武漢大学
WUHAN UNIVERSITY

第8章　基于服务的体系结构

授课教师：　应时

2018-09-01

基于服务的体系结构

- 面向服务的体系结构（Service-Oriented Architecture，SOA）较为典型的定义：
- W3C的定义：SOA是一种应用程序体系结构，在这种体系结构中，所有功能都定义为独立的服务，这些服务带有定义明确的可调用接口，能够以定义好的顺序调用这些服务来形成业务流程。
- Service-architecture.com的定义：服务是精确定义、封装完整、独立于其它服务所处环境和状态的函数。SOA本质上是服务的集合，服务之间彼此通信，这种通信可能是简单的数据传送，也可能是两个或更多的服务协调进行某些活动。服务之间需要某些方法进行连接。
- Gartner的定义：SOA是一种C/S体系结构的软件设计方法，应用由服务和服务使用者组成，SOA与大多数通用的C/S体系结构模型不同之处，在于它着重强调构件的松散耦合，并使用独立的标准接口。

8.1 SOA概述

- SOA是一种在计算环境中设计、开发、部署和管理离散逻辑单元（服务）模型的方法。
- 虽然SOA是基于对象的，但是作为一个整体，它却不是面向对象的。
- 图8-1描述了一个完整的SOA模型。

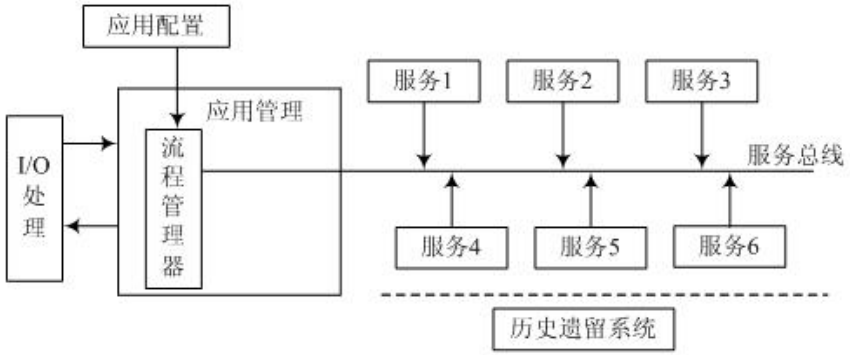

图8-1 SOA模型示例

- 在SOA模型中，所有的功能都定义成了独立的服务。服务之间通过交互和协调完成业务的整体逻辑。所有的服务通过服务总线或流程管理器来连接。
- 这种松散耦合的体系结构使得各服务在交互过程中无需考虑双方的内部实现细节，以及部署在什么平台上。

8.1 SOA概述

- 1、服务的基本结构
- 一个独立的服务基本结构如图8-2所示。

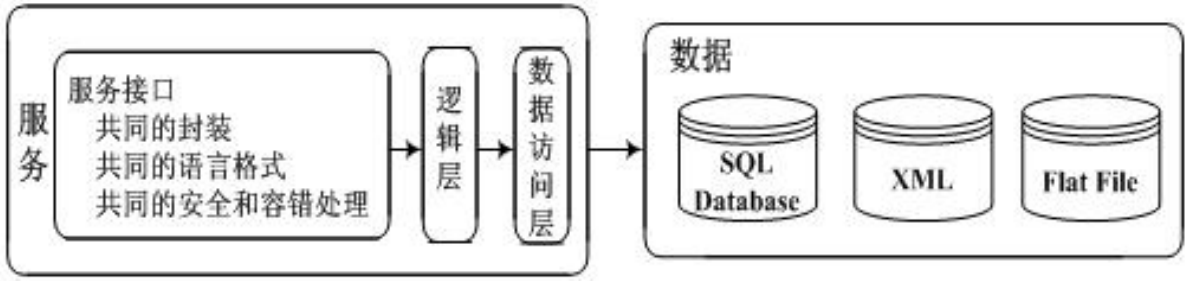

图8-2 单个服务内部结构

- 服务模型的表示层从逻辑层分离出来，中间增加了服务对外的接口层。
- 通过服务接口的标准化描述，使得服务可以提供给在任何异构平台和任何用户接口使用。
- 这允许并支持基于服务的系统成为松散耦合、面向构件和跨技术实现，服务请求者很可能根本不知道服务在哪里运行、是由哪种语言编写的，以及消息的传输路径，而是只需要提出服务请求，然后就会得到所需要的服务。

8.1 SOA概述

- 2、SOA的特征 （1/3）
- SOA是一种粗粒度、松耦合的服务体系结构，其服务之间通过简单、精确定义接口进行通讯，不涉及底层编程接口和通讯模型。这种模型具有下面几个特征：
- **松散耦合**：SOA是松散耦合构件服务，这一点区别于大多数其它的构件体系结构。
  - 松散耦合旨在：将服务使用者和服务提供者，在服务实现和客户如何使用服务方面，隔离开来。
  - 服务提供者和服务使用者间松散耦合背后的关键点是：服务接口作为与服务实现分离的实体而存在。这是服务实现能够在完全不影响服务使用者的情况下进行修改。
  - 大多数松散耦合方法都依靠基于服务接口的消息，基于消息的接口能够兼容多种传输方式（如HTTP、TCP/IP和MOM等），基于消息的接口可以采用同步或异步协议实现。

8.1 SOA概述

- 2、SOA的特征 （1/3）
- **粗粒度服务：**服务粒度（service granularity, coarse-granularity）指的是服务所公开功能的范围，
  - 一般分为细粒度和粗粒度，其中，细粒度服务是那些能够提供少量业务流程可用性的服务。粗粒度服务是那些能够提供高层业务逻辑的可用性服务。
  - 选择正确的抽象级别是SOA建模的一个关键问题。
  - 设计中应该在不损失或损坏相关性、一致性和完整性的情况下，尽可能地进行粗粒度建模。
  - 通过一组有效设计和组合的粗粒度服务，业务专家能够有效地组合出新的业务流程和应用程序。

8.1 SOA概述

- 2、SOA的特征 （3/3）
- **标准化接口**：SOA通过服务接口的标准化描述，从而使得该服务可以提供给在任何异构平台和任何用户接口中使用。
- 这一描述指明了与服务交互需要了解的全部细节，包括：消息格式、传输协议和位置。
- 该接口隐藏了实现服务的细节，允许独立于实现服务基于的硬件或软件平台和编写服务所用的编程语言，使用服务。

8.1 SOA概述

- 3、服务构件与传统构件 （1/2）
- 服务构件体系结构（Service Component Architecture，SCA）是基于SOA的思想，描述服务之间组合和协作的规范，它描述用于使用SOA，构建应用程序和系统的模型。
- 它可简化使用 SOA进行的应用程序开发和实现工作。
- SCA 提供了构建粗粒度构件的机制，这些粗粒度构件由中、细粒度构件组装而成。
- SCA服务构件与传统构件的主要区别在于，服务构件往往是粗粒度的，而传统构件以中粒度居多。

8.1 SOA概述

- 3、服务构件与传统构件 （1/2）
- SCA将传统中间件编程，从业务逻辑分离出来，从而使程序员免受其复杂性的困扰。它允许开发人员集中精力编写业务逻辑，而不必将大量的时间花费在更为底层的实现技术和系统技术上。
- 服务构件的接口描述，使用标准的服务接口描述语言（面向因特网、基于XML语言），而传统构件接口的自描述性及标准化都不够充分。
- 服务构件的实现与语言是无关的，而传统构件常绑定某种特定的语言。
- 服务构件可以通过容器提供QoS的服务，而传统构件则完全由程序代码直接控制。

8.1 SOA概述

- 4、SOA设计原则 （1/3）
- 在SOA体系结构中，继承了来自对象和构件设计的各种原则。例如，封装和自我包含等。
- 那些保证构造单元的灵活性、松散耦合和复用能力的设计原则，对SOA来讲，同样是非常重要的。
- 关于服务，一些常见的设计原则如下：
- **明确定义的接口。**服务请求者依赖于服务规约，来调用服务。
  - 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，减少请求者的不适当使用；
  - 不要让请求者看到服务内部的私有数据。

8.1 SOA概述

- 4、SOA设计原则 （2/3）
- **自包含和模块化。**服务封装了那些在业务上稳定、重复出现的活动和构件，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。
- **粗粒度。**服务数量不应该太多，依靠消息交互，而不是远程过程调用。通常消息量比较大，但是服务之间的交互频度较低。
- **松耦合。**服务请求者可见的是服务的接口，其位置、实现技术、当前状态和私有数据等，对服务请求者而言，是不可见的。

8.1 SOA概述

- 4、SOA设计原则 （3/3）
- **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；
  - 也可以是与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。

8.3 SOA的关键技术

- SOA是一种全新的体系结构，为了支持其各种特性，相关的技术规范不断被推出。
- 服务要以一种可互操作的方式执行发布、发现和绑定这三个操作，必须有一个包含每一层标准的服务栈。
- 因此，整个SOA的技术系列被称之为服务栈，如表8-1所示。

表8-1 服务栈

发现服务层	UDDI、DISCO
描述服务层	WSDL、XML Schema
消息格式层	SOAP、REST
编码格式层	XML
传输协议层	HTTP、TCP/IP、SMTP 等

8.3 SOA的关键技术

- 1、发现服务层
- 发现服务层主要用来帮助客户端应用程序解析远程服务的位置，通过UDDI来实现。
- UDDI规范由Microsoft、IBM和Ariba三家公司在2000年7月提出，它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。
- UDDI规范描述了服务的概念，同时也定义了一种编程接口。
- 通过UDDI提供的标准接口，企业可以发布自己的服务，供其他企业查询和调用，也可以查询特定服务的的描述信息，并动态绑定到该服务上。
- 通过UDDI，Web服务可以真正实现信息的“一次注册到处访问”。

8.3 SOA的关键技术

- 2、描述服务层
- 描述层为客户端应用程序，提供正确地与远程服务交互的描述信息，主要通过WSDL来实现。
- 与UDDI一样，WSDL也是由Microsoft、IBM和Ariba三家公司在2000年7月提出的。
- WSDL为服务提供者提供以XML格式描述服务请求的标准格式，将网络服务描述为能够进行消息交换的通信端点集合，以表达一个服务能做什么，它的位置在哪，如何调用它等信息。

8.3 SOA的关键技术

- 3、消息格式层
- 消息格式层主要用来保证客户端应用程序和服务器端在格式设置上保持一致，一般通过SOAP协议来实现。
- SOAP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。
- SOAP包括三个部分：
  - 定义了描述消息和如何处理消息的框架的封装（SOAP封装）
  - 表达应用程序定义的数据类型实例的编码规则（SOAP编码规则）
  - 描述远程过程调用和应答的的协议（SOAPRPC表示）

8.3 SOA的关键技术

- 4、编码格式层
- 编码格式层主要为客户端和服务器之间提供一个标准的、独立于平台的数据交换编码格式，一般通过XML来实现。
- XML是一种元语言，可以用来定义和描述结构化数据。
- XML使用基于文本的、利用标准字符集的编码方案，从而避开了二进制编码的平台不兼容问题。
- XML有很多优点，包括跨平台支持，公用类型系统和对行业标准字符集的支持，它是服务得以实现的语言基础。服务的其它协议规范都是以XML形式来描述和表达的。

8.3 SOA的关键技术

- 5、传输协议层
- 传输协议层主要为客户端和服务器之间提供两者交互的网络通信协议，一般通过HTTP（Hypertext Transfer Protocol，超文本协议）和SMTP（Simple Mail Transport Protocol，简单邮件传输协议）来实现。
- HTTP是一个在Internet上广泛使用的协议，为服务部件通过Internet交互奠定了协议基础，并具有穿透防火墙的良好特性。
- SMTP则适合于异步通信，如果服务中断，SMTP可以自动进行重试。

8.4 SOA的实现方法

- SOA只是一种概念和思想，需要借助于具体的技术和方法来实现它。
- SOA是一种计算模型，用以支持实现分布式应用。CORBA、DCOM和EJB等都属于这种解决方式。事实上，它们就是SOA的前驱和基础。
- 从逻辑上和高层抽象来看，目前，实现SOA的方法也比较多，其中主流方式有Web Service、企业服务总线和服务注册表。

8.4 SOA的实现方法

- 1、Web Service
- 在Web Service（Web服务）的解决方案中，一共有三种工作角色，其中服务提供者和服务请求者是必须的，服务注册中心是一个可选的角色。它们之间的交互和操作构成了SOA的一种实现体系结构，如图8-4所示。

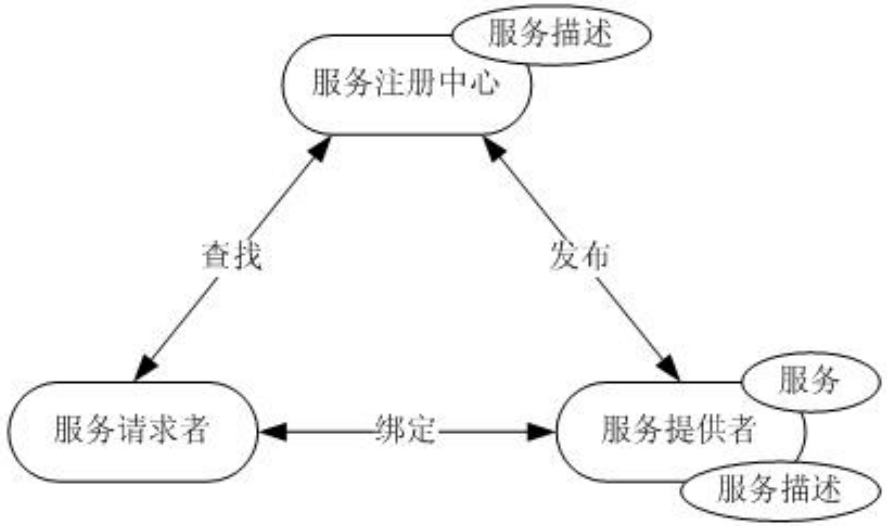

图8-4 Web Service模型

8.4 SOA的实现方法

- （1）服务提供者。
- 服务提供者是服务的所有者，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。

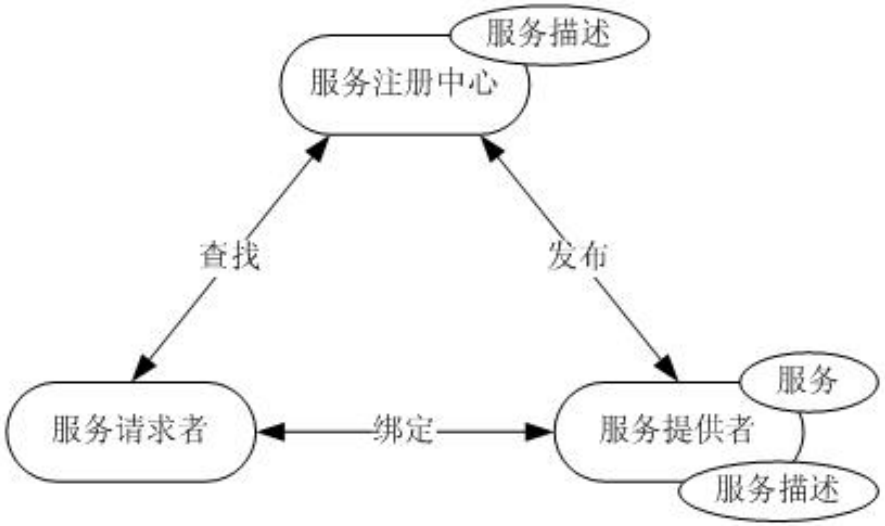

图8-4 Web Service模型

8.4 SOA的实现方法

- （2）服务请求者。
- 服务请求者是服务的使用者，虽然服务面向的是程序，但程序的最终使用者仍然是用户。
- 从体系结构的角度看，服务请求者是查找、绑定并调用服务，或与服务进行交互的应用程序。
- 服务请求者角色可以由浏览器来担当，由人或程序（例如，另外一个服务）来控制。

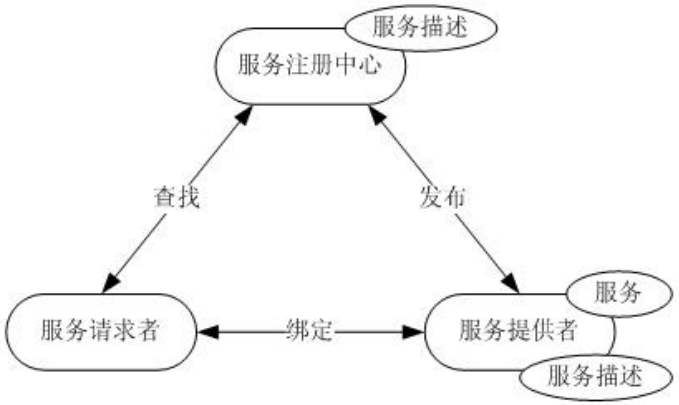

图8-4 Web Service模型

8.4 SOA的实现方法

- （3）服务注册中心。
- 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找他们需要的服务。
- 在某些情况下，服务注册中心是整个模型中的可选角色。
  - 例如，如果使用静态绑定的服务，服务提供者则可以把描述直接发送给服务请求者。

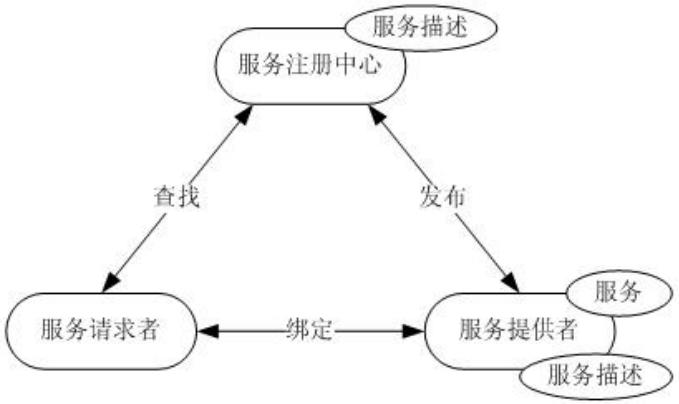

图8-4 Web Service模型

8.4 SOA的实现方法

- Web Service模型中的操作包括：发布、查找和绑定，这些操作可以单次或反复出现。
- （1）发布。为了使用户能够访问服务，服务提供者需要发布服务描述，以便服务请求者可以查找它。
- （2）查找。在查找操作中，服务请求者直接检索服务描述，或在服务注册中心，查询所要求的服务类型。对服务请求者而言，可能会在生命周期的两个不同阶段中涉及到查找操作
  - 在设计阶段，为了程序开发而查找服务的接口描述
  - 在运行阶段，为了调用而查找服务的位置描述。

8.4 SOA的实现方法

- （3）绑定。在绑定操作中，服务请求者使用服务描述中的绑定细节来定位、联系并调用服务，从而在运行时与服务进行交互。
- 绑定可以分为动态绑定和静态绑定。
  - 在动态绑定中，服务请求者通过服务注册中心查找服务描述，并动态地与服务交互。
  - 在静态绑定中，服务请求者已经与服务提供者达成默契，通过本地文件或其他方式直接与服务，进行绑定。

8.4 SOA的实现方法

- 在采用Web Service作为SOA的实现技术时，应用系统大致可以分为六个层次：
- （1）底层传输层。底层传输层主要负责消息的传输机制，HTTP、JMS（Java Messaging Service，Java消息服务）和SMTP，都可以作为服务的消息传输协议，其中HTTP使用最广。
- （2）服务通信协议层。服务通信协议层的主要功能是：描述并定义服务之间进行消息传递所需的技术标准，常用的标准是SOAP和REST协议。
- （3）服务描述层。服务描述层主要以一种统一的方式，描述服务的接口与消息交换方式，相关的标准是WSDL。

8.4 SOA的实现方法

- 在采用Web Service作为SOA的实现技术时，应用系统大致可以分为六个层次：
- （4）服务层。服务层的主要功能是将新服务或遗留系统进行包装，并使之能够通过发布的WSDL接口描述，被服务请求者进行定位和调用。
- （5）业务流程层。业务流程层的主要功能是支持服务发现，服务调用和点到点的服务调用，并将业务流程从服务的底层调用，抽象出来。
- （6）服务注册层。服务注册层的主要功能是：使服务提供者能够通过WSDL，发布服务定义，并支持服务请求者查找所需的服务信息。

8.4 SOA的实现方法

- 2、服务注册表 （1/4）
- 服务注册表（service registry）虽然也具有运行时的功能，但主要在SOA设计时使用。
- 它提供一个策略执行点（Policy Enforcement Point，PEP），在这个点上，服务可以在SOA中注册，从而可以被发现和使用。
- 服务注册表可以包括有关服务和相关构件的配置、依从性和约束文件。
- 概括而言，任何帮助服务注册、发现和查找服务合约、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。
- 大多数商用服务注册产品支持服务注册、服务位置和服务绑定功能。

8.4 SOA的实现方法

- 2、服务注册表 (2/4)

- （1）服务注册。服务注册是指服务提供者，向服务注册表，发布服务的功能（服务合约），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。
- 使用服务注册表实现SOA时，要限制哪些新服务可以向注册表发布、由谁发布以及谁批准和根据什么条件批准等，以便使服务能够有序的注册。

8.4 SOA的实现方法

- 2、服务注册表 （3/4）

- （2）服务定位。服务定位是指：帮助服务使用者，查询已注册的服务，寻找符合要求的服务。
- 这种查找主要是通过检索服务合约，来实现的。
- 在使用服务注册表实现SOA时，需要规定哪些用户可以访问服务注册表，以及哪些服务属性可以通过服务注册表进行公开等，以便服务能得到有效的、经过授权的使用。

8.4 SOA的实现方法

- 2、服务注册表 （4/4）

- （3）服务绑定。服务使用者利用查找到的服务合约，来开发代码。
- 开发的代码将与注册的服务进行绑定，调用注册的服务，以及与它们实现互动。
- 可以利用集成的开发环境，自动将新开发的服务，与不同的新协议、方案和程序间通信所需的其他接口，绑定在一起。

8.4 SOA的实现方法

- 3、企业服务总线ESB
- ESB的概念是从SOA发展而来的，它是一种为连接服务，提供的标准化的通信基础结构。它基于开放的标准，为应用提供了一个可靠的、可度量的和高度安全的环境，并可帮助企业对业务流程进行设计和模拟，对每个业务流程实施控制和跟踪、分析并改进流程和性能。
- ESB是由中间件技术实现并支持SOA的一组基础体系结构，是传统中间件技术与XML、Web Service等技术结合的产物，是在整个企业集成体系结构下的面向服务的企业应用集成机制。

8.4 SOA的实现方法

- 3、企业服务总线ESB
- 在一个复杂的企业计算环境中，如果服务提供者和服务请求者之间采用直接的端到端的交互，那么随着企业信息系统的增加和复杂度的提高，系统之间的关联会逐渐变得非常复杂，形成一个网状结构，这将带来昂贵的系统维护费用，同时也使得IT基础设施的复用变得困难重重。
- ESB提供了一种基础设施，消除了服务请求者与服务提供者之间的直接连接，使得服务请求者与服务提供者之间进一步解耦。

8.4 SOA的实现方法

- 企业服务总线ESB具有以下功能：　（1/2）
- （1）支持异构环境中的服务、消息和基于事件的交互，并且具有适当的服务级别的可管理性。
- （2）通过使用ESB，可以在几乎不更改代码的情况下，以一种无缝的非侵入方式使现有系统具有全新的服务接口，并能够在部署环境中支持任何标准。
- （3）充当缓冲器的ESB（负责在诸多服务之间转换业务逻辑和数据格式）与服务逻辑相分离，从而使不同的系统可以同时使用同一个服务，用不着在系统或数据发生变化时，改动服务代码。

8.4 SOA的实现方法

- 企业服务总线ESB具有以下功能：　（1/2）
- （4）在更高的层次，ESB还提供诸如服务代理和协议转换等功能。允许在多种形式下通过像HTTP、SOAP和JMS总线的多种传输方式，主要是以网络服务的形式，为发表、注册、发现和使用企业服务或界面提供基础设施。
- （5）提供可配置的消息转换翻译机制和基于消息内容的消息路由服务，传输消息到不同的目的地。
- （6）提供安全和拥有者机制，以保证消息和服务使用的认证、授权和完整性。

8.4 SOA的实现方法

- 在企业应用集成方面，与现存专有的集成方案相比，ESB具有以下优势：　（1/2）
- （1）扩展的、基于标准的连接。
  - ESB形成一个基于标准的信息骨架，使得在系统内部和整个价值链中可以容易地进行异步或同步数据交换。
  - ESB通过使用XML、SOAP和其他标准，提供了更强大的系统连接性。
- （2）灵活的、服务导向的应用组合。基于SOA，ESB使复杂的分布式系统（包括跨多个应用、系统和防火墙的集成方案）能够由以前开发测试过的服务组合而成，使系统具有高度可扩展性。

8.4 SOA的实现方法

- 在企业应用集成方面，与现存专有的集成方案相比，ESB具有以下优势：　（1/2）
- （3）提高复用率，降低成本。按照SOA方法构建应用，提高了复用率，简化了维护工作，进而减少了系统总体成本。
- （4）减少市场反应时间，提高生产率。ESB通过构件和服务复用，按照SOA的思想简化应用组合，基于标准的通信、转换和连接来实现这些优点。