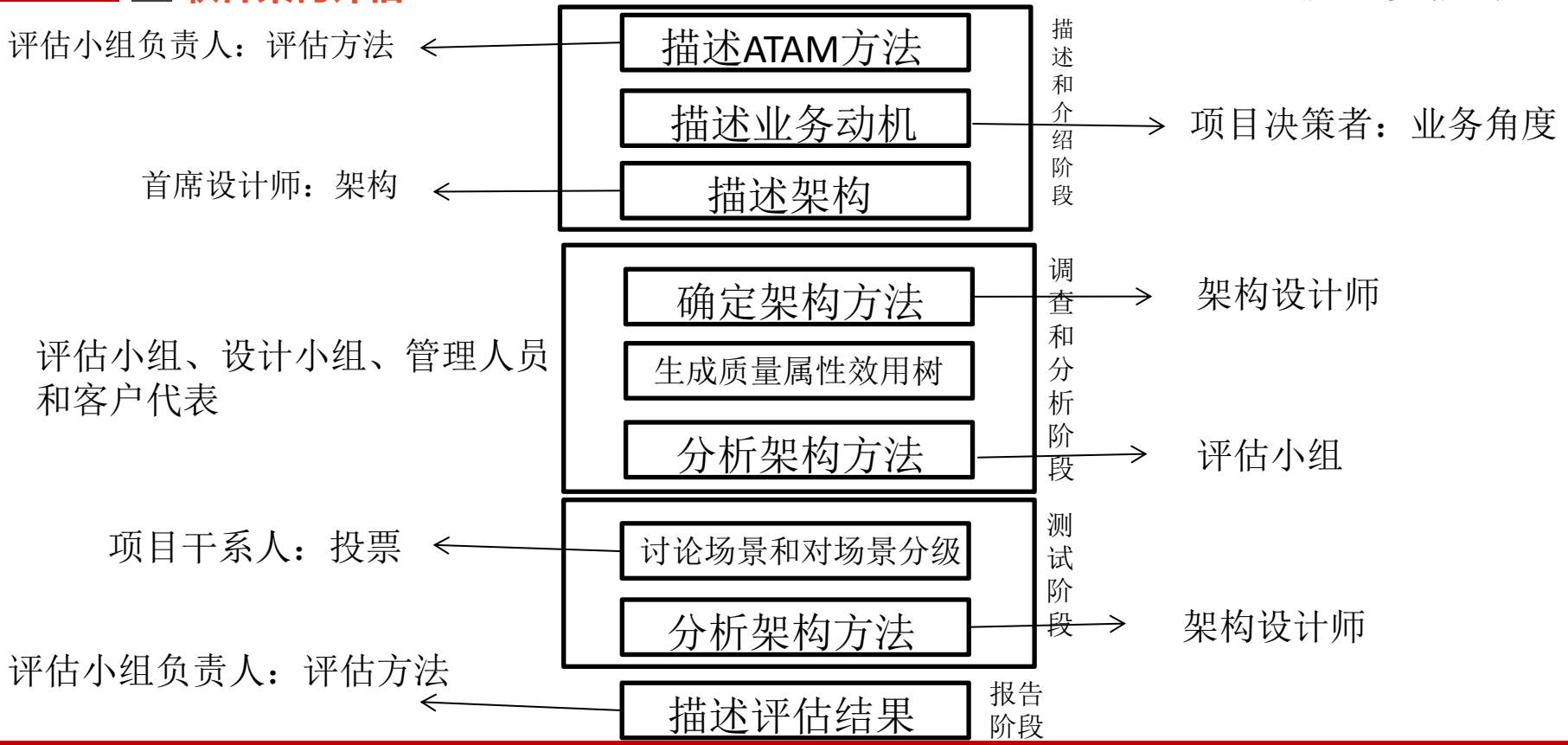
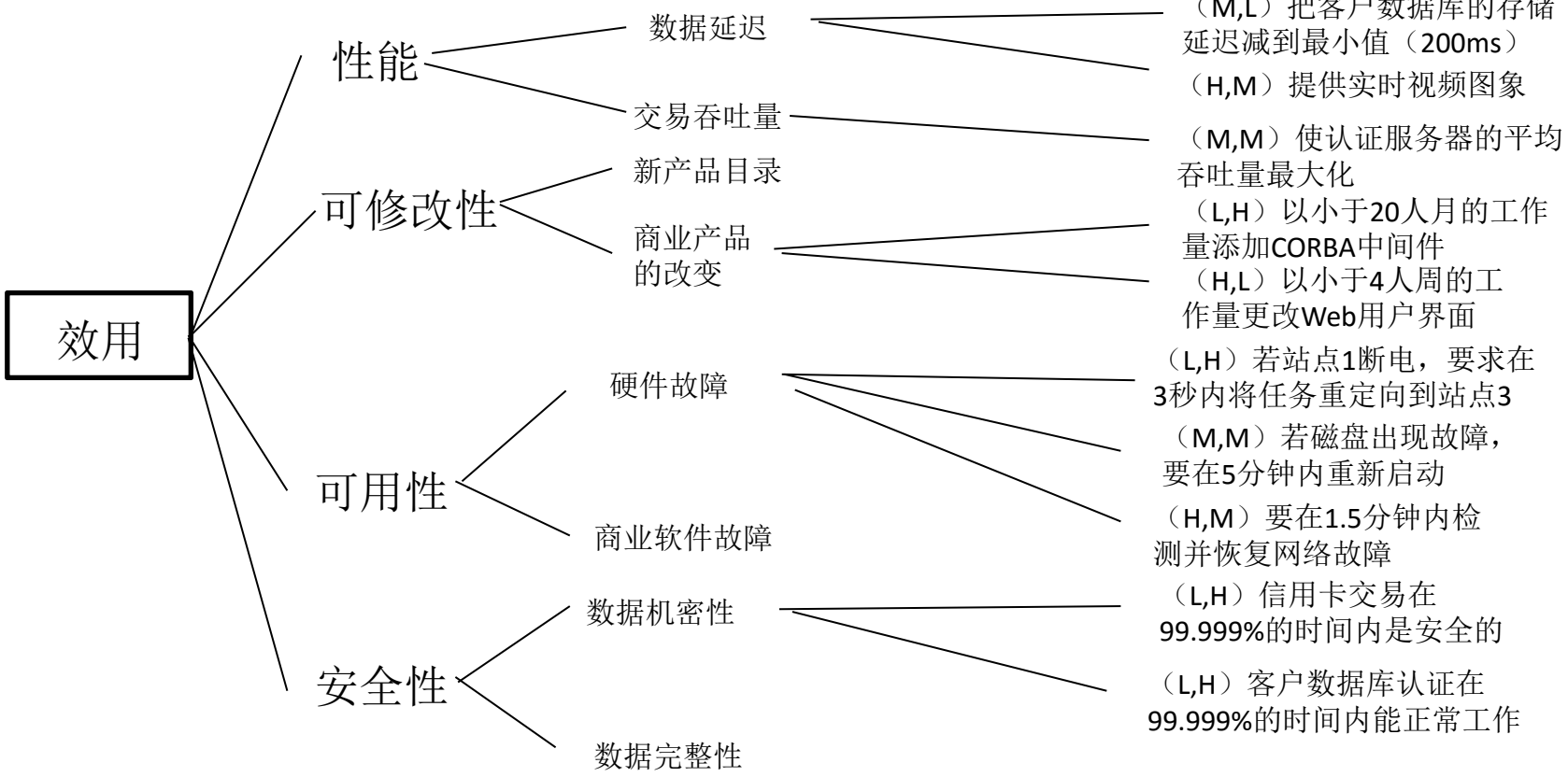
系统架构设计师

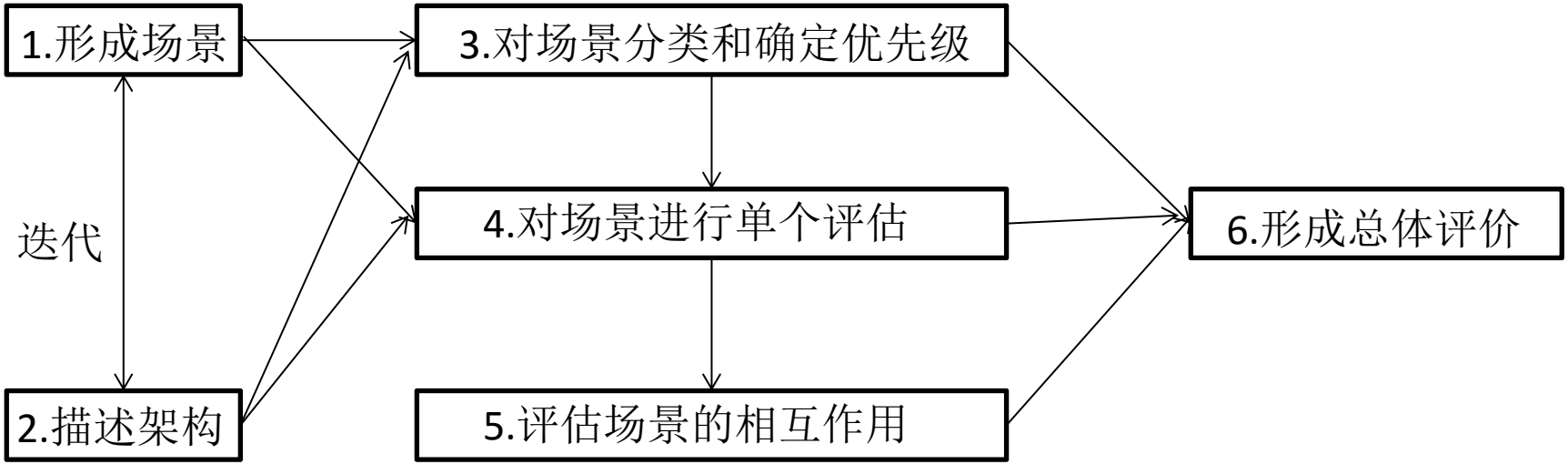
DESIGNER:王川林

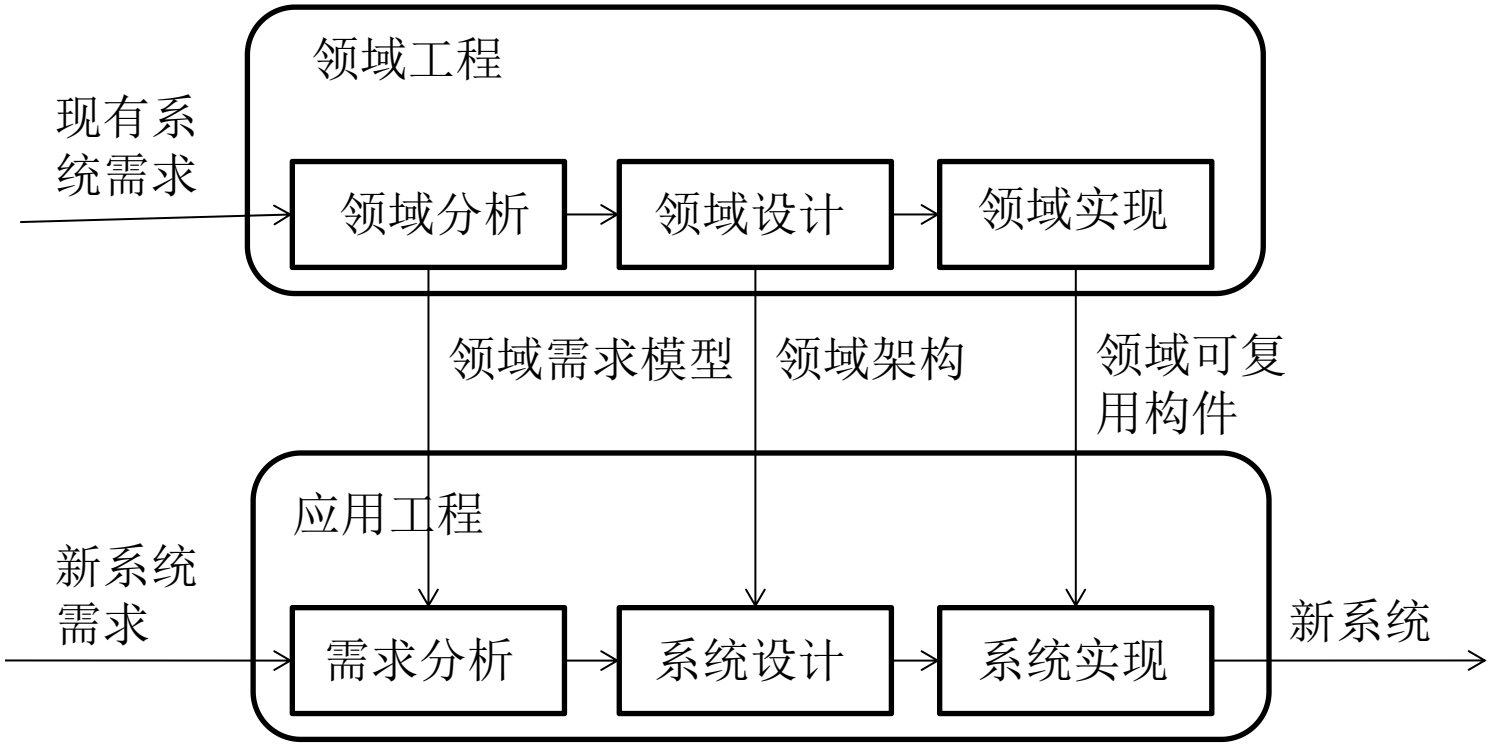
软件架构设计

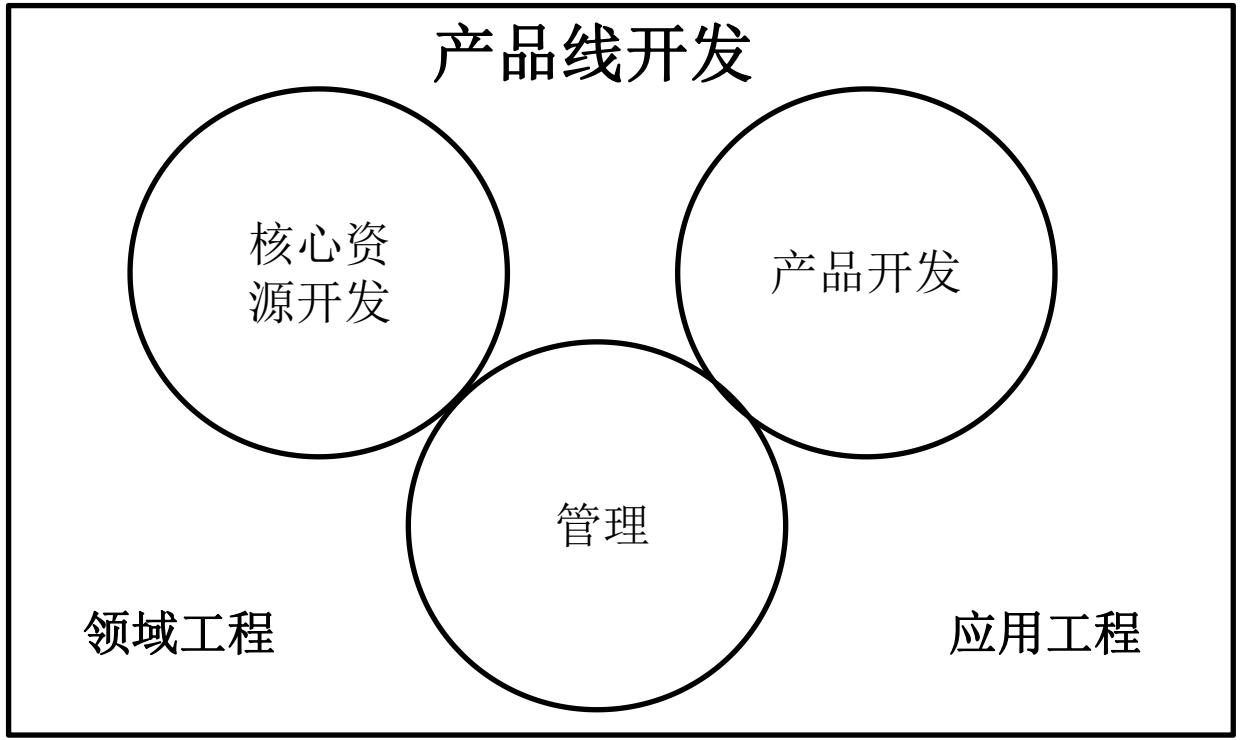

评估小组负责人：评估方法
描述ATAM方法
描述业务动机
首席设计师：架构
描述架构
描述和介绍阶段
项目决策者：业务角度
确定架构方法
生成质量属性效用树
分析架构方法
调查和分析阶段
架构设计师
评估小组
评估小组、设计小组、管理人员和客户代表
项目干系人：投票
讨论场景和对场景分级
分析架构方法
测试阶段
架构设计师
评估小组负责人：评估方法
描述评估结果
报告阶段


效用
性能
数据延迟
交易吞吐量
可修改性
新产品目录
商业产品的改变
可用性
硬件故障
商业软件故障
安全性
数据机密性
数据完整性
（M,L）把客户数据库的存储延迟减到最小值（200ms）
（H,M）提供实时视频图象
（M,M）使认证服务器的平均吞吐量最大化
（L,H）以小于20人月的工作量添加CORBA中间件
（H,L）以小于4人周的工作量更改Web用户界面
（L,H）若站点1断电，要求在3秒内将任务重定向到站点3
（M,M）若磁盘出现故障，要在5分钟内重新启动
（H,M）要在1.5分钟内检测并恢复网络故障
（L,H）信用卡交易在99.999%的时间内是安全的
（L,H）客户数据库认证在99.999%的时间内能正常工作

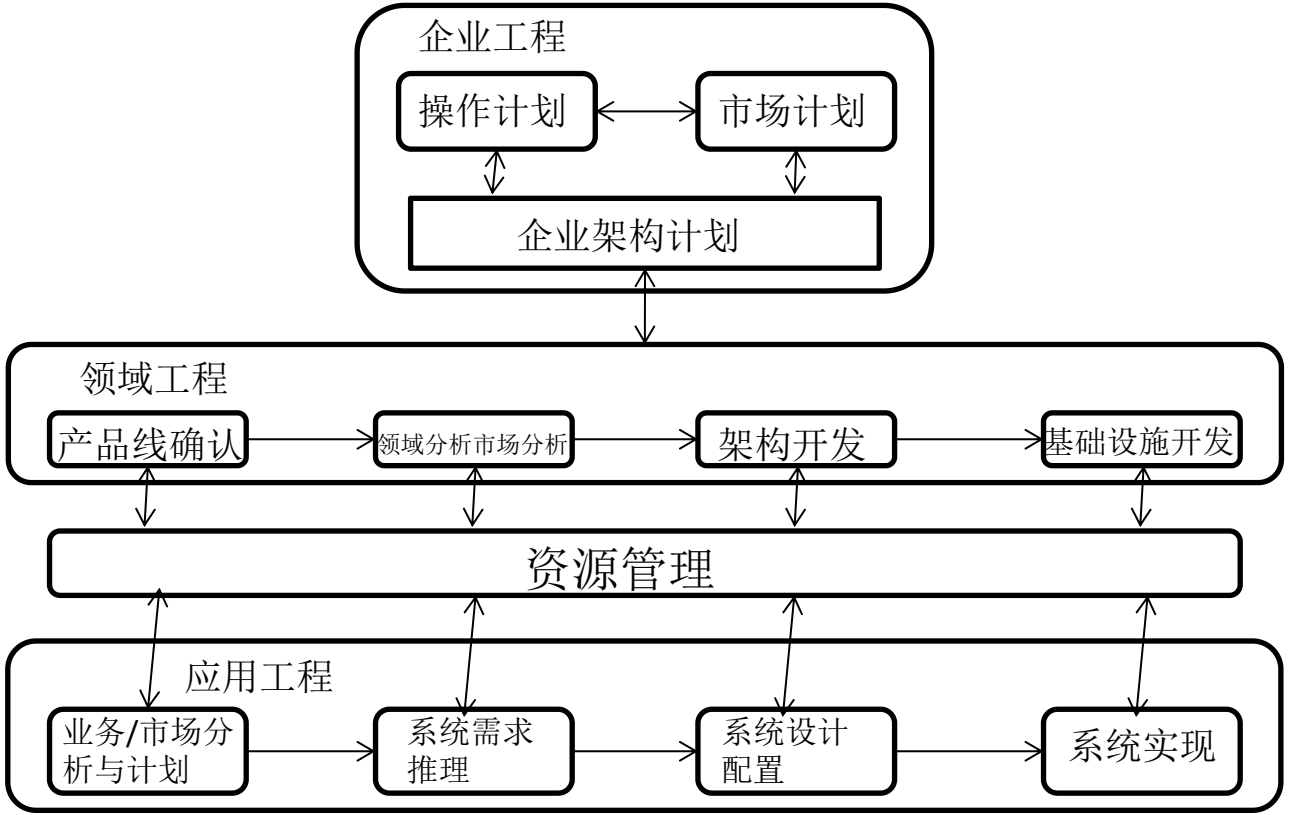
- 整理场景
- 对场景进行求精
- 确定场景的优先级
- 分配效用
- 形成“策略-场景-响应级别”的对应关系
- 确定期望的质量属性响应级别的效用
- 计算各架构策略的总收益
- 根据受成本限制影响的投资报酬率选择架构策略


1.形成场景
3.对场景分类和确定优先级
迭代
4.对场景进行单个评估
6.形成总体评价
2.描述架构
5.评估场景的相互作用


领域工程
现有系统需求
领域分析
领域设计
领域实现
领域需求模型
领域架构
领域可复用构件
应用工程
新系统需求
需求分析
系统设计
系统实现
新系统


产品线开发
核心资源开发
产品开发
管理
领域工程
应用工程

软件产品线—过程模型—三生命周期模型



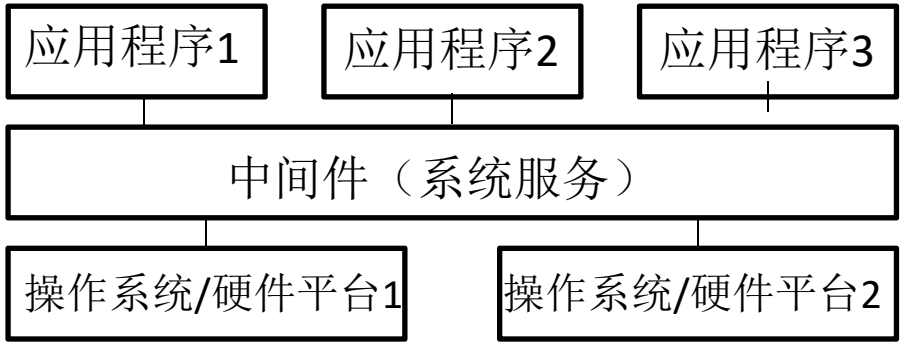
	演化方式	革命方式
基于现有产品	基于现有产品架构设计产品线的架构，经演化现有构件，开发产品线构件	核心资源的开发基于现有产品集的需求和可预测的、将来需求的超集
全新产品线	产品线核心资源随产品新成员的需求而演化	开发满足所有预期产品线成员的需求的核心资源

- 将现有产品演化为产品线
- 用软件产品线替代现有产品集
- 全新软件产品线的演化
- 全新软件产品线的开发

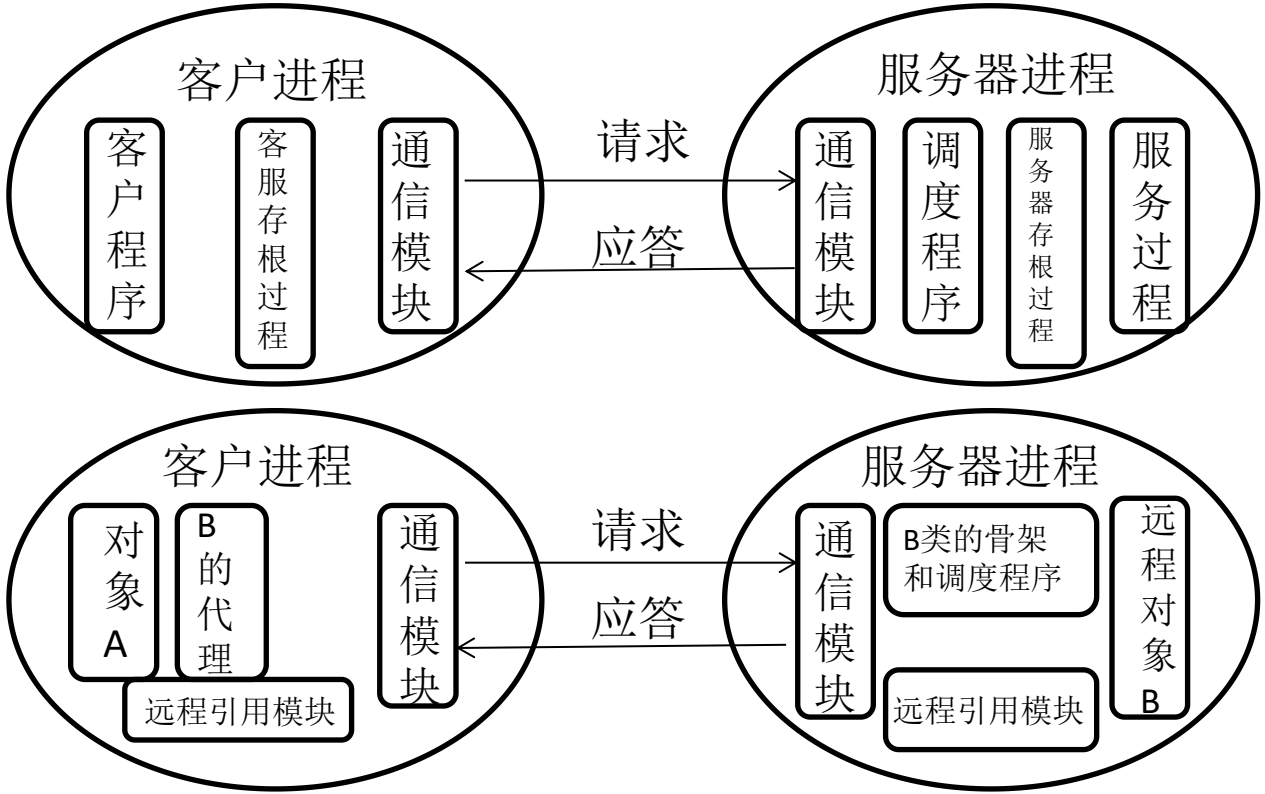
- 设立独立的核心资源小组
- 不设立独立的核心资源小组
- 动态的组织结构

- 对该领域具备长期和深厚的经验
- 一个用于构建产品的好的核心资源库
- 好的产品线架构
- 好的管理（软件资源、人员组织、过程）支持

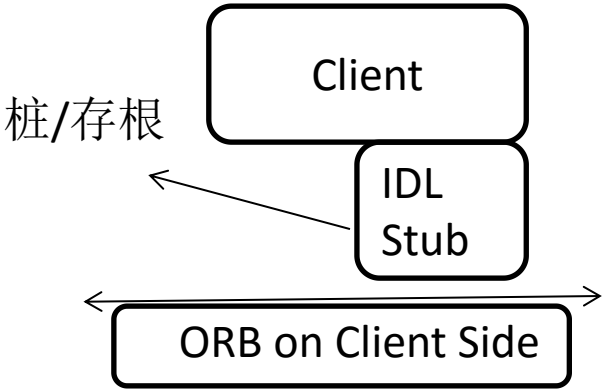
中间件是一种独立的系统软件或服务程序，可以帮助分布式应用软件在不同的技术之间共享资源

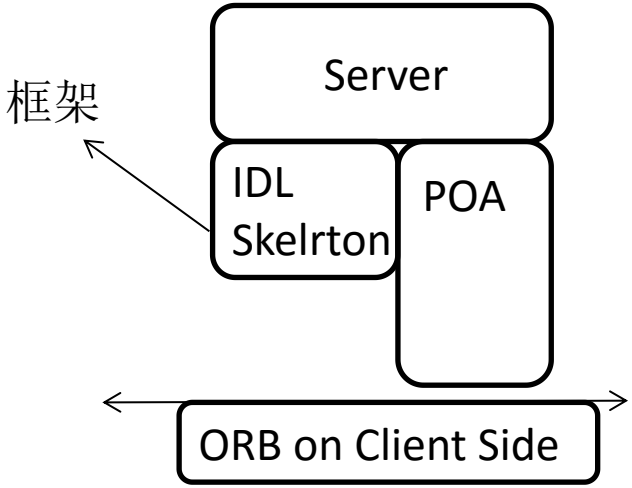


- 负责客户机与服务器之间的连接和通信，以及客户机与应用层之间的高效率通信机制
- 提供应用层不同服务之间的互操作机制，以及应用层与数据库之间的连接和控制机制
- 提供多层构架的应用开发和运行的平台，以及应用开发框架，支持模块化的应用开发
- 屏蔽硬件、操作系统、网络和数据库的差异
- 提供应用的负载均衡和高可用性、安全机制与管理功能，以及交易管理机制，保证交易的一致性
- 提供一组通用的服务去执行不同的功能，避免重复的工作和使应用之间可以协作

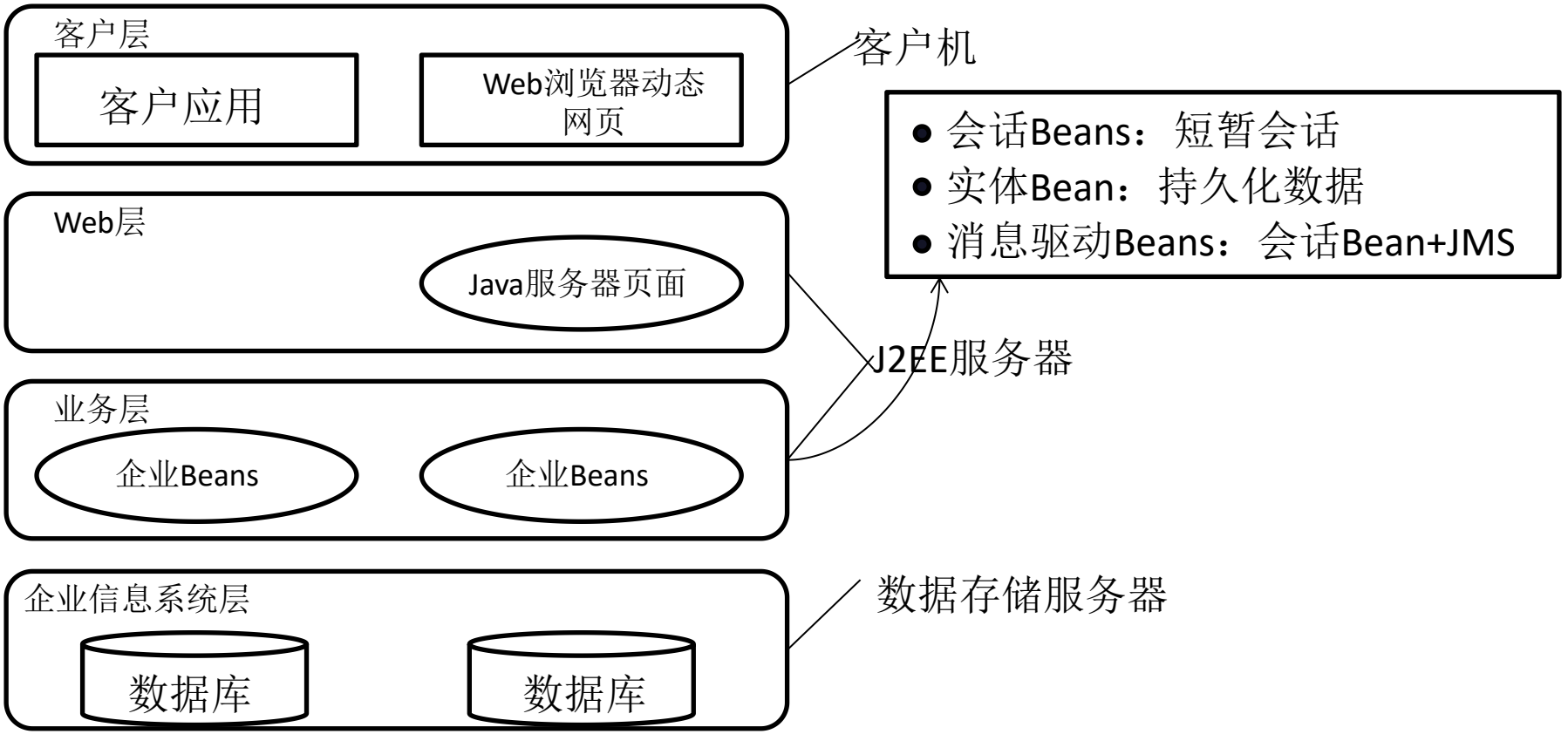


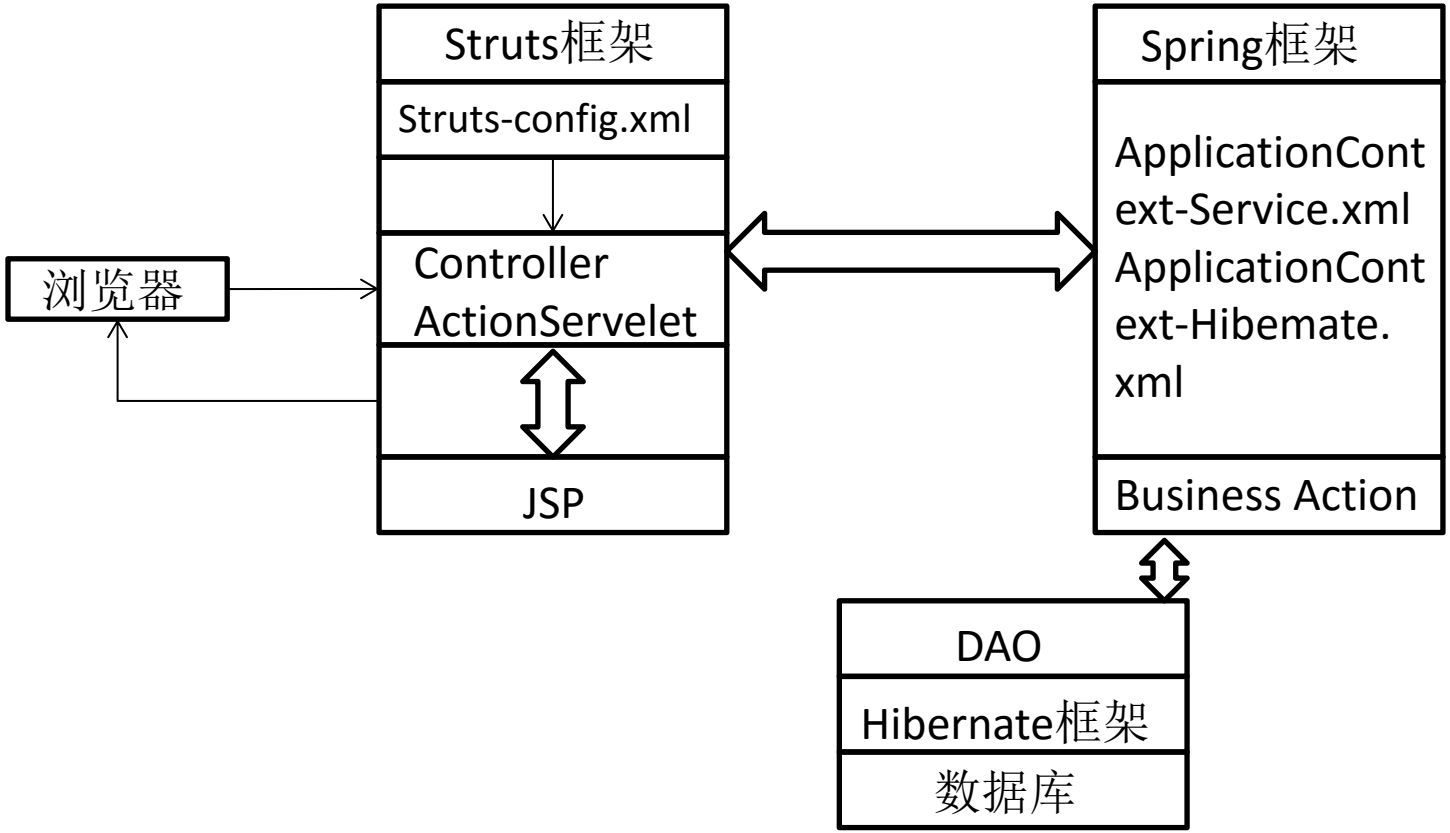
- 远程过程调用
- 对象请求代理
- 远程方法调用
- 面向消息的中间件
- 事务处理监控器


Client
桩/存根
IDL
Stub
ORB on Client Side


Server
框架
IDL
Skelrton
POA
ORB on Client Side

典型应用架构—J2EE—分布式多层应用程序

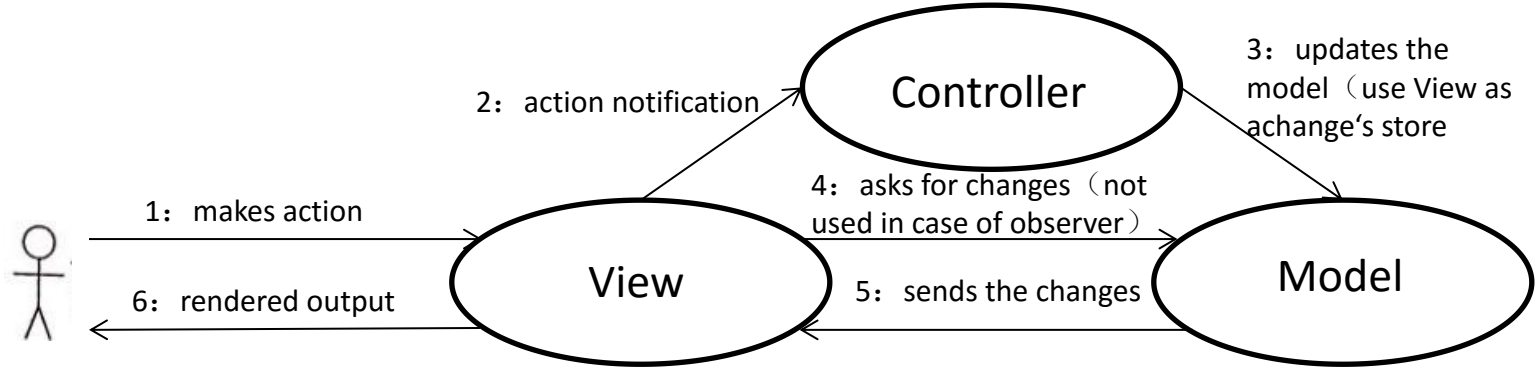



Struts框架
Struts-config.xml
Controller
ActionServelet
JSP
浏览器
Spring框架
ApplicationContext-Service.xml
ApplicationContext-Hibemate.xml
Business Action
DAO
Hibernate框架
数据库

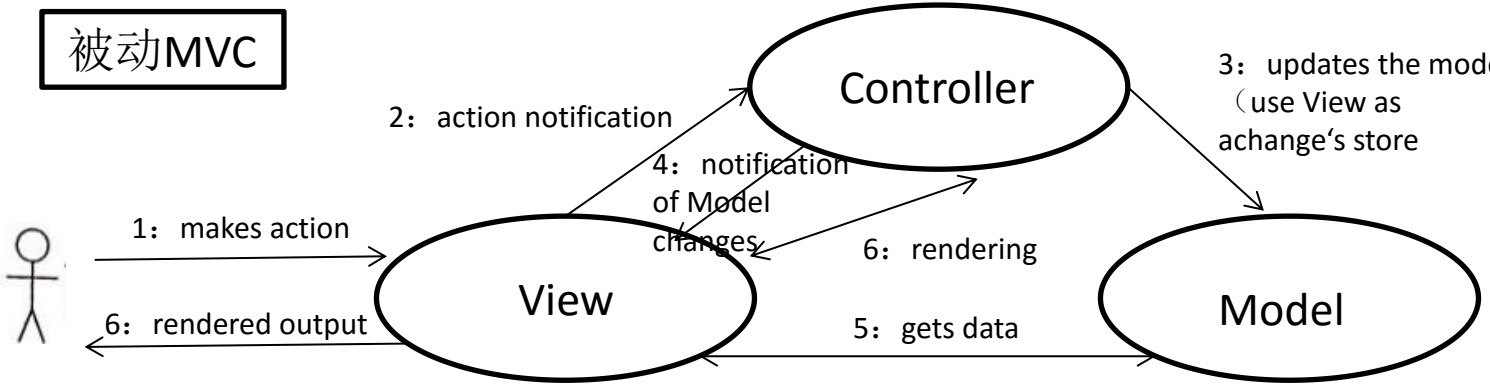
Model（模型）是应用程序中用于处理应用程序数据逻辑的部分。通常模型对象负责在数据库中存取数据
View（视图）是应用程序中处理数据显示的部分。通常视图是依据模型数据创建的。
Controller（控制器）是应用程序中处理用户交互的部分。通常控制器负责从视图读取数据，控制用户输入，并向模型发送数据

J2EE体系结构中：
视图（View）：JSP
控制（Controller）：Servlet
模型（Model）：Entity Bean、Session Bean

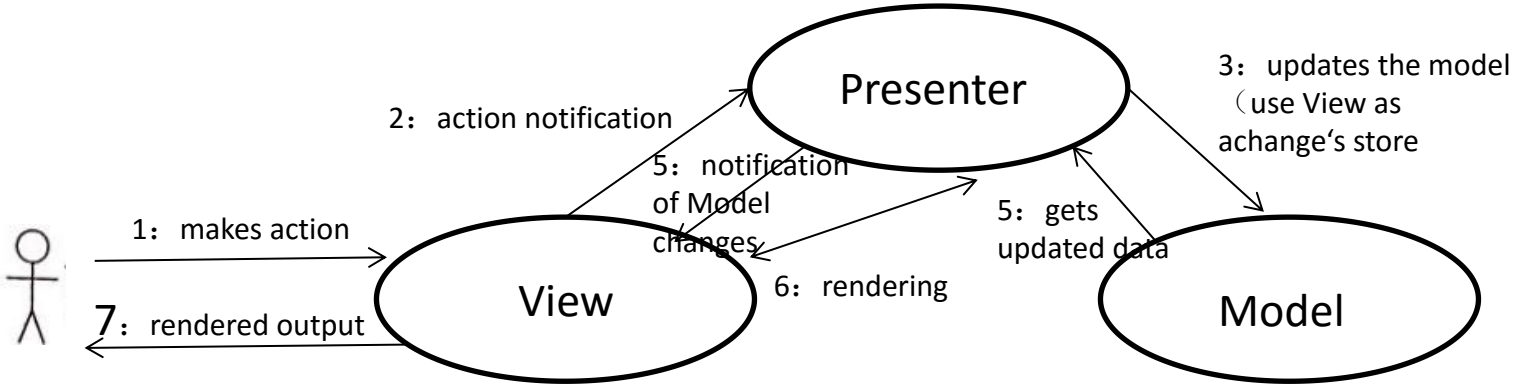
主动MVC

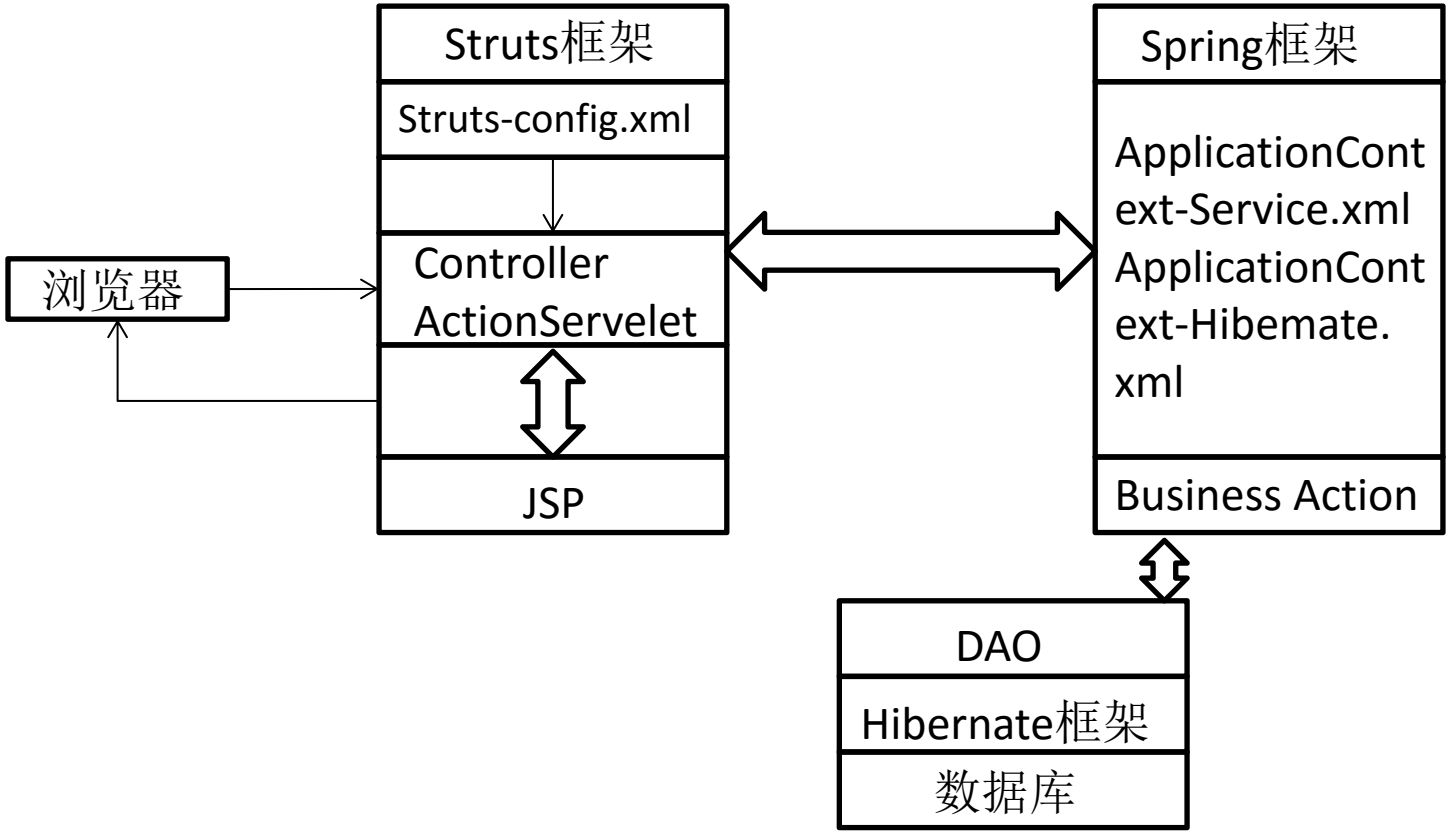

Controller
2：action notification
3：updates the model（use View as achange‘s store
1：makes action
4：asks for changes（not used in case of observer）
View
Model
6：rendered output
5：sends the changes

被动MVC


Controller
2：action notification
3：updates the model（use View as achange‘s store
4：notification of Model changes
1：makes action
6：rendering
View
Model
6：rendered output
5：gets data

- MVP是MVC的变种
- MVP实现了V与M之间的解耦（V不直接使用M，修改V不会影响M）
- MVP更好的支持单元测试（业务逻辑在P中，可以脱离V来测试这些逻辑；可以将一个P用于多个V，而不需要改变P的逻辑
- MVP中V要处理界面事件，业务逻辑在P中，MVC中界面事件由C处理

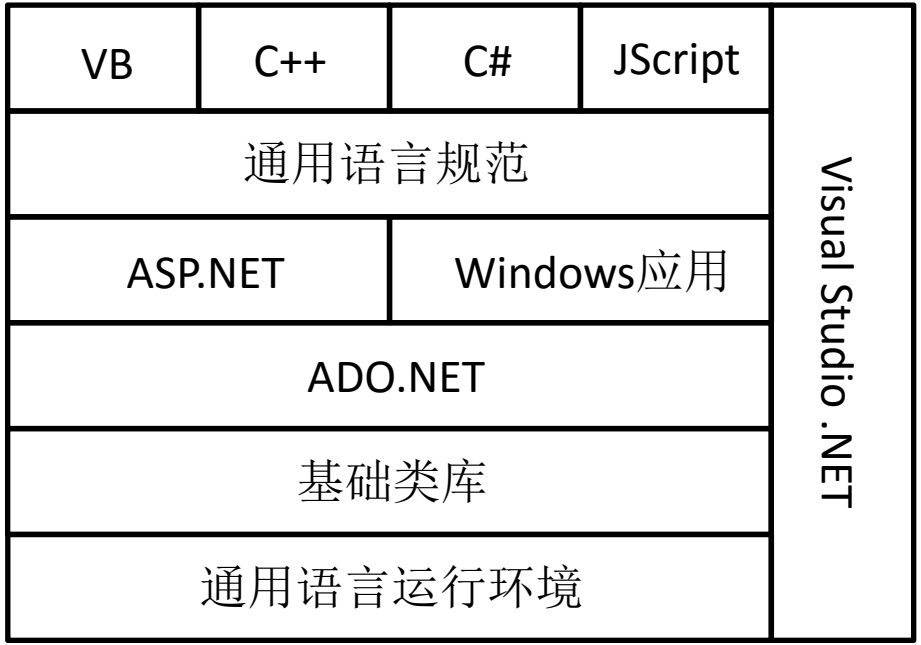


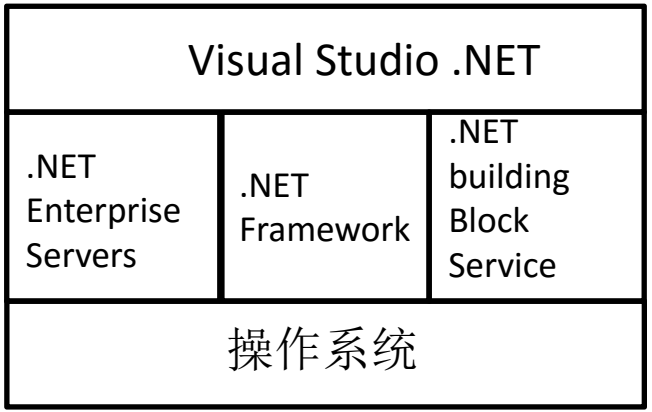

Struts框架
Struts-config.xml
Controller
ActionServelet
JSP
浏览器
Spring框架
ApplicationContext-Service.xml
ApplicationContext-Hibemate.xml
Business Action
DAO
Hibernate框架
数据库

Struts是一个基于J2EE平台的MVC框架，主要采用Serviet和JSP技术来实现。在Struts中，M由实现业务逻辑的JavaBean构成，C由ActionServiet和Action来实现，V由一组JSP文件构成

Spring通过RMI或Web Service远程访问业务逻辑，允许自由选择和组装各部分功能，还提供和其他软件集成的接口。Spring本身是个容器，管理构件的生命周期、构件的组态。依赖注入等，并可以控制构件在创建时以原型或单例模式来创建

Hibernate是一个对象关系映射框架，提供了Java对象到数据库表之间的直接映射，它对JDBC进行了非常轻量级的对象封装，使得Java程序员可以使用对象编程思维来操作数据库。在Hibernate中，ORM机制的核心是一个XML文件，该文件描述了数据库模式是怎么与一组Java类绑定在一起的


VB
C++
C#
JScript
通用语言规范
ASP.NET
Windows应用
ADO.NET
基础类库
通用语言运行环境
Visual Studio .NET


Visual Studio .NET
.NET Enterprise Servers
.NET Framework
.NET building Block Service
操作系统

- JVM与CLR
- 对多层分布式应用的支持
- 安全性
- 应用程序的部署
- 可移植性
- 外部支持

DESIGNER:王川林

软件架构设计

THANK YOU