软件工程-软件设计与实现

- 面向对象设计
- 实现中有关问题
- 开源开发
- 嵌入式系统的设计与实现

资料整理
火龙果软件
荐

- 软件设计和实现是软件工程过程中的一个阶段。设计和实现活动是交叉进行的。
- 设计是创造性活动，基于客户的需求识别出系统组件及其关系，实现将设计转变为程序。
- 设计的细节层次取决于待开发系统的类型以及是否使用计划驱动或敏捷开发方法。
- 本章主要介绍如何使用面向对象方法进行设计以及实现中的有关重要问题。
- 嵌入式软件不一般的特性也是软件设计涉及到的一个重要方面。

资料整理
火龙果软件
荐

7.1 面向对象设计

- 面向对象设计将面向对象分析创建的分析模型变换为设计模型，它将作为软件实现的蓝图。但由于面向对象分析与设计活动是一个迭代与演化的过程，概念与表示方法的一致性使得分析与设计阶段平滑过渡。

- 传统的设计方法将问题域分解成一系列功能模块来完成，这些模块形成过程式软件的基本结构。面向对象方法把问题域分解为一系列相互作用的对象，在此基础上构造出基于对象及其交互的软件系统结构。

资料整理
火龙果软件
荐

本教材作者Sommerville认为面向对象设计的一般过程应有以下几个阶段：

（1）了解并定义系统上下文和与系统的外部交互

（2）设计系统体系结构

（3）识别出系统中的主要对象

（4）开发设计模型

（5）定义对象接口

上述活动交替进行，彼此影响，不断精化。

以下通过一个例子介绍面向对象的设计过程。

资料整理
火龙果软件
荐

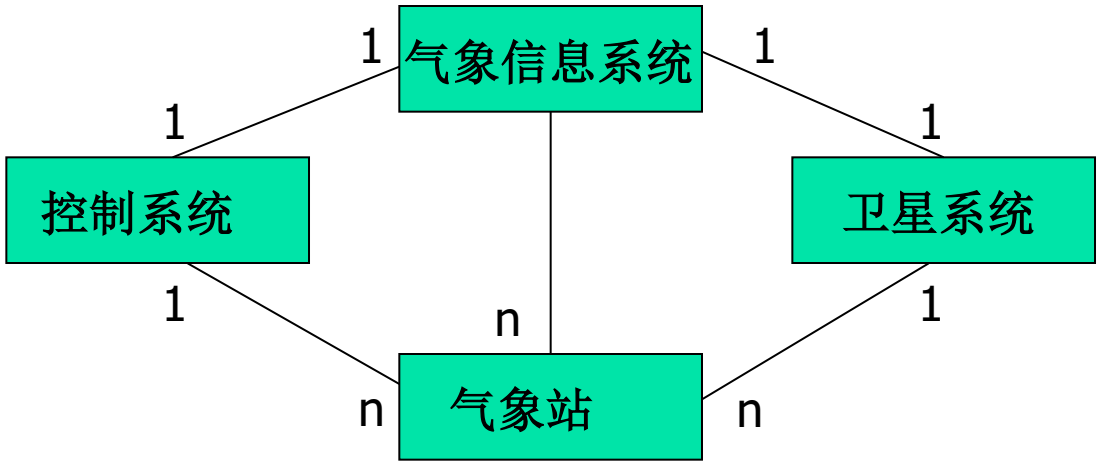
7.1.1 系统上下文与交互

例：气象站系统：气象站部署在远程的区域，每个气象站记录当地的气象信息，并通过卫星将这些信息定期传输给气象信息系统。

- 软件设计首先了解待开发系统与外部环境之间的关系，帮助确定系统提供哪些功能以便有效地与外部环境进行通信。
- 有两个模型可以表达系统与环境之间的关系：

资料整理
火龙果软件
荐

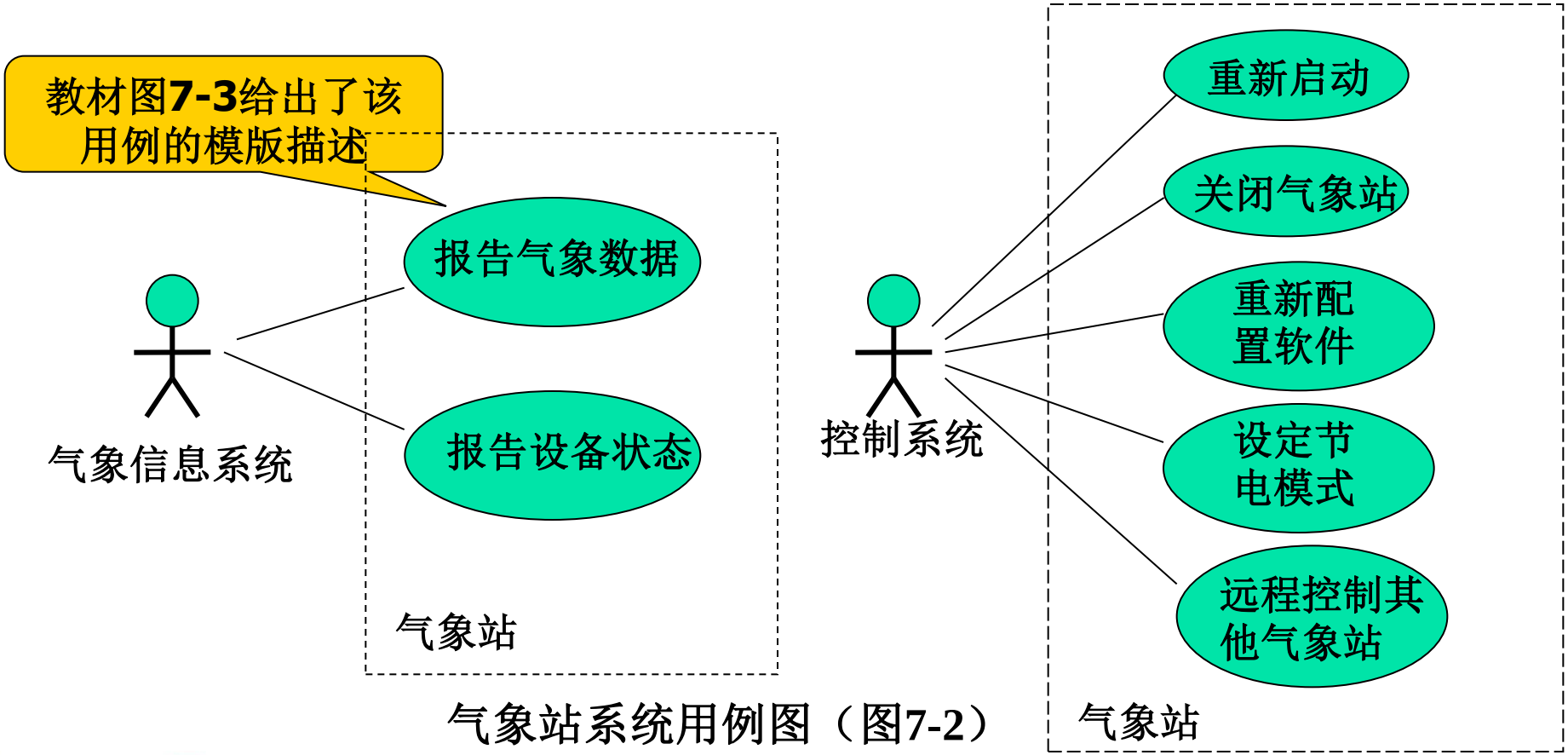
• 系统上下文模型，是结构模型，描述系统环境中的实体及其关联。上下文模型可用关联（ER）图来表示，方框表示系统或子系统。也可用UML的包图来表示。下图说明了每个气象站所在环境中都包含一个气象信息系统，一个机载卫星系统和一个控制系统。



气象站系统上下文

资料整理
火龙果软件
荐

- 交互模型，描述系统与外部角色有哪些交互。UML的用例模型是使用抽象方法表示系统与外部角色之间的交互。



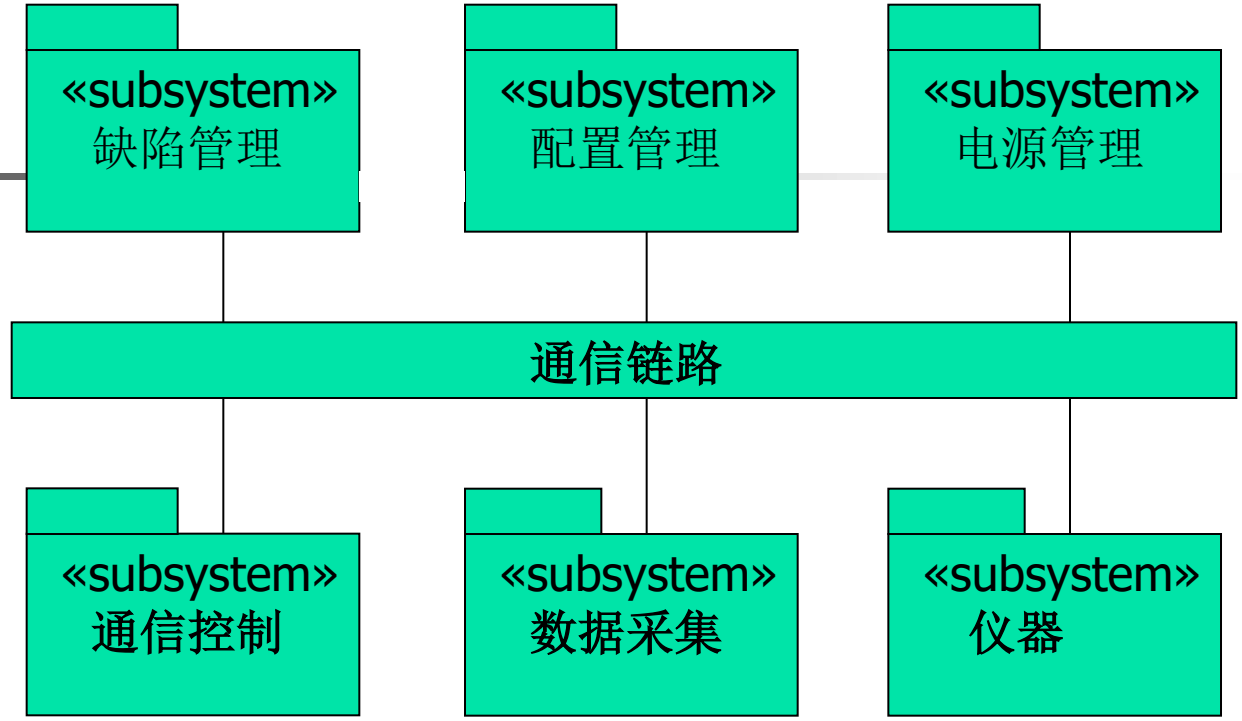
气象站系统用例图（图7-2）

资料整理 火龙果软件 荐

7.1.2 体系结构设计

- 系统与环境间交互的定义可以作为系统体系结构设计的基础，还要结合业务知识和设计的一般知识进行体系结构的设计。
- 首先识别出系统的主要组件及它们之间的关系，然后运用一种体系结构模式来组织这些组件。
- 气象站系统运用了广播模型。

资料整理
火龙果软件
荐



当该子系统接收到“关机”命令，其它与该命令相关的子系统也能关机

气象站系统高层体系结构

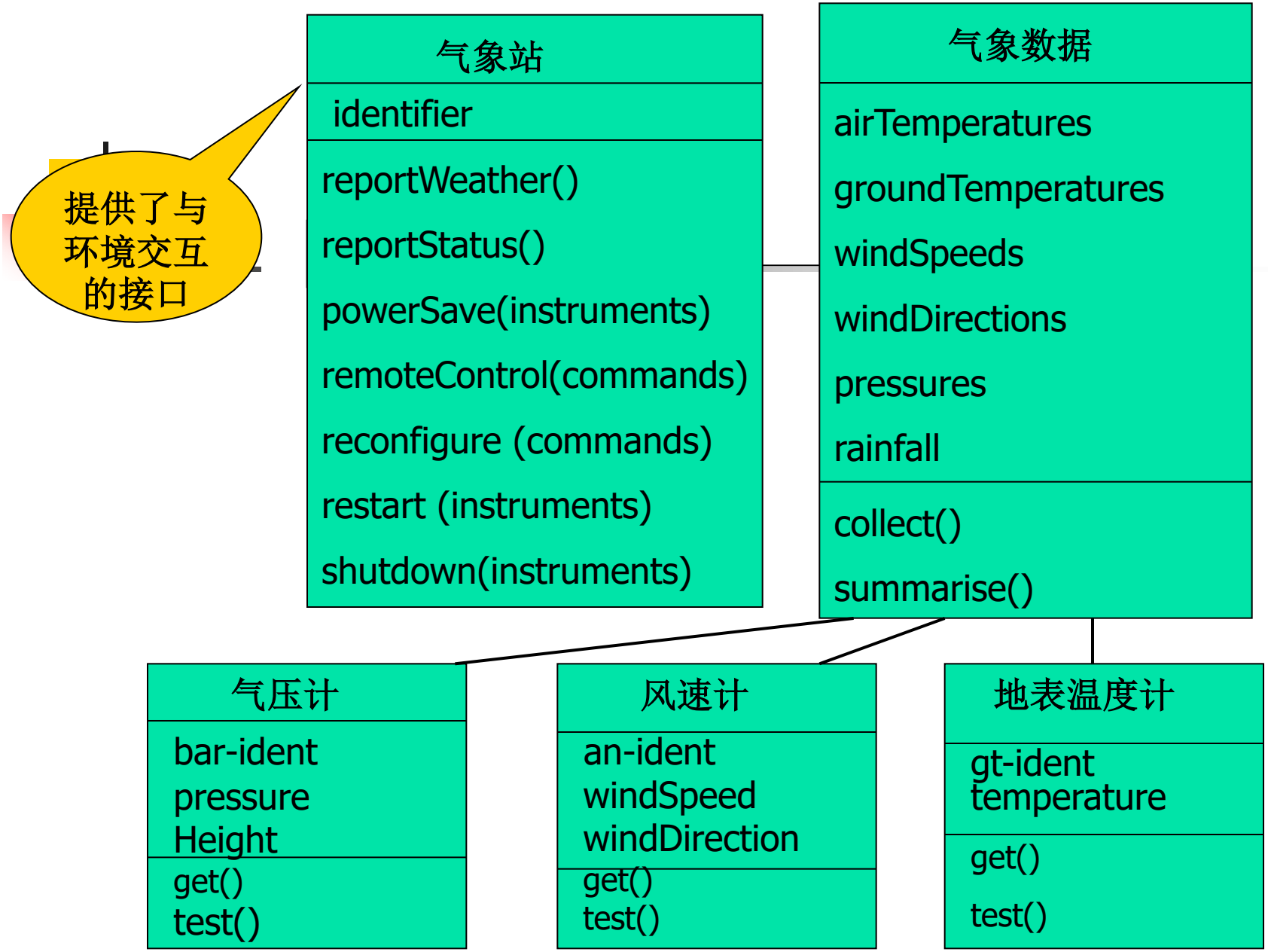
资料整理
火龙果软件
荐

7.1.3 对象识别

- 面向对象分析识别问题域的对象，面向对象设计是识别求解域的对象。求解域的对象一部分来自于问题域，一部分来自于与设计相关的系统描述。
- 如何识别对象？方法同面向对象分析.

根据以上思想识别出的对象见下图。

资料整理
火龙果软件
荐



气象站系统的对象（类）

资料整理 火龙果软件 荐

- 气象站类提供与环境之间的接口，它的操作反映了用例模型中给出的交互。
- 气象数据类负责处理报告气象数据指令，它的相关操作包括对数据的收集和汇总，源数据是不同仪器采集到的。
- 其他三个类与系统中的仪器直接对应，操作就是对这些硬件的控制。这些对象以某些特定频率自动采集数据并在本地存储，在得到请求时将数据传输给“气象数据”对象。

资料整理
火龙果软件
荐

7.1.4 开发设计模型

- 设计模型包含了对象类以及它们之间不同类型的关系，是系统需求和系统实现之间的桥梁。
- 设计过程中的一个重要活动是决定需要什么样的设计模型和设计模型的细节层次，依赖所开发的系统类型。
- 不一定为系统建立所有的模型.
- 一般要进行两类模型的设计：
  - 静态模型　描述系统的静态结构（如气象站系统的类图）
  - 动态模型　描述系统的动态结构（交互、状态变化）。

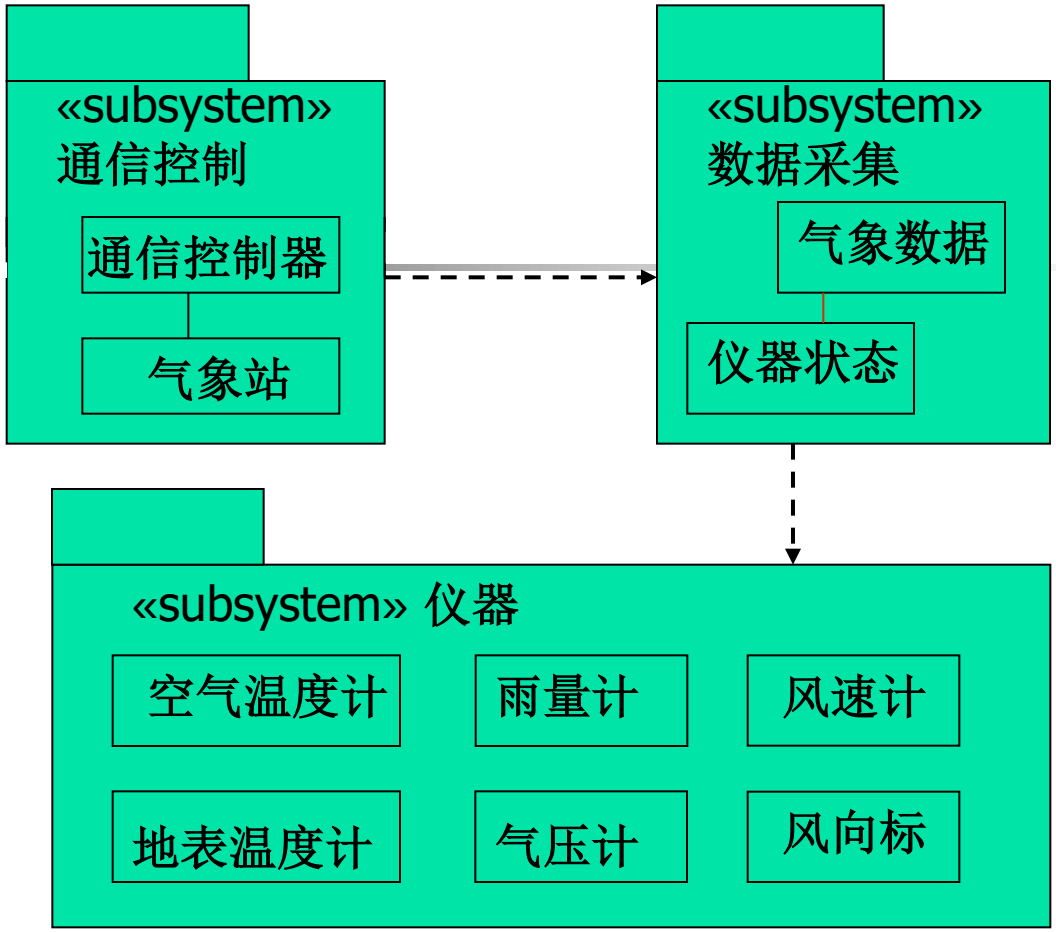
资料整理
火龙果软件
荐

本教材作者认为有以下3个模型有助于为用例和体系结构模型增加细节：

- 子系统模型：给出系统的逻辑分组，每个分组构成一个子系统，可使用包图来表示。子系统模型说明如何能将设计组织成逻辑上相关的对象群。
- 时序模型：说明对象交互的时序，使用时序图或协作图来表示。
- 状态机模型：说明某个对象如何响应事件来改变它们的状态，使用状态图来表示。

下图给出了气象站系统包图以及关系。包图加上类图描述了系统的逻辑分组。

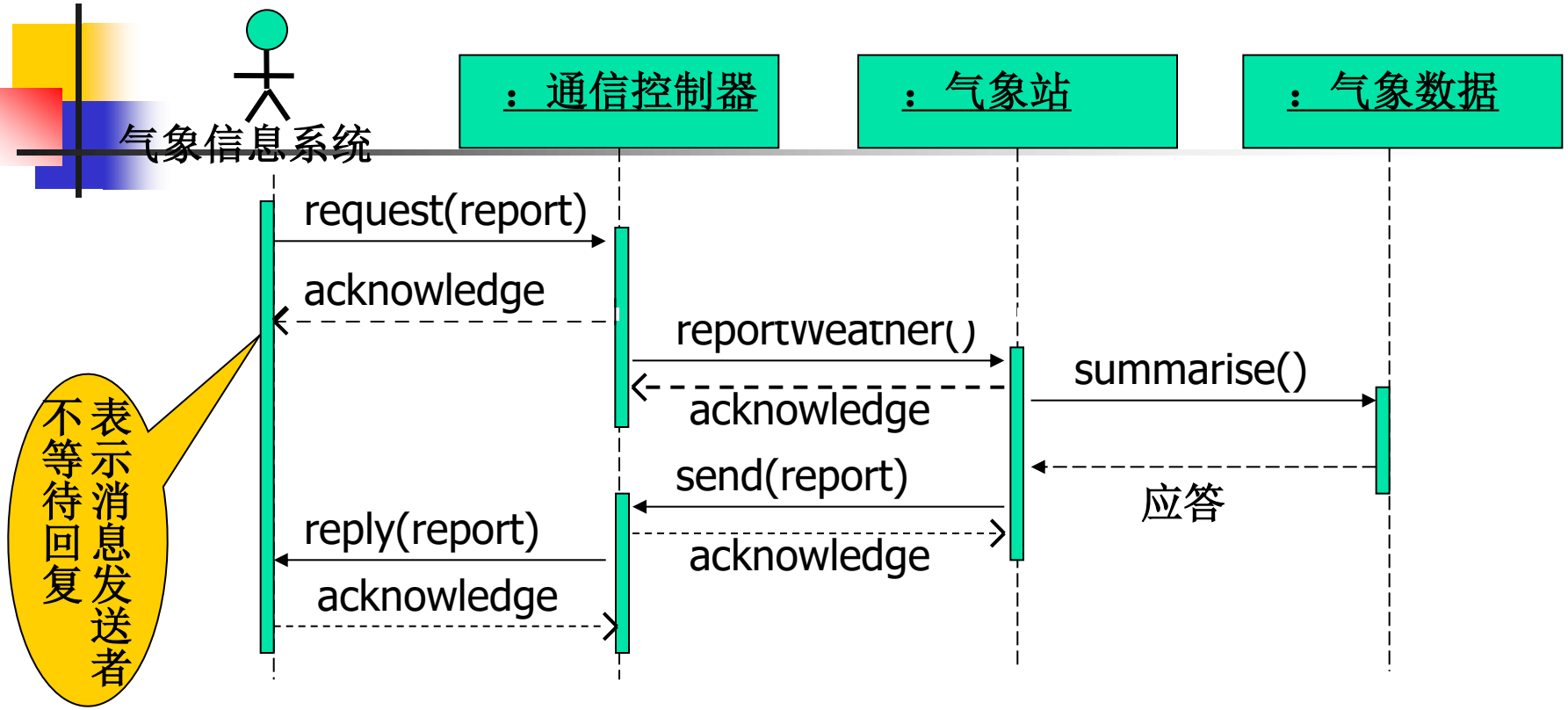
资料整理
火龙果软件
荐



气象站系统（包图加上类图描述了系统逻辑分组）

下图给出了气象信息系统向气象站请求汇总数据时发生的的交互序列：

资料整理
火龙果软件
荐

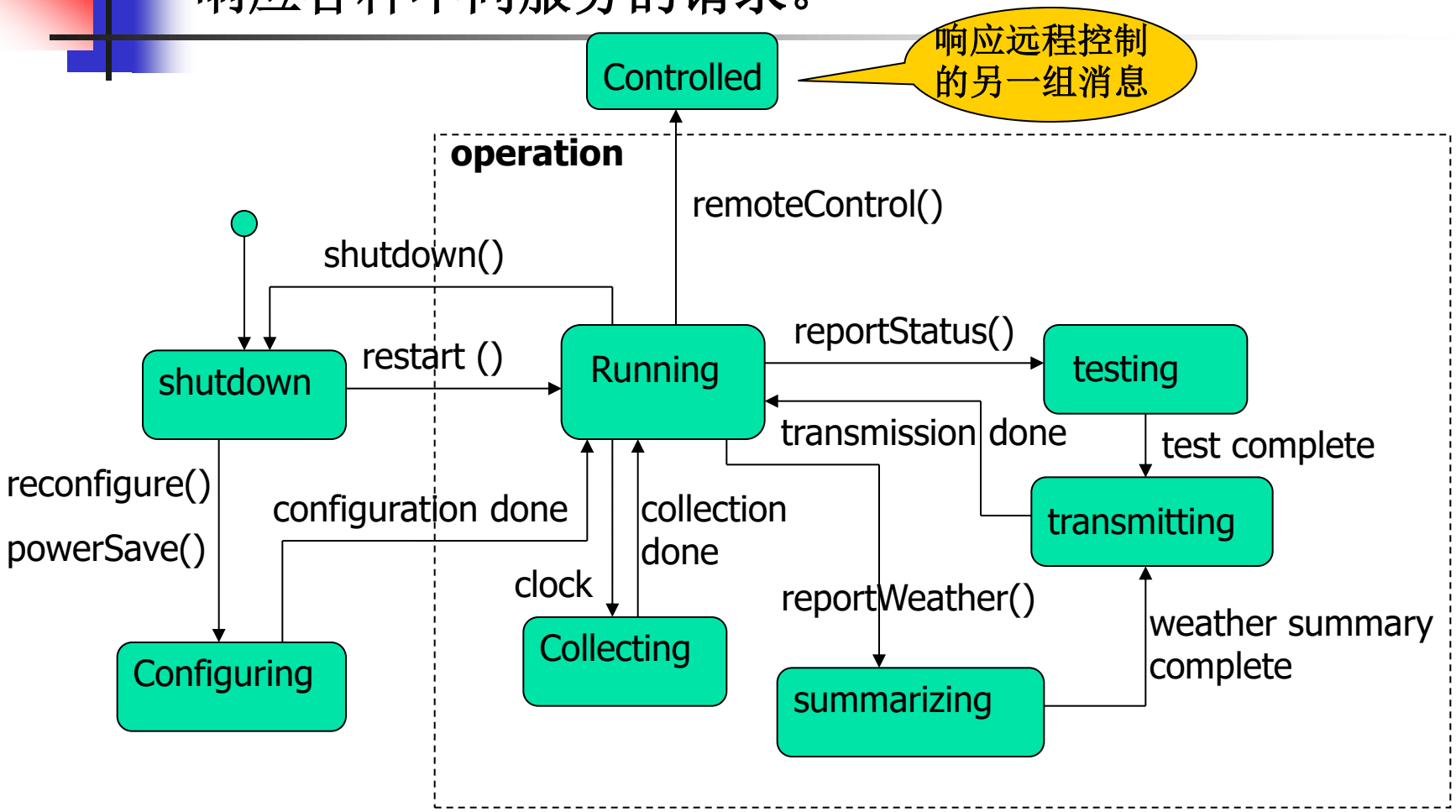


设计的时候，应该为每个重要的交互（用例模型中的用例）创建一个时序图或协作图。

资料整理
火龙果软件
荐

时序图用来对一组对象的交互行为建模，状态机模型表示某个对象对不同事件的响应行为。

下图是气象站对象的状态图，给出了该对象如何响应各种不同服务的请求。

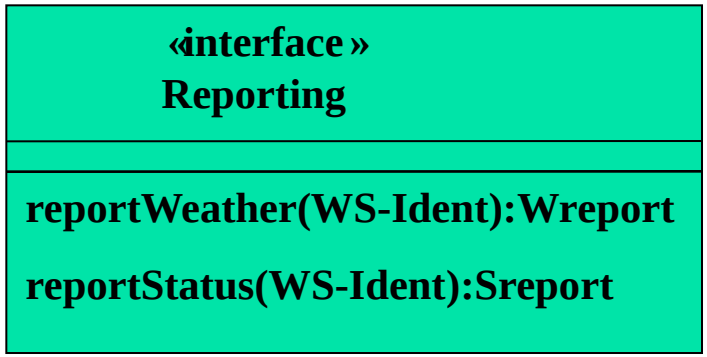


7.1.5 接口描述

- 接口提供了一种方法，把组件基于操作的功能说明与具体实现区分开来，使得任何依赖和使用接口的客户程序不必依赖于接口的具体实现，有利于接口实现的替换。
- 接口要精确的定义，以便能由其他对象使用。
- 接口描述可以用UML类图模板，省略属性说明，《interface》要包含在类名部分中。
- 也可以用程序设计语言来定义接口，以便用编译器来发现接口描述中的错误和不一致。

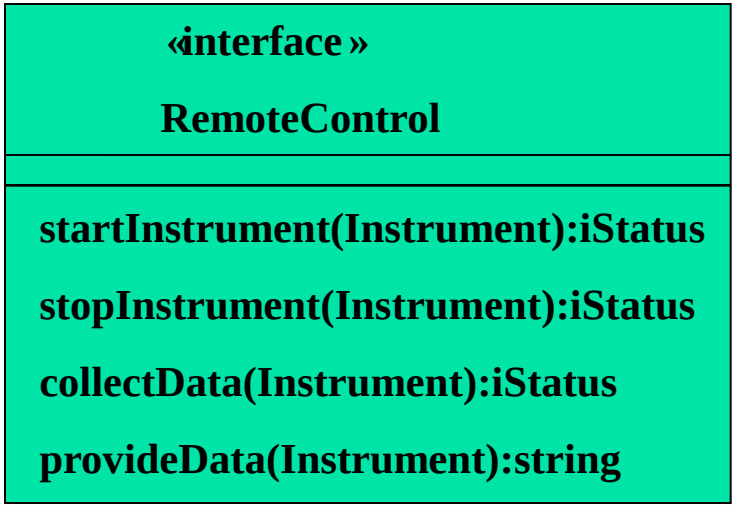
资料整理
火龙果软件
荐

- 对象与接口不是一对一的关系，一个对象会有若干接口。同样，一组对象可以通过同一个接口访问。
- 下图是气象站系统中定义的两个接口：



报告接口

直接映射到气象站对象中的操作



远程控制接口

每一个操作都编码成气象站对象中remoteControl方法所关联的一个命令字符串

资料整理
火龙果软件
荐

7.2 设计模式

- 模式(Pattern)的概念最初来自于建筑学领域，用模式描述建筑物的建筑元素（Alexander，1977），概括了被认为是好的设计的实践经验。
- 模式是对问题和解决方案的基本内容的描述。模式是积累的经验和智慧的描述，可以帮助人们在软件开发过程中对于经常重复出现的问题制定成功解决的方案，即解决方案可以在不同的设置下被复用。

资料整理
火龙果软件
荐

- 94年Gamma等4人（简称“Gang of Four”)合著的《设计模式：可复用面向对象软件的基础》提出了用设计模式解决重复出现的软件设计问题，并对设计模式进行了分类描述和解释。
- 96年由Buschmann等5人合著的《面向模式的软件体系结构》将模式跨越不同的抽象层次，提出了高层的体系结构模式、中层的设计模式和低层的习惯用法。 本节主要针对设计模式进行讨论。

资料整理
火龙果软件
荐

Gamma等人定义了模式的四个基本要素：

（1）模式名称

（2）问题描述

描述设计模式所解决的问题，什么情况下可以应用该模式，或者说使用该模式可以在设计中避免的一些缺点。

（3）解决方案

通常以图形的方式描述设计方案的组成成分（对象）以及它们之间的关系、职责和协作方式。它不是一个具体的设计描述，而是一个设计方案的模版，可以用不同的方式实例化。

（4）结果陈述

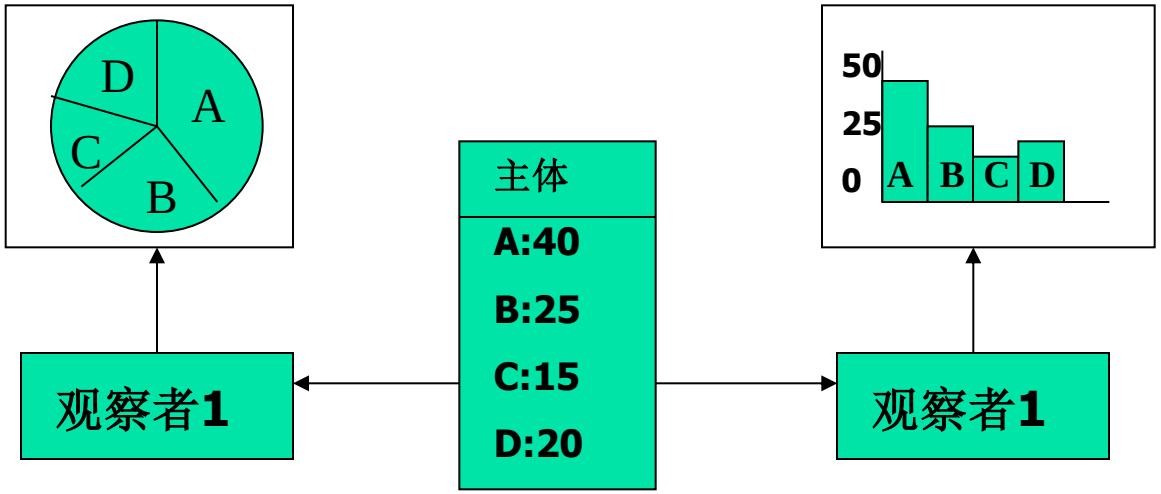
说明应用该模式的结果和副作用，帮助设计者了解该模式在特定的环境条件下用是否有效。

教材P121图7-10展示了观察者模式的定义。

资料整理
火龙果软件
荐

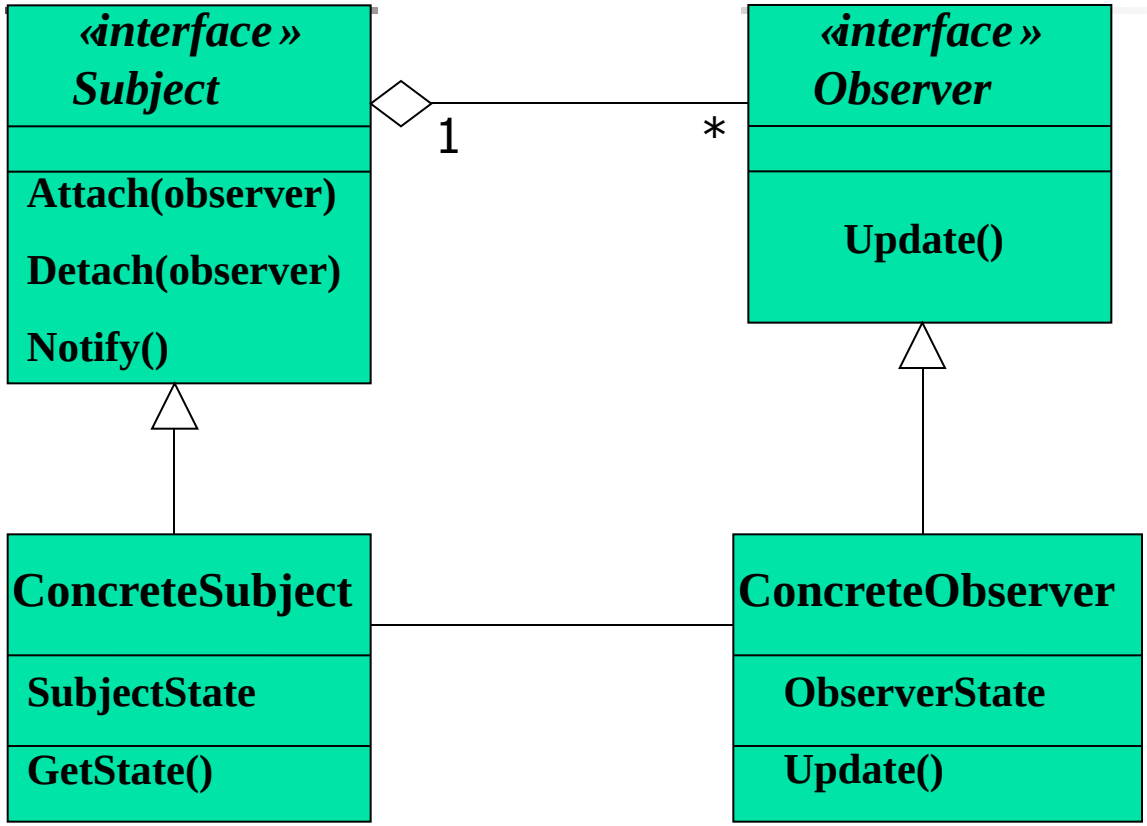
例1：观察者模式

- 定义对象间的一种一对多的依赖关系，当一个对象的状态发生变化时，所有依赖它的对象都得到通知并被自动更新。



一个数据集的两个图形显示

资料整理
火龙果软件
荐



观察者模式的UML模型

资料整理
火龙果软件
荐

- **Observer将自己注册到主体对象Subject中，主体对象将观察者存放在一个容器（Container）里。**
- 主体对象状态发生了变化，从容器中得到所有注册过的观察者，将变化通知观察者。
- 观察者模式通常与 **MVC**模式有关系。观察者模式可被用来降低 **model** 与 **view** 的耦合程度 , 主体便是模型加控制器，而观察者便是视图。
- 当一个对象数据更新时需要通知其他对象，不知道具体有多少个对象或者不希望互相间形成紧耦合，适合使用该模式。

资料整理
火龙果软件
荐

- 设计模式的使用是一个非常有用的复用形式。由于模式的复杂性，为了有效的使用，设计者需要对许多设计模式有详细的了解。
- 使用一个模式时，主要复用该模式的思想，根据所开发系统中需要解决的问题调整自己的实现方案。
- 模式不是一个可执行组件，不像组件那样只需了解组件的接口。有经验的程序员才能辨别出一个模式所能适用的一般条件。因此，有效地使用设计模式需要积累经验。

资料整理
火龙果软件
荐

7.3 实现问题

- 复用　开发软件时，尽可能的考虑现有组件或系统的复用。
- 配置管理　开发软件是一个团队进行的项目，每个生成的组件都会有很多版本，若不很好的管理这些版本，将会造成混乱。
- 基于宿主机（host computer)-目标机开发

  宿主机开发的软件运行在目标机上经常有完全不同的环境。开发平台与运行平台的问题。

资料整理
火龙果软件
荐

7.3.1 复用（参考第16、17章）

- 软件复用的好处是降低总体开发成本、增加了系统的可靠性并可与标准兼容，加快了软件开发速度。
- 软件复用可以运作在不同层次上：：
  - 抽象层 成功经验的复用，抽象知识的复用，如设计模式、体系结构模式。
  - 对象层 直接复用开发平台库中的对象，要分析对象和方法是否提供所需的功能，代替自己编写代码。
  - 组件层 组件提供标准接口和通用的功能，通常往往利用框架（Frameworks）对组件进行调整与扩展。
  - 系统层 通过添加代码或者使用系统的配置界面来复用商业现成产品。

资料整理
火龙果软件
荐

复用需考虑的关键因素

- 软件的开发进度：如要快速，考虑复用现成系统而不是单个组件。可能需适当修改需求。
- 所预计的软件生命期：长生命期系统要关注可维护性，为适用新需求需改变组件及它们的使用方式。如果不能得到源代码，避免使用外部供应商提供的组件或系统。
- 开发团队的背景、技术和经验。
- 软件的可靠性及其他非功能性需求。
- 应用领域：某些应用领域可复用的产品较多，可重新配置。
- 系统运行的平台：如 .NET是Microsoft平台专用的。

资料整理
火龙果软件
荐

支持软件复用的方法（教材P273图16-4）

- 体系结构模式
- 设计模式
- 基于组件的开发
- 应用框架
- 遗留系统包装
- 面向服务的系统：
- 应用产品线
- 面向方面的软件开发
- 商业现成产品（COTS）集成
- ERP系统
- 可配置垂直应用
- 程序库
- 程序生成器
- 模型驱动开发

资料整理
火龙果软件
荐

应用框架

框架（**Frameworks**）框架是一个一般化的结构（有时称为“半成品”），它包含抽象类和具体类的集合以及这些类之间的接口。

应用系统的构建是通过增加组件和通过对框架中的抽象类提供具体的实现（特化）来完成。

资料整理
火龙果软件
荐

Fayad and Schmidt 提出了三种类型的框架：

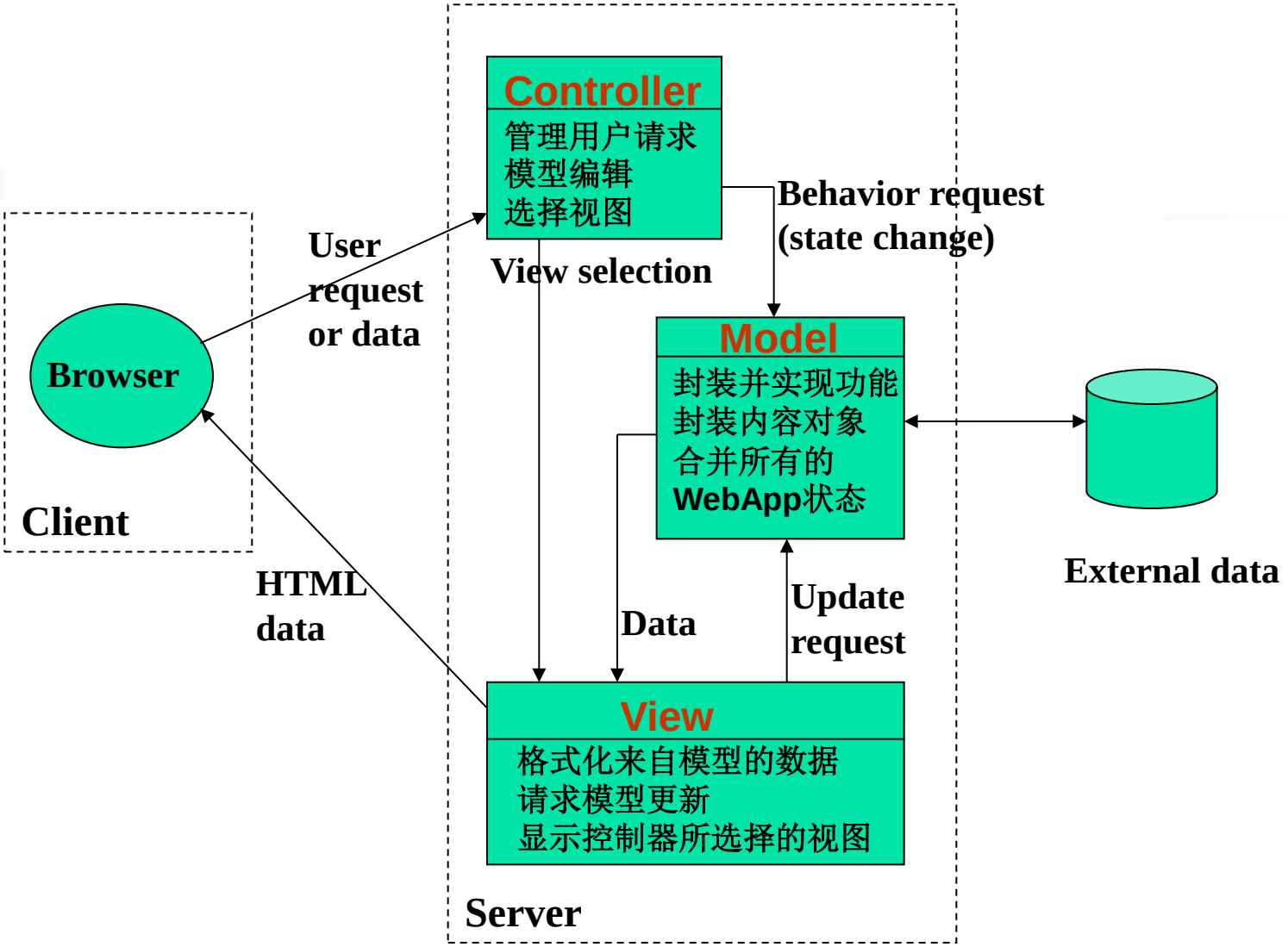
（1）系统基础实施框架

（2）中间件集成框架

（3）企业应用框架:框架嵌入了应用域知识并支持用户应用的开发。

如Web应用框架以MVC模式为基础，支持动态网站的创建。

资料整理
火龙果软件
荐



Web应用系统一般的体系结构

应用系统复用

指的是对整个系统的复用，大粒度的软件产品的复用。

（1）商业现成产品的复用

商业现成产品（COTS）是无需购买者作任何修改即可使用的软件系统。如桌面软件、数据库管理系统等。

资料整理
火龙果软件
荐

（2）软件产品线

是相关联的一组应用系统，具有相同的领域相关的体系结构。应用族的核心部分在每次构建一个新应用时都要复用。

对软件产品线各类特化处理需要做的工作有：

- 平台特化： 为不同的平台开发应用程序的不同版本。如一个应用程序可能存在Windows 、Linux等各种平台上的版本，程序的功能不变，修改与硬件和操作系统的接口。

资料整理
火龙果软件
荐

- 功能特化： 为不同需求的客户创建不同的应用程序版本。
- 过程特化：调整系统使之与特殊的业务过程配套。
- 环境特化： 创建应用程序的版本来处理特殊的操作环境和外部设备。

资料整理
火龙果软件
荐

基于组件的复用（参考第17章）

组件也称为构件，是一个独立的软件单元，可以与其它组件构成新的软件系统。下表列出了组件的主要特征：

组件特性	描述
标准化	符合某种组件模型的标准。
独立性	可以独立地进行组合与部署。若需要外部提供服务，在它的需求接口中声明。
可组合性	所有外部交互通过公共定义的接口进行，并提供对自身信息的外部访问。
可部署性	能作为一个独立的实体在组件平台上运行。
文档化	组件有定义接口的语法和语义，便于用户理解组件是否满足需求

资料整理
火龙果软件
荐

- 基于组件的开发为进行系统化的复用创造了机会。

  可复用组件有两个关键特性:

  （1）组件是可独立运行的的实体。它可以作为外部服务来引用，也可能直接包含在一个程序中。

  （2）组件所提供的服务通过其接口得到，而且所有的交互都是通过接口实现的。组件接口表现为参数化的过程，其内部状态是不会暴露出来的。

资料整理
火龙果软件
荐

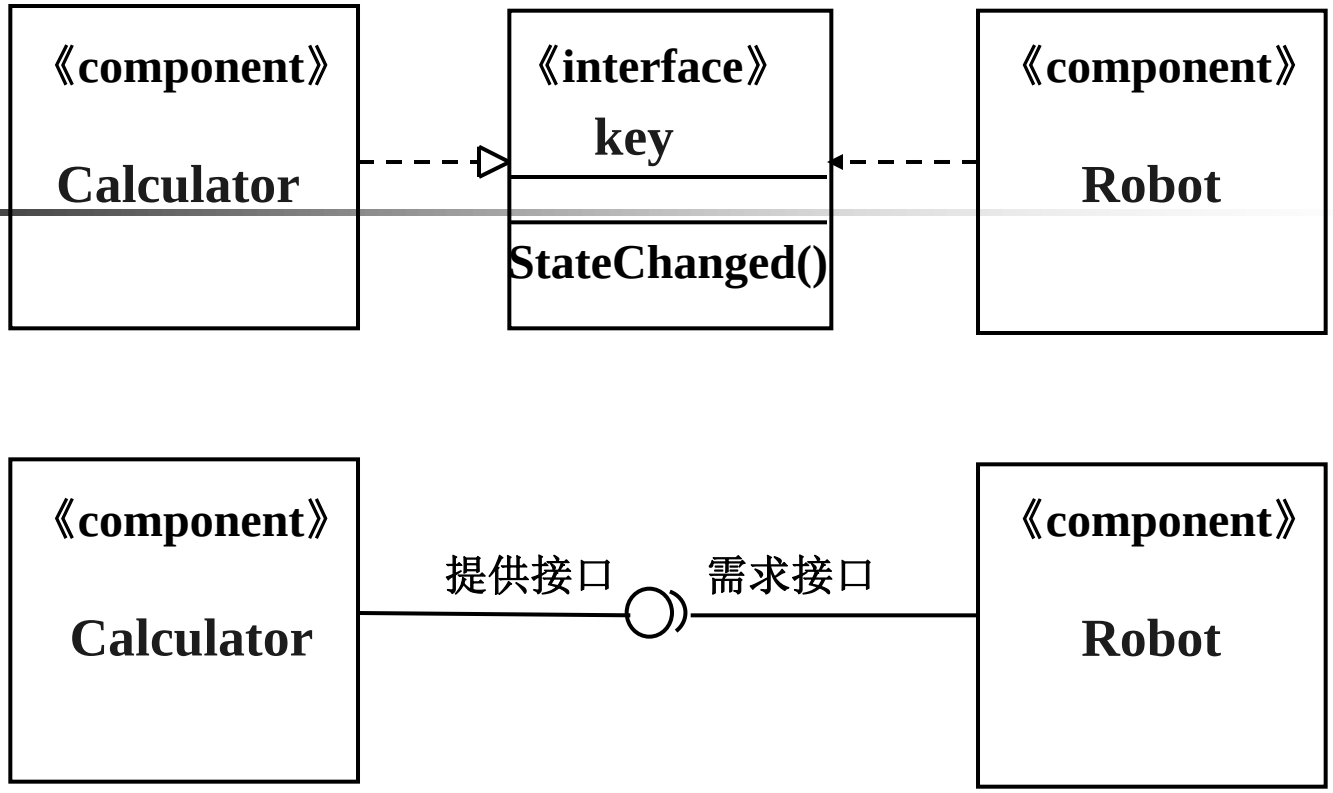
UML 组件图

- 组件在UML指的是一个实际文件，它可以是数据文件、源代码文件、二进制文件、可执行文件、动态连接库等，UML2.0称为工件（artifact)。
- 组件图用来说明这些组件之间在编译、链接或执行时的依赖关系，利用这种依赖关系能容易的分析一个组件的变化对其他组件的影响。
- 组件图中有接口和依赖关系：

表示组件接口，对外提供可见的操作。

表示依赖关系

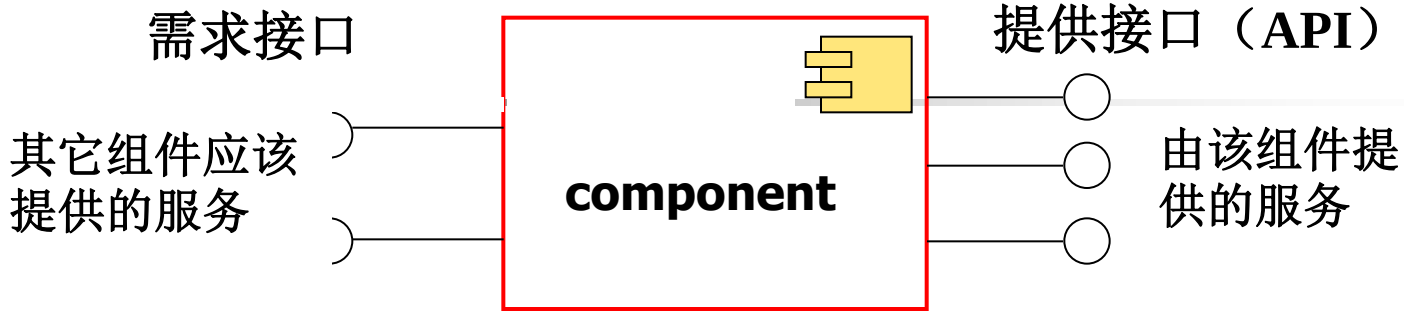
资料整理
火龙果软件
荐



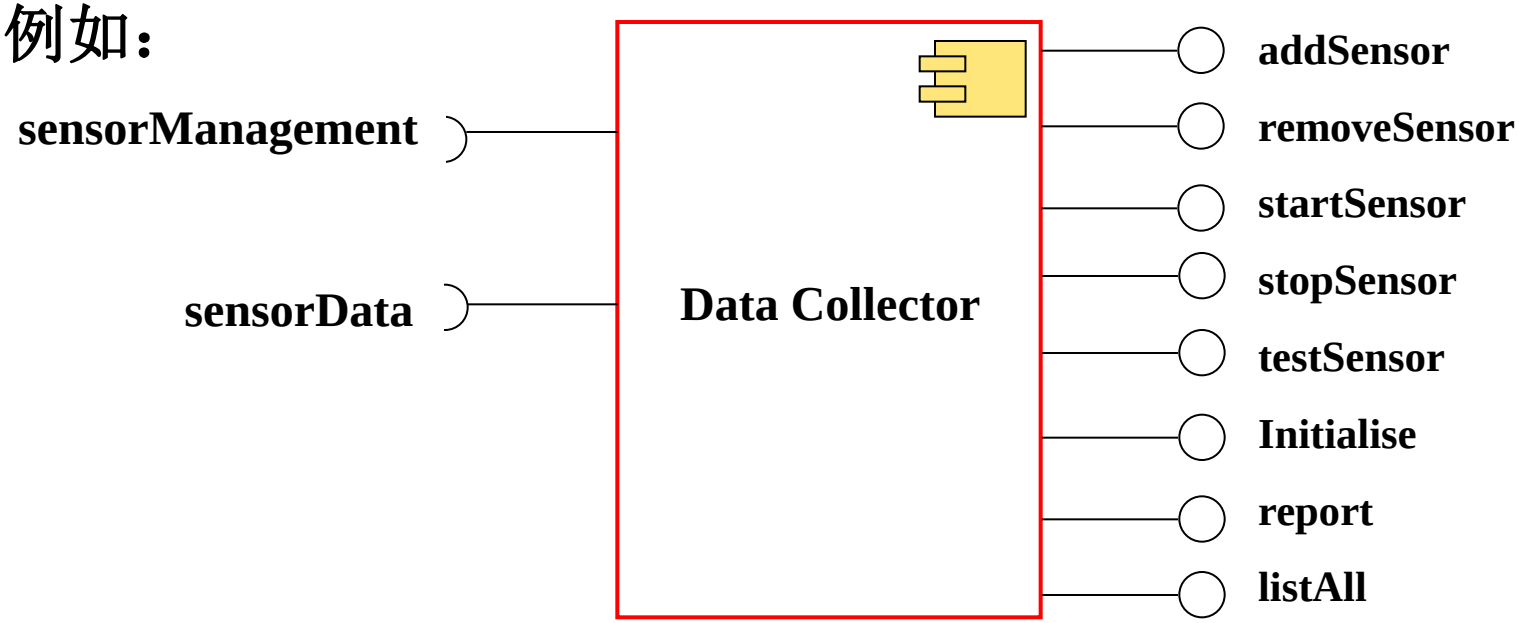
组件图中表示实现关系和依赖关系的两种方式

资料整理
火龙果软件
荐

组件有两类接口：



例如：



数据采集组件的接口

资料整理
火龙果软件
荐

组件模型

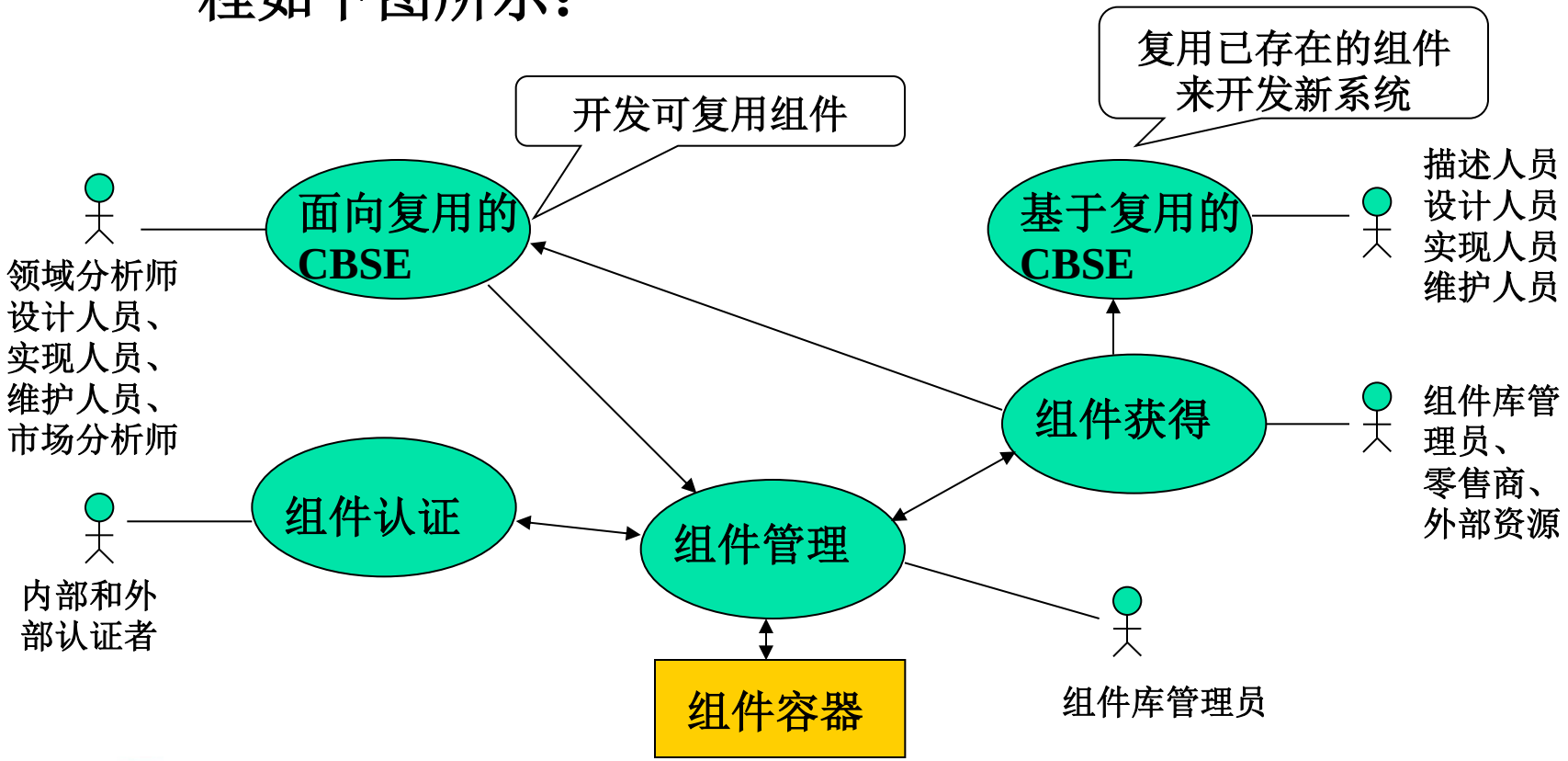
组件模型定义了组件实现、文档书写以及部署的标准。这些标准为开发者确保组件的互操作性而设立。

现在最为关注的组件模型有：**WebServices**模型，企业**Java Bean**（**EJB**）模型，**.Net**模型。

资料整理
火龙果软件
荐

基于组件的开发（CBSE过程）

- CBSE （Component-Based Software Engineering）过程如下图所示：



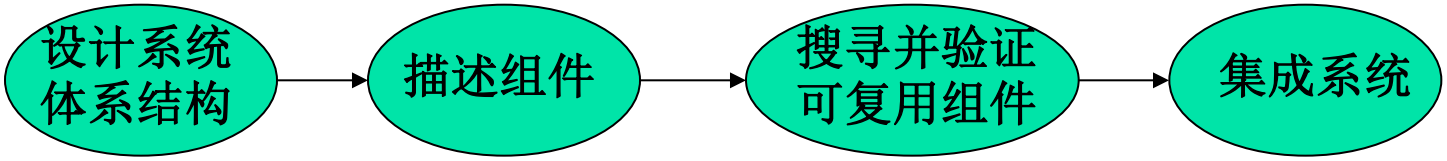
- 组件获得：得到面向复用的组件或开发成一个可复用组件的过程，该组件是内部开发或从外部资源找到。
- 组件管理：对组件进行正确的分类与存储，以便可复用。
- 组件认证：检查组件是否符合它的描述。

资料整理
火龙果软件
荐

基于组件开发的两种思路：

- 完成高层设计，对设计中的组件给出描述，以便找出可复用的组件，这些组件可在体系结构层次上加入或更详细的设计层次上加入。
- 先根据需求搜寻可复用组件，再将设计建立在获得的组件基础上。

两种思路可结合起来。



先完成架构设计的复用

复用驱动设计

资料整理
火龙果软件
荐

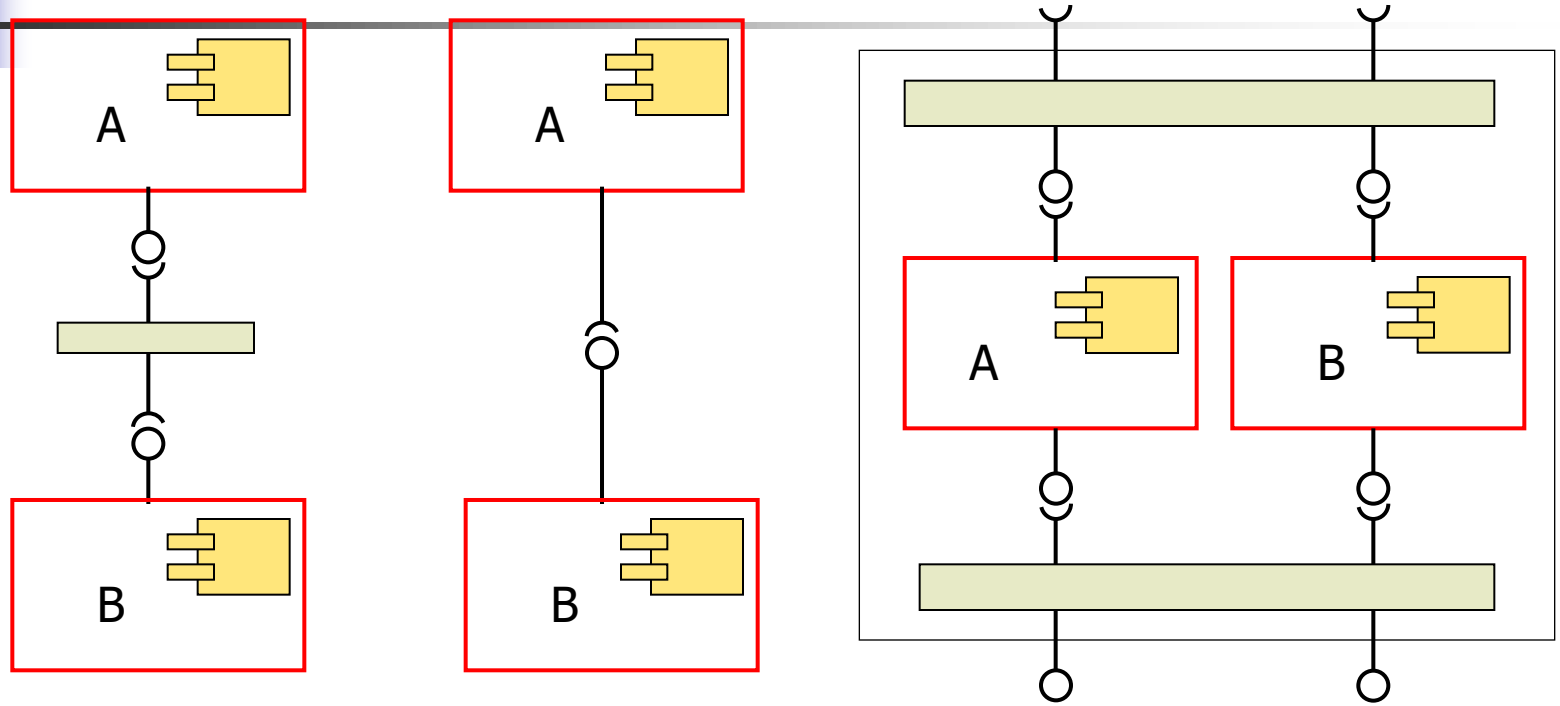
组件合成

组件合成是通过组件组装来创建新系统的过程。合成的类型包括：

- 顺序合成：各个组件的提供接口组合在一起，可能需要编写一些额外的代码来连接这些组件。
- 层次合成：一个组件直接调用另一个组件所提供的服务。即一个组件的需求接口与另一个组件的提供接口直接组合。
- 叠加合成：多个组件接口集中在一起创建一个新组件。重复操作需要删除。

（见下页图）

资料整理
火龙果软件
荐



a）顺序合成　　b）层次合成　　c）叠加合成

组件合成的类型

资料整理
火龙果软件
荐

- 在合成独立开发的可复用组件时，会经常面临接口不兼容的问题，如：
  - 参数不兼容：接口每一边的操作有相同的名字，但参数的类型和数目不同。
  - 操作不兼容：提供接口和需求接口操作名不同。
  - 操作不完备：一个组件的提供接口是另一个组件的需求接口的子集，或者相反。
- 对所有情况，都必须通过编写适配器组件来解决不兼容的问题，可将一个组件的接口转换为另一个组件的接口。

资料整理
火龙果软件
荐

7.3.2 配置管理

- 配置管理是系统开发中管理软件变更的过程，当一个团队的人员协同开发软件时这个过程尤为重要。
  - 一个程序员作出的修改会不会覆盖另一个人提交到系统中的工作成果？
  - 如何保证每个人访问的组件是最新版本？
  - 新版本出现问题时，系统组件可以退回到哪个正常版本？

  …
- 配置管理的目标是支持系统集成过程，有以下基本活动：

资料整理
火龙果软件
荐

- 版本管理　对组件不同版本的追踪提供支持。
- 系统集成　帮助开发人员定义在创建每个系统版本时所用的组件版本，这些描述可以用于编译连接需要的组件，以自动构建一个系统。
- 问题追踪　提供支持允许用户报告缺陷及其他问题，允许查看谁在修复这些问题以及修复完成的时间。

配置管理活动需要自动工具的支持。

资料整理
火龙果软件
荐

7.3.3 基于宿主机-目标机开发

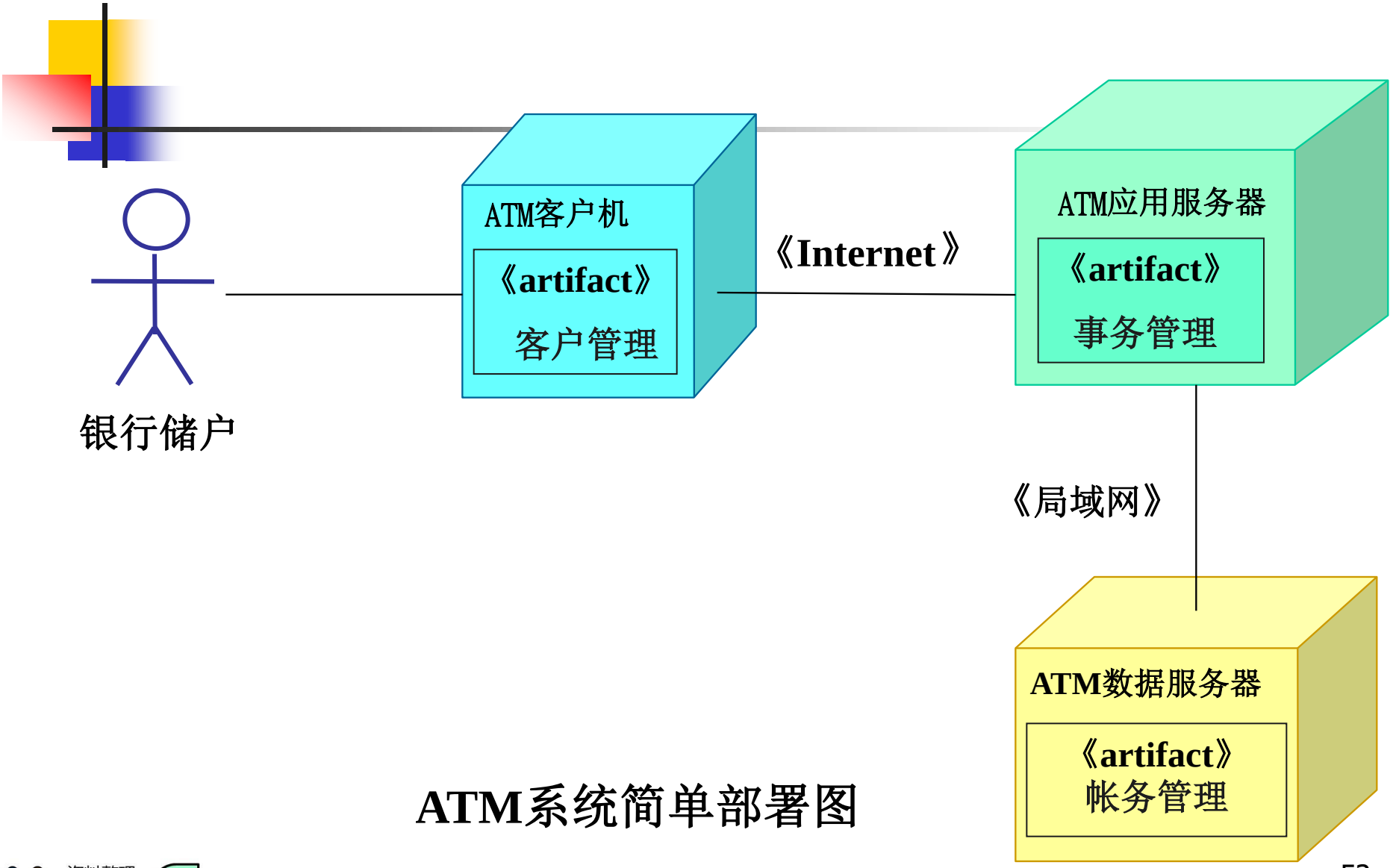
- 开发平台或运行平台除了硬件设备，还包括安装的OS、DBMS及其它支持软件。
- 若二者不同，需要将开发好的软件迁移到运行平台进行整个系统级的测试，或者在开发平台上搭建运行环境（如实时系统中的传感器）。
- 集成开发环境（IDE）支持软件开发，它将工具和集成机制整合在一起协同工作。

资料整理
火龙果软件
荐

- 对于分布式系统，需要将组件部署到目标平台上，UML部署图用来描述软硬件部署的决策。
- UML部署图描述系统中硬件设备的配置、通信以及在各硬件设备上组件的配置。
- UML2.0对该图中硬件设备定义为执行工件（artifact）的节点，节点之间的连接为通信路径（不一定是电线或电缆），路径上一般要标明通信协议或网络类型。

下图是银行ATM系统的简单部署图。

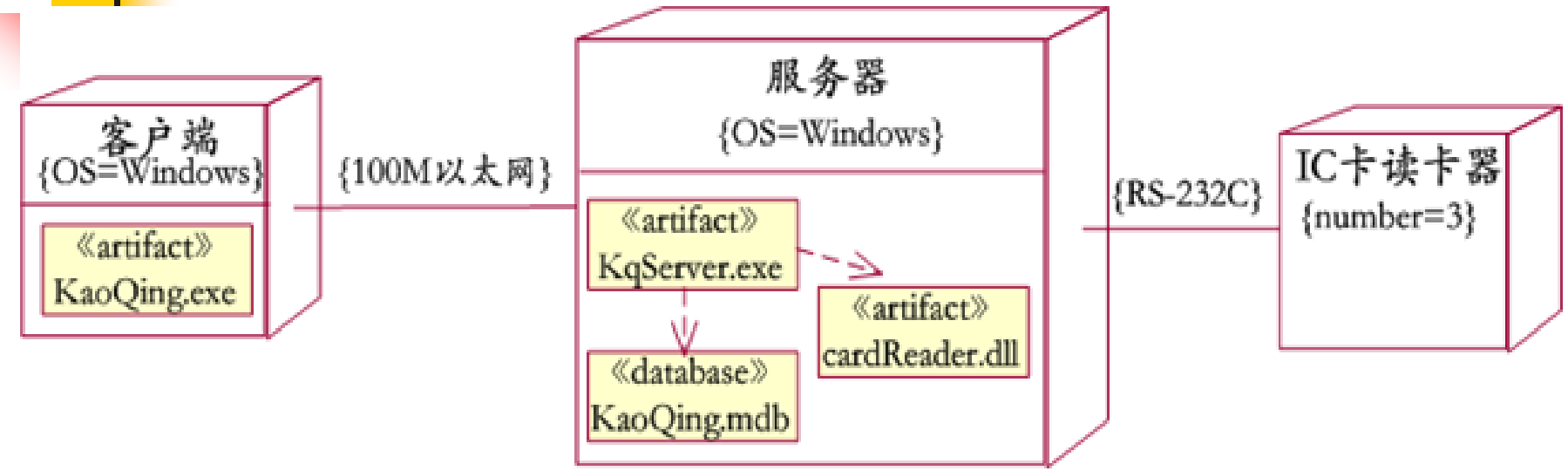
资料整理
火龙果软件
荐



ATM系统简单部署图

资料整理
火龙果软件
荐

读卡考勤系统部署图



- **IC**卡读卡器：**3**个，提供给员工刷卡用，它将收集刷卡的时间信息，传给应用系统，并存入数据库中。
- 应用服务器：用来负责从**IC**卡读卡器中收集信息，并对管理人员提供员工设置、考勤查询等功能。
- 数据库服务器：由于该系统比较小，因此在物理上数据库服务器与应用服务器合并，用来存储考勤数据。
- 客户端软件：提供给管理人员使用，连接服务器，完成相应操作。

资料整理
火龙果软件
荐

7.4 开源开发

- 开源开发指将软件系统的源代码公布出来，允许志愿开发人员加入到开发过程中。
- 开源软件的版权持有人在软件协议的规定之下保留一部分权利允许用户学习、修改、增进提高软件的质量。源代码修改后，可在依照同一许可条件下再发布。

资料整理
火龙果软件
荐

按照使用条件的不同，开源软件许可条件分为三类：

1. **GNU 通用公共许可GPL**：也称“互惠”许可。即使用该类许可开源软件搭建的软件，源代码也必须以相同许可证公开。

2. **GNU 宽松的通用公共许可LGPL**：使用该类许可开源软件搭建的软件，可以不公开自己的代码，但若对开源代码有修改，则需要将自己的代码以相同许可证公开。

3. **Berkley标准许可BSD**：

使用该类许可开源软件搭建的软件（包括修改）再公开时，没有特殊限制，只需要申明许可。代表许可类型还有MIT等 。

资料整理
火龙果软件
荐

7.5 嵌入式软件的设计与实现（参考第20章）

- 嵌入式软件系统是对环境中的事件做出响应的软件/硬件系统的一部分，通常都是实时系统。
- 实时系统是一个必须实时地对事件作出响应的软件系统。它的功能正确与否取决于系统产生的结果，以及产生这个结果所花的时间。
- 除了包含实施响应的特点外，嵌入式软件和其它软件还有以下不同：

资料整理
火龙果软件
荐

- 嵌入式软件往往保持持续运行直到硬件被关闭。
- 和系统环境的交互是不可预测和控制的。设计是基于多进程的并发的。
- 存在影响系统设计的物理限制（如空间、重量）。
- 要和各种硬件设备交互，有的没有设备驱动软件。
- 安全性、可靠性占据主要地位。需使用成熟的技术。

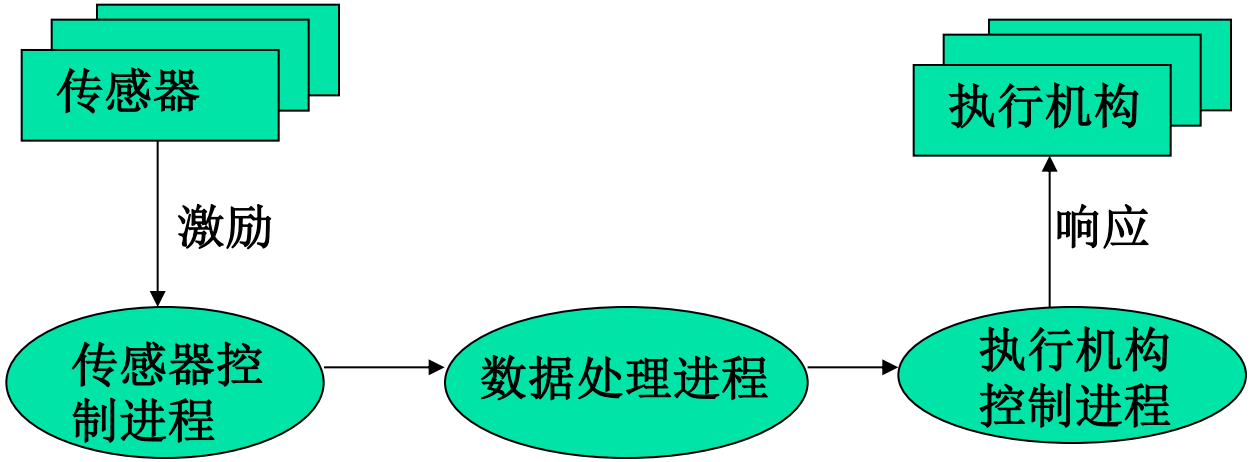
资料整理
火龙果软件
荐

1. 嵌入式系统设计

- 嵌入式系统的设计是一个系统工程的过程，要将系统能力分解为硬件实现和软件实现的组件。
- 嵌入式实时软件设计设计的一般方法时基于激励-响应模型。激励（stimulus）是事件，响应是发送的信号或消息。
- 响应时间必须在设计过程早期考虑。激励/响应或事件（而不是对象或功能）是系统设计的核心内容。
- 激励有两类：
  - 周期性的激励　在每一个时间间隔内发生。如系统每隔50ms检查一次传感器，根据传感器的值采取响应行动。
  - 非周期性的激励　激励不规则的发生，如中断。

资料整理
火龙果软件
荐

- 实时系统需要响应不同时间里发生的激励。实时系统一般被设计成一组并发协作的进程。见图：



传感器和执行器进程

资料整理
火龙果软件
荐

- 设计过程：

（1）平台选择：采用的硬件和实时操作系统。（影响选择的因素包括时间和资源限制、价格、经验等）

（2）激励-响应识别：识别系统必须处理的一些激励和相关的响应。

（3）时序分析：给出每个激励和响应的时间约束。

（4）进程设计：将激励和响应聚集到几个并发进程中，为实时系统建模。

（5）算法设计：对每个激励和响应的运行设计算法。

（6）数据设计：定义进程间交换的信息，设计相应的数据结构来管理信息交换。

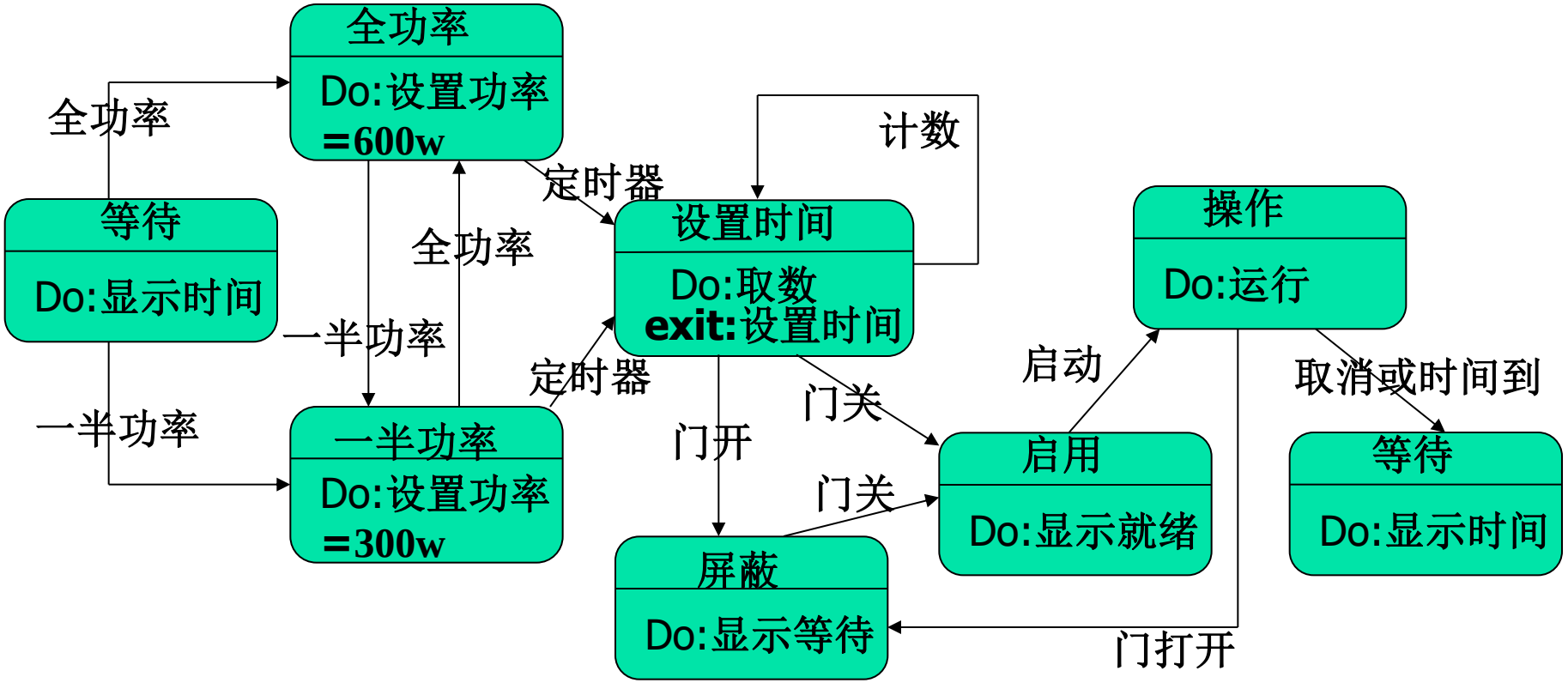
（7）进程调度：设计调度程序，确保进程都能及时启动并满足时限要求。

该过程是一个迭代过程。

资料整理
火龙果软件
荐

2. 实时系统建模

实时系统必须响应在不规则时间间隔内发生的事件，这些事件（或激励）引起系统转换到不同的状态。因此状态机模型能很好的反映实时系统的行为。



简单微波炉的状态机模型

资料整理
火龙果软件
荐

3. 实时编程

- 语言必须具有访问系统硬件的能力以及能满足时序的要求，如汇编语言。
- 使用“系统调用”提供对软件并发性、多线程运行的支持。如C语言通过调用开发平台提供的多线程库函数接口，就能使用操作系统的一些功能。
- 实时Java、Ada，包含了线程机制、异步事件处理机制和时序描述。

资料整理
火龙果软件
荐