中间件技术
Middleware Technology

第十章　Web 服务

赖永炫　博士/教授
厦门大学　软件工程系

大 纲

- Web服务的历史
- Web服务的概念
- 主要概念介绍
  - SOAP
  - WSDL
  - UDDI
- 基于SOAP的Web服务
- RestFul Web服务
- Web Service安全
- Web 服务和SOA
- 小结

Web服务的历史

- DCE/RPC采用典型的客户端/服务器结构。客户端调用服务器的方法，参数从客户端传递给服务器，返回值从服务器端传回。
- DCE框架采用"平台和语言中立"的原则，但实践过程中主要偏向类C语言为主。

DCE/RPC

- DCE/RPC提供了一些工具以屏蔽底层通信等方面的复杂性。比如，提供了接口定义语言IDL，作为提供调用和服务的合同。
- 一个典型的例子如下：

```
/* echo.idl */
[uuid(2d6ead46-05e3-11ca-7dd1-426909beabcd), version(1.0)]
interface echo {
    const long int ECHO_SIZE = 512;
    void echo(
        [in]              handle_t h,
        [in,  string]     idl_char from_client[ ],
        [out, string]     idl_char from_server[ECHO_SIZE]
    );
}
```

IDL接口

- 文档定义了名为echo的IDL接口，确定用机器生成的UUID（通用唯一标识符），声明一个同名方法echo。
- IDL的语法基本上是C语法，其可以看作是WSDL（Web服务描述语言，Web Services Description Language）文档的前身。

轻量级的RPC系统XML-RPC

- XML-RPC支持一些基本的数据类型和一些简单的命令，采用XML格式的文档，并基于HTTP协议进行序列化的传输。
  - XML-RPC的消息载荷是文本的，而DCE RPC的消息是二进制数据
  - XML-RPC使用HTTP协议进行传输，用户只需1个标准的HTTP库，以及XML相关的处理库即可。

- XML-RPC也是采用请求-反馈模式，一个典型的调用和消息如下：

```
<?xml version="1.0">
<methodCall>
   <methodName>fib<methodName>
   <params>
     <param><value><i4>11</i4></value></param>
   </params>
</methodCall>
```

WSDL是XML格式

- WSDL是XML格式的用于描述服务端所提供服务的文档。
- WSDL描述了服务端提供的服务，提供的调用方法，以及调用时所要遵循的格式，比如调用参数和返回值的格式等等。
- WSDL很像COM编程里的IDL(Interface Description Language)，是服务器与客户端之间的契约，双方必须按契约严格行事才能实现功能。

大 纲

- Web服务的历史
- Web服务的概念
- 主要概念介绍
  - SOAP
  - WSDL
  - UDDI
- 基于SOAP的Web服务
- RestFul Web服务
- Web Service安全
- Web 服务和SOA
- 小结

Web服务

- Web服务（Web service）是一个平台独立的，低耦合的，自包含的、基于可编程的web的应用程序，可使用开放的XML标准来描述、发布、发现、协调和配置这些应用程序，用于开发分布式的互操作的应用程序。
- Web service平台是一套标准，它定义了应用程序如何在Web上实现互操作性。通过Web服务技术，运行在不同机器上的不同应用可相互交换数据或进行集成，而无须借助附加的、专门的第三方软件或硬件。

Web服务

- Web Service是自描述、自包含的可用网络模块。基于常规的产业标准以及已有的技术，诸如XML和HTTP，非常容易部署。
- 依据Web Service规范实施的应用之间都可以相互交换数据，无论它们所使用的语言、平台或内部协议是什么。
- 常见的两种Web Service处理方式包含
  - ✓ 1）基于WSDL/SOAP的方式；
  - ✓ 2）基于REST（表述性状态转换）风格的方式

什么是XML

- XML（Extensible Markup Language）即可扩展标记语言，它与HTML一样，都是SGML (Standard Generalized Markup Language,标准通用标记语言)。Xml是Internet环境中跨平台的，依赖于内容的技术，是当前处理结构化文档信息的有力工具。扩展标记语言XML是一种简单的数据存储语言，使用一系列简单的标记描述数据。

- XML与HTML的设计区别是：XML是用来存储数据的，重在数据本身。而HTML是用来定义数据的，重在数据的显示模式。

W3C's Definition

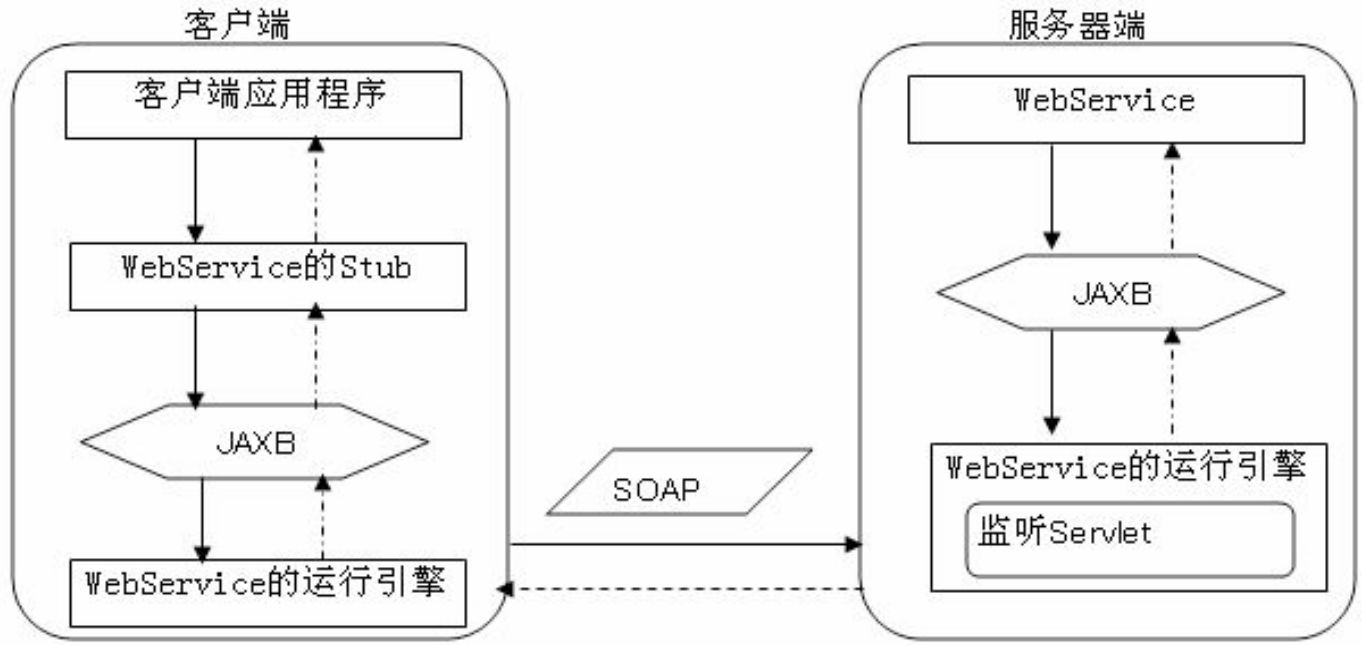
- A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.

Web服务框架

- Web service的工作原理分为两步：首先是在服务器上创造服务并将其在一台"目录服务器"上注册；然后客户端查找并调用该服务。
  - ✓ 1）服务器按规格生成服务的类和方法（比如java bean），使其能够接受和响应SOAP消息
  - ✓ 2）撰写WSDL文件用于描述此Web Services，服务器向"目录服务器"注册自己的这些方法。
  - ✓ 3）将此WSDL发布到目录服务器上，对外发布一个可调用的方法（服务）目录；
  - ✓ 4）客户端向"目录服务器"查找这些服务，获得这些服务的地址之后再去请求服务。

Web Service框架

- 服务器，客户端和目录服务器是三个必需的角色，客户端和服务器都需要知道目录服务器的地址。



大 纲

- Web服务的历史
- Web服务的概念
- 主要概念介绍
  - SOAP
  - WSDL
  - UDDI
- 基于SOAP的Web服务
- RestFul Web服务
- Web Service安全
- Web 服务和SOA
- 小结

SOAP

- Simple Object Access Protocol
- 简单对象访问协议(SOAP)提供了标准的远程过程调用(RPC)方法来调用Web service。
- SOAP规范定义了SOAP消息的格式，以及怎样通过HTTP协议来使用SOAP。SOAP也是基于XML和XSD的，XML是SOAP的数据编码方式。客户端和服务端之间的方法调用请求和结果返回值都放在这些消息里。

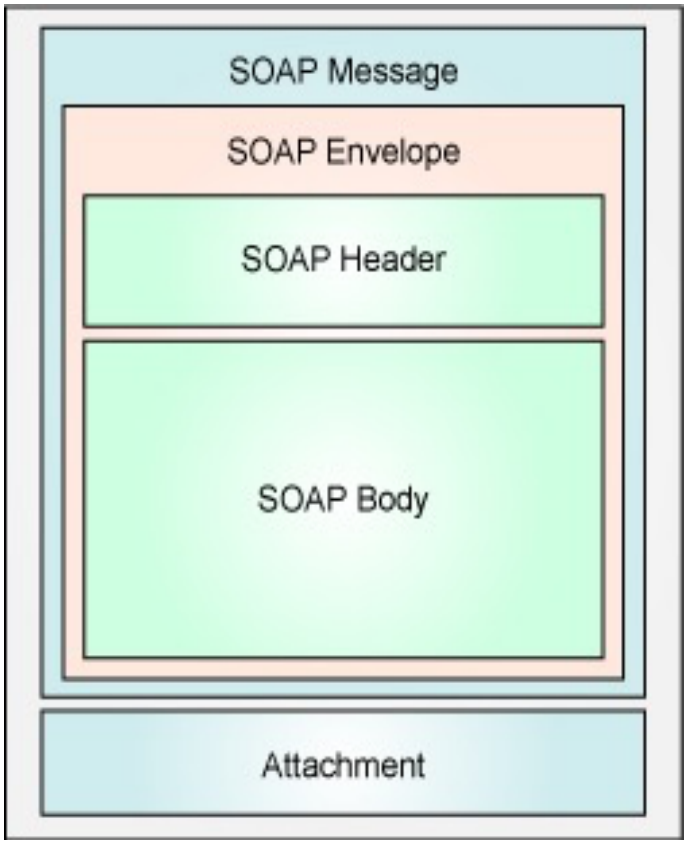
SOAP—概述

- 简单对象访问协议
  - ➢其中"O"—对象的含义逐步淡化
  - ➢重心从对象转移到通用的 XML 消息处理框架上
- SOAP是一种轻量级协议，用于在分散型、分布式环境中交换结构化信息。SOAP 利用 XML 技术定义一种可扩展的消息处理框架，它提供了一种可通过多种底层协议进行交换的消息结构。这种框架的设计思想是要独立于任何一种特定的编程模型和其他特定实现的语义
- 在协议栈中，SOAP XML 位于用来发送消息的传输协议之上，而位于特定领域的 XML 文档之下

SOAP体系结构

- SOAP envelop，描述SOAP消息格式
- SOAP encoding rules,定义了一组对数据类型进行编码的规则
  - ➢定义应用程序中需要使用的数据类型，SOAP1.2将其作为一个可选项处理；
  - ➢对于编程人员来说，没有这个编码更加灵活；
- SOAP RPC，定义了SOAP消息如何执行远程调用
- SOAP binding，SOAP绑定，定义了一个使用底层传输协议来完成在结点间交换SOAP信封的约定

SOAP消息

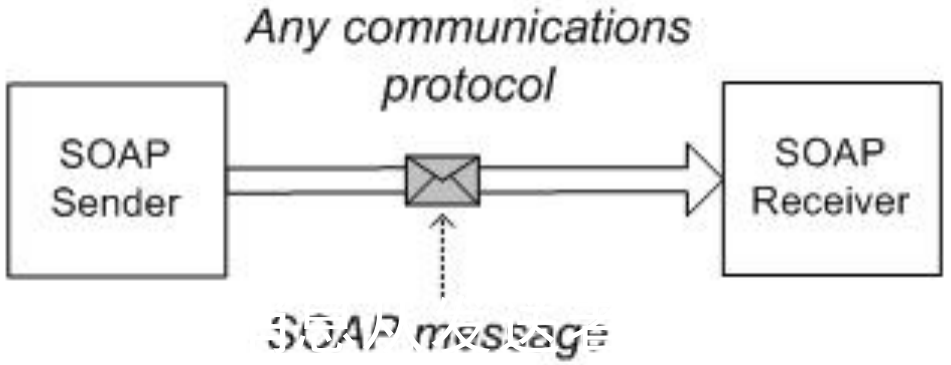


- SOAP <Envelope> 元素是 SOAP 消息的根元素，包含一个可选的 SOAP Header 和一个必需的 SOAP Body 元素
- 可选的并且可扩展的 <Header> 元素，用于描述元数据（metadata），比如安全性、事务处理和会话状态信息
- 必需的 <Body> 元素，包含发送者的 XML 文档（文档形式和RPC形式）
- 在接收者（SOAP中介）返回的响应信息中，可能包含<faults> 的元素，用来描述在阅读 SOAP 消息时遇到的任何异常情况
- W3C Note 指定了一种在 SOAP 消息中嵌入和描述 附件（*attachment*，格式不限）的方式

SOAP编码

- 描述在SOAP消息中如何表示特定种类的数据的规则（可选）
  - 不是强制性的，开发人员可以选择其它任意一种编码方法；
  - SOAP Header或Body元素可以具有SOAP Encoding Style属性，该属性包含一个映射到编码规则的URI，它规定了对SOAP消息内部各数据元素的数据类型进行定义的规则

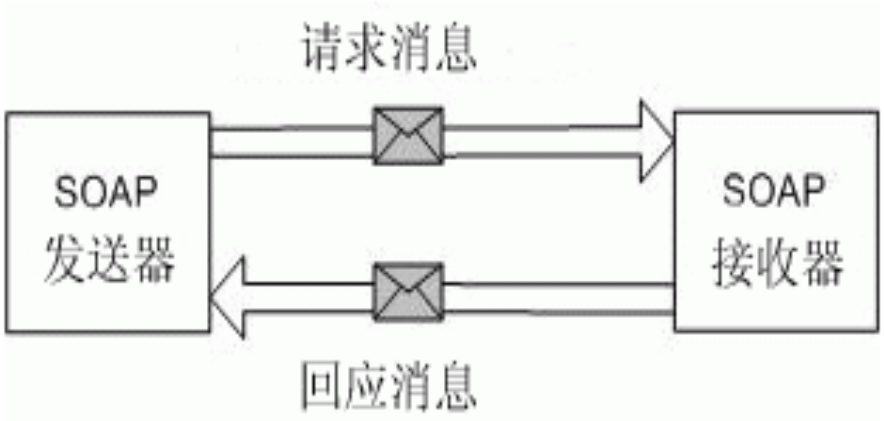
SOAP—消息交换模型



- 发送者，创建和发送 SOAP 消息给最终的 SOAP 接收者
- 最终 SOAP 接收者，SOAP 发送者为发送的 SOAP 消息规定的最终目的地（不是中介），在这个模型中，接收者不反馈信响应信息，信息的发送是单向的

- 可扩展，该协议是简单的，缺少分布式系统的许多特征，如安全性、路由及可靠性等，允许分层扩展
- 可通过多种底层网络协议使用及 独立于编程模型

SOAP—消息交换模型（Cont）



- 请求/响应模型
  - 接收方向发送方发送响应消息

SOAP—消息交换模型（Cont）

- 增加可选的“SOAP中介（intermediary）”

用来在 SOAP 发送者和最终 SOAP 接收者之间截取 SOAP 消息。在将消息发送给最终 SOAP 目的地之前，截取 SOAP 消息的任意中介都可以分析它，以执行过滤、记录和缓存等操作。SOAP 中介可以看作是发送者和接收者。

XML和XSD

- 可扩展的标记语言(XML)是Web Service平台中表示数据的基本格式。除了易于建立和易于分析外，XML主要的优点在于它既是平台无关的，又是厂商无关的。
- W3C制定的XML Schema(XSD)定义了一套标准的数据类型，并给出了一种语言来扩展这套数据类型。Web service平台就是用XSD来作为其数据类型系统的。

WSDL

- Web Services Description Language
- WSDL网络服务描述语言是Web Service的描述语言，它包含一系列描述某个web service的定义。
- 包括服务端提供的服务，提供的调用方法，以及调用时所要遵循的格式，比如调用参数和返回值的格式等等。

WSDL简介

- 用一种与平台无关的语言（XML）来描述一个或多个服务。它描述了服务、访问服务的方式以及需要返回的响应的类型（如果有的话）
- 可以私下交换 WSDL 文档，也可以将其发送到 UDDI 注册中心（公共的或者私有的），以允许更广泛的访问
- 基于 XML 的文件格式，用来描述类型（Type）、消息（Message）、操作（Operation）和接口（称为 端口类型（PortTypes））、定位和协议绑定
- 可以用 WSDL 来把 Web 服务描述成一组运行在消息上的端点

WSDL简介

- 消息描述了客户端和服务之间的通信（通过交换的数据类型来描述）
- 操作包括输入和输出消息
- *PortTypes* 包括一组操作。而且，PortTypes 被约束在某些协议上，这称为绑定（*binding*）
- WSDL 是可扩展的，可以与其他类型的网络协议和消息格式一起进行使用

JAXB

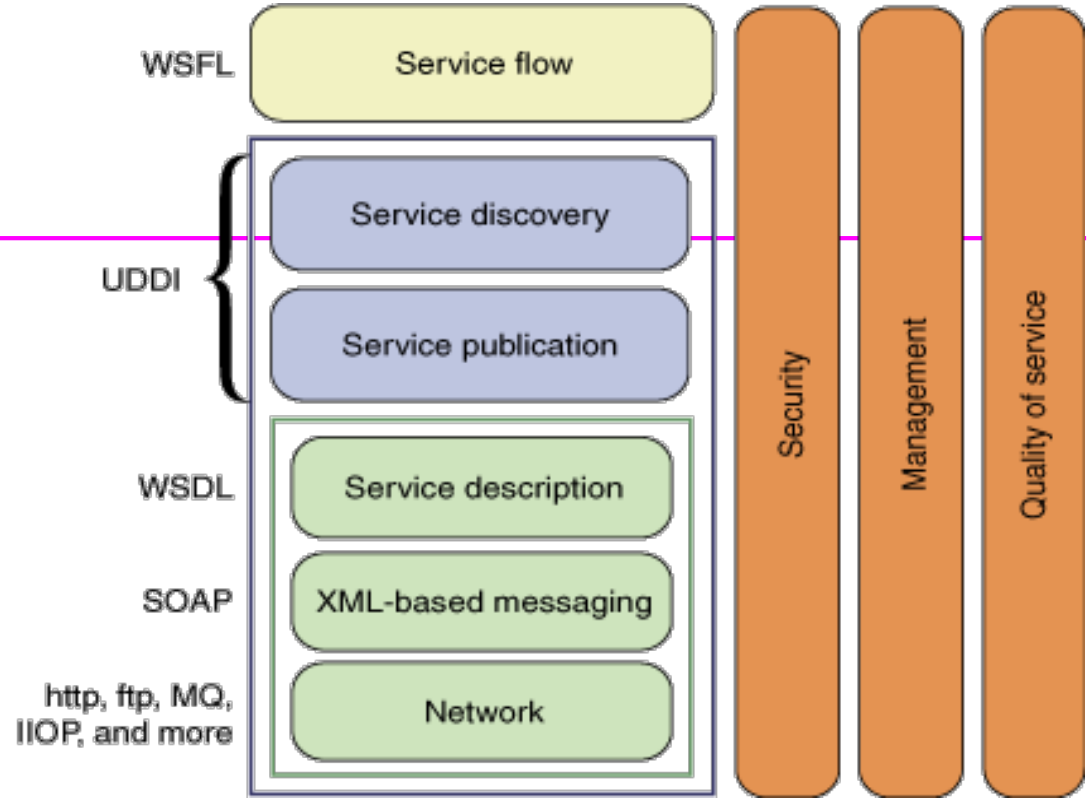
- JAXB（Java Architecture for XML Binding） 是一个业界的标准，是一项可以根据XML Schema产生Java类的技术；也允许将JAVA类映射为XML表示方式。

UDDI（统一描述发现和集成）

- Universal Description, Discovery and Integration

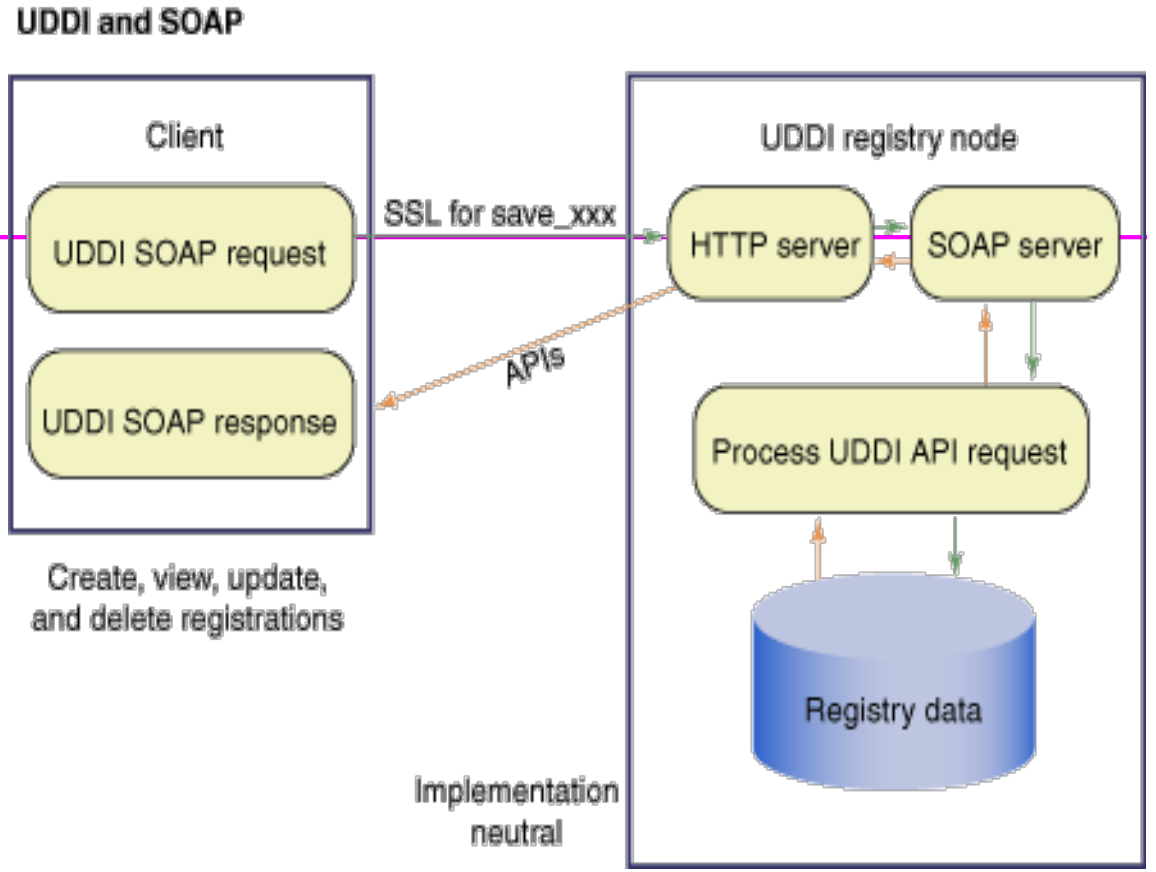
- 提供一种发布和查找服务描述的方法。UDDI数据实体提供对定义业务和服务信息的支持。WSDL 中定义的服务描述信息是UDDI注册中心信息的补充。

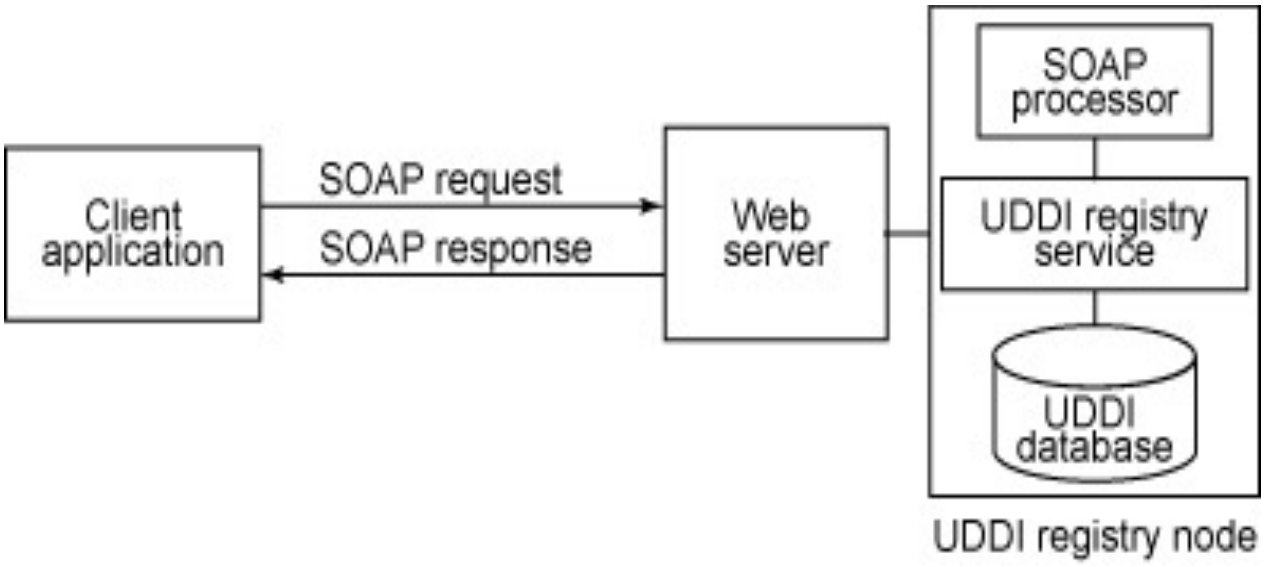
UDDI简介



- 统一描述、发现和集成（Universal Description, Discovery, and Integration）
- 一个提供注册和定位Web服务（商业）的开放框架，既是一个规范，又是若干企业间的伙伴关系

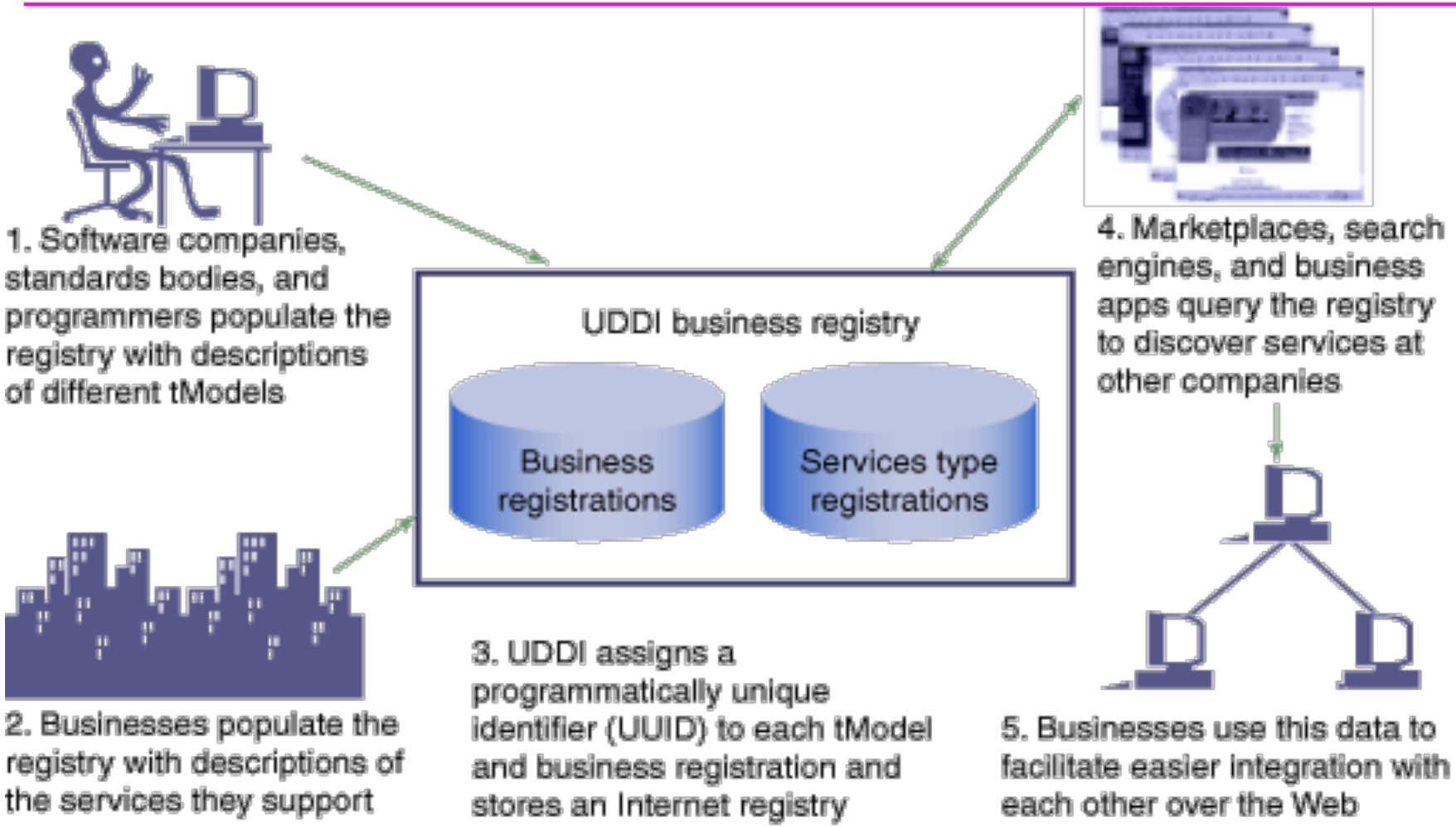
UDDI工作原理





通过 HTTP 从客户机的 SOAP 请求传到注册中心节点，然后再反向传输

UDDI工作原理（Cont.）



如何往UDDI注册中心送入数据，顾客又如何能发现和使用这一信息。UDDI注册中心建立在顾客提供的数据的基础之上。

WSML

- Web Services Meta Language

- 用于描述WSDL里提供的方法与实现该方法的COM对象之间的映射关系。该文件是Microsoft的实现中特有的，不是SOAP标准的一部分。一般情况下，该文件只在服务端存在。

Web Service的特点

- Web Service的主要目标是跨平台的可互操作性。
- 为了达到这一目标，Web Service完全基于XML（可扩展标记语言）、XSD （XML Schema）等独立于平台、独立于软件供应商的标准，是创建可互操作的、分布式应用程序的新平台。

Web Service的主要优势

- １．异构平台的互通性
  - ✓ 理论上， Web Service 最大的优势是提供了异构平台的无缝街接技术手段。由于不同的用户使用不同的硬件平台，不同的操作平台，不同的软件，不同的协议通信，这就产生了互相通信的需求。
  - ✓ Web Service 使任何两个应用程序，只要能读写ＸＭＬ，那么就能互相通信。

- 2．更广泛的软件复用
  - ✓ 软件的复用技术通过组合已有模块来搭建应用程序，能大幅度提高软件的生产效率和质量。用户只要获得了描述 Web Service 的 WSDL 文件，就可以方便地生成客户端代理，并通过代理访问 Web Service 。
- 3. 普通的通信能力
  - ✓ Web Service 可用基于 XML 的 SOAP 来表示数据和调用请求。并且通过 HTTP 协议传输 XML 格式的数据。

- 4. 迅捷的软件发行方式 SAAS
  - ✓ Web Service 将彻底地改变软件的发行方式。软件供应商可以把软件分解成若干 Web Service 模块构成的系统，直接在 Web 上发布软件。
- 5. 方便的商务集成
  - ✓ 企业通过把业务软件的核心模块以Web Service的形式向其合作伙伴发布，这样既保留了原有的数据和软件，又方便了彼此的联系。

大 纲

- Web服务的历史
- Web服务的概念
- 主要概念介绍
  - SOAP
  - WSDL
  - UDDI
- 基于SOAP的Web服务
- RestFul Web服务
- Web Service安全
- Web 服务和SOA
- 小结

基于SOAP的Web服务

- JAX-WS规范是一组XML web services的JAVA API。
- JAX-WS允许开发者可以选择RPC-oriented或者message-oriented来实现自己的web services。
  - ✓ 在 JAX-WS中，一个远程调用可以转换为一个基于XML的协议例如SOAP。
  - ✓ 开发者不需要编写任何生成和处理SOAP消息的代码，JAX-WS的运行时实现会将这些API的调用转换成为对应的SOAP消息

- 在服务器端，用户只需要通过Java语言定义远程调用所需要实现的接口SEI（service endpoint interface），并提供相关的实现，通过调用JAX-WS的服务发布接口就可以将其发布为WebService接口。
- 在客户端，用户可以通过JAX-WS的API创建一个代理（用本地对象来替代远程的服务）来实现对于远程服务器端的调用。

代码展示Demo

大 纲

- Web服务的历史
- Web服务的概念
- 主要概念介绍
  - SOAP
  - WSDL
  - UDDI
- 基于SOAP的Web服务
- RestFul Web服务
- Web Service安全
- Web 服务和SOA
- 小结

Rest样式的Web服务

- REST (REpresentation State Transfer) 描述了一个架构样式的网络系统，它首次出现在 2000 年 Roy Fielding 的博士论文中。
- REST 指的是一组架构约束条件和原则，满足这些约束条件和原则的应用程序或设计就是 RESTful。

- 具体来说，REST是一种轻量级的Web Service架构风格，其实现和操作明显比SOAP和XML-RPC更为简洁，可以完全通过HTTP协议实现，还可以利用缓存Cache来提高响应速度，性能、效率和易用性上都很有优势。

对资源的定义和连接

- REST中的资源所指的不是数据，而是数据和表现形式的组合。
- 资源标识符就是URI(Uniform Resource Identifier)，不管是图片，Word还是视频文件，甚至只是一种虚拟的服务，也不管你是XML（标准通用标记语言下的一个子集）格式、txt文件格式还是其它文件格式，全部通过URI对资源进行唯一的标识。
  - ✓ “最新访问的10位会员” vs “最活跃的10位会员”

资源的一致命名

- 对资源使用一致的命名规则（naming scheme）最主要的好处就是你不需要提出自己的规则——而是依靠某个已被定义，在全球范围中几乎完美运行，并且能被绝大多数人所理解的规则。
- 使用URI表示链接的优雅之处在于，链接可以指向由不同应用、不同服务器甚至位于另一个大陆上的不同公司提供的资源
  - ✓ 因为URI命名规范是全球标准，构成Web的所有资源都可以互联互通。任何可能的情况下，使用链接可以访问和指向被标识的资源。

标准的操作方法

- 所有的资源都支持同样的接口，一套同样的方法集合。
- REST架构遵循了CRUD原则，CRUD原则对于资源只需要四种行为：
  - ✓ Create（创建）、Read（读取）、Update（更新）和Delete（删除）
- 正好对应HTTP协议提供的GET、POST、PUT、DELETE方法

标准的操作方法

- 可以想象，RESTful HTTP方案中的所有资源都继承自类似于这样的一个类：
- class Resource {
  - ✓ Resource(URI u);
  - ✓ Response get();
  - ✓ Response post(Request r);
  - ✓ Response put(Request r);
  - ✓ Response delete();
- }

无状态通信

- Web服务也应该是无状态的请求。客户端和服务器之间的交互在请求之间是无状态的。
- 从客户端到服务器的每个请求都必须包含理解请求所必需的信息。
- 虽然REST包含无状态性（statelessness）的观念，但这并不是说暴露功能的应用不能有状态。REST要求状态要么被放入资源状态中，要么保存在客户端上。

- 服务器端不能保持除了单次请求之外的，任何与其通信的客户端的通信状态。
- 这样做的最直接的理由就是可伸缩性——
  - ✓ 如果服务器需要保持客户端状态，那么大量的客户端交互会严重影响服务器的内存可用空间（footprint）。
  - ✓ 更为重要的是：无状态约束使服务器的变化对客户端是不可见的。客户端不会察觉到后台的服务器是否已经改变。

REST设计准则

- REST架构是针对Web应用而设计的，其目的是为了降低开发的复杂性，提高系统的可伸缩性。
- REST提出了如下设计准则：
- 所有事物都被抽象为资源，每个资源对应唯一的标识符；
- 通过标准的方法对资源进行操作；
- 所有的操作都是无状态的，对资源的各种操作不会改变资源标识符。

分层的原则

- REST架构还就有分层的原则。
- 组件无法了解它与之交互的中间层以外的组件。
- 通过将系统知识限制在单个层，可以限制整个系统的复杂性，促进了底层的独立性。

RESTful Web 服务的例子

https://spring.io/guides/gs/rest-service/

Rest vs. Soap Web服务

- Http协议作为传输协议现今被广泛利用，但当初的设计者其实是把HTTP协议的作为应用协议来考虑的。
- REST并不是什么协议也不是什么标准，而是将Http协议的设计初衷作了诠释。
- REST从资源的角度来观察整个网络，分布在各处的资源由URI确定，而客户端的应用通过URI来获取资源的表征。

Rest vs. Soap Web服务

- 获得这些表征致使这些应用程序转变了其状态。随着不断获取资源的表征，客户端应用不断地在转变着其状态，所谓表征状态转移（Representational State Transfer）。
- Rest Service具备Web Service的所有特点：平台独立、松耦合、互操作性。而且，Rest更轻量级，更简单。

Rest vs. Soap Web服务

- 从基本原理层次上说，REST 样式和 SOAP 样式 Web Service的区别取决于应用程序是面向资源的还是面向活动的。
  - ✓ 例如，在传统的WebService中，一个获得天气预报的webservice会暴露一个WebMethod：string GetCityWether（string city）。
  - ✓ 而RESTful WebService暴露的不是方法，而是对象（资源），通过Http GET, PUT, POST 或者 DELETE来对请求的资源进行操作。

Rest vs. Soap Web服务

- 可以认为REST架构定义的Web Service实际上定义了一个接口的规范。
- REST Service是Web Service的一种实现，而不是替代。

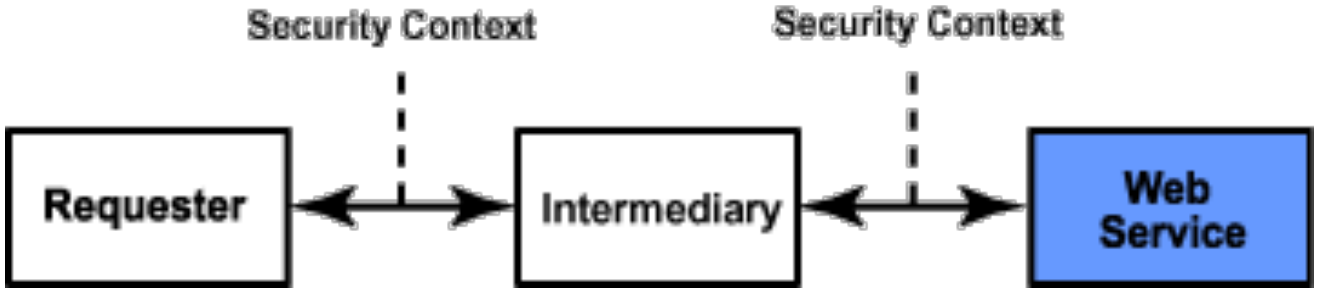
大 纲

- Web服务的历史
- Web服务的概念
- 主要概念介绍
  - SOAP
  - WSDL
  - UDDI
- 基于SOAP的Web服务
- RestFul Web服务
- Web 服务的安全
- Web 服务和SOA
- 小结

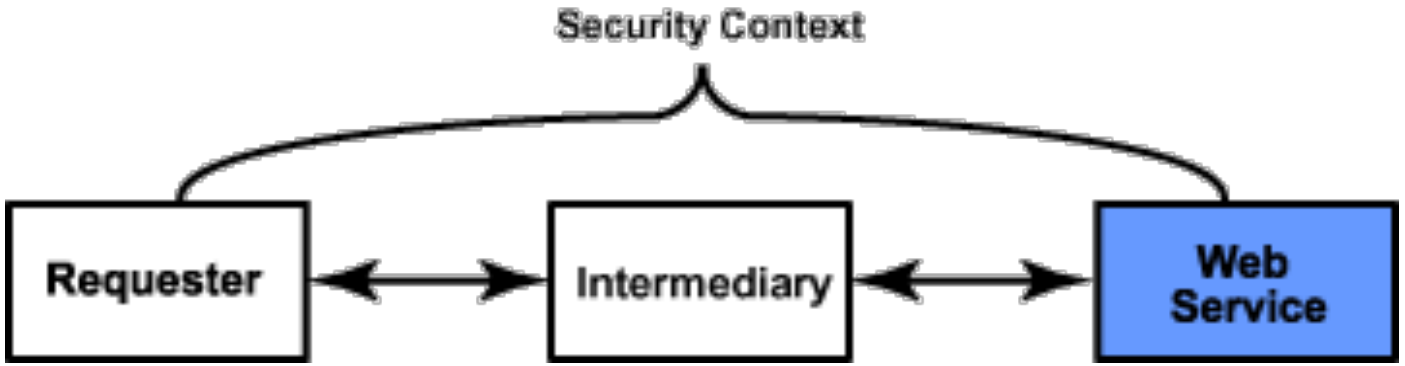
Web服务安全概述

- 保护Web服务安全性目标可以分解为几个子目标
  - 提供用来保护消息完整性和机密性的工具；
  - 提供用来确保服务只对表达策略所需声明的消息中的请求起作用的工具；
- 安全套接字层（SSL）和实际的传输层安全性（Transport Layer Security，TLS）一起被用于为 Web 服务应用程序提供传输级别的安全性
- IPSec是另一个用于传输安全性的网络层标准
- 传输层之外的中介体接收并转发数据时，数据的完整性和任何随数据流动的安全性信息都可能会丢失
  - Web服务应用程序将许多动态的系统组织在一起；
  - 消息传递依赖中介，传输层安全措施不能保证其安全性；

点对点配置，中介处的安全性得不到保证



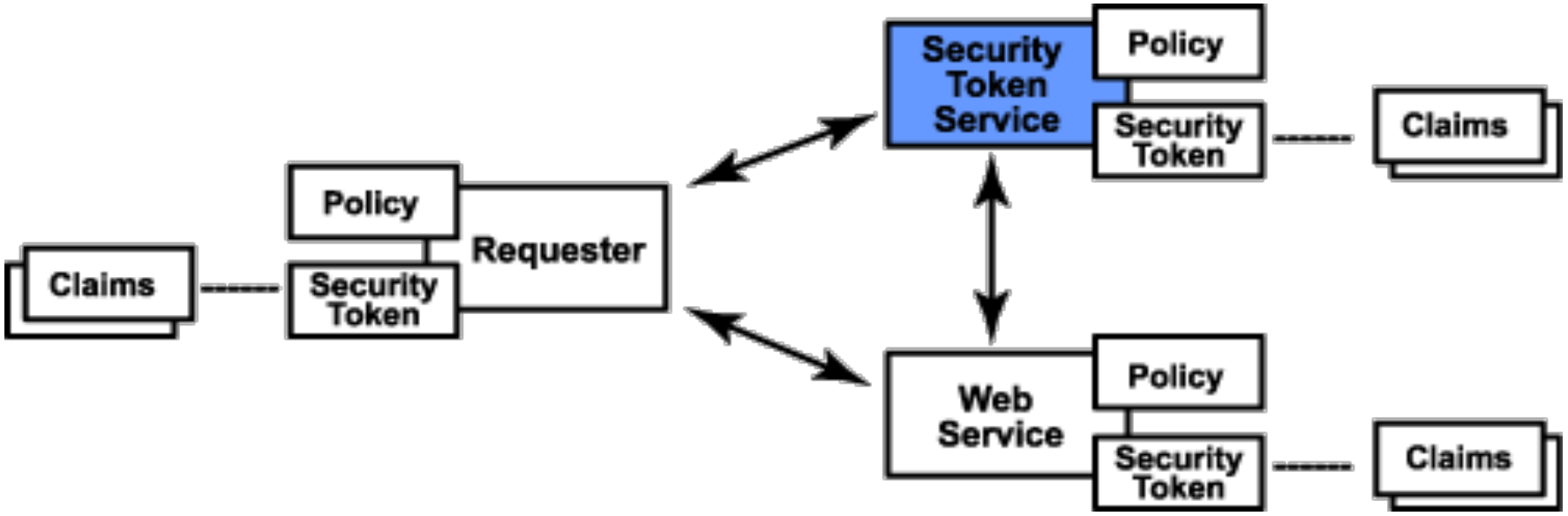
全面的 Web 服务安全性体系架构中所需要的是一个提供端到端安全性的机制－Web服务安全性模型



Web服务安全概述（Cont.）

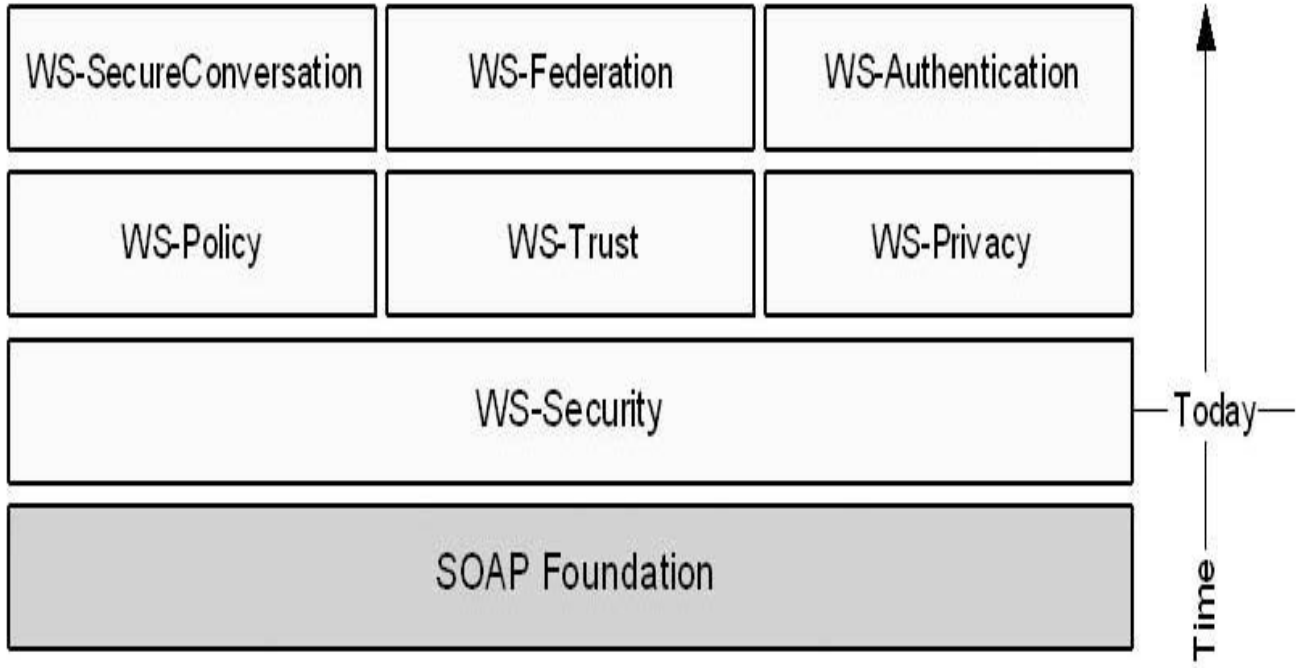
- Web 服务安全性模型使我们能够通过一个过程达到保证安全性的目的，在这个过程中：
  - Web 服务可以要求进来的消息证明一组 声明（例如，名称、密钥、许可、性能等等）
  - 请求者可以通过把 安全性令牌与消息关联起来发送带必需声明的证明的消息
  - 如果请求者没有必需的声明，那么请求者或它们的代表可以通过与其它 Web 服务联系设法获得必需的声明，其它的 Web 服务指的是 安全性令牌服务（security token service），可以接下来要求它们自己的一组声明，安全性令牌服务通过签发安全性令牌代理不同信任域之间的信任

安全性令牌服务模型



这个常规的消息传递模型 — 声明、策略和安全性令牌 — 包含并支持几个更特殊的模型，比如基于身份的安全性、访问控制列表和基于性能的安全性。它允许使用现有的技术如 X.509 公用密钥证书、Kerberos 共享秘密的票据甚至密码摘要

Web服务安全



- Web服务的安全性构建在一整套的安全协议的基础上；
- WS－Security是所有安全的基础
  - 描述如何向 SOAP 消息附加签名和加密报头；
  - 描述如何向消息附加安全性令牌（包括二进制安全性令牌，如 X.509 证书和 Kerberos 票据）
- 另一个问题是如何把Web服务安全性与现有的安全性模型关联起来

大 纲

- Web服务的历史
- Web服务的概念
- 主要概念介绍
  - SOAP
  - WSDL
  - UDDI
- 基于SOAP的Web服务
- RestFul Web服务
- Web Service安全
- Web 服务和SOA
- 小结

SOA的概念

- 面向服务的体系结构SOA（Service-Oriented Architecture）是一个组件模型。
- 它通过定义良好的接口和契约，将应用程序的不同功能单元（称为服务）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。
- 这使得构建在各种各样的系统中的服务可以使用一种统一和通用的方式进行交互。

SOA的概念

- 通过SOA，松散耦合的、粗粒度的应用组件可根据需求进行分布式的部署、组合和使用。
  - ✓ 应用和服务之间通过简单、精确定义接口进行通讯，不涉及底层编程接口和通讯模型。
- SOA可以看作是B/S模型、XML、Web Service技术之后的自然延伸
  - ✓ 帮助软件工程师们站在一个新的高度理解企业级架构中的各种组件的开发、部署形式，它将帮助企业系统架构者以更迅速、更可靠、更具重用性架构整个业务系统。
  - ✓ 较之以往，以SOA架构的系统能够更加从容地面对业务的急剧变化。

SOA 与Web服务

- 从本质上来说，SOA是一种架构模式，而Web服务是利用一组标准实现的服务。Web服务是实现SOA的方式之一。就目前而言，Web Service是最适合实现SOA的一些技术的集合，但SOA和Web Service不能混为一谈。

附：SOA and Web Service的区别

- WebServices are self describing services that will perform well defined tasks and can be accessed through the web.
- Service Oriented Architecture (SOA) is (roughly) an architecture paradigm that focuses on building systems through the use of different Web Services, integrating them together to make up the whole system.

SOA and Web Service?

- Service-Oriented Architecture is an IT business system design methodology that is focused on reusability, breaking down silos, and enabling rapid and on-going optimization of business processes.
- An SOA can be implemented using any number of technologies. However, in practice the Web services standards and technologies (XML, SOAP, WSDL, WS-Security, WS-Policy, etc.) are at the heart of forming every organization's SOA implementation.

小结

- Web服务的历史、概念
- 主要概念介绍
- 基于SOAP的Web服务
- RestFul Web服务
- Web Service安全
- Web 服务和SOA

小结

- A Web service: is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.

Thanks

Q&A