极客时间训练营

系统架构设计师

第15章 面向服务架构设计理论与实践

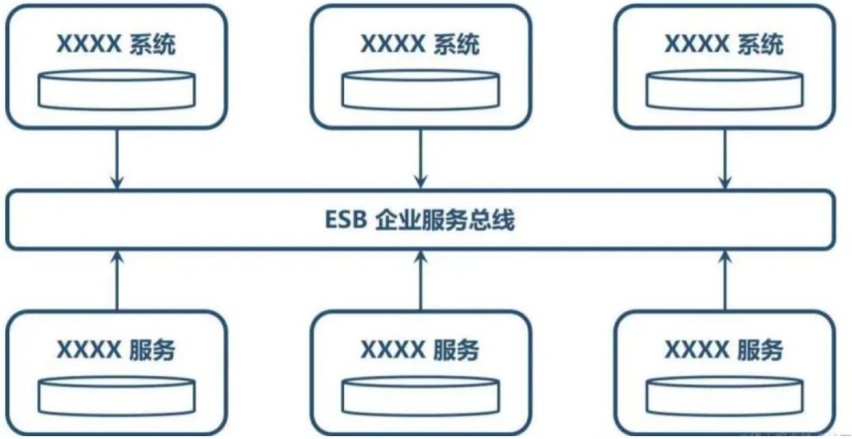
授课：王建平

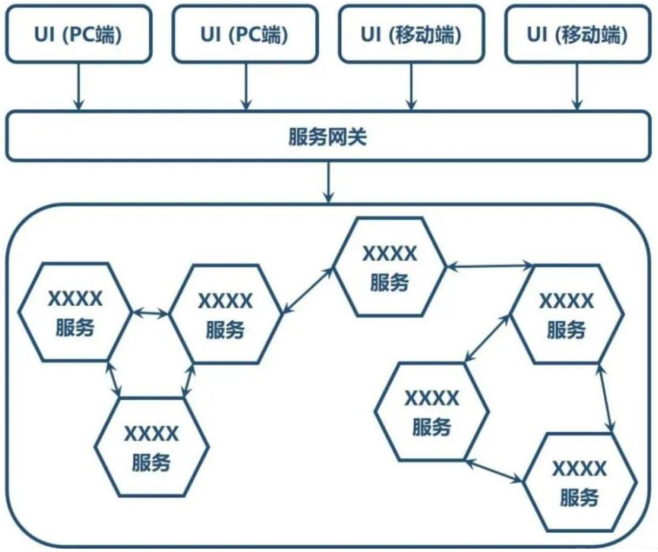
目录

目录

架构发展

软件架构的演化过程大概分成以下阶段：1、单体架构； 2、SOA 架构；3、微服务架构（教程无）





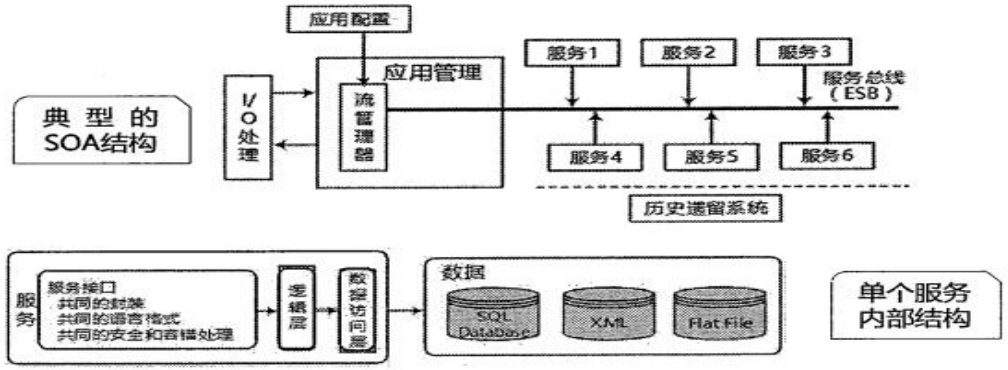
SOA 的相关概念

◆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。（★★）

◆ BPEL：面向Web 服务的业务流程执行语言，是一种使用Web服务定义和执行业务流程的语言。使用BPEL，用户可以通过组合、编排和协调Web服务自上而下地实现面向服务的体系结构。BPEL目前用于整合现有的Web Services，将现有的Web Services按照要求的业务流程整理成为一个新的Web Services，在这个基础上，形成一个从外界看来和单个Service一样的Service。



SOA 的相关概念

◆ SOA的微服务化发展
◆ SOA架构向更细粒度、更通用化程度发展，就成了所谓的微服务了。SOA与微服务的区别在于如下几个方面：
(1)微服务相比于SOA更加精细，微服务更多地以独立的进程的方式存在，互相之间并无影响；
(2)微服务提供的接口方式更加通用化，例如HTTP RESTful方式，各种终端都可以调用，无关语言、平台限制；
(3)微服务更倾向于分布式去中心化的部署方式，在互联网业务场景下更适合。

◆ SOA 架构是一个面向服务的架构，可将其视为组件模型，其将系统整体拆分为多个独立的功能模块，模块之间通过调用接口进行交互，有效整合了应用系统的各项业务功能，系统各个模块之间是松耦合的。SOA架构以企业服务总线链接各个子系统，是集中式的技术架构，应用服务间相互依赖导致部署复杂，应用间交互使用远程通信，降低了响应速度。

◆微服务架构是SOA架构的进一步优化，去除了ESB企业服务总线，是一个真正意义上去中心化的分布式架构。其降低了微服务之间的耦合程度，不同的微服务采用不同的数据库技术，服务独立，数据源唯一，应用极易扩展和维护，同时降低了系统复杂性。

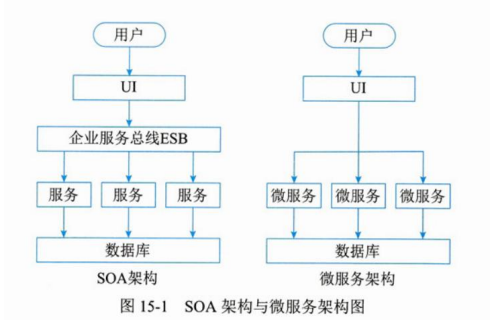


图 15-1　SOA 架构与微服务架构图

SOA 的设计模式-微服务模式

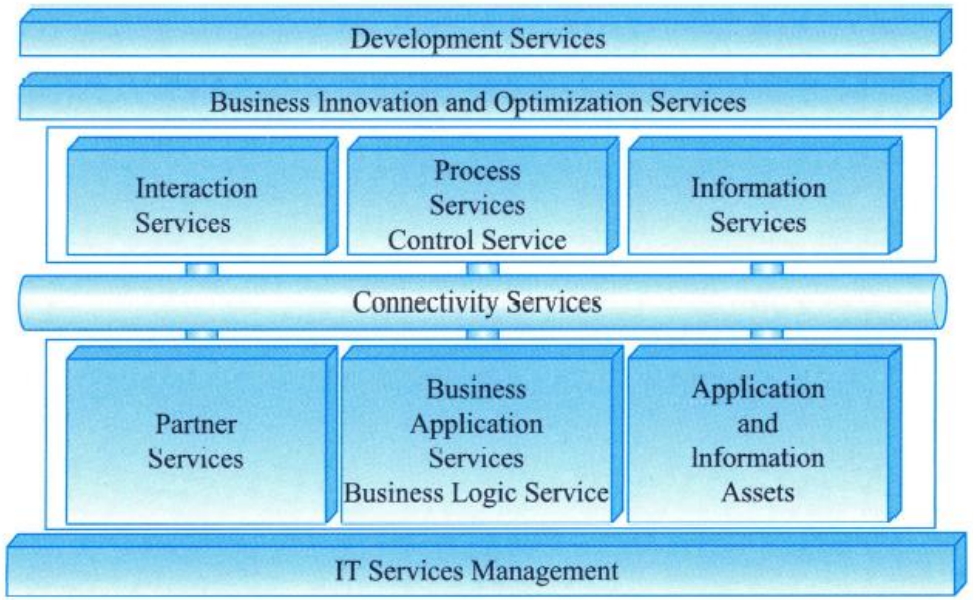
SOA和微服务区别（★★★）

SOA	微服务
是整体的服务，所有服务放一起打包管理	能拆分就拆分
水平多层次性	纵向业务划分
按层次划分不同部门的组织责任	由单一的组织负责
粗粒度	细粒度
描述架构非常复杂	两句话就可以描述清楚
存在较复杂的组件	组件小
业务逻辑跨越多个业务领域	业务逻辑存在每个服务中
企业服务总线（ESB）充当了服务之间通信的角色	使用轻量级的通信方式，比如HTTP
企业级，自上而下开展实施	团队级，自底向上开展实施
集成方式复杂（ESB/WS/SOAP）	简单的集成方式（HTTP/REST/JSON）

SOA的参考架构

◆典型的以服务为中心的企业集成架构如下图所示，采用“关注点分离” 的方法规划企业集成中的各种架构元素，同时从服务视角规划每种架构元素提供的服务，以及服务如何被组合在一起完成某种类型的集成。可划分为六大类：（★★★）

(1)业务逻辑服务: 包括用于实现业务逻辑的服务和执行业务逻辑的能力。
(2)控制服务: 包括实现人、流程和信息集成的服务，以及执行这些集成逻辑的能力。
(3)连接服务: 通过提供企业服务总线提供分布在各种架构元素中服务间的连接性。
(4)业务创新和优化服务: 用于监控业务系统运行时服务的业务性能，采取措施适应变化的市场。
(5)开发服务: 贯彻整个软件开发生命周期的开发平台。
(6)IT服务管理: 支持业务系统运行的各种基础设施管理能力或服务。



SOA的参考架构

1.连接服务——企业服务总线：
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。

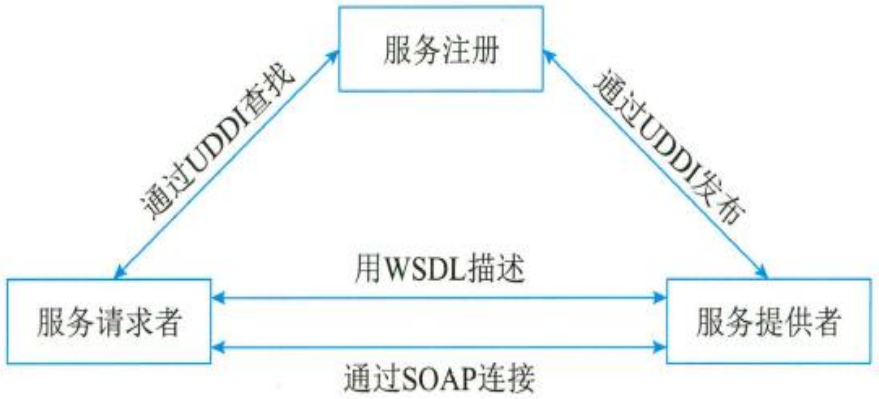
2.业务逻辑服务
1)整合已有应用——应用和信息访问服务：实现对已有应用和信息的集成，主要有两类访问服务：可接入服务、事件发现服务。
2)整合新开发的应用——业务应用服务：实现新应用集成，主要有三类业务应用服务：组件服务（可重用）、核心服务（运行时）、接口服务。
3)整合客户和业务伙伴(B2C/B2B)——伙伴服务：提供与企业外部的B 2 B 的集成能力，包括：社区服务、文档服务、协议服务。

SOA的参考架构

3.控制服务
1)数据整合——信息服务：提供集成数据的能力，目前主要包括如下集中信息服务：联邦服务（不同类型数据聚合）、复制服务（远程数据本地访问）、转换服务（格式转换）、搜索服务。
2)流程整合——流程服务：完成业务流程集成，包括：编排服务（预定义流程顺序）、事务服务（保证ACID）、人工服务（人工活动集成到流程中）。
3)用户访问整合——交互服务：实现用户访问集成，包括：交付服务（运行时交互框架）、体验服务、资源服务（运行时交互组件的管理）。
4.开发服务：开发环境和工具中为不同开发者的角色提供的功能被称为开发服务。根据开发过程中开发者角色和职责的不同，有如下4类服务：建模服务、设计服务、实现服务、测试服务。
5.业务创新和优化：以业务性能管理(BPM)技术为核心提供业务事件发布、收集和关键业务指标监控能力。包括以下服务：
(1)公共事件框架服务: 通过一个公共事件框架提供IT和业务事件的激发、存储和分类等。
(2)采集服务通过基于策略的过滤和相关性分析检测感兴趣的服务。
(3)监控服务: 通过事件与监控上下文间的映射，计算和管理业务流程的关键性能指标。
6.IT 服务管理：为业务流程和服务提供安全、高效和健康的运行环境，包括：安全和目录服务、系统管理和虚拟化服务。

目录

案例引入



在项目开发中，我们除了发布WebService供客户调用外，也经常需要调用一些客户或者第三方的WebService服务

1、寻找天气预报服务。百度“天气预报服务”，这里以http://www.webxml.com.cn/WebServices/WeatherWebService.asmx中会提供一些方法getWeatherbyCityName、getSupportDataSet的方法

2、添加一个空的MVC项目。然后引入Web服务，在引用中加入URL.

3、添加Home控制器和两个GetWeather方法和GetWeather视图

4、运行测试即可

SOA主要协议和规范

Web 服务作为实现 SOA 中服务的最主要手段。Web 服务最基本的协议包括UDDI、WSDL和SOAP，通过它们，可以提供直接而又简单的 Web Service 支持。（★★★）

一、UDDI协议

UDDI(统一描述、发现和集成协议)是一种用于描述、发现、集成Web Service的技术，它是Web Service协议栈的一个重要部分。通过UDDI，企业可以根据自己的需要动态查找并使用Web服务，也可以将自己的Web服务动态地发布到UDDI注册中心，供其他用户使用。

二、WSDL规范

WSDL (Web 服务描述语言)，是一个用来描述 Web 服务和说明如何与 Web 服务通信的XML语言。也就是描述与目录中列出的Web服务进行交互时需要绑定的协议和信息格式。WSDL描述Web服务的公共接口。

通过WSDL，可描述Web 服务的三个基本属性。

(1)服务做些什么--服务所提供的操作 (方法)。

(2)如何访问服务--和服务交互的数据格式以及必要协议。

(3)服务位于何处--协议相关的地址，如 URL。

WSDL文档以端口集合的形式来描述 Web 服务，WSDL服务描述包含对一组操作和消息的一个抽象定义，绑定到这些操作和消息的一个具体协议，和这个绑定的一个网络端点规范。WSDL文档被分为两种类型：服务接口和服务实现。

SOA主要协议和规范

三、SOAP协议

SOAP(简单对象访问协议)是在分散或分布式的环境中交换信息的简单协议，是一种轻量的、简单的、基于XML的协议。包括4部分：

- ✓ 封装(Envelop)：定义了一个框架，该框架描述了消息的内容是什么，是谁发送的，谁应当接收并处理以及如何处理它；
- ✓ 编码规则：定义了一种序列化的机制，用于表示应用程序需要使用的数据类型的实例；
- ✓ SOAP RPC 表示：定义了用于表示远程过程调用和应答的协定；
- ✓ 绑定(Binding)：定义了SOAP使用哪种协议交换信息。HTTP/TCP /UDP协议都可以。

四、REST 规范：为了让不同的软件或者应用程序在任何网络环境下都可以进行信息的互相传递。微服务对外就是以 RESTAPI 形式暴露给调用者。RESTful 即 REST 式的，是对遵循 REST 设计思想同时满足设计约束的一类架构设计或应用程序的统称，这一类都可称为 RESTful，可以理解为资源表述性状态转移：

(1)资源：将互联网中一切暴露给客户端的事物都可以看作是一种资源。

(2)表述：REST 中用表述描述资源在 Web 中某一个时间的状态。

(3)状态转移：分为两种，应用状态——对某个时间内用户请求会话相关信息的快照，保存在客户端。资源状态——在服务端保存，是对某个时间资源请求表述的快照。

(4)超链接：是通过在页面中嵌入链接和其他资源建立联系。

目录

SOA设计标准要求

SOA设计标准要求（★）
1.文档标准化。SOA 服务具有平台独立的自我描述 XML 文档。Web 服务描述语言是用于描述服务的标准语言。
2.通信协议标准。SOA 服务用消息进行通信，该消息通常使用 XML Shema 来定义。
3.应用程序统一登记与集成。在一个企业内部，SOA 服务通过一个扮演目录列表 (Directory Listing)角色的登记处(Registry)来进行维护。应用程序在登记处(Registry)寻找并调用某项服务。统一描述、定义和集成是服务登记的标准。
4.服务质量（QoS）。主要包括：
(1)可靠性：服务消费者和服务提供者之间传输文档时的传输特性（仅且仅仅传送一次、最多传送一次、重复消息过滤、保证消息传送）。
(2)安全性：Web 服务安全规范用来保证消息的安全性。
(3)策略：服务提供者有时候会要求服务消费者与某种策略通信。例如，服务提供商可能会要求消费者提供 Kerberos 安全标示才能取得某项服务。
(4)控制：在 SOA 中，进程是使用一组离散的服务创建的。BPEL4WS 或者 WSBPEL (Web Service Business Process Execution Language ）是用来控制这些服务的语言。
(5)管理：针对运行在多种环境下的所有服务，必须有一个统一管理系统，以便系统管理员能够有效管理。任何根据 WSDM 实现的服务都可以由一个 WSDM 适应(WSDM-compliant ）的管理方案来管理。

SOA设计原则

SOA的设计原则如下：（★★★）

(1)无状态。避免服务请求者依赖于服务提供者的状态。

(2)单一实例。避免功能冗余。

(3)明确定义的接口。服务的接口由 WSDL定义，用于指明服务的公共接口与其内部专用实现之间的界线。WS-Policy 用于描述服务规约，XML模式用于定义所交换的消息格式(即服务的公共数据)。使用者依赖服务规约调用服务，所以服务定义必须长时间稳定，一旦公布，不能随意更改；服务的定义应尽可能明确，减少使用者的不适当使用；不要让使用者看到服务内部的私有数据。

(4)自包含和模块化。服务封装了那些在业务上稳定、重复出现的活动和组件，实现服务的功能实体是完全独立自主的，独立进行部署、版本控制、自我管理和恢复。

(5)粗粒度。服务数量不应该太大，依靠消息交互而不是远程过程调用(RPC)，通常消息量比较大，但是服务之间的交互频度较低。

SOA设计原则

SOA的设计原则如下：

(6)服务之间的松耦合性。服务使用者看到的是服务的接口，其位置、实现技术和当前状态等对使用者是不可见的，服务私有数据对服务使用者是不可见的。

(7)重用能力。服务应该是可以重用的。

(8)互操作性、兼容和策略声明。为了确保服务规约的全面和明确，策略成为一个越来越重要的方面。这可以是技术相关的内容，例如一个服务对安全性方面的要求；也可以是跟业务有关的语义方面的内容，例如需要满足的费用或者服务级别方面的要求，这些策略对于服务在交互时是非常重要的。
WS-Policy 用于定义可配置的互操作语义来描述特定服务的期望、控制其行为。在设计时，应该利用策略声明确保服务期望和语义兼容性方面的完整和明确

目录

SOA的设计模式-服务注册表模式

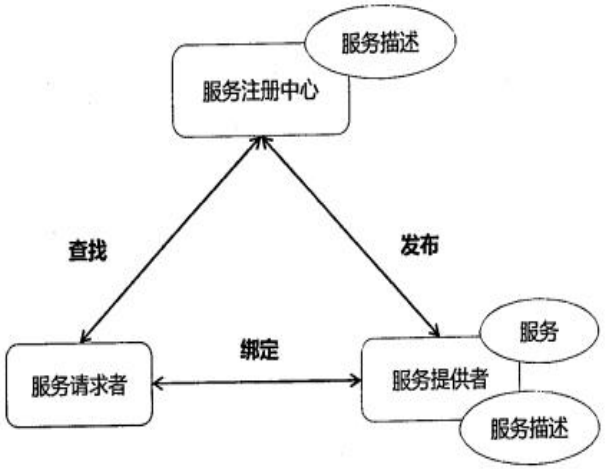
一、服务注册表模式（★★★）

服务注册表(Service Registry)主要在 SOA设计时使用，虽然它们常常也具有运行时段的功能。注册表包括有关服务和相关软件组件的配置、遵从性和约束配置文件。任何帮助注册发现和检索服务合同、元数据和策略的信息库、数据库、目录或其他节点都可以被认为是一个注册表。

(1)服务注册：应用开发者，也叫服务提供者，向注册表公布他们的功能。他们公布服务合同，包括服务身份、位置、方法、绑定、配置、方案和策略等描述性属性。

(2)服务位置：也就是服务应用开发，帮助他们查询注册服务，寻找符合自身要求的服务。注册表让服务的消费者检索服务合同。

(3)服务绑定：服务的消费者利用检索到的服务合同来开发代码，开发的代码将与注册的服务绑定、调用注册的服务以及与它们实现互动。



SOA 的设计模式-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

◆微服务架构将一个大型的单个应用或服务拆分成多个微服务，可扩展单个组件而不是整个应用程序堆栈，从而满足服务等级协议。微服务架构围绕业务领域将服务进行拆分，每个服务可以独立进行开发、管理和迭代，彼此之间使用统一接口进行交流，实现了在分散组件中的部署、管理与服务功能，使产品交付变得更加简单，从而达到有效拆分应用，实现敏捷开发与部署的目的。

◆微服务的特点（★★★）

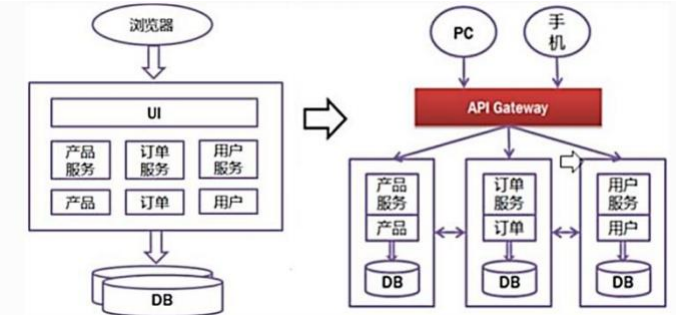
(1)复杂应用解耦：微服务架构将单一模块应用分解为多个微服务，同时保持总体功能不变。

(2)独立：微服务在系统软件生命周期中是独立开发、测试及部署的。

(3)技术选型灵活：微服务架构下系统应用的技术选型是去中心化的，每个开发团队可根据自身应用的业务需求发展状况选择合适的体系架构与技术。

(4)容错：由于各个微服务相互独立，故障会被隔离在单个服务中，并且系统其他微服务可通过重试、平稳退化等机制实现应用层的容错，从而提高系统应用的容错性。

(5)松耦合，易扩展：微服务架构中每个服务之间都是松耦合的，可以根据实际需求实现独立扩展，体现微服务架构的灵活性。

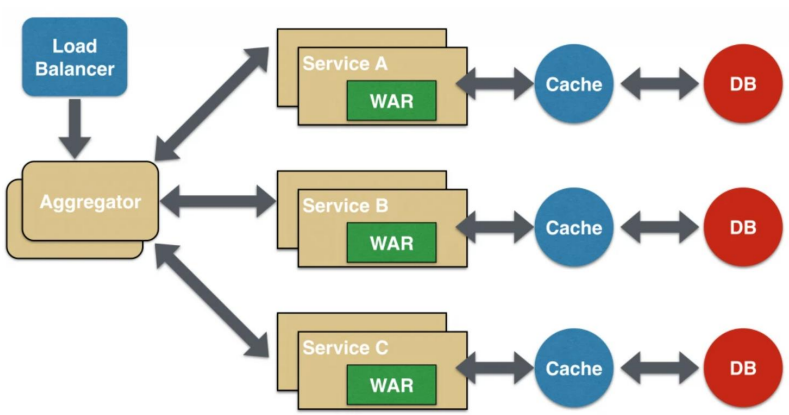


SOA 的设计模式-微服务模式

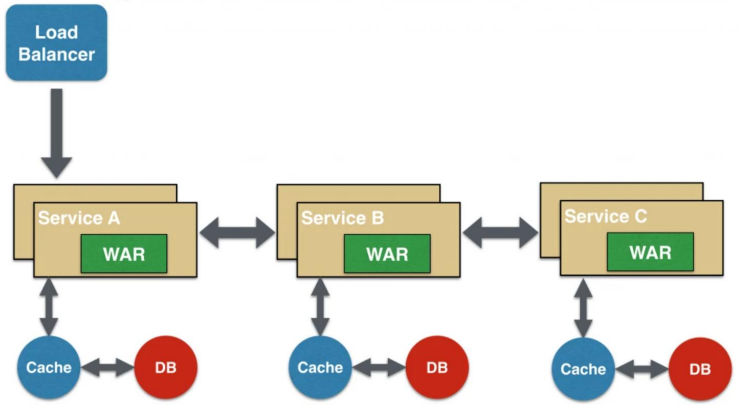
◆微服务架构模式方案。（★★）
主要包括：
（1）聚合器微服务：聚合器充当流程指挥者，调用多个微服务实现系统应用程序所需功能。
（2）链式微服务：客户端或服务在收到请求后，会发生多个服务间的嵌套递归调用，返回经过合并处理的响应。
（3）数据共享微服务：该模式适用于在单体架构应用到微服务架构的过渡阶段，服务之间允许存在强耦合关系，例如存在多个微服务共享缓存与数据库存储的现象。
（4）异步消息传递微服务：对于一些不必要以同步方式运行的业务逻辑，可以使用消息队列代替 REST 实现请求、响应，加快服务调用的响应速度。

◆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

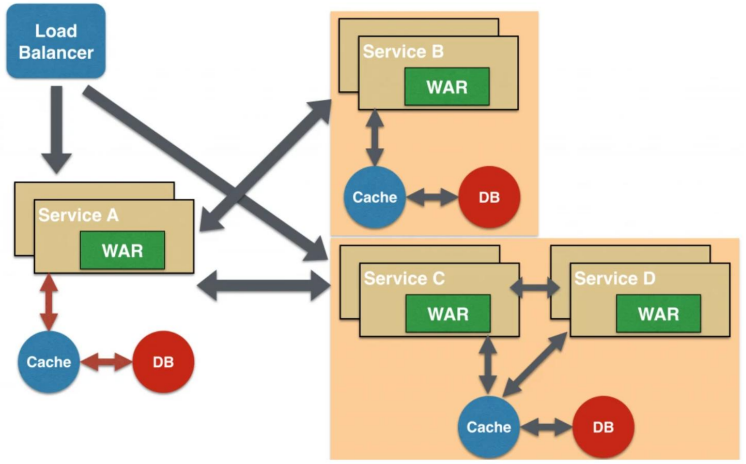
SOA 的设计模式-微服务模式（教程无）



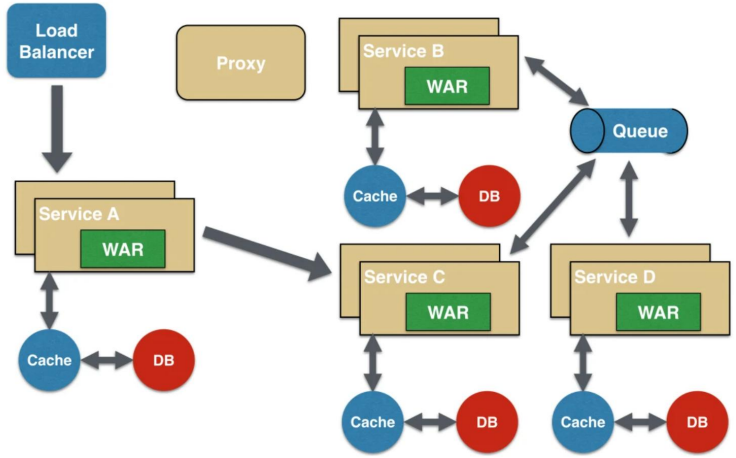
聚合器微服务



链式微服务



数据共享微服务



异步消息传递微服务

构建 SOA 架构时应该注意的问题

◆原有系统架构中的集成需求。包括：应用程序集成的需求、终端用户界面集成的需求、流程集成的需求以及已有系统信息集成的需求。

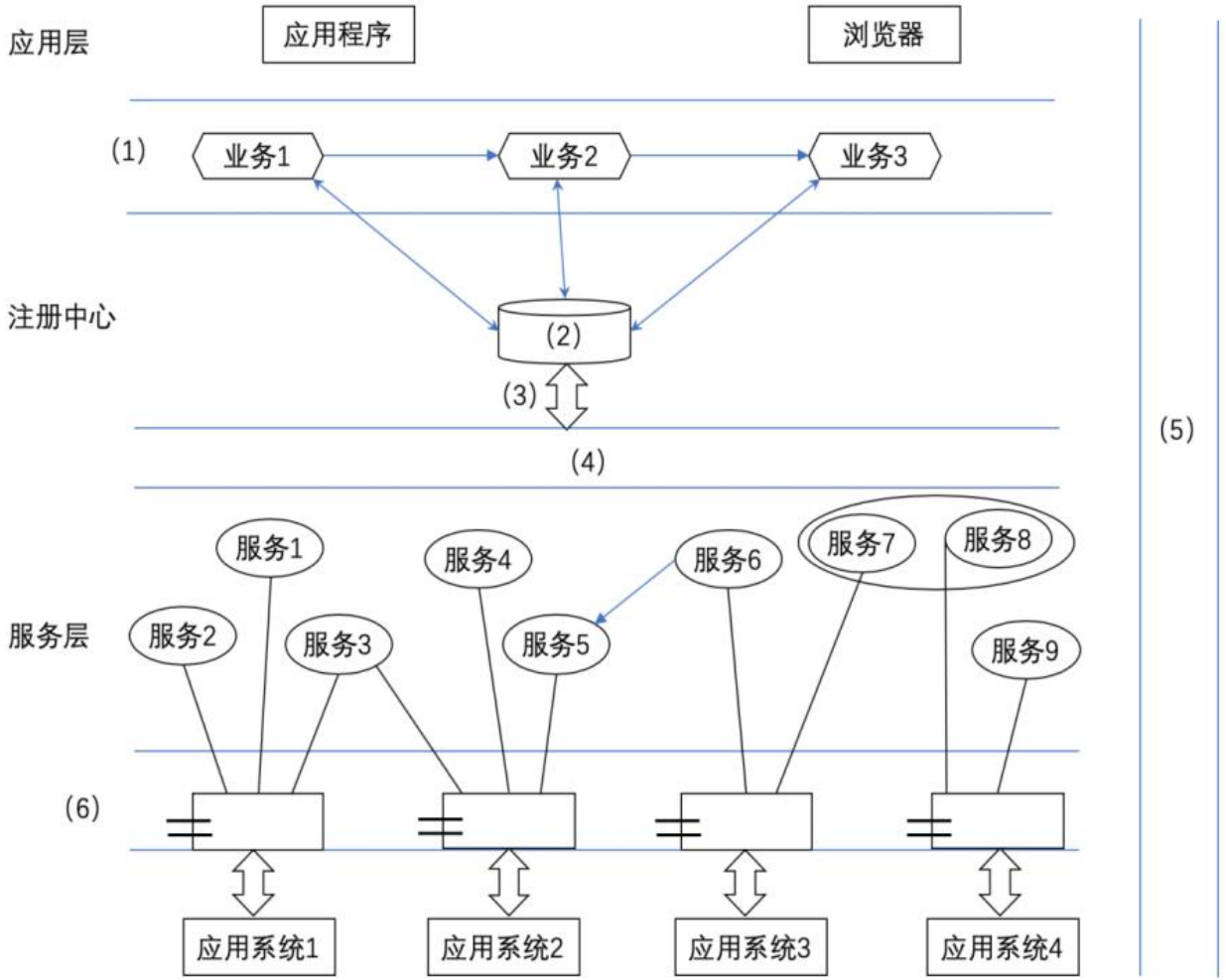
◆服务粒度的控制以及无状态服务的设计。（★）
（1）服务粒度的控制：对于将暴露在整个系统外部的服务推荐使用粗粒度的接口，而相对较细粒度的服务接口通常用于企业系统架构的内部。
（2）无状态服务的设计：SOA 系统架构中的具体服务应该都是独立的、自包含的请求，在实现这些服务的时候不需要前一个请求的状态，也就是说服务不应该依赖于其他服务的上下文和状态，即 SOA 架中的服务应该是无状态的服务。

SOA 实施的过程

◆选择 SOA 解决方案。解决方案主要从以下三个方面选择：（★）
（1）尽量选择能进行全局规划的方案。
（2）选择时充分考虑企业自身的需求。
（3）从平台、实施等技术方面进行考察。

◆业务流程分析。业务流程分析主要关注：（★）
（1）建立服务模型：自顶向下分解法，业务目标分析法，自底向上分析法
（2）建立业务流程：建立业务对象（实体、过程、事件等业务对象）、建立服务接口、建立服务流程。

典型真题



请从(a) ~ (j) 中选择相应内容填入图中(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

1、 业务层
2、UDDI
3、publish
4、企业服务总线ESB
5、安全验证和质量管理
6、组件层

典型真题

下列关于 SOA 与微服务的描述，有错误的是（ ）。
A. 微服务相比于 SOA 更加精细，微服务更多地以独立的进程的方式存在，互相之间并无影响
B. 微服务提供的接口方式更加通用化，例如 HTTP RESTful 方式，各种终端都可以调用，无关语言、平台限制
C. 微服务更倾向于分布式去中心化的部署方式，在互联网业务场景下更适合
D. 微服务更容易实现出高并发的特性，有助于实现互联网业务的秒杀促销活动.
解析：微服务在实现高并发方面是局限的。只有没有调用关系的微服务，相对于单体服务来说，才有并发性的提升。
答案：D

下列选项哪个不是关于 SOA 的服务架构（ ）。
A．业务逻辑服务　B．中间件服务.　C．连接服务　D．控制服务
解析：SOA 的参考架构中包括业务逻辑服务(Business Logic Service)、控制服务(Control Service)、连接服务(Connectivity Service)、业务创新和优化服务(Business Innovation and ptimization Service)、开发服务(Development Service)、IT 服务管理(IT Service Management)。
答案：B

典型真题

SOA 的设计原则为无状态、单一实例、明确定义的接口、（ ）、粗粒度、服务之间的松耦合性、重用能力、互操作性。
A．复用性和构件化　　B．自包含和模块化．　　C．独立性和构件化　D．隔离性和归一化
解析：SOA 的设计原则为无状态、单一实例、明确定义的接口、自包含和模块化、粗粒度、服务之间的松耦合性、重用能力、互操作性。
答案：B

微服务架构将一个大型的单个应用或服务拆分成多个微服务，可扩展单个组件而不是整个应用程序堆栈，从而满足服务等级协议。微服务架构围绕业务领域将服务进行拆分，每个服务可以（ ），彼此之间使用统一接口进行交流，实现了在分散组件中的部署、管理与服务功能，使产品交付变得更加简单，从而达到有效拆分应用，实现敏捷开发与部署的目的。
A．独立进行开发、管理、迭代．　　B．独立进行部署、运维、升级
C．独立进行测试、交付、验收　　D．独立进行发布、发现、访问
解析：微服务架构围绕业务领域将服务进行拆分，每个服务可以独立进行开发、管理和迭代，彼此之间使用统一接口进行交流，实现了在分散组件中的部署、管理与服务功能。
答案：A

本章重点回顾

1、SOA参考架构
2、SOA设计原则
3、SOA设计模式

THANKS