系统架构设计师

架构知识2

姜美荣

目录

直播内容：架构知识

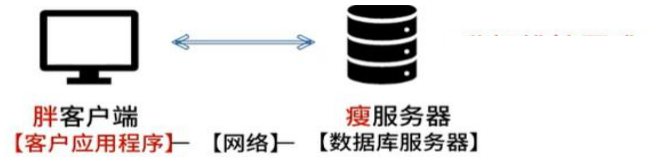
1层次型架构相关案例☆☆☆☆☆

- 表现层设计
- 业务逻辑层组件设计
- 数据访问层设计
- 物联网架构设计

2面向服务的架构相关案例☆☆☆☆☆

- SOA相关概念
- SOA实现模式
- SOA关键技术
- SOA案例

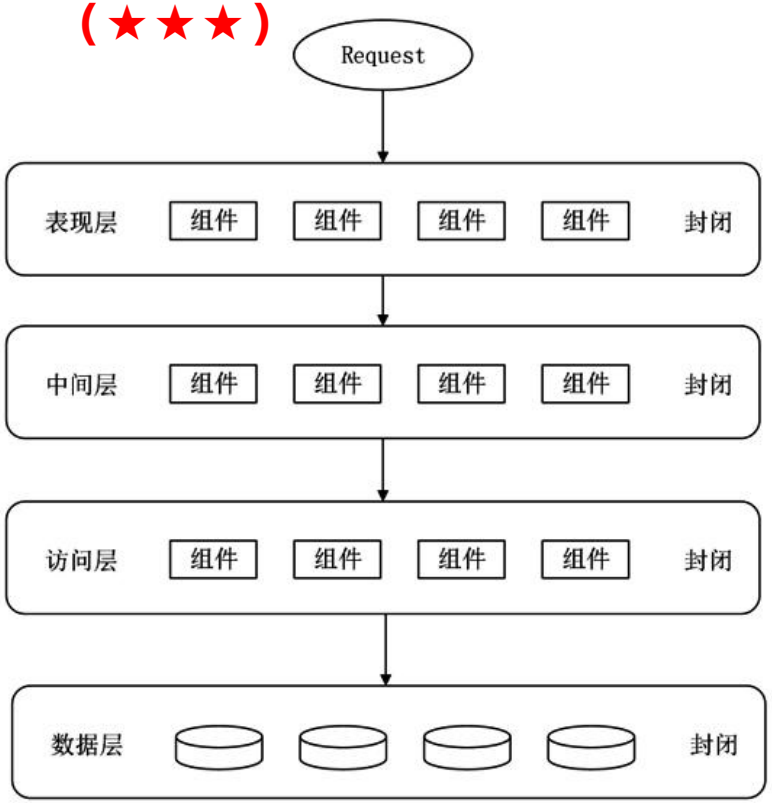
层次式体系结构设计

层次架构	结构	特点
两层C/S	 胖客户端（业务逻辑） 服务器（数据处理）	升级维护困难
三层C/S	 瘦客户端（界面输入输出） 应用服务器（业务逻辑） 服务器（数据库）	业务逻辑变化不再需要 更新客户端
B/S架构	 浏览器 WEB应用服务器（业务逻辑） 服务器（数据库）	完全不需要客户端 升级维护容易
RIA富联网技术	 客户端（Ajax、Flex、 小程序、HTML5）等	临时下载客户端 反应速度快、易于传播、 交互性强

层次式体系结构设计

层次式体系结构设计将系统组成一个层次结构，每一层为上层服务，并作为下层客户。内部的层接口只对相邻的层可见，**每一层最多只影响两层，只要给相邻层提供相同的接口**，允许每层用不同的方法实现，为软件重用提供了强大的支持。

层次式架构也称N层架构模式，分成**表现层(展示层)、中间层(业务层)、数据访问层(持久层) 和数据层。（★★★）**



MVC---MVP---MVVM

ORM

但设计时要注意以下两点：（★）

(1)要注意**污水池反模式**

污水池反模式，就是请求流简单地穿过几个层，每层里面基本没有做任何业务逻辑，或者做了很少的业务逻辑。如果请求超过20%，则应该考虑让一些层变成开放的。

(2)需要考虑的是**分层架构可能会让你的应用变得庞大**。

即使你的表现层和中间层可以独立发布，但它的确会带来一些潜在的问题，比如：分布模式复杂、健壮性下降、可靠性和性能的不足，以及代码规模的膨胀等。

分层架构的一个特性就是**关注分离**。（★）

MVC架构风格

MVC 模式

MVC 强制地把一个应用的输入、处理、输出流程按照**视图、控制、模型**的方式进行分离，形成了控制器、模型、视图三个核心模块。

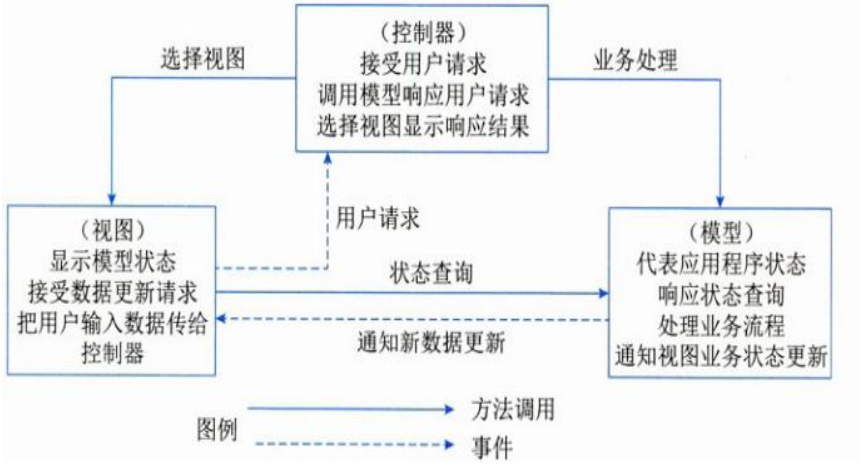
(1)视图 (View)：用户看到并与之交互的界面。视图向用户显示相关的数据，并接收用户输入的数据，但是它并不进行任何实际的业务处理。

(2)控制器(Controller)

该部分是用户界面与模型的接口。解释来自于视图的输入，将其解释成为系统能够理解的对象，同时它也识别用户动作，并将其解释为对模型特定方法的调用；另一方面，它处理来自于模型的事件和模型逻辑执行的结果，调用适当的视图为用户提供反馈。

(3)模型(Model)

应用程序的主体部分。模型表示**业务数据和业务逻辑**。**一个模型能为多个视图提供数据**。同一个模型可以被多个视图重用，提高应用的可重用性。



使用MVC 模式来设计表现层，可以有以下的优点。

1、允许多种用户界面的扩展
2、易于维护
3、功能强大的用户界面。
4、增加应用的可拓展性、强壮型、灵活性。

J2EE体系结构中：

1、视图：JSP
2、控制：Servlet
3、模型：Entity Bean、Session Bean

MVP架构风格

MVP模式

MVP模式中Model提供数据，**View负责显示，Presenter/Controller负责逻辑的处理**。（★★）

MVP与MVC 有一些显著的区别，MVC模式中允许 View 和 Model直接进行"交流"，在MVP 模式中是不允许的。在MVP 中 View 并不直接使用 Model，它们之间的通信是通过 Presenter (MVC中的Controller)来进行的，所有的交互都发生在 Presenter 内部，而在MVC中View会直接从Model 中读取数据而不是通过Controller。

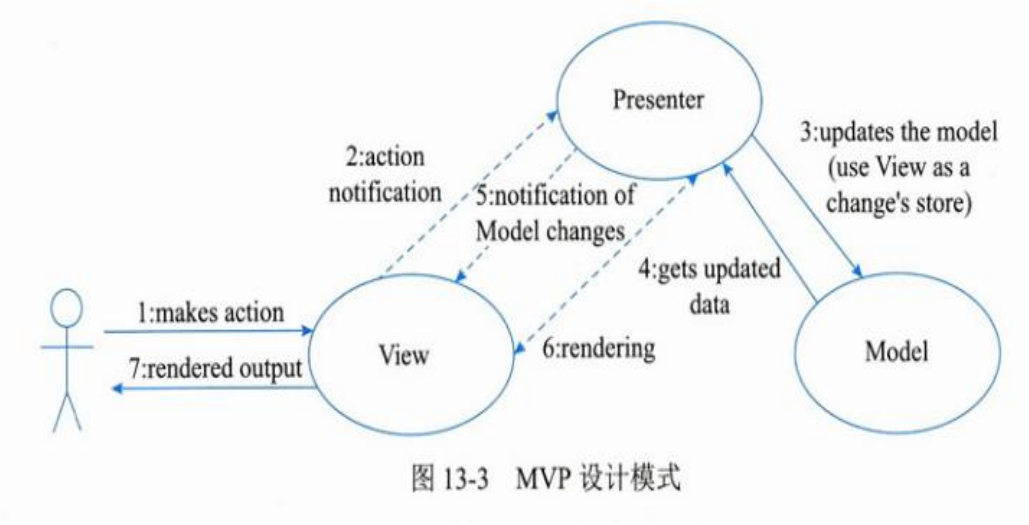


图 13-3　MVP 设计模式

使用MVP模式来设计表现层，可以有以下的优点。

(1 ）**模型与视图完全分离**，可以修改视图而不影响模型。

(2 ）可以更高效地使用模型，**因为所有的交互都发生在一个地方--Presenter 内部**。

(3 ）**可以将一个 Presenter 用于多个视图**，而不需要改变 Presenter 的逻辑。这个特性非常的有用，因为视图的变化总是比模型的变化频繁。

(4 ）如果把逻辑放在 Presenter 中，就可以脱离用户接口来测试这些逻辑（单元测试）。

目前，MVP 模式被更多地用 在 Android 开发当中。

MVVM架构风格

MVVM模式（★★）

MVVM模式正是为**解决MVP中UI种类变多**，接口也会不断增加的问题而提出的。MVVM模式全称是模型-视图-视图模型 (Model-View-ViewModel)，它和MVC、MVP类似，主要目的都是为了实现视图和模型的分离，不同的是 MVVM 中，View 与Model 的交互通过 ViewModel来实现。ViewModel是MVVM的核心，它通过**DataBinding**实现 View 与Model之间的**双向绑定**，其内容包括数据状态处理、数据绑定及数据转换。例如，View 中某处的状态和Model中某部分数据绑定在一起，这部分数据一旦变化将会反映到 View 层。而这个机制通过 ViewModel来实现。

ViewModel，即视图模型，是一个专门用于数据转换的控制器，它可以把对象信息转换为视图信息，将命令从视图携带到对象。View和ViewModel之间使用DataBinding及其事件进行通信。View的用户接口事件仍然由 View自身处理，并把相关事性映射到ViewModel，以实现View中的对象与视图模型内容的同步，且可通过**双向数据**绑定进行更新。

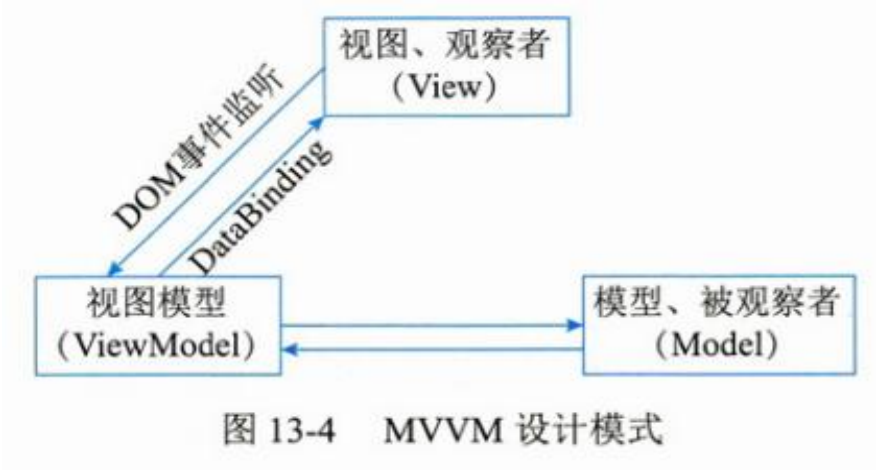


图 13-4　MVVM 设计模式

业务逻辑层框架

业务逻辑层框架。**业务框架位于系统架构的中间层**，是实现系统功能的核心组件。采用**容器**的形式，便于系统功能的开发、代码重用和管理。在业务容器中，业务逻辑是按照 Domain Model-Service-Control 思想来实现的。

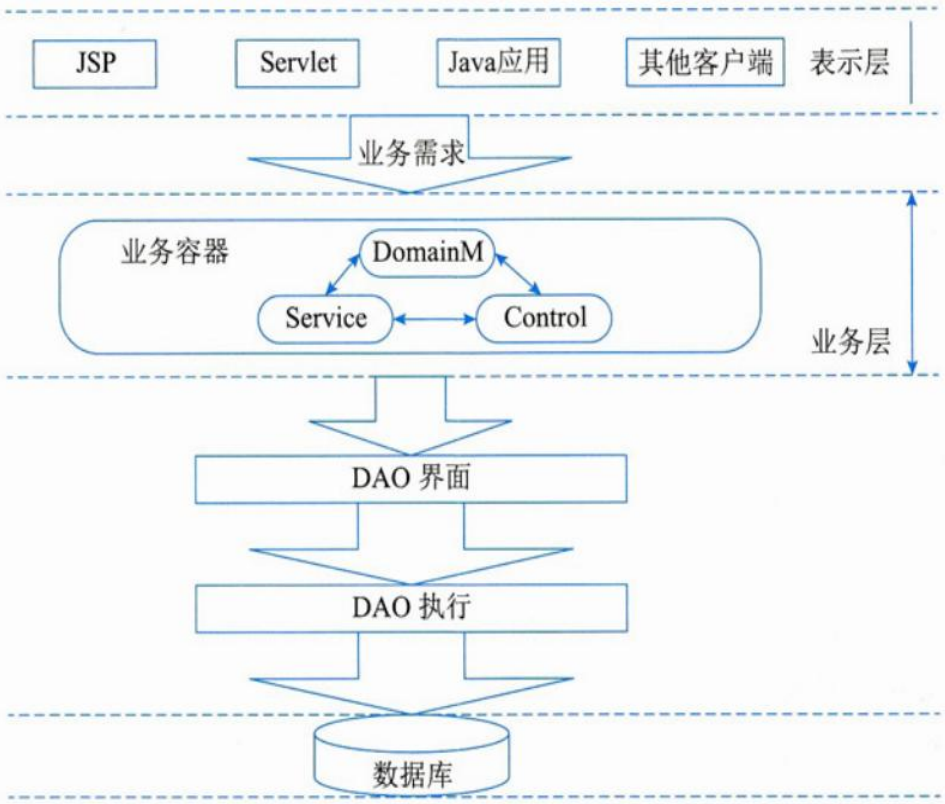


图 13-10　业务框架在整个系统架构中的位置

其中：

业务层采用业务容器的方式存在于整个系统当中，采用此方式可以大大降低业务层和相邻各层的耦合，表示层代码只需要将业务参数传递给业务容器，而不需要业务层多余的干预。如此一来，可以有效地防止业务层代码渗透到表示层。

在业务容器中，业务逻辑是按照**Domain Model–Service–Control**思想来实现的。（★★★）

(1)Domain Model是领域层业务对象，它仅仅包含业务相关的属性。

(2)**Service**是业务过程实现的组成部分，是应用程序的不同功能单元，通过在这些服务之间定义良好的接口和契约联系起来。

(3)**Control**服务控制器，是服务之间的纽带，不同服务之间的切换就是通过它来实现的。

数据访问层设计

5种数据访问模式

模式	特点
在线访问	每个数据库操作都会通过数据库连接不断地与后台的数据源进行交互
Data Access Object	底层数据访问操作 与 高层业务逻辑 分离
Data Transfer Object	跨不同的进程或是网络的边界来传输数据
离线数据模式	从数据源获取数据后，存放到本地处理
对象 / 关系映射	应用系统中的对象与数据库中的数据表形成映射关系

数据访问层设计

对象/关系映射(O/R Mapping)（★★）

对象/关系映射的基本思想来源于这样一种现实：**多数应用的数据都是存在关系型数据库中**，而这些应用程序中的数据在开发或是运行时是以对象的形式组织起来的，那么**对象/关系映射**就提供了这样一种工具或平台，能够将应用程序中的数据转换成关系型数据库中的记录，或是将关系数据库中的记录转换成应用程序中的代码便于操作的对象。

面向对象	关系数据库
类(class)	数据库的表(table)
对象(object)	记录(record,行数据)
对象的属性(attribute)	字段(field)

技术实现	Hibernate	MyBatis(iBatis)
简单对比	强大，复杂，间接，SQL无关	小巧，简单，直接，SQL相关
可移植性	好(不关心具体数据库)	差(根据数据库SQL编写)
复杂多表关联	不支持	支持

物联网架构

物联网架构

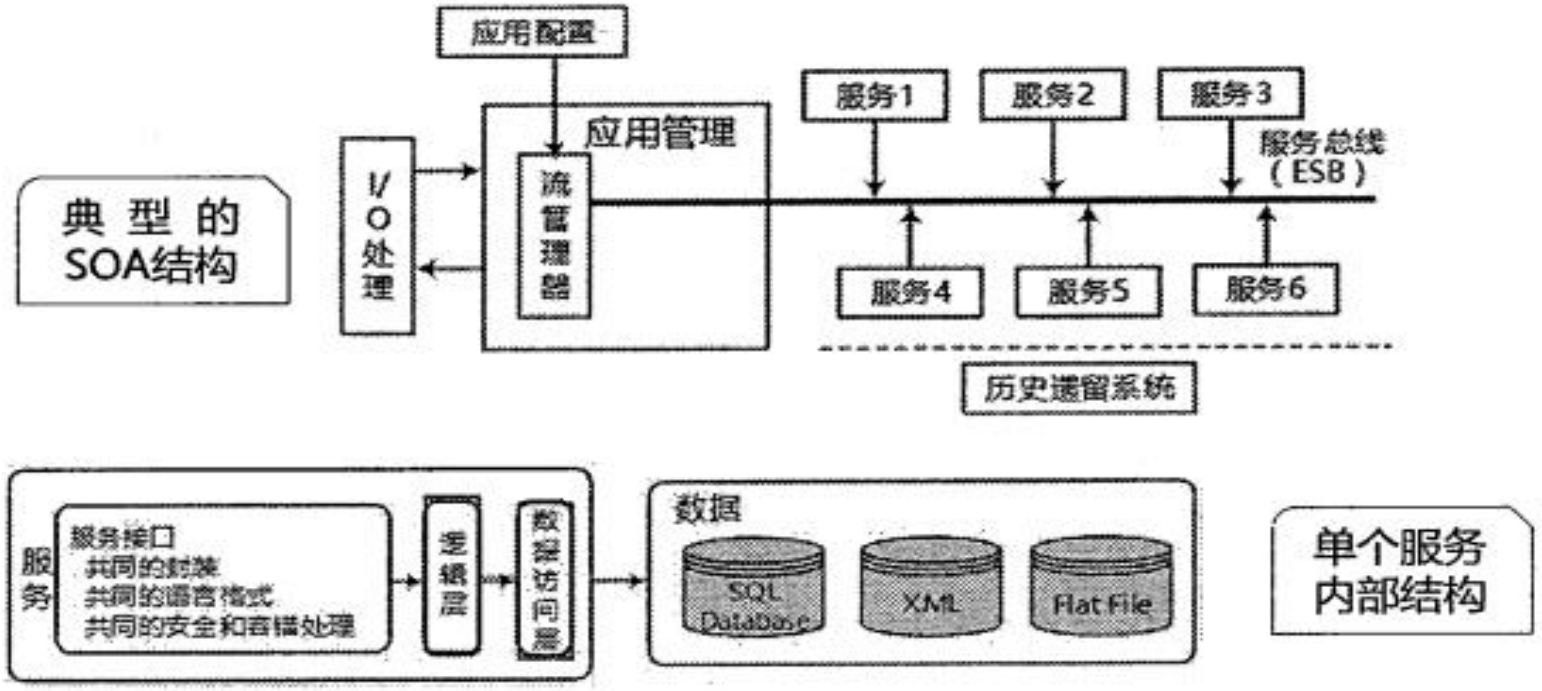
应用层	解决信息处理和人机交互问题 应用服务 智能终端
平台层	操作系统软件开发设备管理平台连接管理平台
网络层	传递信息和处理信息 网络 通信标准/协议
感知层	解决数据获取问题 传感器 芯片 通信模组

SOA 的相关概念

从软件的基本原理定义，可以认为SOA是一个组件模型，它将**应用程序的不同功能单元(称为服务)**通过这些服务之间定义良好的接口和契约联系起来。接口是采用中立的方式进行定义的，它应该独立于实现服务的硬件平台、操作系统和编程语言。

SOA是一种**粗粒度、松耦合**服务架构，**标准化接口**接口进行通信，不涉及底层编程接口和通信模型。

业务流程是指为了实现某种业务目的行为所进行的流程或一系列动作。



SOA 的实现模式-WEB Service

一、WEB Service：服务提供者、服务注册中心（中介，提供交易平台，可有可无）、服务请求者。

服务描述

服务注册中心是连接服务提供者和服务请求者的纽带，服务提供者在此发布他们的服务描述，而服务请求者在服务注册中心查

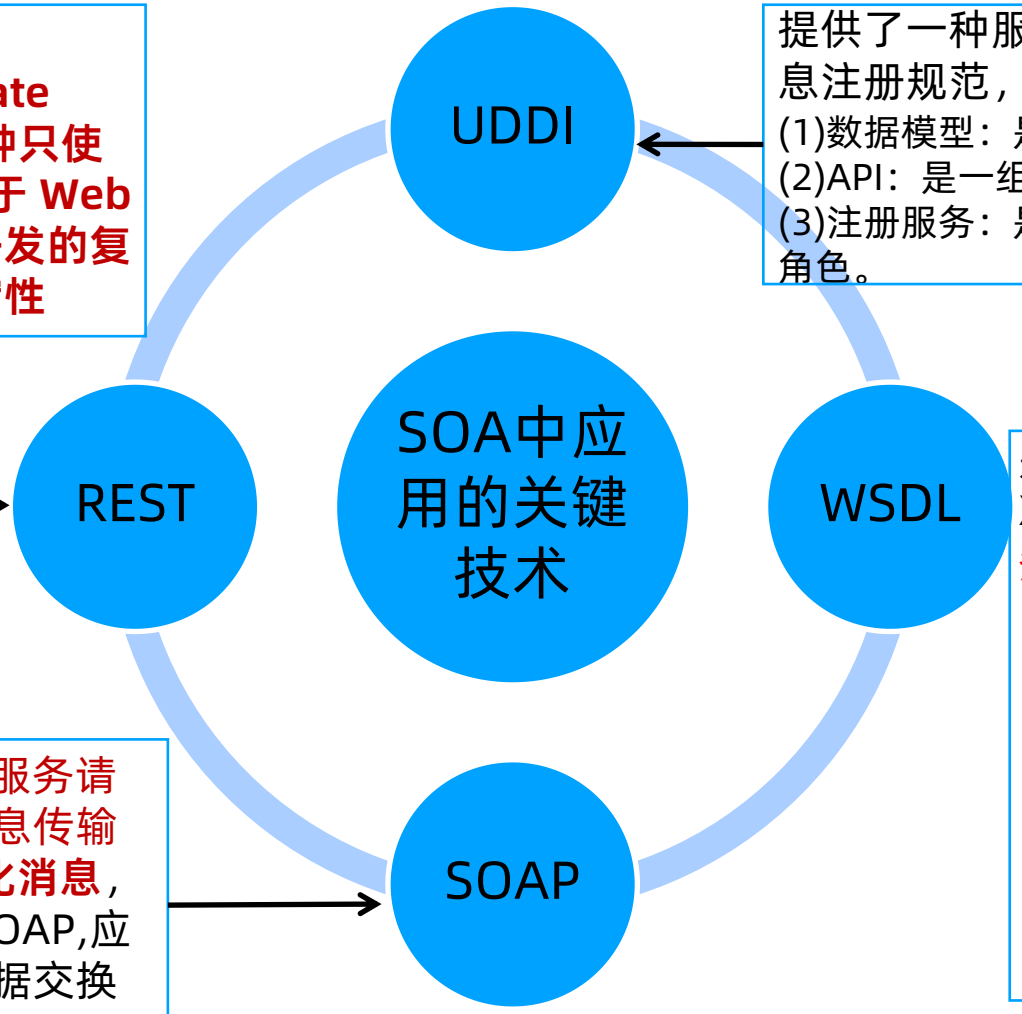
服务提供者是服务的服务所有者，该角色负责定义并实现服务，使用WSDL对服务进行详细、准确、规范的描述，并将该描述发布到服务注册中心，供服务请求者查找并绑定使用

查找、绑定并调用服务，或与服务进行交互的应用程序

WSDL就是WebService接口对应的WSDL文件，该文件通过xml格式说明如何调用，可以看作 WebService的接口文档(使用说明书)。

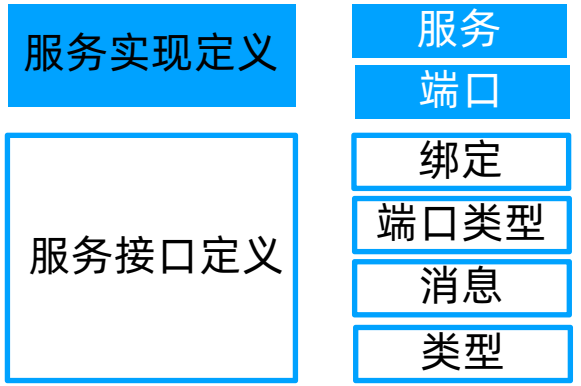
面向服务的架构风格-SOA关键技术

表述性状态转移（Representational State Transfer，REST）是一种只使用 HTTP 和 XML 进行基于 Web 通信的技术，可以降低开发的复杂性，提高系统的可伸缩性



提供了一种服务发布、查找和定位的方法，是服务的信息注册规范，以便被需要该服务的用户发现和使用它
(1)数据模型：是一个用于描述业务组织和服务的XML Schema。
(2)API：是一组用于查找或发布UDDI数据的方法，基于SOAP。
(3)注册服务：是SOA中的一种基础设施，对应着服务注册中心的角色。

是对服务进行描述的语言，它有一套基于XML的语法定义。WSDL描述的重点是服务，它包含服务实现定义和服务接口定义



简单对象访问协议：定义了服务请求者和服务提供者之间的消息传输规范。SOAP**用XML来格式化消息**，**用HTTP来承载消息**。通过SOAP,应用程序可以在网络中进行数据交换和远程过程调用

表述性状态转移

REST(Representational State Transfer,表述性状态转移)是一种通常使用HTTP和XML进行基于Web通信的技术，可以降低开发的复杂性，提高系统的可伸缩性。

REST的5个原则（★★★）
(1)网络上的所有事物都被抽象为资源。
(2)每个资源对应一个唯一的资源标识。
(3)通过通用的连接件接口对资源进行操作。
(4)对资源的各种操作不会改变资源标识。
(5)所有的操作都是无状态的。

REST API

命令	用途
GET	从指定的资源请求数据，只进行**数据检索**，不进行其他操作
POST	将数据发送到服务器进行**创建**，通常用于上传文件或提交表单
PUT	更新目标资源的所有当前表示，使用上传的内容进行**替换**
DELETE	**删除**指定的资源
HEAD	与GET方法类似，但**只传输状态行和头部信息**
PATCH	对资源进行**部分修改**

SOA 的实现模式-企业服务总线模式

二、企业服务总线模式

企业服务总线 (ESB)，提供一种标准的软件底层架构，各种程序组件能够以服务单元的方式“插入”到该平台上运行，并且组件之间能够以标准的消息通信方式来进行交互。

ESB 本质上是以中间件形式支持服务单元之间进行交互的软件平台。各种程序组件以标准的方式连接在该“总线”上，并且组件之间能够以格式统一的消息通信的方式来进行交互。

服务总线事实上实现了组件和应用系统的位置透明和协议透明。技术人员可以通过开发符合 ESB标准的组件(适配器)将外部应用连接至服务总线，实现与其他系统的互操作。同时，ESB 以中间件的方式，提供服务容错、负载均衡、QoS 保障和可管理功能。

ESB的核心功能如下：

(1)提供**位置透明的消息路由和寻址服务**。

(2)提供**服务注册和命名的管理功能**。

(3)支持**多种消息传递范型**(如请求/响应、发布/订阅等

(4)支持**多种可以广泛使用的传输协议**。

(5)支持**多种数据格式及其相互转换**。

(6)提供**日志和监控功能**。

SOA 的设计模式-微服务模式

三、微服务模式

优点	说明	缺点
复杂应用解耦	小服务(且专注于做一件事情) 化整为零，易于小团队开发	微服务架构面临的问题与挑战 （1）服务发现与服务调用链跟踪变得困难 （2）很难实现传统数据库的强一致性，转而追求最终一致性 （3）测试复杂性（服务间依赖测试） （4）运维更加复杂 （5）并非所有系统都可拆分为微服务 （6）微服务的部署也更加困难
独立	独立开发 独立测试及独立部署(简单部署) 独立运行(每个服务独立在其独立进程中)	
技术选型灵活	支持异构(如：每个服务使用不同数据库)	
容错	故障被隔离在单个服务中，通过重试、平稳退化等机制实现应用层容错	【微服务与SOA的区别】 微服务比SOA更精细，可以独立进程方式存在。 微服务接口更通用化，如用HTTP 、RESTful,各种终端都可调用，语言无关，平台无关。 更倾向于分布式部署，互联网场景更适合。
松耦合，易扩展	可根据需求独立扩展	

SOA 的设计模式-微服务模式

微服务架构模式方案。（★★）

主要包括：

（1）**聚合器微服务**：聚合器充当流程指挥者，调用多个微服务实现系统应用程序所需功能。

（2）**链式微服务**：客户端或服务在收到请求后，会发生多个服务间的嵌套递归调用，返回经过合并处理的响应。

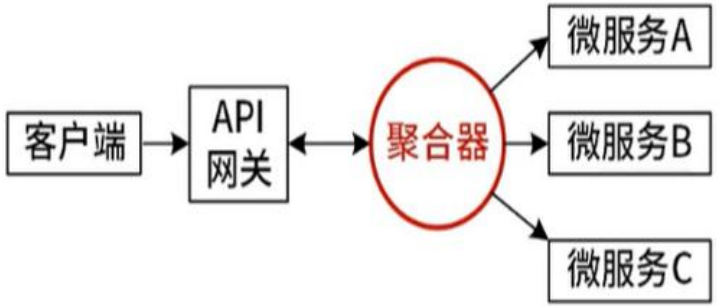
（3）**数据共享微服务**：该模式适用于在单体架构应用到微服务架构的过渡阶段，服务之间允许存在强耦合关系，例如存在多个微服务共享缓存与数据库存储的现象。

（4）**异步消息传递微服务**：对于一些不必要以同步方式运行的业务逻辑，可以**使用消息队列**代替 REST 实现请求、响应，加快服务调用的响应速度。

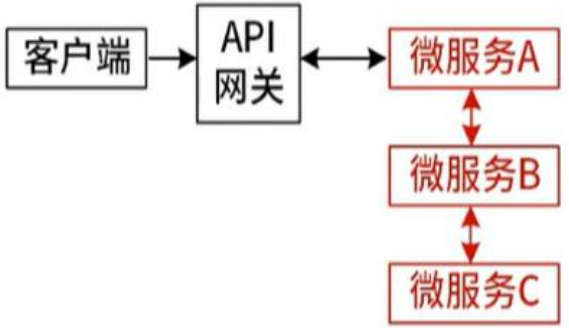
SOA 的设计模式-微服务模式

【微服务架构模式方案】

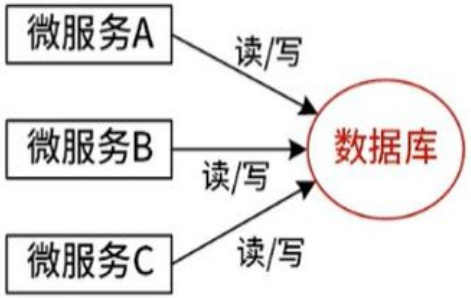
聚合器微服务



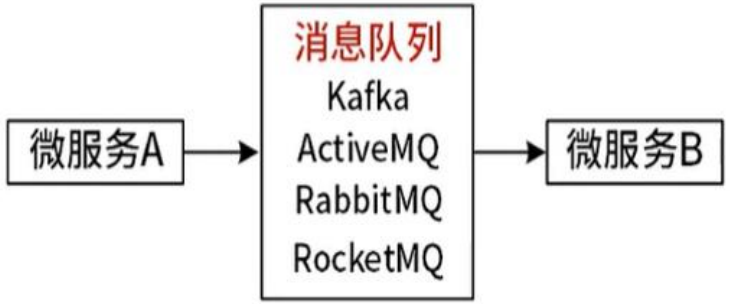
链式微服务



数据共享微服务



异步消息传递微服务



典型例题

阅读以下关于 Web 系统设计的叙述，在答题纸上回答问题1至问题3。

【 说 明 】

某银行拟将以分行为主体的银行信息系统，全面整合为由总行统一管理维护的银行信息系统，实现统一的用户账户管理、转账汇款、自助缴费、理财投资、贷款管理、网上支付、 财务报表分析等业务功能。但是，由于原有以分行为主体的银行信息系统中，多个业务系统采用异构平台、数据库和中间件，使用的报文交换标准和通信协议也不尽相同，使用传统的 EAI 解决方案根本无法实现新的业务模式下异构系统间灵活的交互和集成。因此，为了以最小的系统改进整合现有的基于不同技术实现的银行业务系统，该银行拟采用基于ESB的面向服务架构 (SOA) 集成方案实现业务整合。

【问题1】(7分)

请分别用200字以内的文字说明什么是面向服务架构 (SOA) 以及ESB 在 SOA 中的作用与特点。

典型例题

【问题1】参考答案

面向服务的体系架构 (SOA) 是一种粗粒度、松耦合服务架构，服务之间通过简单、精确定义接口进行通信。它可以根据需求通过网络对松散耦合的粗粒度应用组件进行分布式部署、组合和使用。

SOA 能帮助企业系统架构设计者以更迅速、更可靠、更高重用性设计整个 业务系统架构，基于 SOA 的系统能够更加从容地面对业务的急剧变化。

企 业 服 务 总 线 (ESB) 是由中间件技术实现的全面支持面向服务架构的基础软件平台， 支持异构环境中的服务以及基于消息和事件驱动模式的交互，并且具有适当的服务质量和可管理性。

典型例题

【问题2】(12分)
基于该信息系统整合的实际需求，项目组完成了基于SOA 的银行信息系统架构设计方 案。该系统架构图如图5-1所示。请从(a)～(j) 中选择相应内容填入图5-1的(1)～(6)，补充完善架构设计图。
(a) 数据层
(b) 界面层
(c) 业务层
(d) bind
(e) 企业服务总线ESB
(f)XML
(g) 安全验证和质量管理
(h) publish
(i)UDDI
(j) 组件层
(k)BPEL

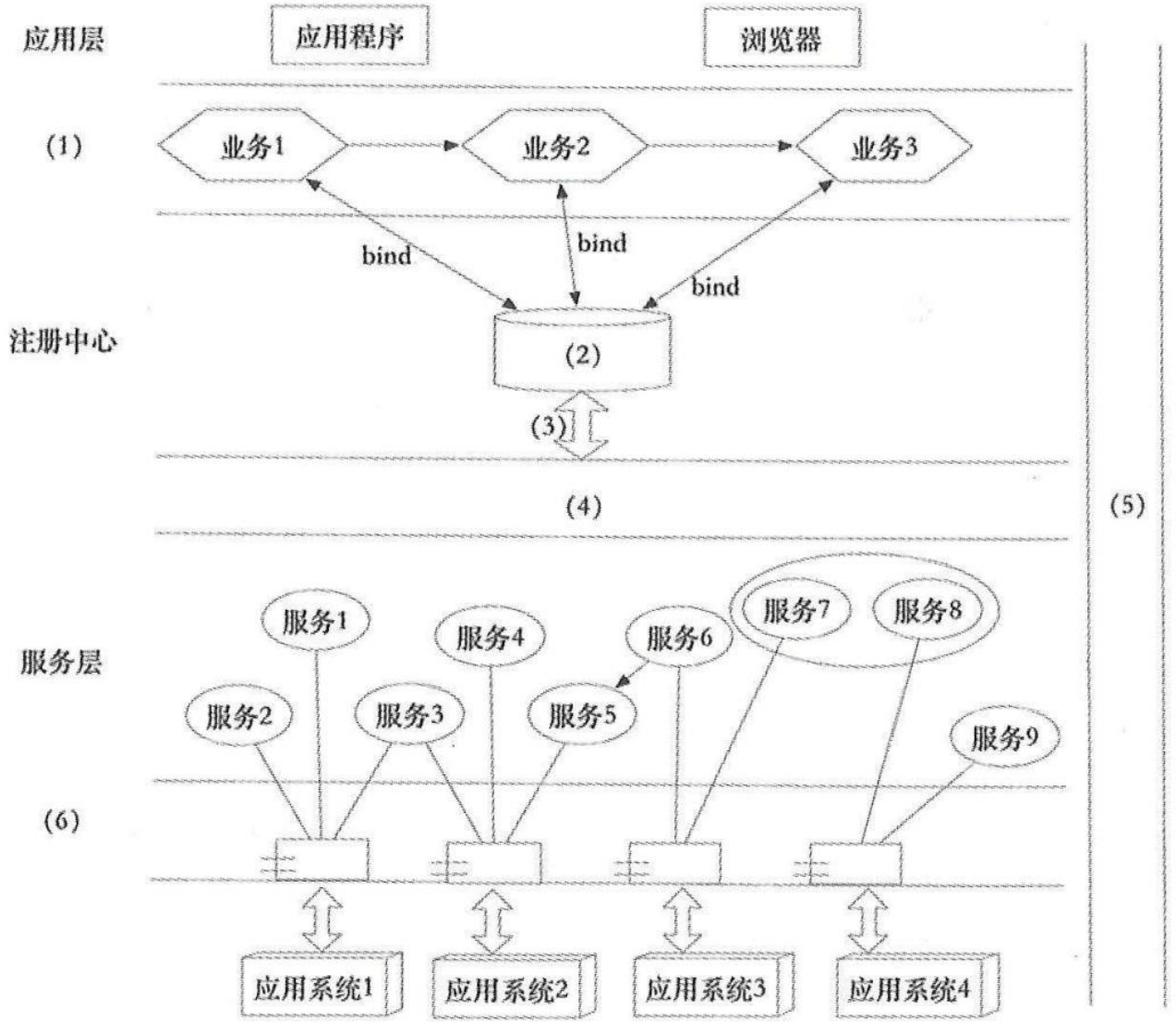


图 5-1 基于 SOA 的银行信息系统架构设计

典型例题

【问题2】(12分)
基于该信息系统整合的实际需求，项目组完成了基于SOA 的银行信息系统架构设计方 案。该系统架构图如图5-1所示。请从(a)～(j) 中选择相应内容填入图5-1的(1)～(6)，补充完善架构设计图。
(a) 数据层
(b) 界面层
(c) 业务层
(d) bind
(e) 企业服务总线ESB
(f)XML
(g) 安全验证和质量管理
(h) publish
(i)UDDI
(j) 组件层
(k)BPEL

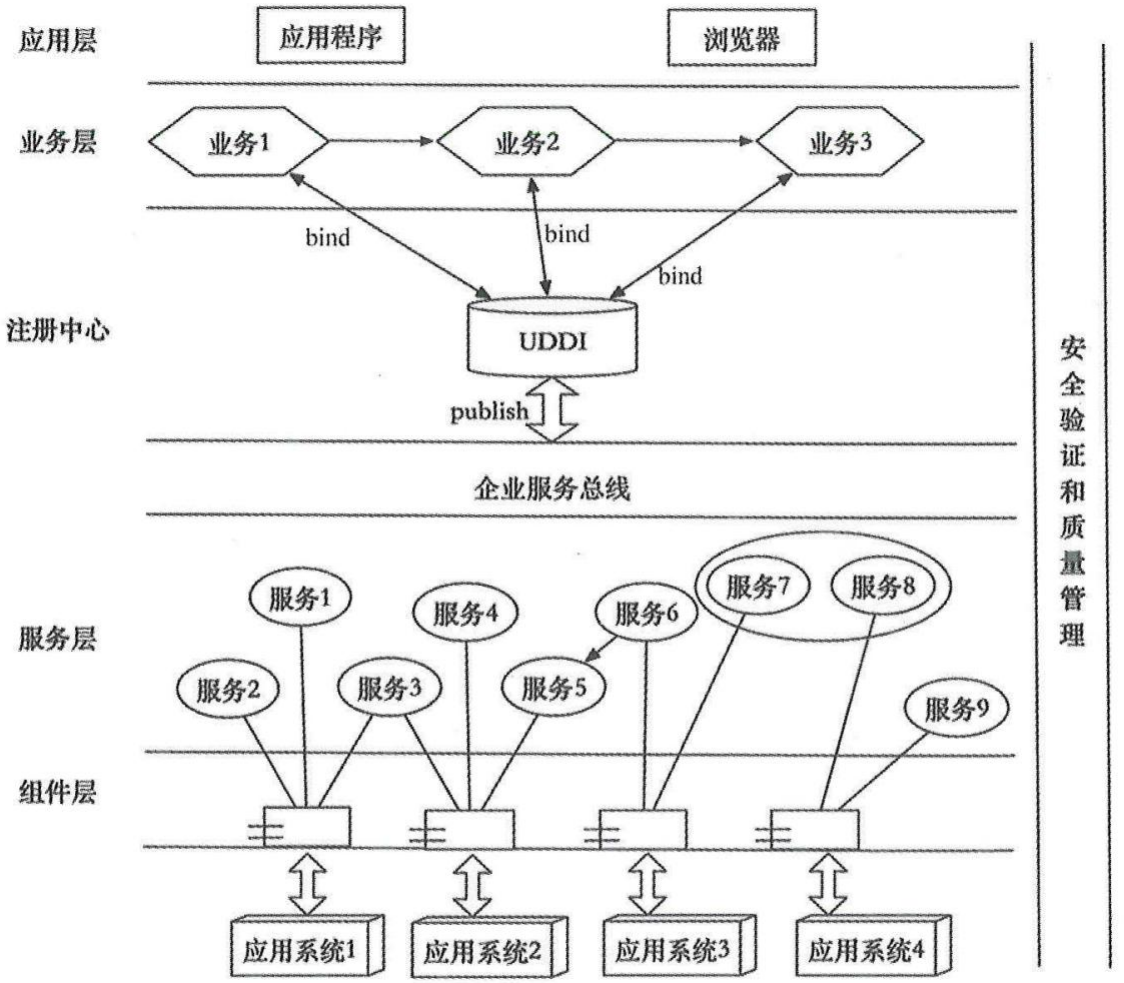
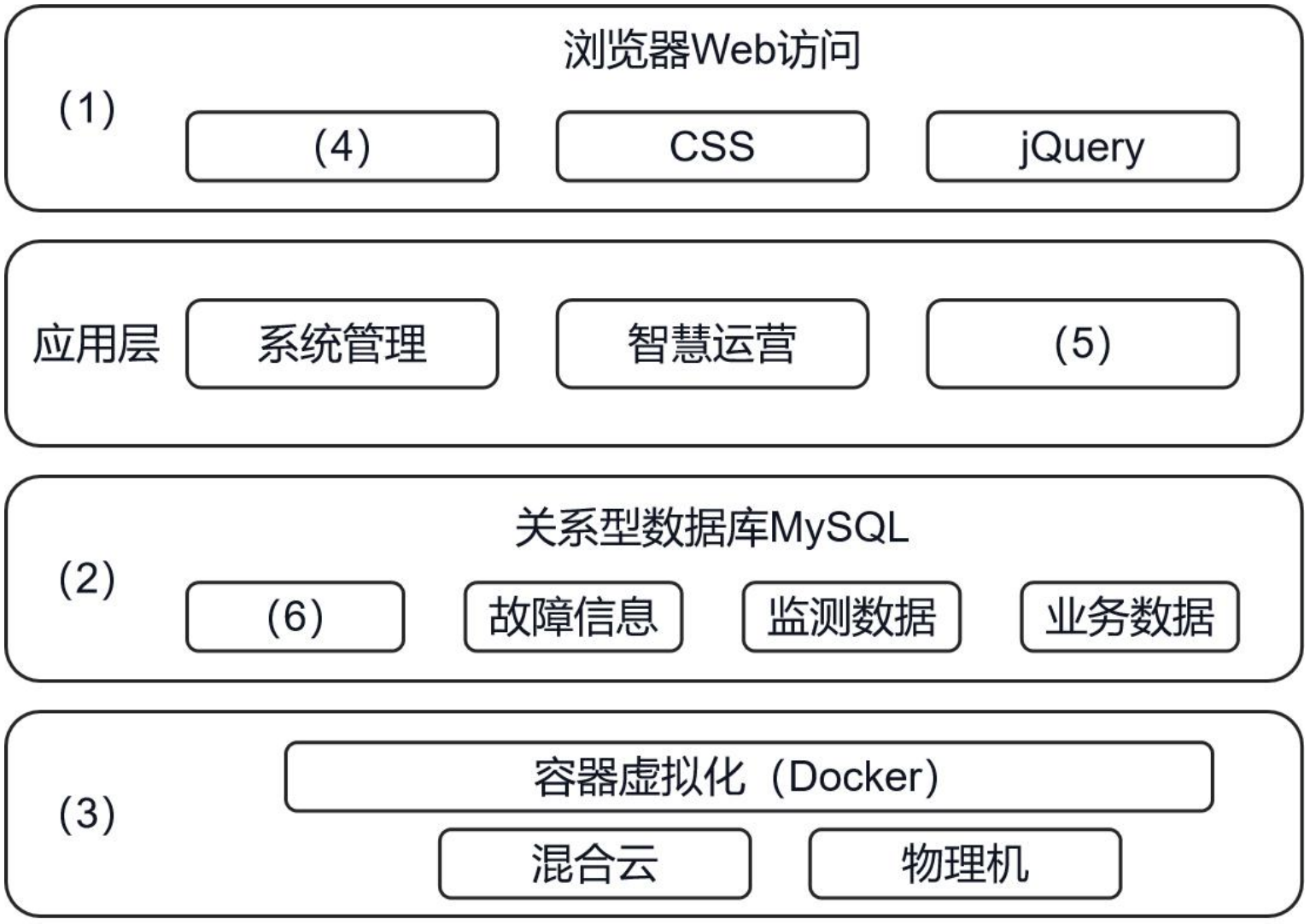


图 5-2　基于 SOA 的银行信息系统完整架构设计

典型例题（2025系分真题）

案例分析五
阅读以下关于 Web 应用系统的描述，回答问题 1 至问题 3。
【说明】
为了保障市民出行安全，提高出行效率，建设绿色交通，某城市委托某公司开发一个基于 Web 的地铁站智慧管理系统。该系统的主要需求分析如下：
（1）智慧指挥。对地铁站日常指挥工作提供所需信息，包括车站端和车辆端。其功能有视频巡检、客流热力图、列车位置、摄像头分布、预案管理和执行结果反馈；
（2）智慧运营。帮助站内调度人员实现各调度角色之间的调度命令，降低劳动强度，提高劳动效率，包括交接班管理、检修工作管理、工作日志显示、文档管理和事件查询等；
（3）系统管理。提供了角色管理、部门管理、摄像头管理、岗位管理、用户管理和联动管理等六个功能；
（4）地铁业务有明显的日常和峰值流量，涉及市民出行安全，应该注意系统和存储的高并发性、可扩展性和安全性。
该公司针对上述需求组建了项目组，经项目组讨论，决定采用 MVC 架构，使用 SSM 框架实现该系统。

典型例题



系统MVC架构图

典型例题

【问题 1】（9 分）
请用 200 字以内文字说明
(1) 什么是 MVC 架构？
(2) 如何利用 SSM 框架来实现 MVC 架构。

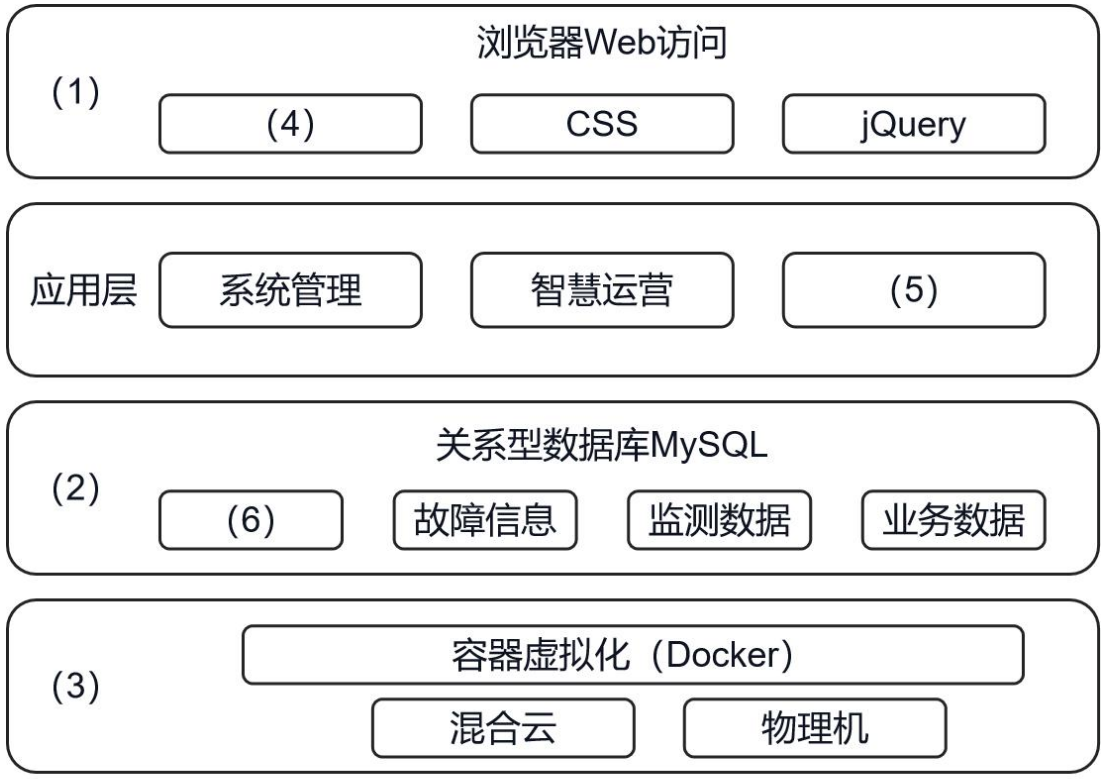
参考答案：
(1) MVC架构
模型（Model）：负责管理数据和业务逻辑，直接与数据库交互，处理数据的存储、检索和更新。
视图（View）：用于展示数据和用户界面，仅负责呈现内容，不包含业务逻辑。
控制器（Controller）：作为桥梁连接模型和视图，接收用户输入，调用模型处理数据，并根据结果选择对应的视图进行展示。

(2) 如何利用 SSM 框架来实现 MVC 架构。
SSM 框架由 Spring、Spring MVC 和 MyBatis 三部分组成。
Model（模型）：表示数据层，通常由 MyBatis 负责实现与数据库的交互（持久层）。
View（视图）：负责数据的展示，通常用 JSP、HTML 或前端框架（如 Vue、Thymeleaf 等）。
Controller（控制器）：负责业务处理和请求分发，通常由 Spring MVC 控制器（如 @Controller 类）实现。

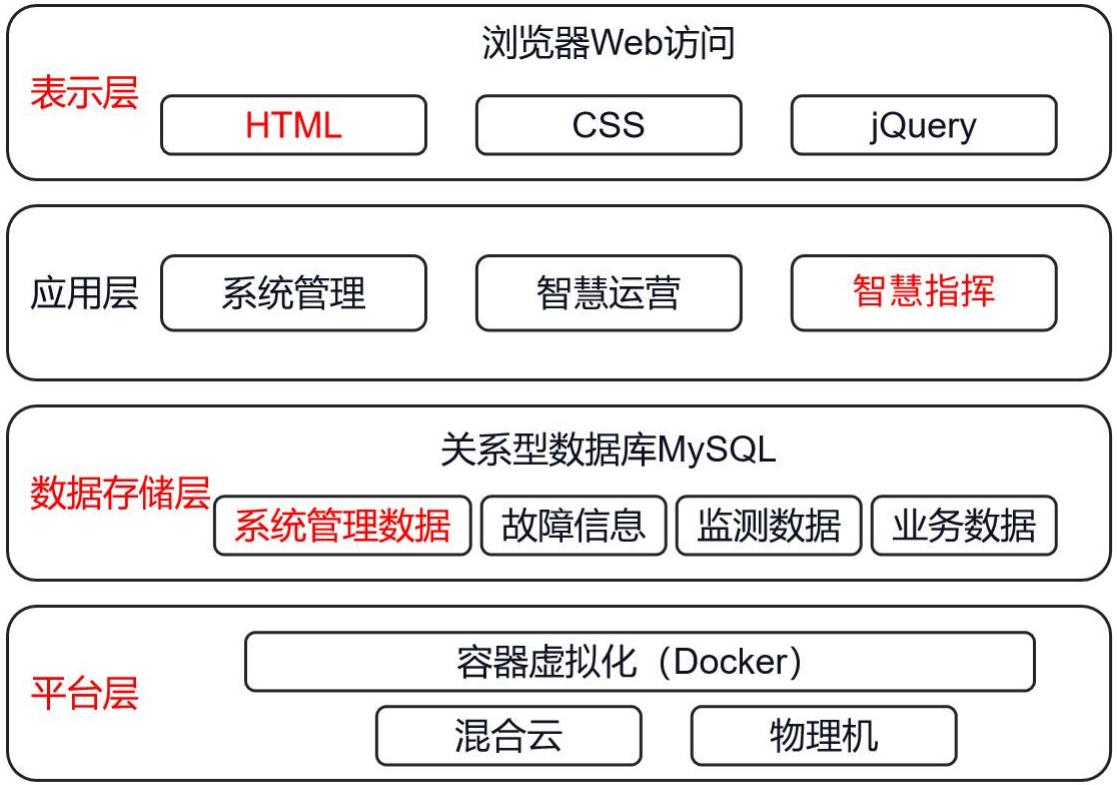
典型例题

【问题 2】（6 分）

请基于对 SSM 框架的了解，补充完善图 1 的空白处 (1) ~ (6)，完成系统的软件架构图。



系统MVC架构图



系统MVC架构图

典型例题

【问题 3】（10 分）

云存储是一种通过互联网将数据存储在远程服务器上的技术，用户可按需访问和管理数据。项目组在讨论采用哪种云存储时，张工建议使用公有云、李工建议使用私有云，经讨论最终决定使用混合云，请比较公有云、私有云和混合云的区别，填写表 1 中的 (1) ~ (9)，并说明选用混合云的理由。

	公有云	私有云	混合云
所有权	(1)	(4)	(7)
安全性	(2)	(5)	(8)
扩展性	(3)	(6)	(9)

	公有云	私有云	混合云
所有权	公开	单个组织	部分公开、部分私有
安全性	低	高	中等
扩展性	高	低	中等

作业

2025.11【问题 1】(7分)
简要说明微服务的概念以及优势与挑战。

202505真题（作业）

可选题1:医院知识图谱

公司想要建设一个医院的智能问答系统，建立医院知识图谱，某工认为互联网的关键字检索无法满足大量的数据需求，建议建立知识图谱，采用爬虫技术。

1、【问题1】图1展示了基于知识图谱的医药领域智能问答系统的架构设计方案，请从给出的(a)~(p)中选择，补充完善图1中(1)~(10)空处的内容。（10分）
比如业务层和知识层内部细节比如知识获取知识管理， 结构化数据。数据清洗，文本向量化等等。【可选项：(a)网络层、(b)数据层 、(c)业务层、(d)知识层、(e)网页数据、(f) 结构化数据、(g）数据采集、(h)知识获取、(i)知识清洗、(j)数据清洗、(k)知识管理、(l)实体获取、(m)关系获取、(n)意图识别、 (o)语句解析、(p)知识检索。
图1 基于知识图谱的医药领域智能问答系统的架构设计方案

202505真题（作业）

可选题1:医院知识图谱

公司想要建设一个医院的智能问答系统，建立医院知识图谱，某工认为互联网的关键字检索无法满足大量的数据需求，建议建立知识图谱，采用爬虫技术。

1、【问题1】图1展示了基于知识图谱的医药领域智能问答系统的架构设计方案，请从给出的(a)~(p)中选择，补充完善图1中(1)~(10)空处的内容。（10分）
比如业务层和知识层内部细节比如知识获取知识管理， 结构化数据。数据清洗，文本向量化等等。【可选项：(a)网络层、(b)数据层 、(c)业务层、(d)知识层、(e)网页数据、(f) 结构化数据、(g）数据采集、(h)知识获取、(i)知识清洗、(j)数据清洗、(k)知识管理、(l)实体获取、(m)关系获取、(n)意图识别、 (o)语句解析、(p)知识检索。

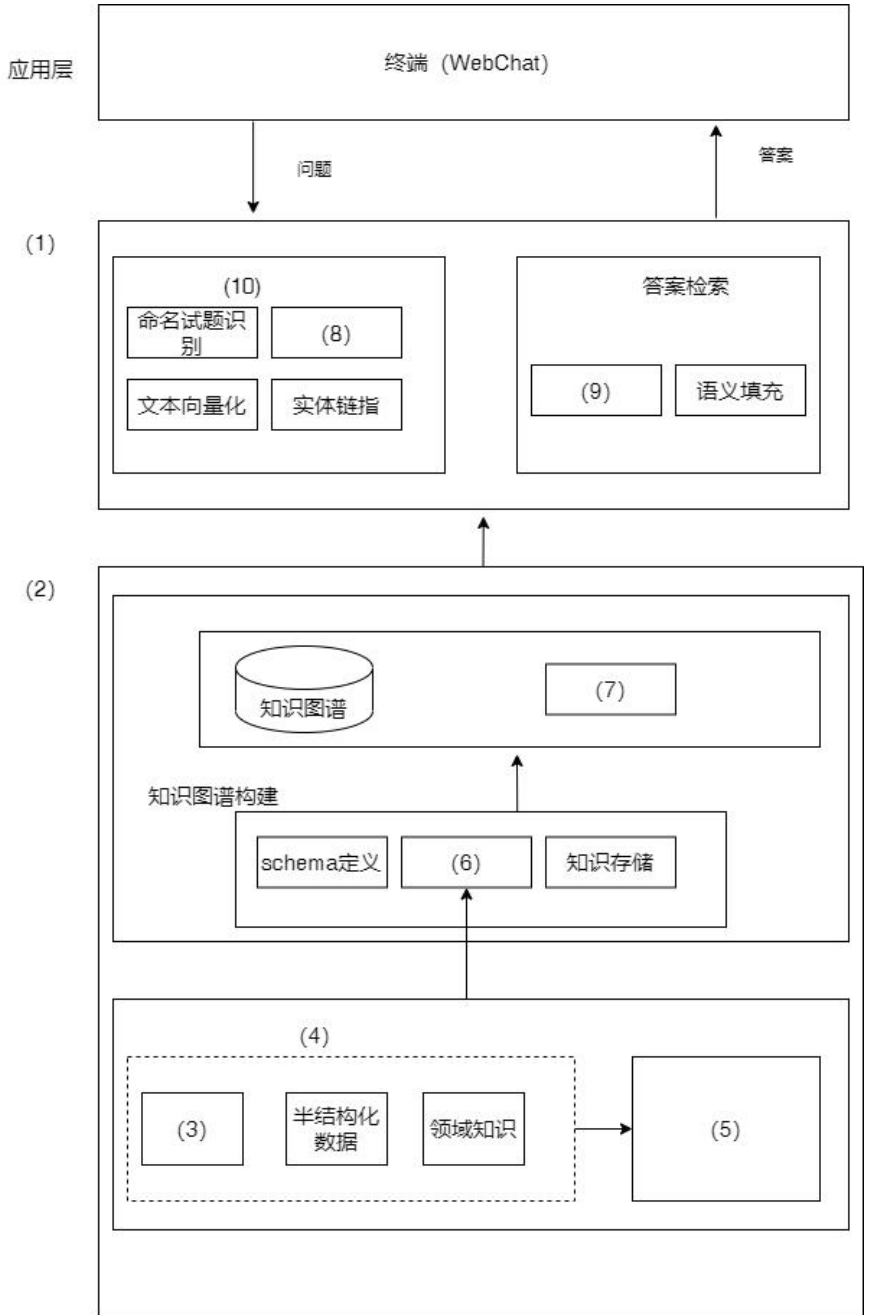


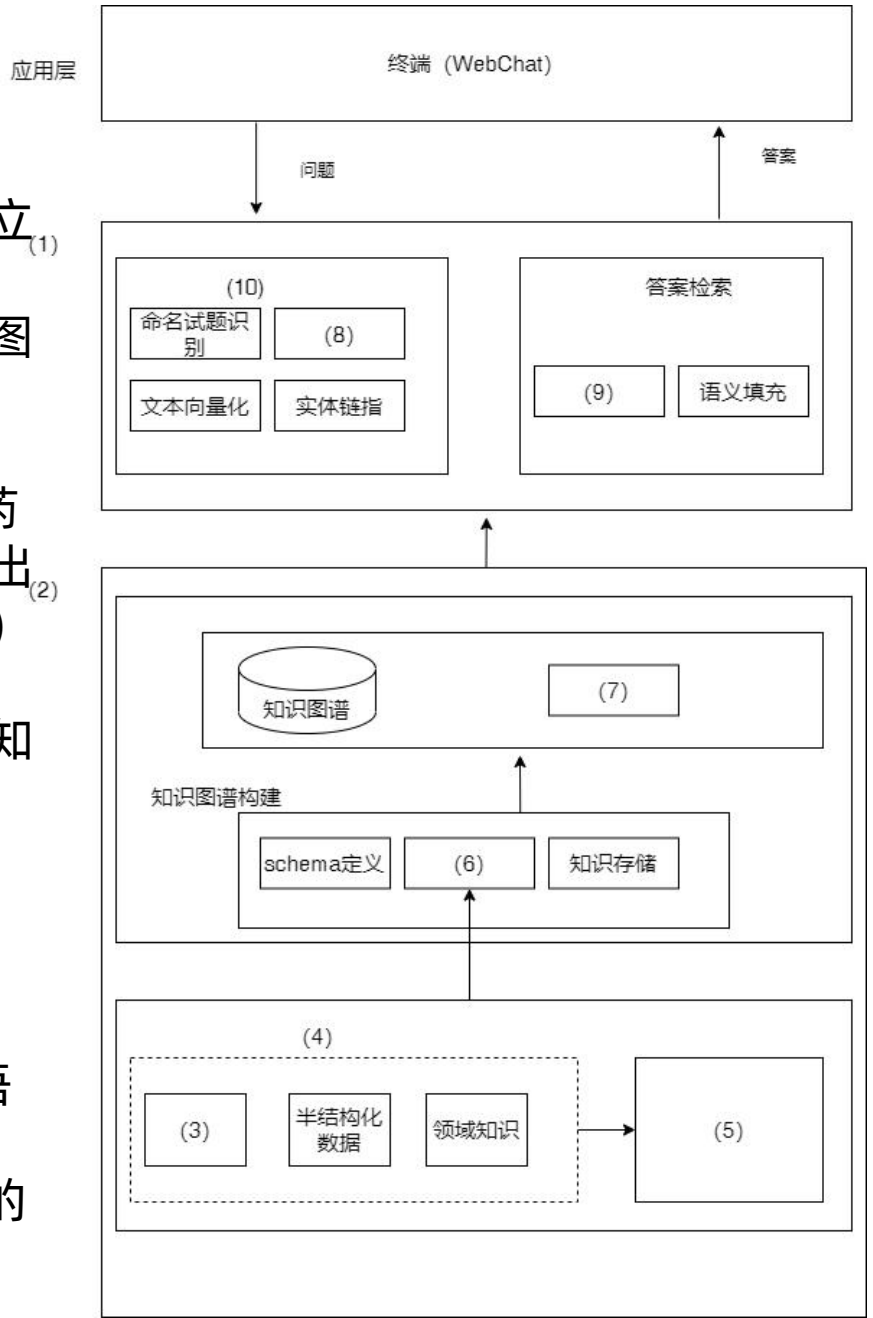
图1 基于知识图谱的医药领域智能问答系统的架构设计方案

202505真题（作业）

可选题1:医院知识图谱

公司想要建设一个医院的智能问答系统，建立医院知识图谱，某工认为互联网的关键字 检索无法满足大量的数据需求，建议建立知识图谱，采用爬虫技术。

1、【问题1】图1展示了基于知识图谱的医药领域智能问答系统的架构设计方案，请从给出的(a)~(p)中选择，补充完善图1中(1)~(10)空处的内容。（10分）
比如业务层和知识层内部细节比如知识获取知识管理， 结构化数据。数据清洗，文本向量化等等。【可选项：(a)网络层、(b)数据层 、(c)业务层、(d)知识层、(e)网页数据、(f) 结构化数据、(g）数据采集、(h)知识获取、(i)知识清洗、(j)数据清洗、(k)知识管理、(l)实体获取、(m)关系获取、(n)意图识别、 (o)语句解析、(p)知识检索。
图1 基于知识图谱的医药领域智能问答系统的架构设计方案



【参考答案】：
（1）：（c）业务层
（2）：（b）数据层
（3）：（f）结构化数据
（4）：（g）数据采集
（5）：（j）数据清洗
（6）：（h）知识获取
（7）：（k）知识管理
（8）：（l）实体获取
（9）：（p）知识检索
（10）：（n）意图识别

作业

【参考答案】
微服务是一种架构风格，将单体应用划分成一组小的服务，服务之间相互协作，实现业务功能每个服务运行在独立的进程中，服务间采用轻量级的通信机制协作(通常是HTTP/JSON)，每个服务围绕业务能力进行构建，并且能够通过自动化机制独立地部署。
1、微服务有以下优势：
（1）通过分解巨大单体式应用为多个服务方法解决了复杂性问题。它把庞大的单一模块应用分解为一系列的服务，同时保持总体功能不变，但整体并发却得到极大提升。
（2）让每个服务能够独立开发，开发者能够自由选择可行的技术，提供API服务，数据库的结构。
（3）微服务架构模式是每个微服务独立的部署。开发者不再需要协调其他服务部署对本服务的影响。这种改变可以加快部署速度。
（4）微服务使得每个服务独立扩展。可以根据每个服务的规模来部署满足需求的规模，甚至可以使用更适合于服务资源需求的硬件。
2、微服务架构带来的挑战如下：
（1）并非所有的系统都能转成微服务。
（2）部署较单体架构更加复杂：系统由众多微服务搭建，每个微服务需要单独部署，从而增加部署的复杂度，容器技术能够解决这一问题。
（3）性能问题：由于微服务注重独立性，互相通信时只能通过标准接口，可能产生延迟或调用出错。
（4）数据一致性问题：作为分布式部署的微服务，在保持数据一致性方面需要比传统架构更加困难。

202505综合知识

65.下面关于三层C/S架构的特点描述不正确的是（）。
A.合理地划分三层的功能，使整个系统的逻辑结构更为清晰，能提高系统的可维护性和可扩展性
B.B/S架构是一种特殊的两层C/S架构
C.与两层C/S架构相比，在三层C/S架构中，增加了一个应用服务器
D.三层C/S架构将应用系统分成表示层、功能层和数据层三个部分
【解析】：
三层C/S 结构如图12-2 (a）所示。
目前最典型的基于三层C/S 结构的应用模式便是我们最熟悉、较流行的B/S (Brower/Server，浏览器／服务器）模式，如图12-2 (b）所示。所以B错误。
【参考答案】：B

66.SOA中（）进一步解耦了服务请求者和服务提供者。
A.WebService　　B.ESB（企业服务总线）　　C.服务注册表　　D.RMI
【解析】：
ESB 的基本特征和能力包括：描述服务的元数据和服务注册管理；在服务请求者和提供者之间传递数据，以及对这些数据进行转换的能力，并支持由实践中总结出来的一些模式如同步模式、异步模式等；发现、路由、匹配和选择的能力，以支持服务之间的动态交互，解耦服务请求者和服务提供者。高级一些的能力，包括对安全的支持、服务质量保证、可管理性和负载平衡等。
【参考答案】：B

202505综合知识

68.关于架构脆弱性，描述错误的是（）。
A.分层架构需要穿过多层通信，可能导致性能问题
B.黑板风格可能没得到解，也没有合适的知识来源继续执行
C.管道过滤器无法并发
D.事件驱动中触发事件顺序不可控
【解析】：
管道过滤器支持并发处理，所以C错误。
【参考答案】：C

69.影响软件质量的三组因素是（）。
A.产品运行、产品修改、产品转移
B.产品运行、产品修改、产品重构
C.产品修改、产品转移、产品重构
D.产品运行、产品重构、产品健壮
【解析】：
从管理角度出发，可以将影响软件质量的因素划分为3 组，分别反映用户在使用软件产品时的3 种不同倾向和观点。这3 组分别是：产品运行、产品修改和产品转移。
【参考答案】：A

THANKS