第4章 面向对象的软件分析与设计

4.1 4+1模型及UML语言实现
4.2 面向对象的软件工程
4.3 用例图
4.4 活动图
4.5 用户界面设计
4.6 类图
4.7 交互图
4.8 包图
4.9 系统与模型
4.10 部署图

第4章 面向对象的软件分析与设计

4.8 包图(package diagrams)

4.8.1 为什么要有包?

4.8.2包图的组成要素

4.8.3 包图的常见类型

4.8.4 包的关系

4.8.5 包图的制作步骤

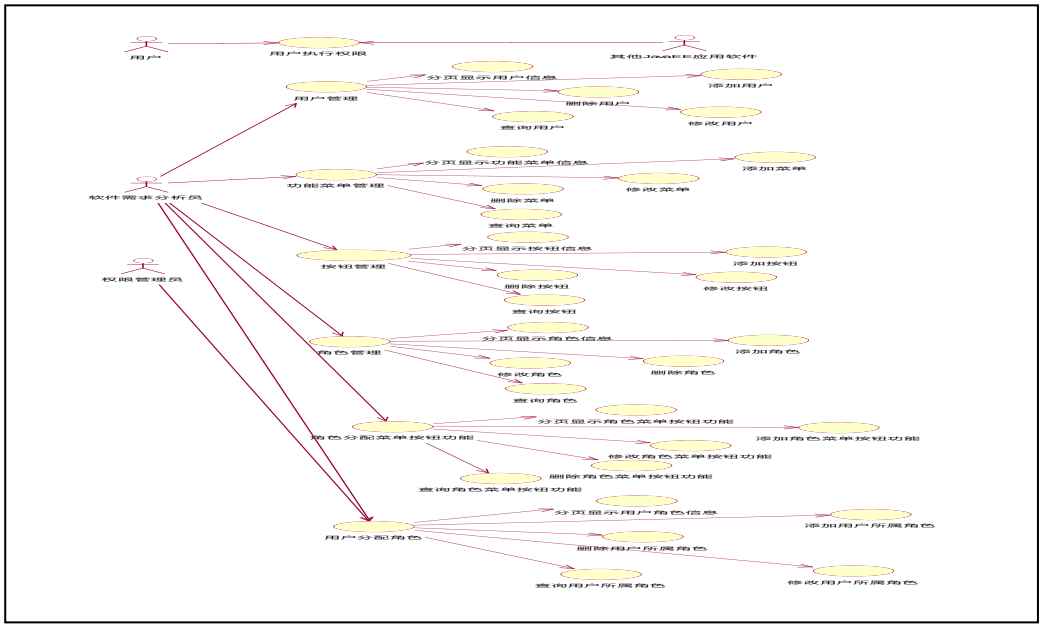
4.8.6 Case:图书管理系统包图建模

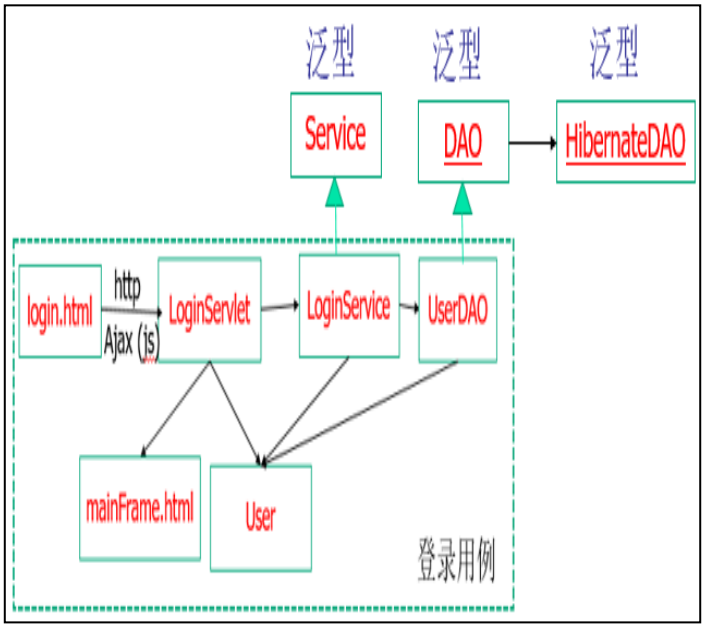
4.8.7 软件目录结构

4.8 包图(package diagrams)

4.8.1 为什么要有包?

- 对复杂系统中的模型元素进行**分组**，并且给分好组的元素提供一个命名空间。



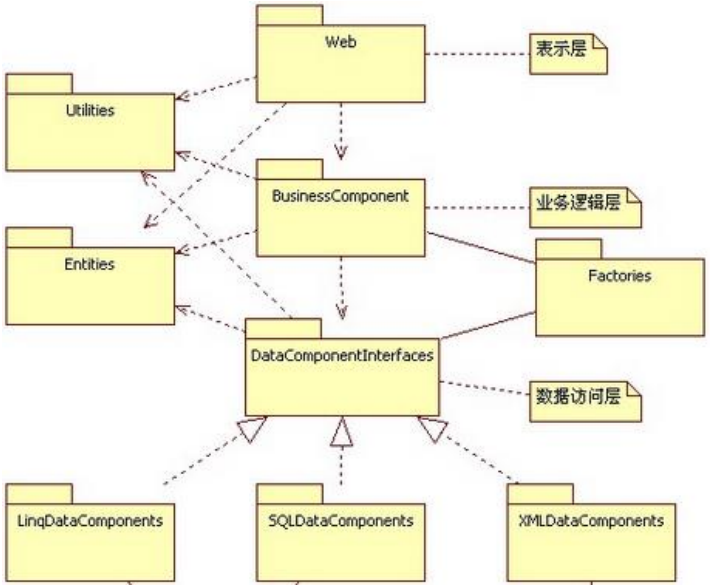


- 能够对一个大系统进行分解，降低系统的复杂度。
- 方便团队成员的分工。
- 可以减小因模块内部的变化而引起模块间相互影响的可能

因此，我们在软件设计中引入了包的概念

4.8.1 为什么要有包?

- 在UML的建模机制中，**模型的组织**是通过包（Package）来实现的。
  - 包可以把所建立的各种模型（包括静态模型和动态模型）组织起来，形成各种功能或用途的模块，并可以控制包中元素的可见性，以及描述包之间的依赖关系。



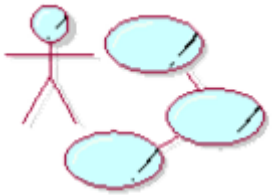
4.8 包图(package diagrams)

4.8.2包图的组成要素

- 类
- 接口
- 组件
- 节点
- 协作
- 用例
- 图
- 其他包

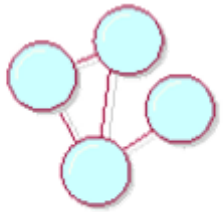
4.8 包图(package diagrams)

4.8.3 包图的常见类型



BusinessUseCaseModelC

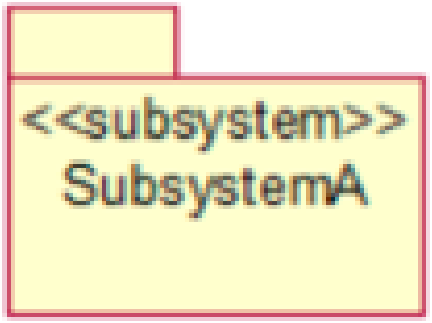
用例包

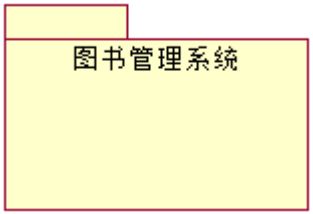


BusinessAnalysisModelA

系统

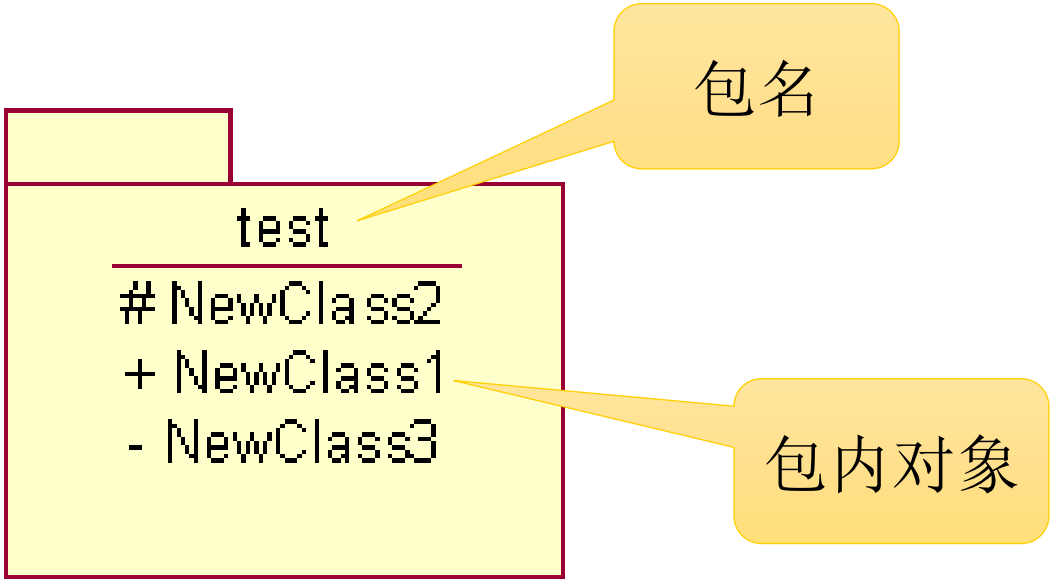
BusinessDesignModelB





4.8.3 包图的常见类型

- 包的示例



可见性：

+ public
# protect
- private

4.8.3 包图的常见类型

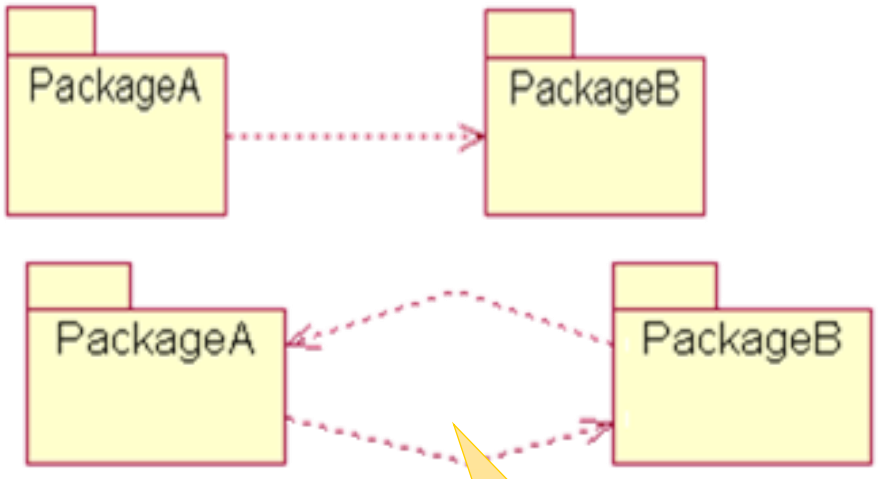
- 说明
  - 一个模型元素不能被一个以上的包所拥有
  - 如果包被撤销，其中的元素也要被撤销

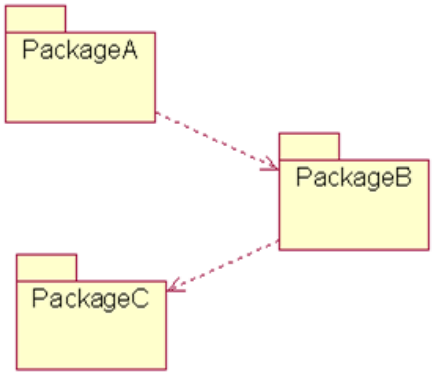
4.8 包图(package diagrams)

4.8.4 包的关系

- 包的依赖关系

❖包的依赖关系通常是指这两个包所包含的模型元素之间存在一个和多个依赖的关系



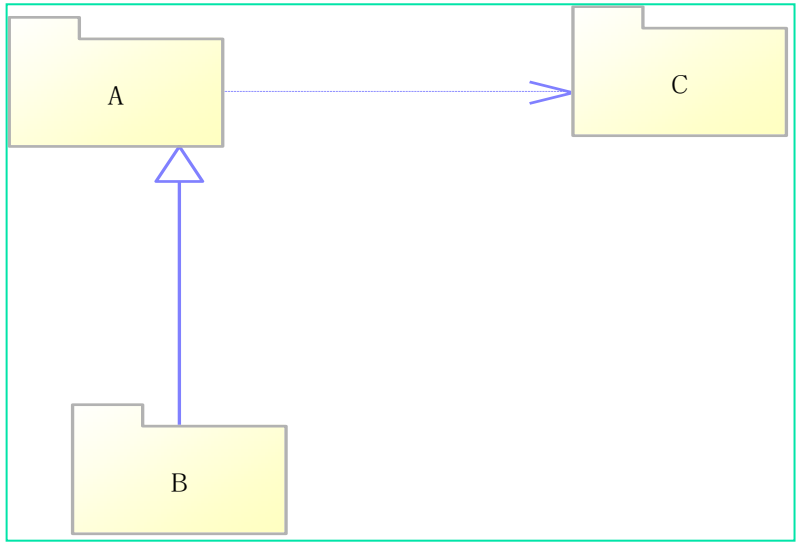


包的循环依赖关系示例

4.8.4 包的关系

包的泛化关系

- 包之间的泛化关系与对象类之间的泛化关系十分类似。如果一个包继承了另外一个包的接口，我们就说这个包与另外一个包有泛化关系。
- 使用  符号
- 包的泛化关系不常用



4.8 包图(package diagrams)

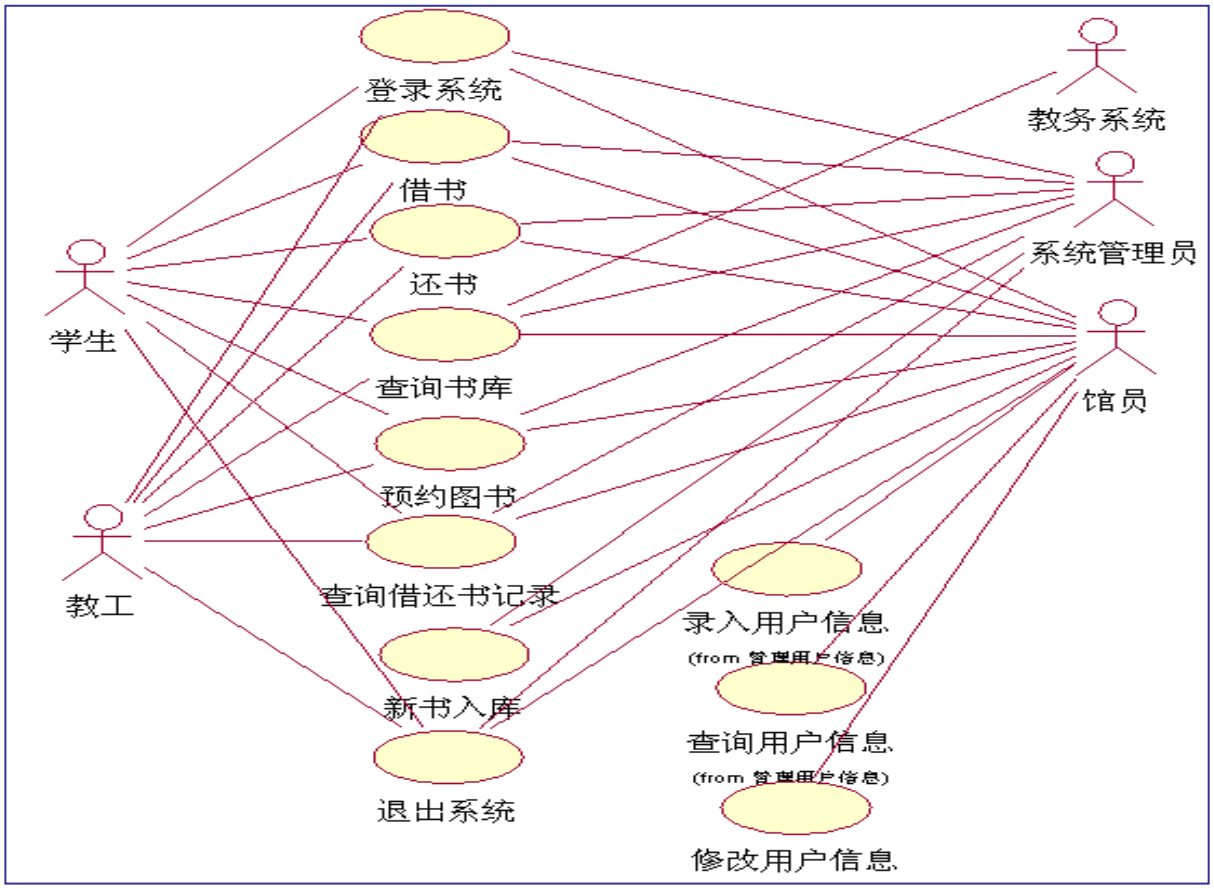
4.8.5 包图的制作步骤

- 分析系统的模型元素，把概念或语义上相近的元素归入同一个包
- 对于每个包，标出其模型元素的可视性，确定元素的访问属性是公共、保护或者私有
- 确定包之间的依赖联系
- 绘制包图，对结果进行细化

4.8 包图(package diagrams)

4.8.6 Case:图书管理系统包图建模

- 用例图建模

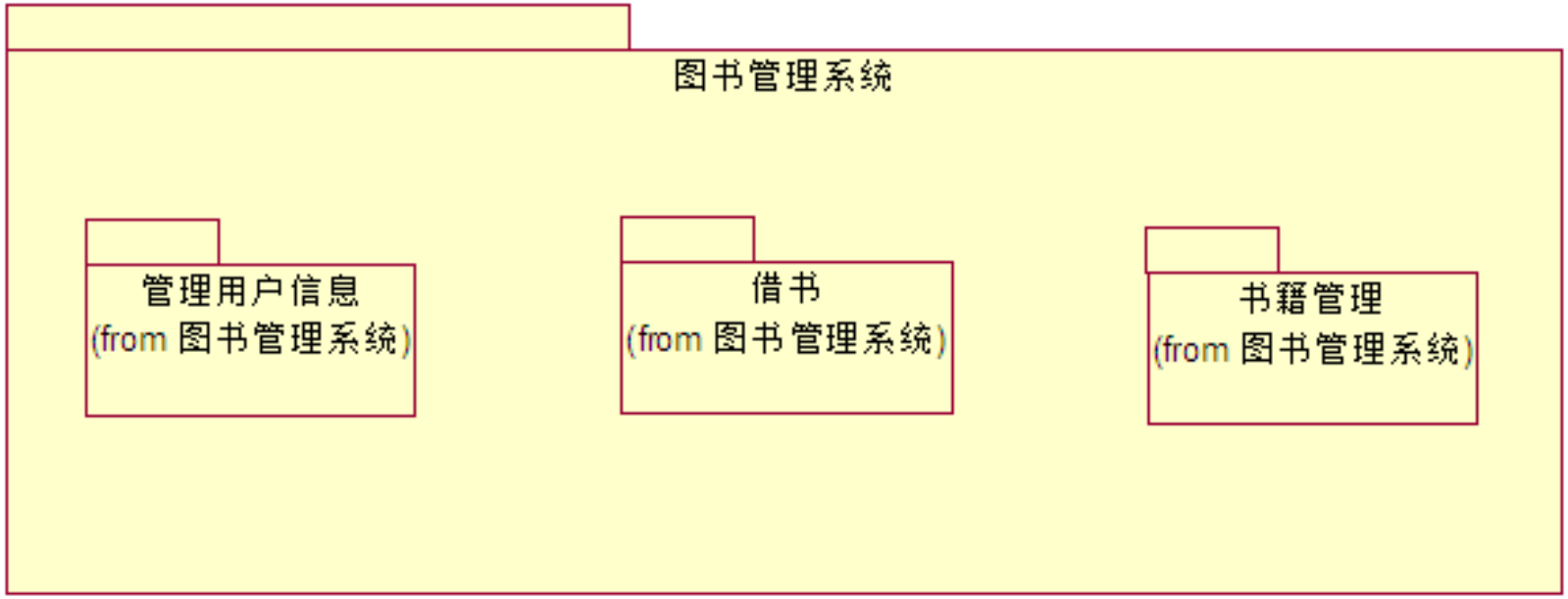


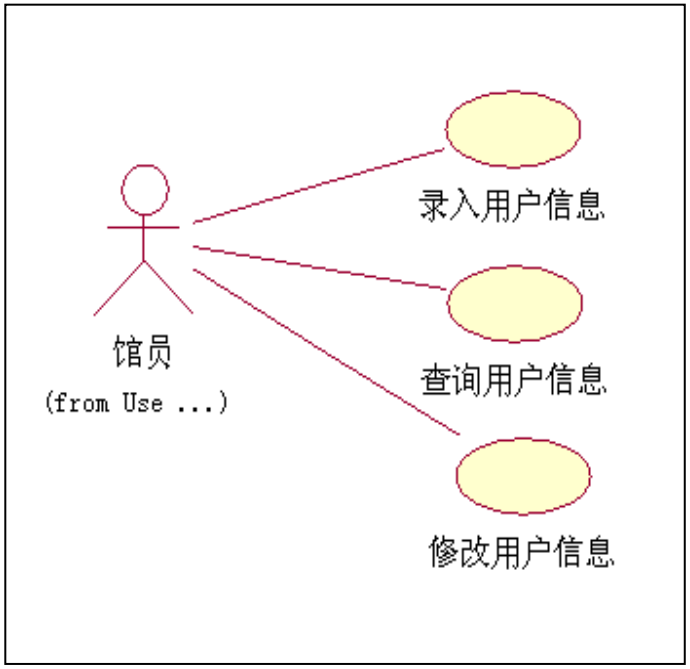
- 分析系统的模型元素，把概念或语义上相近的元素归入同一个包
- 对于每个包，标出其模型元素的可视性，确定元素的访问属性是公共、保护或者私有
- 确定包之间的依赖联系
- 绘制包图，对结果进行细化

4.8.6 Case:图书管理系统包图建模

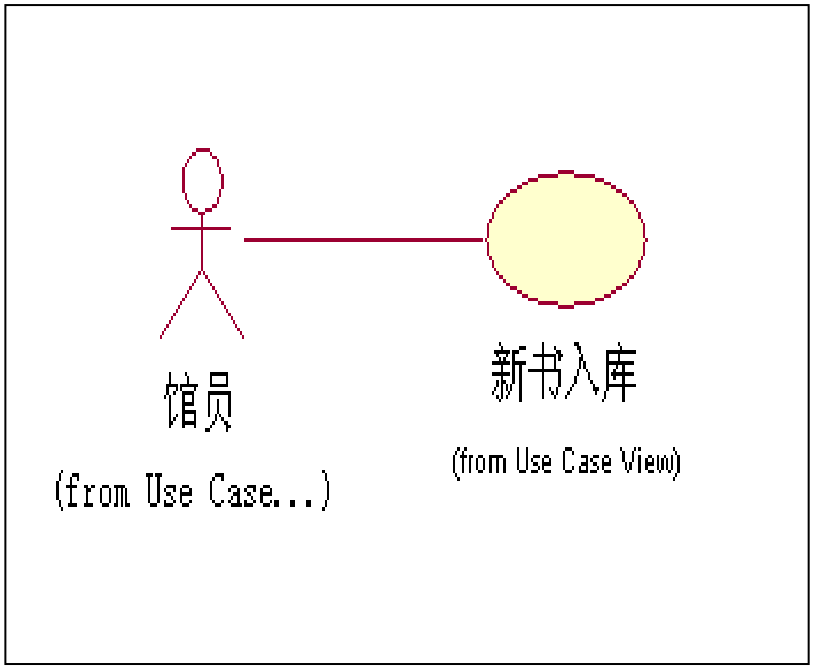
- 用例图建模
  - ❖把概念或语义上相近的元素（用例）归入同一个包

图书管理系统的用例包



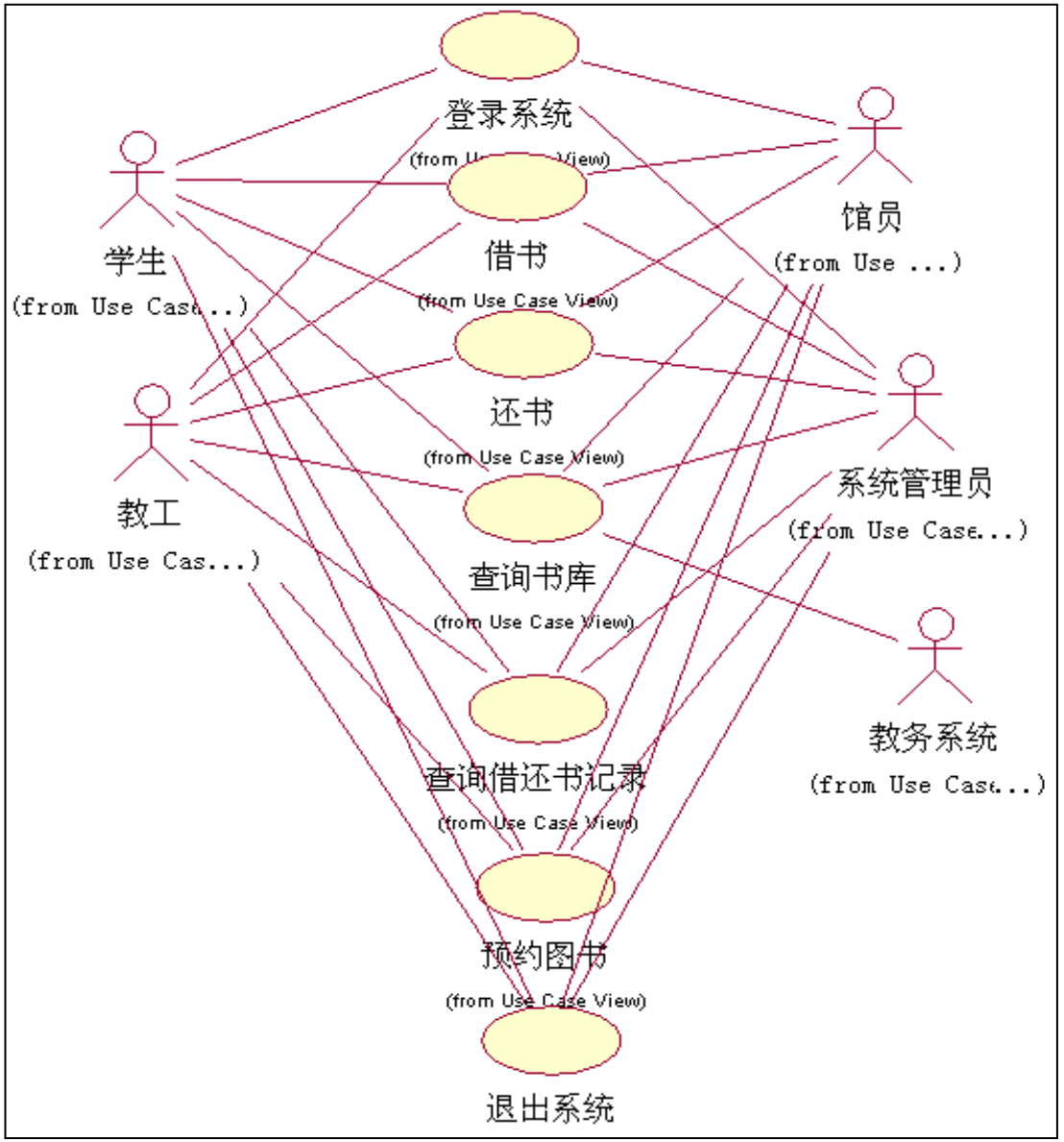


管理用户信息用例图



书籍管理用例图

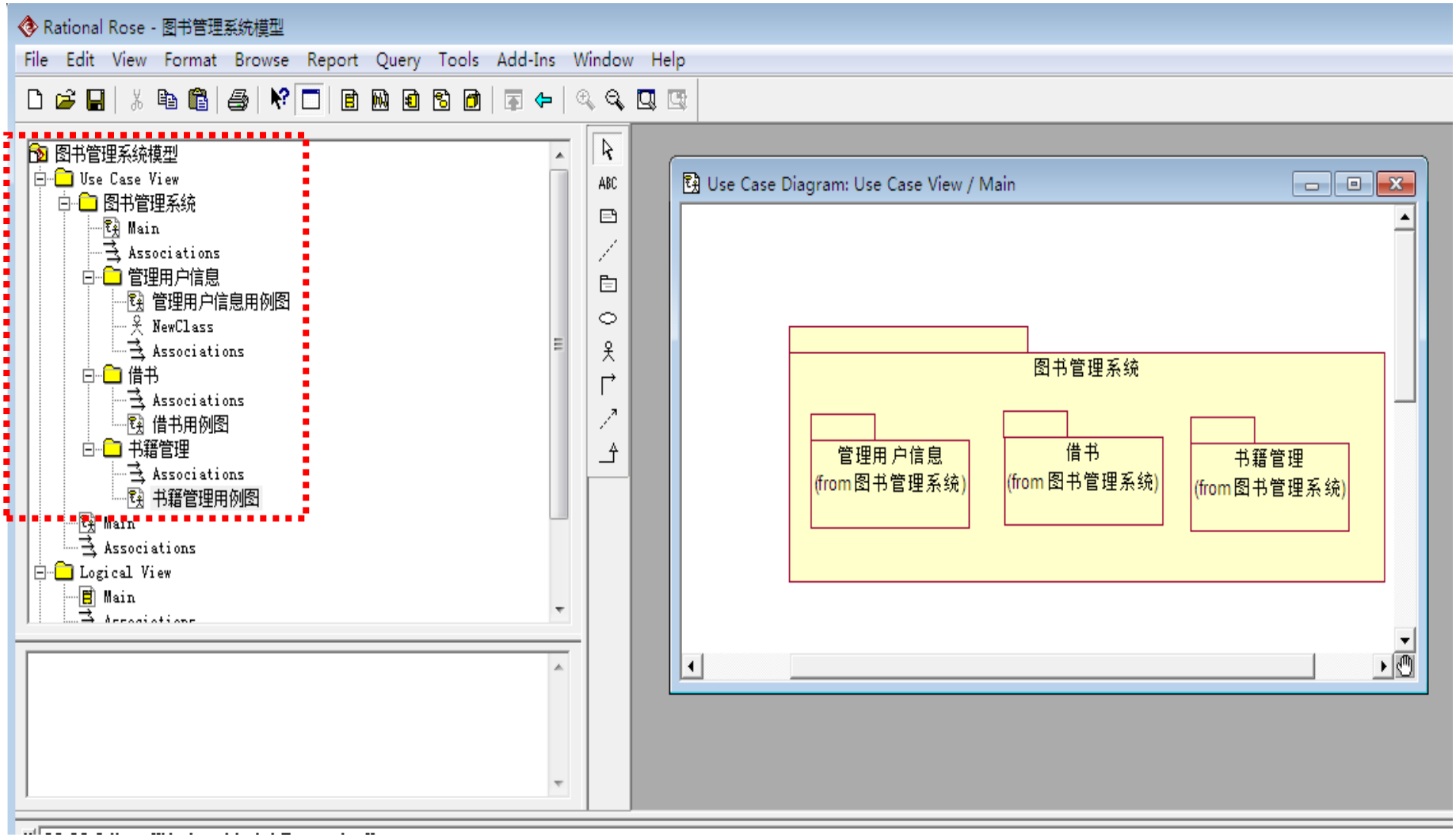
借书用例图


登录系统
(from U /iew)
借书
(from Use Case View)
还书
(from Use Case View)
查询书库
(from Use Case View)
查询借还书记录
(from Use Case View)
预约图书
(from Use Case View)
退出系统
学生
(from Use Case...)
教工
(from Use Cas...)
馆员
(from Use ...)
系统管理员
(from Use Case...)
教务系统
(from Use Case...)

4.8.6 Case:图书管理系统包图建模

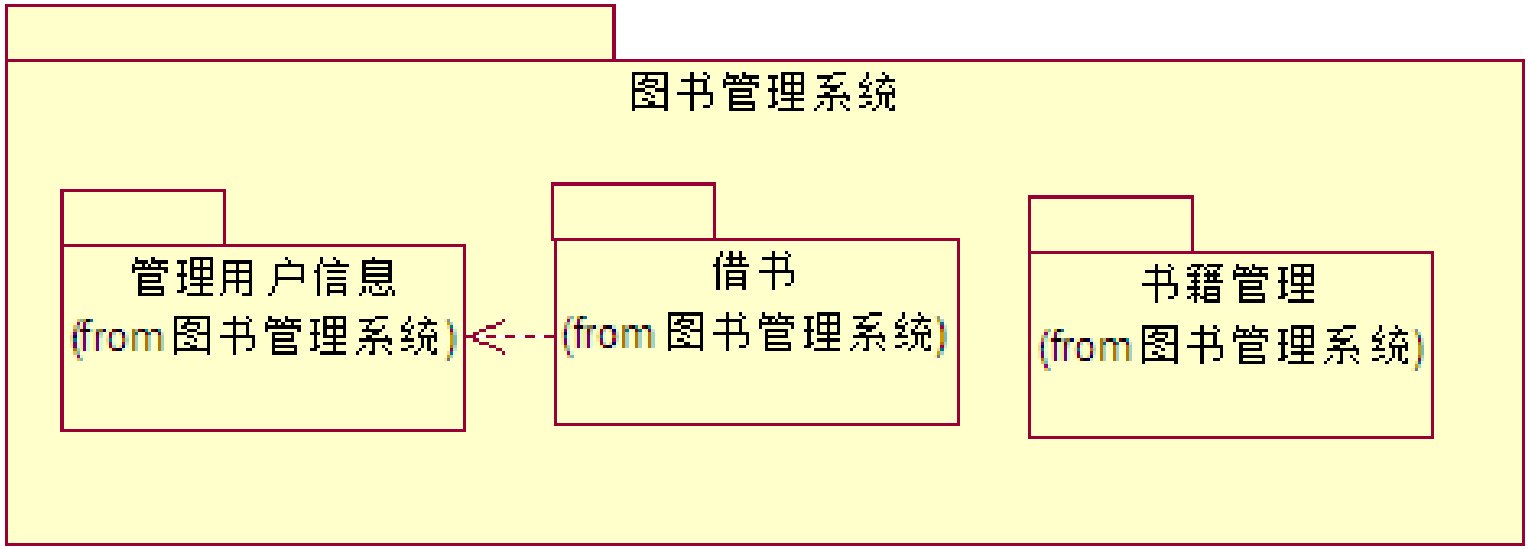
用例图建模

- Rational rose建模



4.8.6 Case:图书管理系统包图建模

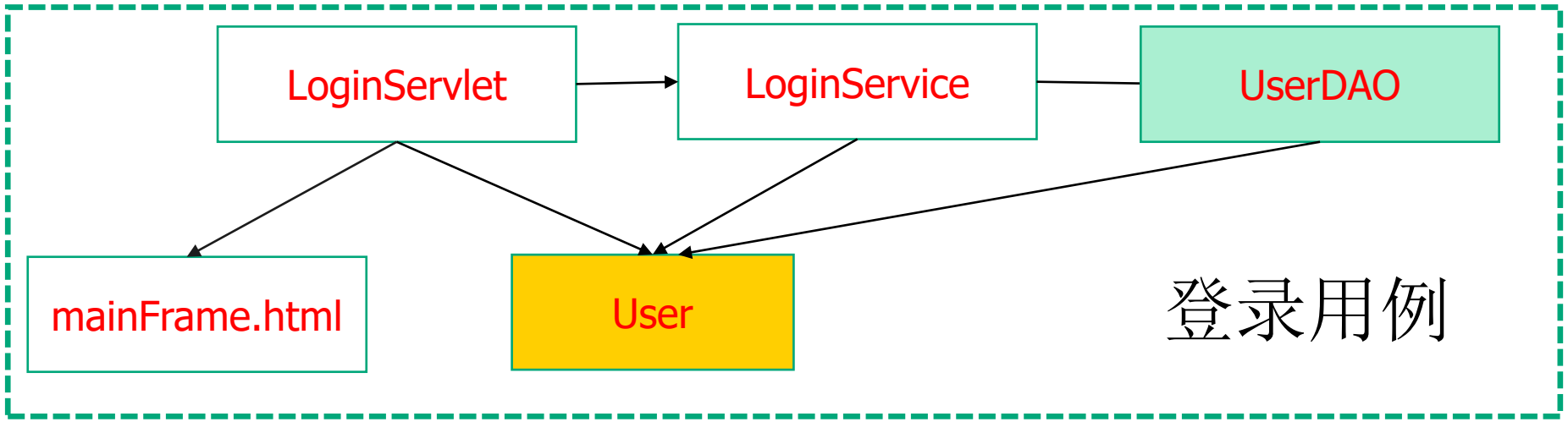
- 用例图建模
  - 确定包之间的依赖联系

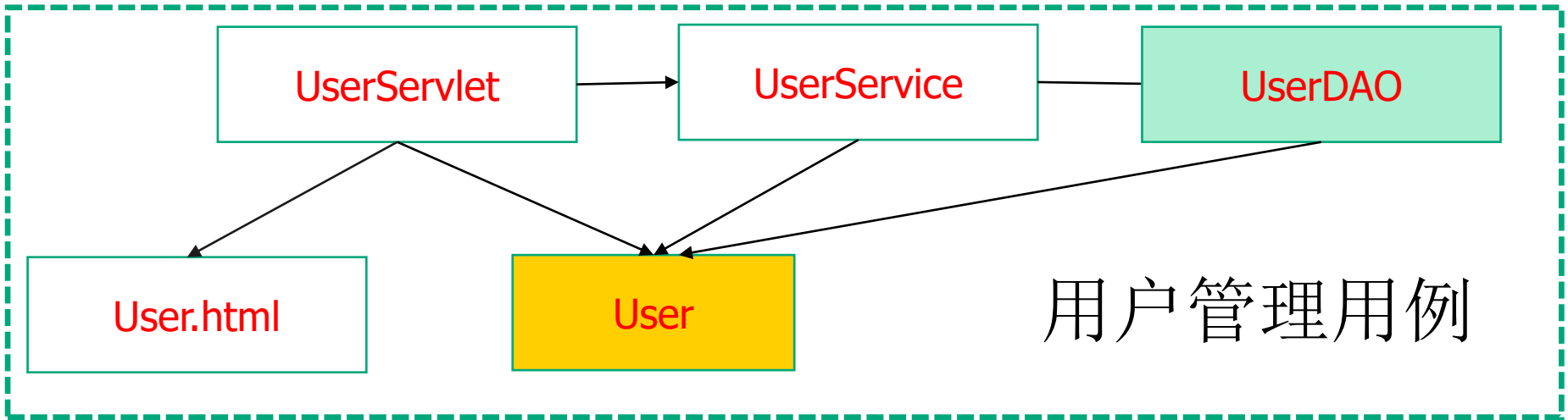


4.8.6 Case:图书管理系统包图建模

类图建模

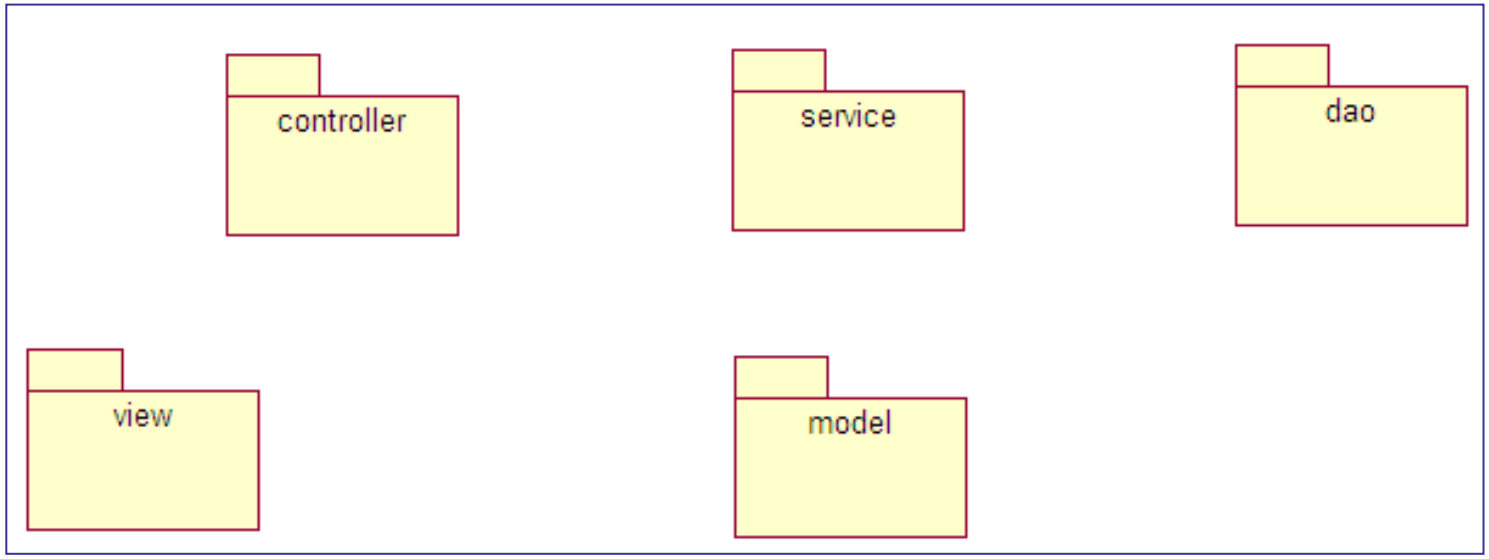
- 分析系统的模型元素，把概念或语义上相近的元素归入同一个包
- 对于每个包，标出其模型元素的可视性，确定元素的访问属性是公共、保护或者私有
- 确定包之间的依赖联系
- 绘制包图，对结果进行细化





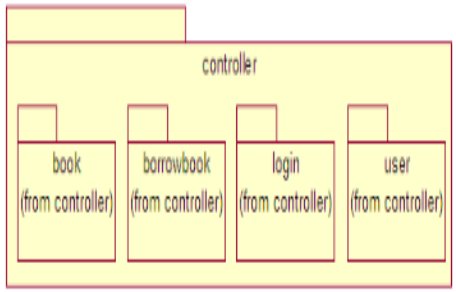
4.8.6 Case:图书管理系统包图建模

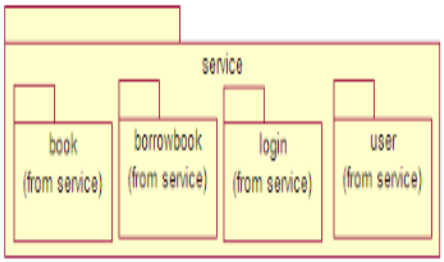
- 类图建模
  - ❖把概念或语义上相近的元素（类）归入同一个包
    - 按照MVC设计模式整个系统包为下图所示：

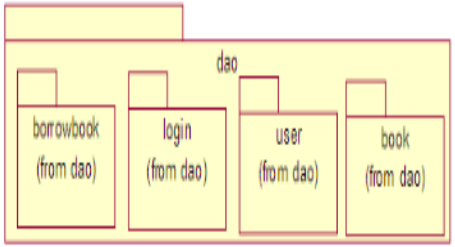


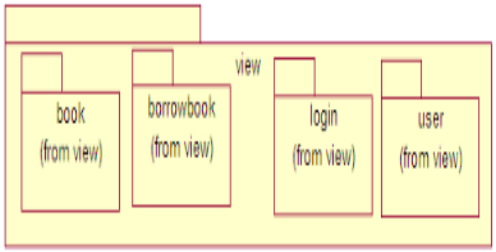
4.8.6 Case:图书管理系统包图建模

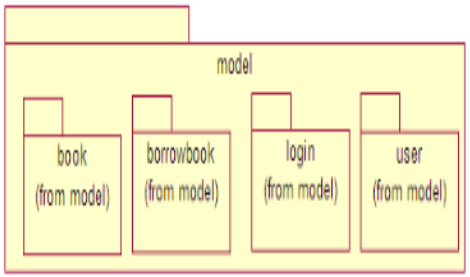
- 类图建模
  - ❖把概念或语义上相近的元素（类）归入同一个包
    - 对上述包图中的每个包按照用例进一步细化：







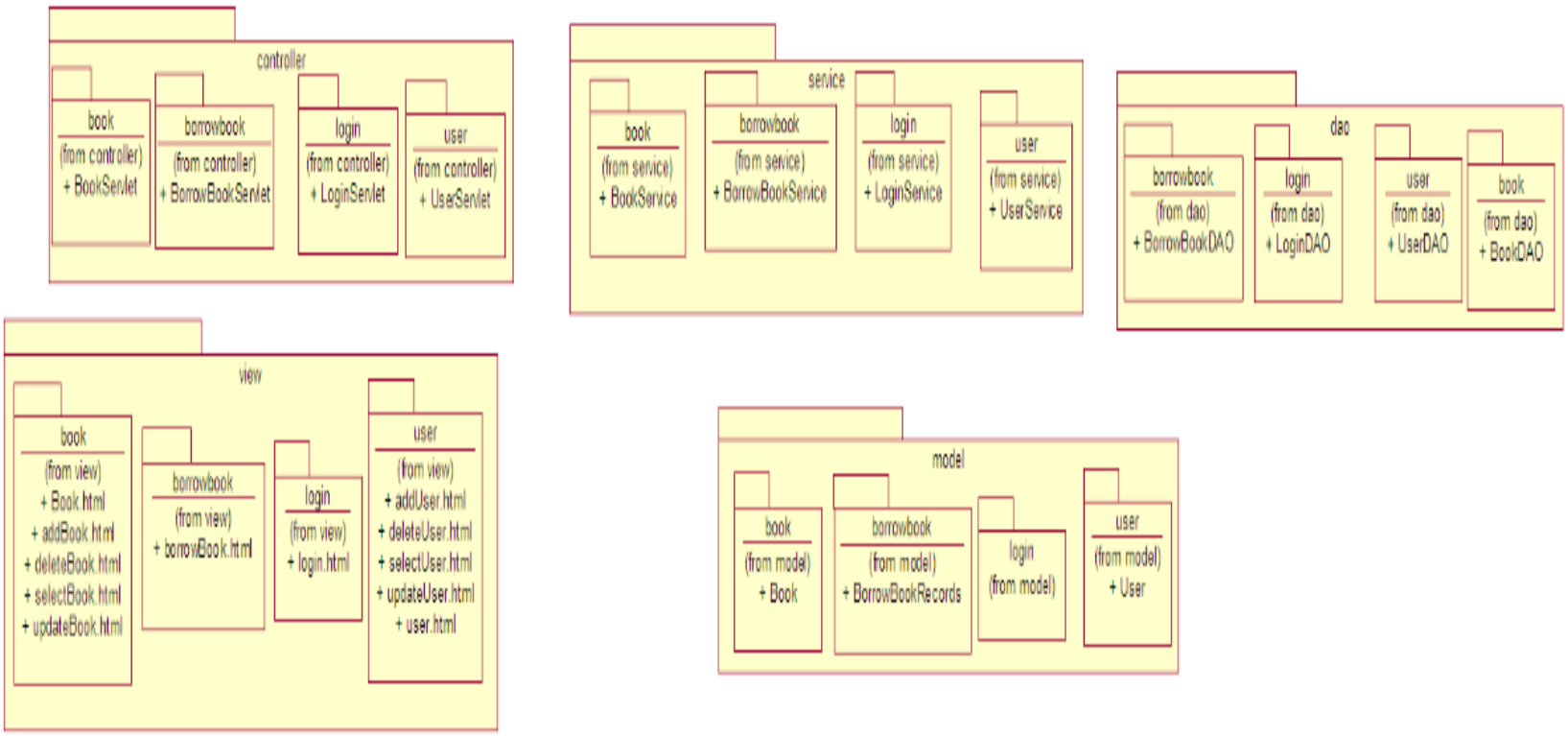




package_library2.png

4.8.6 Case:图书管理系统包图建模

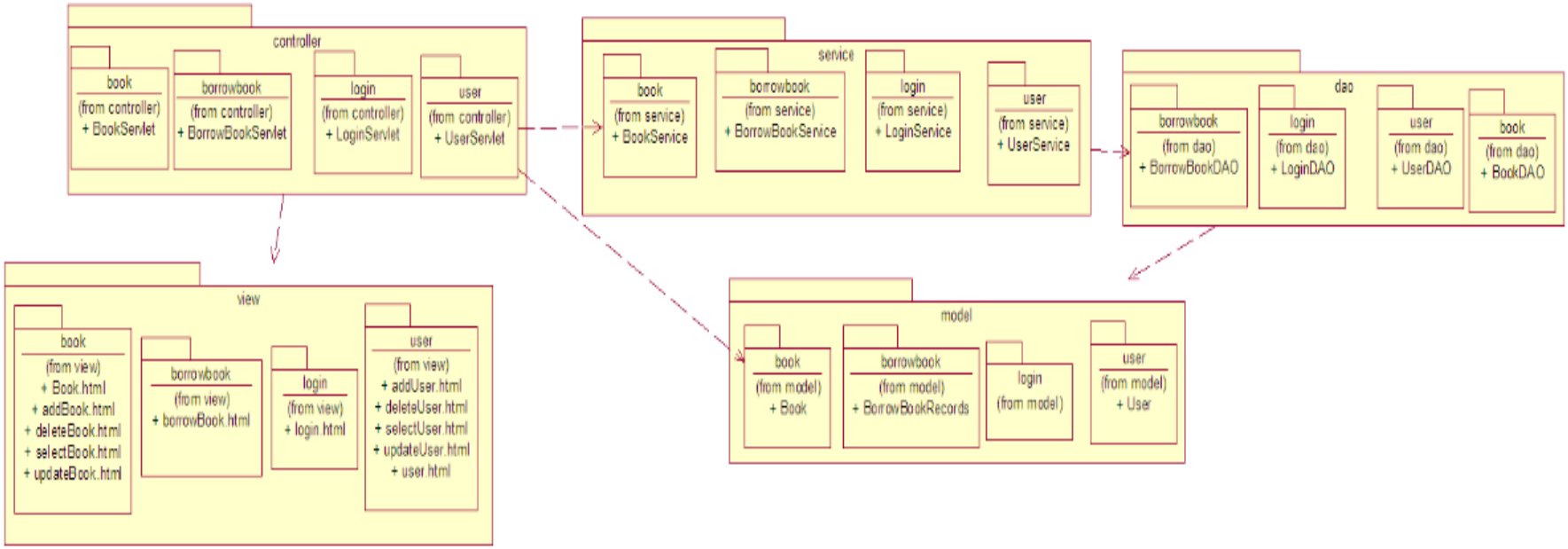
- 类图建模
  - ❖把概念或语义上相近的元素（类）归入同一个包
    - 显示每个包中的类



package_library3.png

4.8.6 Case:图书管理系统包图建模

- 类图建模
  - 确定包之间的依赖联系



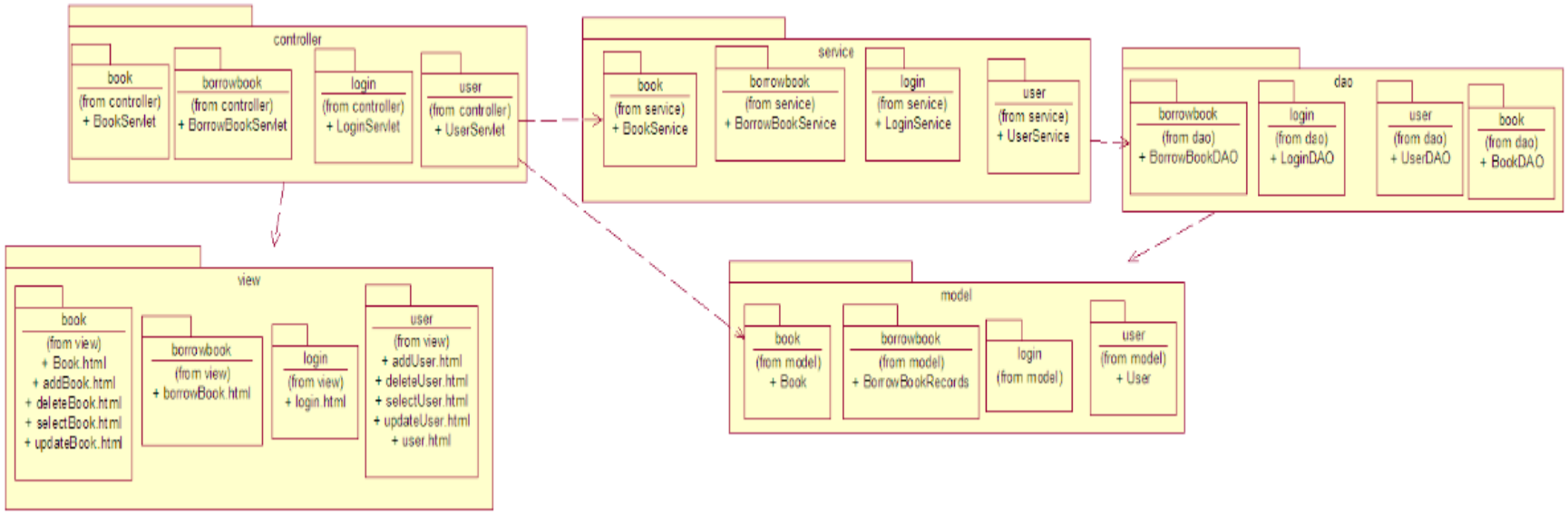
package_library4.png

4.8.6 Case:图书管理系统包图建模

- 类图建模
  - ❖确定包之间的依赖联系

    绘制包图，对结果进行细化

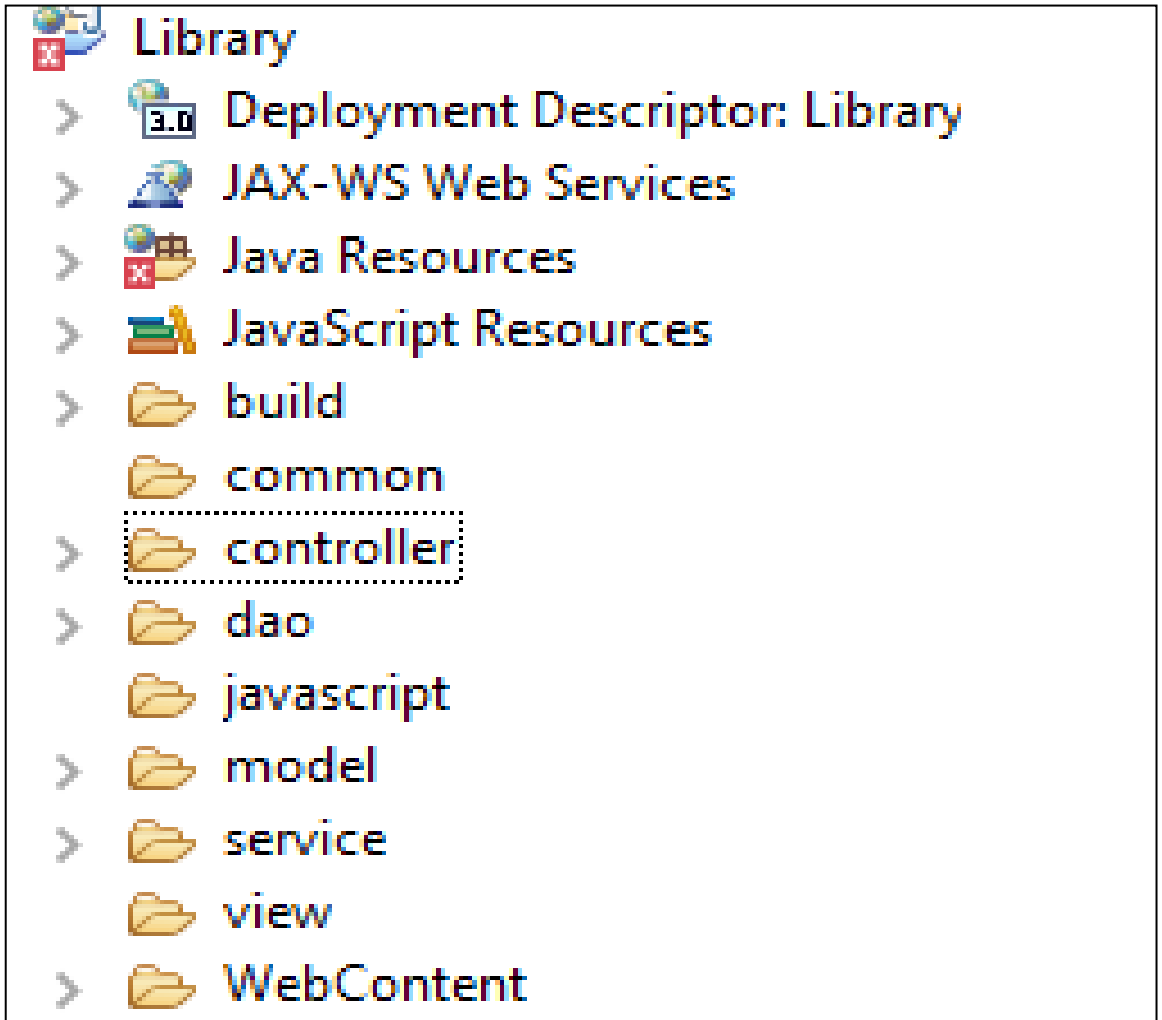
package_library4.png



4.8 包图(package diagrams)

4.8.7 软件目录结构

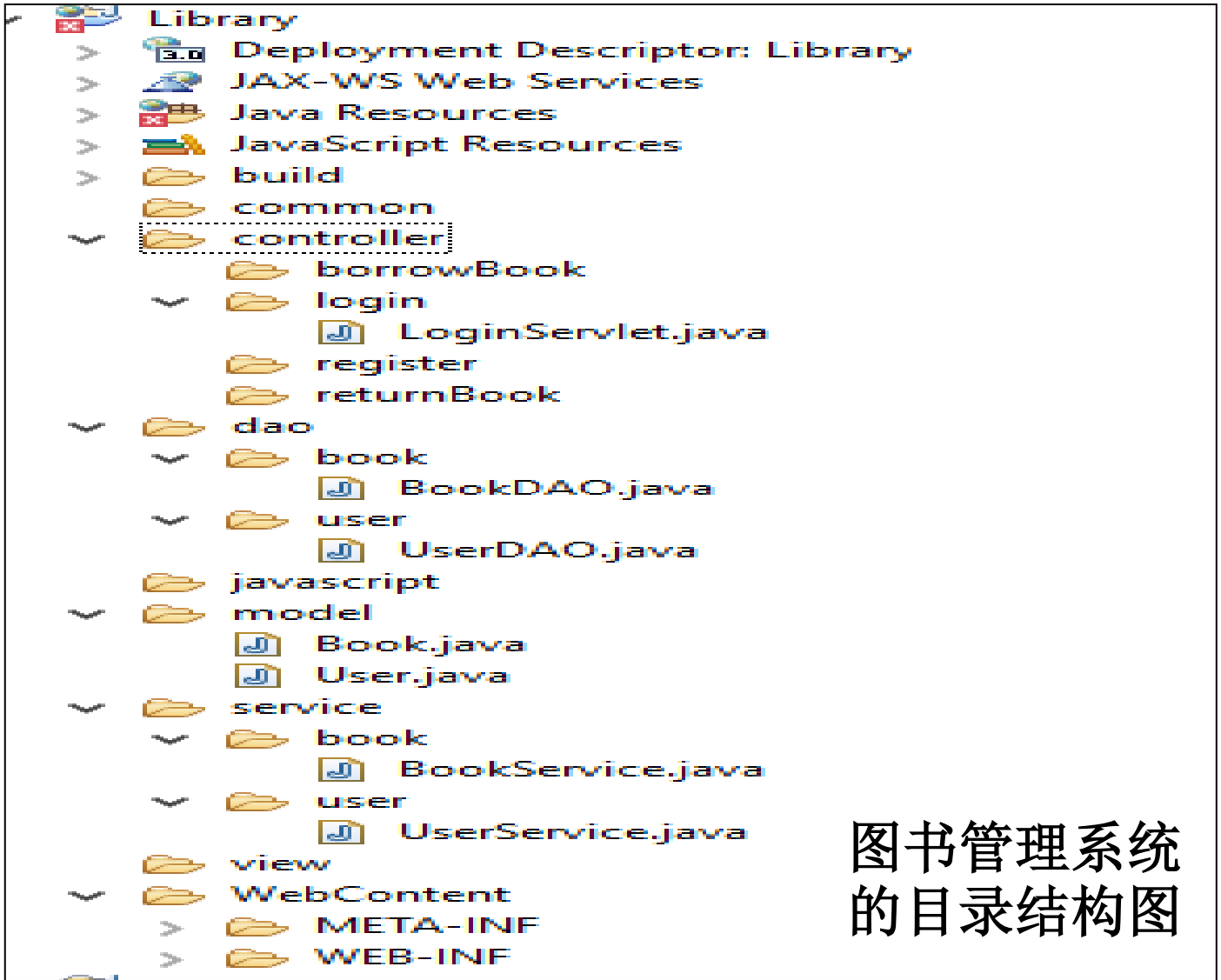
- 包对应目录（文件夹）



图书管理系统的目录结构图（MVC模式）

4.8.7 软件目录结构

- 软件目录结构



图书管理系统的目录结构图

thanks