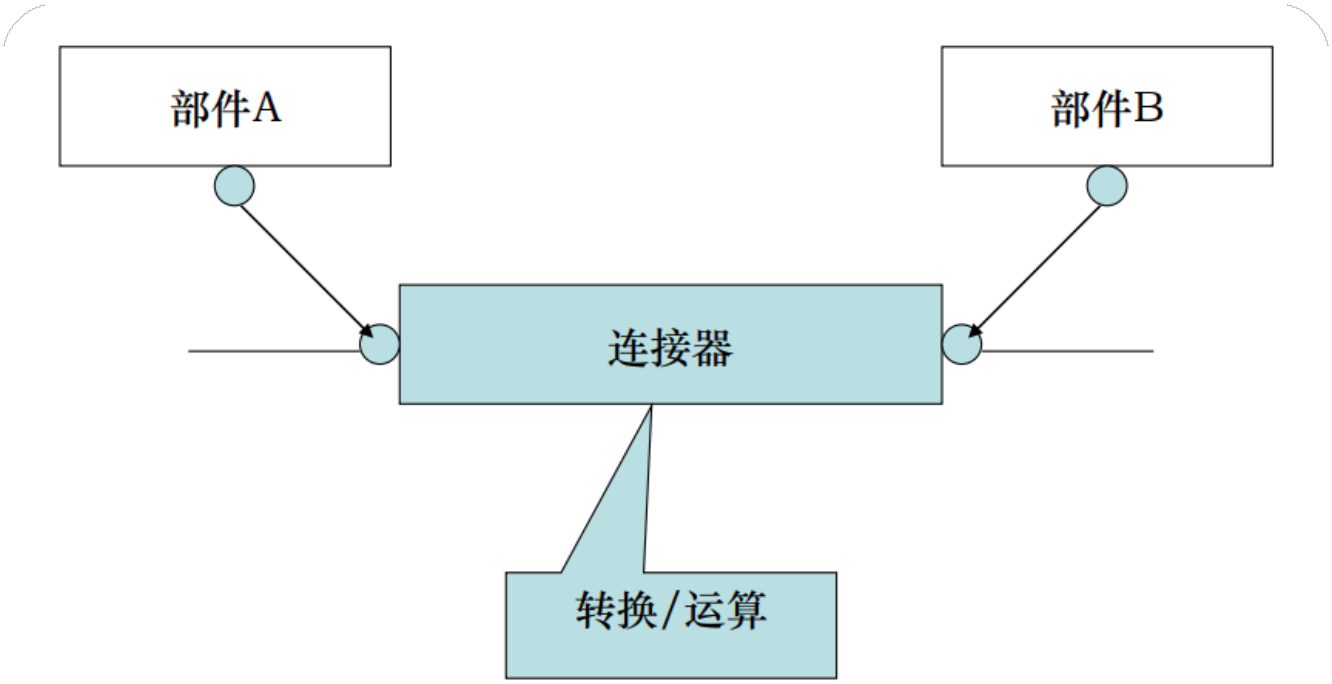
5.3 连接器

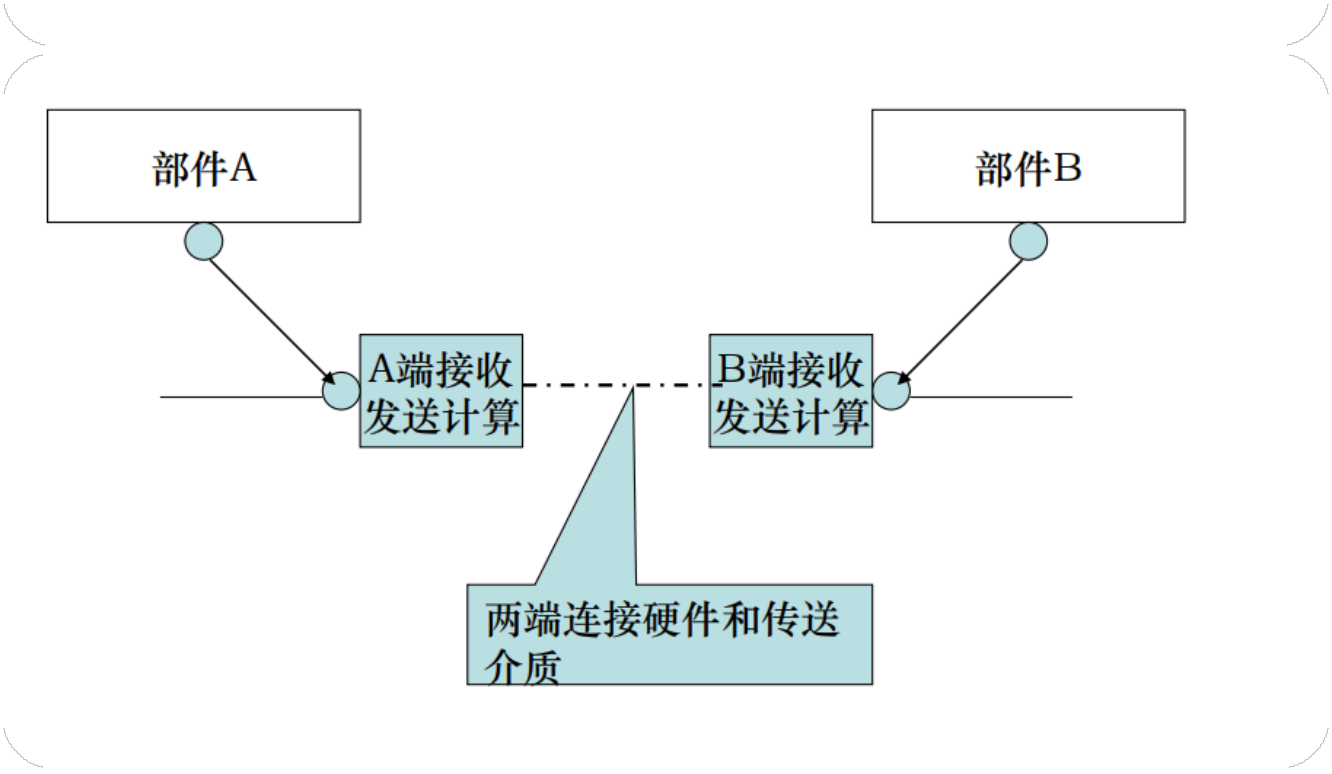
连接器是实现部件与部件之间联系的特殊机制或特殊部件

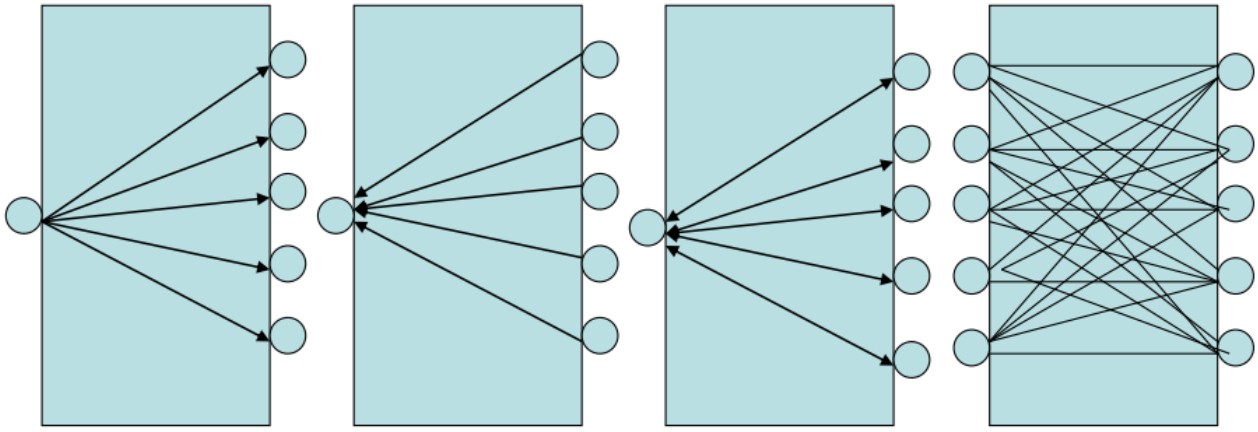
（1）连接器的层次性

- （2）连接器的表达形式




部件A
部件B
连接器
转换/运算


部件A
部件B
A端接收
发送计算
B端接收
发送计算
两端连接硬件和传送
介质



多端连接器端口之间的连同关系

（3）连接器的特性

- 连接的关系 1：1、1：n、n：1、m：n
- 连接的角色和方向
- 连接的交互方式（信号、语言式）
- 连接的可扩展性
- 连接的互操作性
- 连接请求响应特性
- 连接请求的处理策略
- 连接的代价、处理速度或能力

Chapter 6 体系结构的一般描述

通用的一般的描述方法

（1）主子程序

（2）数据抽象和面向对象

- 数据抽象和面向对象设计是在主程序和子过程设计基础上建立和发展起来的重要的软件描述方法。
- 数据抽象 是面向对象设计的理论基础
- 类和对象是该描述方法的基础粒度，而非模块或者包 。

本质上没有逃出主子程序的思想

（3）层次结构

- windows NT

理论的形式化方法

（1）Z Notation

（2）CSP（通信顺序进程）

（3）类属理论

（4） CHAM（化学抽象机模型）

软件体系结构集成环境

（1）UniCon

（2）Darwin

（3）Wright

（4）ACME

appendix 风格 && oo法则

体系结构风格

体系结构风格

- 数据流、调用和返回、独立构件、虚拟机、数据中心 得知道有这些

面向对象法的设计法则说了四个

*

```
1 ![](image/image_JKKr_59SWh.png)
```