**评估(对设计原则中的模块化、接口、信息隐蔽、增量式开发、抽象与通用性这几个方面进行描述)**

负责人：尹国泰

负责模块：管理员模块

**模块化：**

|  |  |
| --- | --- |
|  | 概况描述 |
| 耦合度 | 管理员模块的耦合度较高，因为EmployeeController、Employee、EmployeeServiceImpl等类都直接或间接地依赖于数据库访问层EmployeeDaoImpl。修改或扩展EmployeeDaoImpl可能会对其他类造成影响。  可以考虑减少管理员模块与其他模块的直接依赖，可以通过引入中介者模式或使用事件驱动的方式来解耦。 |
| 内聚度 | 管理员模块的内聚度一般，因为EmployeeController、Employee、EmployeeServiceImpl等类都处理与管理员相关的请求或业务逻辑，但不一定只处理管理员相关的内容。  可以考虑将管理员模块拆分成更小的模块，每个模块只负责单一的职责，提高内聚性。 |

**接口：**

相关说明：管理员模块中缺少面向接口编程的实践。服务层接口（IEmployeeService）和数据访问层接口（IEmployeeDao）的使用虽然提供了一定程度的松耦合，但是控制器（EmployeeController）中依然直接依赖于服务层实现类（EmployeeServiceImpl）和数据访问层实现类（EmployeeDaoImpl），导致代码可维护性不佳。

考虑引入面向接口编程的实践，定义服务层接口（IEmployeeService）和数据访问层接口（IEmployeeDao），管理员模块的其他类（EmployeeController）依赖于接口而不是具体实现类。

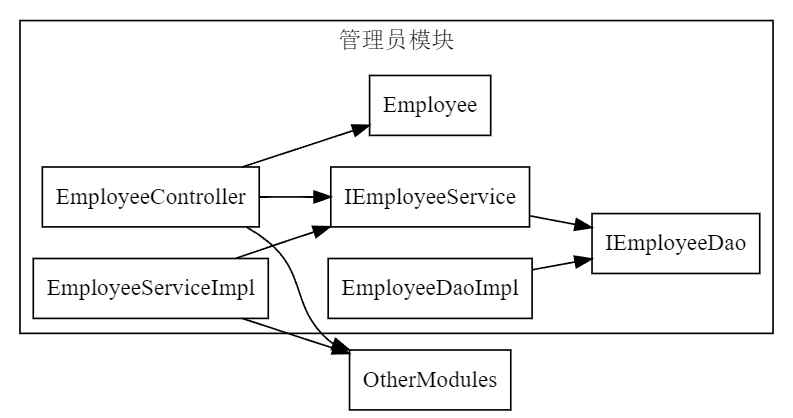
**信息隐蔽：**

描述：管理员模块中有些内容没有很好地实现信息隐蔽原则。EmployeeController、Employee、EmployeeServiceImpl等类直接操作管理员信息，没有对外界提供接口或抽象层，使得管理员信息容易被误用或滥用。

考虑重新封装管理员信息，提供良好的访问控制，只通过特定的接口或方法来访问和修改管理员信息，避免直接操作管理员信息。

**增量式开发：**

使用图：



通过模块化和接口设计，可以实现增量式开发。每个模块都提供清晰的接口定义，可以独立地进行扩展或修改，而不会对整个系统造成影响。

**抽象：**

管理员模块中缺少良好的抽象设计。管理员相关的类（EmployeeController、Employee、EmployeeServiceImpl）没有提供良好的抽象层，导致类之间的依赖性较强，可维护性和可扩展性都比较差。考虑提供良好的抽象层，定义接口或抽象类来表示管理员相关的功能和行为。各个类通过依赖注入或工厂模式获取相应的实现类，降低类之间的耦合度。

**通用性：**

EmployeeController、Employee、EmployeeServiceImpl等类只处理管理员相关的请求和业务逻辑，没有提供通用的功能和接口。如果需要增加新的通用功能，需要在系统中各个模块中都进行添加，造成重复劳动。

可以考虑在管理员模块中引入通用的功能和接口，例如权限管理、日志记录等，使其不仅仅局限于管理员的相关操作。可以通过定义通用的接口和抽象类来实现可重用性，让其他模块也能够使用这些通用功能。