SOA之道

思想、技术、过程与实践

胡德华 著

上海交通大學出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS

SOA之道

思想、技术、过程与实践

胡德华 **著**

上海交通大学出版社

内 容 提 要

SOA 即面向服务的架构(Service Oriented Archtecture)。本书作者以对 SOA 的独到见解,通过各种真实案例,重点从 SOA 思想、技术架构和软件工程三个方面系统地阐述了如何应用 SOA 进行 IT 系统规划、应用研发和过程管理。

全书思想脉络清晰,语言生动,所引用各类材料、案例均来自作者所从事领域的实际经验,具备很强的实践性、指导性,适合各类企事业单位 IT 系统规划人员、软件架构师、软件开发和研发管理人员学习、参考和借鉴。

图书在版编目(CIP)数据

SOA 之道:思想、技术、过程与实践/胡德华著. —上海:上海交通大学出版社,2011

ISBN 978-7-313-07720-2

Ⅰ. S... Ⅱ. 胡... Ⅲ. 互联网络—网络服务器—研究 Ⅳ. TP368.5

中国版本图书馆 CIP 数据核字(2011)第 183868 号

SOA 之道

思想、技术、过程与实践

胡德华 著

上海交通大学出版社出版发行

(上海市番禺路 951 号 邮政编码 200030)

电话:64071208 出版人:韩建民

常熟市大宏印刷有限公司 印刷 全国新华书店经销

开本:787mm×1092mm 1/16 印张:13.5 字数:324 千字

2011 年 10 月第 1 版 2011 年 10 月第 1 次印刷

ISBN 978-7-313-07720-2/TP 定价:60.00 元

序　一

世界上任何技术的发展都是为满足人们日益增长的物质与文化的需求。软件技术也不例外。软件技术的发展可以极大地提升社会生产效率。时至今日，软件技术领域需要解决的是如何最大限度地消除业务与软件技术之间的鸿沟，使得软件能够比以往任何时候更加快速地响应来自客户（业务）的需求，实现业务需求驱动 IT 系统的构建，达到真正的“技术”无关性。

SOA 已经在国内各行各业的信息系统建设中被广泛应用，但这并不意味着我们广大 IT 系统规划和软件从业人员就理解、掌握和应用真正的 SOA 来构建我们的各类 IT 系统。

其实，问题的本质是：什么是 SOA，真正的 SOA？有没有一个完整的答案可以用来评判某个项目或应用系统是基于真正的 SOA？我想，关于这个问题的解答至少不能用几句简单的文字可以描述的。

看了胡德华先生所著的《SOA 之道－思想、技术、过程与实践》，我认为，书中所有的文字给出了答案。这个答案不是空洞的理论，更不是虚无的概念，而是作者经历了在教育、通讯、互联网、金融和 SaaS 等领域几十个研发或实验项目的实践基础上，为读者展示了一整套 SOA 的思想、技术和过程与实践。

本书思想脉络清晰，语言生动，娓娓道来，很适合对 SOA 有兴趣的各类软件从业人员阅读。书中所引用的各类素材、案例，均来自作者所从事领域的实际经验，具有很强的实践性、指导性，适合各类企事业单位中的 IT 系统规划人员、软件架构师、软件开发和研发管理人员学习、参考和借鉴。

本书的可贵之处在于，作者用自己多年的丰富实践所著的关于 SOA 独到的思想和指导，呈现给大家的是自成体系的，可复用的并可执行的 SOA 技术架构 SOTEA 和过程模型 SODEP。

本书不仅适用于软件从业人员，而对于各类企事业单位 IT 系统规划和应用系统建设人员更具有价值。

浙江大学电子服务研究中心主任、博导


教授

2011 年 7 月于杭州

序　二

如何满足业务快速发展的需求，如何快速地支持业务发展，是横亘在信息部门面前的一道鸿沟。软件系统架构的每一次变革，都让我们把这道鸿沟予以缩小；每一次新的架构的出现，都让我们更加向实现梦想靠近。这个梦想就是能够真正达到技术与业务的分离，达到“业务”与技术的无关性。

SOA 架构的推出，是 IT 技术发展的成果积累。目前 SOA 架构已经在各个行业得到应用。但真正的理解、规划并建设起一个 SOA 的企业业务环境，还有很多的问题需要去解决。

常言道：“知易行难”，但现实往往是我们并没有真正理解那些知识。在针对 SOA 架构的实践中也是这样，SOA 的真正要素是什么？如何评判某个项目或应用系统是基于真正的 SOA 架构？如何通过相应的开发过程来保证和实现真正的 SOA？

得益于作者拥有教育、通信、互联网、金融和 SaaS 等领域几十个研发或实施项目的实践，本书以各种真实案例为基础，为读者展示了一整套 SOA 的思想、技术以及过程与实践。同时，作者在深入的总结和思考的基础上给出了自己关于 SOA 独到的思想和指导，呈现了自成体系的、可复用的以及可执行的 SOA 技术架构 SOTEA 和过程模型 SODEP。

本书对于抽象的概念以简洁明快的语言予以深入浅出的解释，所引用各类材料、案例，均来自作者所从事领域的实际经验，具备很强的实践性、指导性，可以帮助读者建立起一整套关于 SOA 的理论体系以及构建 SOA 环境所需的技术和指导，适合各类企事业单位 IT 系统规划人员、软件架构师、软件开发和研发管理人员学习、参考和借鉴。

恒生电子股份有限公司　CTO　范径武

2011 年 7 月

序　三*

2004年我在中国上海。在我的那次旅行中，我很荣幸的一件事情就是在一个酒店大堂认识了Tony。酒店的休息室里面，灯光昏暗，桌上有台电脑，闪烁着微弱光束的屏幕，时不时地照亮着黄色的沙发。那个下午，一大瓶德国喜力啤酒，我俩一起探讨着软件开发。自从我们第一次碰面至今，Tony一直是我要好的朋友，我有很多次很棒的机会与他一起分享时光。

什么是SOA？一些人认为它只是一个软件架构。有些人可能要说SOA是软件设计的方法。Tony常说他们都说的对。SOA是一种思想，体现在软件开发的技术和过程中。自从我们于2004年第一次见面以来，Tony就一直勤奋地工作在这个领域。他的工作和实践已经对软件架构领域产生了重要的贡献，你们可以在《SOA之道》这本书里面阅读和学习，与Tony一道成长。本书涵盖了来自Tony关于SOA架构的第一手的经验和思想的总结，你们可以通过这本书来学习，让Tony给你做向导。他有丰富的软件研发和项目实施经验。在过去的经历中，他曾参与了几十个不同领域的软件研发和实施、运营项目，这些领域包括金融、通信、互联网和E-Learning。

读者应该以一种保守的态度来读这本书。所有的软件架构都是高度依赖于具体的细节。所有的解决方案应该被加以裁剪以满足实际的需求。没有一个具体的方案能够满足所有的需求。一个成功的架构师总是试图找到一个平衡各方面因素的架构实现。一个好的架构师当然不会只依赖于这本书，而是把这本书作为一个资源和参考，用心选择你的设计，仅用此书不能替代你独立的思考。

Professor Sean，Sheridan College，Oakville，Ontario，Canada

2011年7月

---

* 注：根据英文原稿翻译

前　言

记得上小学的时候，我们有个教授植物课程的老师，他叫王成福。王老师每周一次的课程总能够给我们安排丰富多彩的校外授课，譬如到田间地头，到空旷的原野上。王老师利用校外的机会结合所见到的植物，给我们讲授自然知识，授课内容新颖，言语幽默真切。上王老师课的经历成为了我小学阶段关于上课最为美好的记忆，恐怕也是我学生时代最为美好的上课记忆了吧。

王老师的“伟大”之处其实是因为他说过的一句话，“世界上的任何想象都可以成为现实”。这句话就像一个神灵从小到大，一直陪伴着我。我其实真不相信这是什么真理，无外乎是老师逗我们这帮小孩子乐呵而已。可随着我年龄的增长，学识的加深，我越来越信奉王老师的这句话，至此，某种程度上讲，它已经是我的信仰了。

软件技术一开始就被冠以“绝对的高科技”而存在，甚至有点神秘。单机系统、C/S、组件化、分布式等等，这些技术都曾经是软件高手们自以为站在了时代最前沿的核心内容。我们甚至有这样一种感觉，那就是“谁的技术水平越高，谁就离业务人员越远”。可业务人员是谁？他们是软件从业者的上帝，软件的使用方，这种技术与业务的距离难道真的就是那横在牛郎与织女之间永远不可跨越的鸿沟吗？

技术是为业务服务的，我们一直以来所追求的东西就是如何让技术更好地服务于业务。有没有可能做到这样的情况，即我们不需要总是抵制业务的需求，总是想方设法阻止业务的变更想法，业务也不需要总是失落，总是被技术拒绝，总是有一张无形的所谓“技术”的墙挡住了业务创新的欲望。

SOA 为消除这一切带来了希望。我们可以做到业务和技术双方心平气和地坐下来潜心沟通业务真正需求的是什么，而且他们之间有着可以相互理解与沟通的语言，以此来坦诚交流。业务从此可以酣畅淋漓地说出“我要这个，我要那个，这么改改，那么弄弄”。技术也坦然接受，呵呵，这就是所谓的“和谐世界”。

SOA 是万能的主吗？如果是，那我可以这么认为，“太神奇了，我居然可以从上海飞到洛杉矶”。没错，是人们借助飞机这个工具实现了飞翔。SOA 可以做到业务驱动和技术无关性，是因为技术本身的发展，XML，WSDL，SOAP 等技术提供了“技术无关性”的可能性，而 ESB 和流程引擎等基础设施为业务无关性提供了可能性。

SOA 往往被理解为只是继 EAI 之后的另外一种异构系统之间的整合思路和技术而已，其实，SOA 也是软件系统架构模式的一种。基于 SOA 架构开发实现的系统其自身的服务单元短期内并没有被其他系统复用，但可以被系统内其他模块或子系统复用。由此可见，SOA 所代表的范围是非常广泛的。

有了 SOA 思想和技术，还得有面向 SOA 的过程模型。这就是软件工程的演进理由。我们还没有忘记当年由于不重视开发过程，或者采取了不合适的过程模型而出现的“软件危机”。本书的重点之一就是运用基于迭代增量的基础模型概念，结合 SOA 的特征，倡导了可执行的、

面向SOA架构的过程模型，即SODEP。

技术架构是可以被抽象和复用的。好的技术架构结合适用的过程模型就构成了业务驱动下的快速构建复杂应用系统的一种敏捷方法。对于一个软件公司或者是大型企业，这些也构成了企业的核心技术资产。

本书不是纯粹的个人杜撰。它是结合了业界一些优秀的东西，在大量创新实践案例的经验基础之上的一种认识、总结和再创新。作者的目的就是要给予那些需要SOA知识的人们一些可执行式的启发、借鉴和参照。

在此书撰写过程中，我得到了来自浙江大学陈德人教授、恒生电子股份有限公司总裁方汉林、CTO范径武与银行事业部各位领导和同事们的帮助，他们是高远、何占文、谢晶晶、杨军、安连武等。在此向他们深表感谢！

谨以此书献给我深爱的这个行业和在这个行业里打拼的可爱的人们！

谨以此书献给我亲爱的家人！

胡德华

2011年6月10日于上海

目　录

第 1 章　何为 SOA …… 1

1.1　SOA 思想 …… 1
1.2　SOA 技术 …… 4
1.3　SOA 过程 …… 9

第 2 章　SOA 之落地 …… 16

2.1　概述 …… 16
2.2　企业系统规划 …… 18
2.3　厂商产品规划 …… 20

第 3 章　基于 SOA 的可执行技术架构—SOTEA …… 26

3.1　架构技术需求 …… 26
3.2　SOTEA 设计和实现 …… 34

第 4 章　SODEP 过程模型 …… 151

4.1　RUP 和 SOMA …… 151
4.2　SODEP 过程 …… 153

第 5 章　SOA 之实践 …… 167

5.1　业务建模 …… 167
5.2　系统需求 …… 185
5.3　业务服务 …… 188
5.4　系统服务 …… 190
5.5　系统设计 …… 191

附录　SODEP 过程文档模板 …… 193

第1章　何为SOA

何为SOA？直译之，面向服务的架构(Service Oriented Archtecture)。有人进一步给出解释，SOA是一种架构思想，也对；有人说SOA是一种技术，也不错。而且，我们不仅需要SOA思想和技术，我们还需要适应SOA的软件工程方法。

我们可以应用基于SOA的架构思想进行大到整个IT系统的规划，小到构建一个业务产品应用系统。SOA思想的落地不折不扣的需要适应SOA架构的技术。换句话说，SOA架构的技术落地使得SOA思想成为“神马不是浮云”。同时，落地的过程也需要SOA化，否则，可能这个落地结果不一定扎实，或者说可能会非常痛苦。

1.1　SOA思想

SOA里面的S即代表Service(服务)，在Service之前最常见的是“组件”这个词，在组件之前可能是“对象”，而对象之前是“模块”。换句话说，今天所说的“服务”是进化于模块、对象和组件。它们之间的共同点就在于都是描述能够提供某个功能东西。它们之间的区别在于从模块到对象，从对象到组件，抽象化程度越来越大。再深入点讲，它们都是收集、汇总信息和行为，对于外界而言，隐含内部的处理逻辑，给外部提供服务接口。就区别来讲，对象是通过使用抽象数据类型或抽象数据来表述，对象和组件是通过类或类继承层次关系来组织，而服务是能够被单独、层次组合方式或协助方式被发布、发现和消费的。

那什么是面向服务的架构(Service Oriented Architecture)呢？SOA是某种架构，也可能是某种实践或设计实现模式，这些使得某种应用的功能或功能集能够被提供、发现和消费。而且，这些服务或者服务集合往往是以接口的形式被抽象，具体的内部实现方式并不重要，对于消费端而言，也不需要知道内部的实现技术和方法。换句话说，这些服务都具备透明性和可复用性。

SOA具有的特征：一是复用性。真正复用的是服务本身，而不是通过拷贝代码或模块文件进行复用。这个复用性也不是传统的对象或是组件内部行为的复用，而是基于不同粒度和策略的、可被发布、发现和消费的服务级别的复用。这个很重要，传统的代码或内部行为的复用被提升到了外部基于抽象行为级别的复用。而如果该服务承担流程的驱动职能，其本身能够驱动业务流程，那么在业务领域，该服务则可以被理解为解决流程级别的复用。二是抽象性。服务被消费时，消费端不需要关心该服务的具体实现逻辑。这就为采用基于SOA架构进行应用系统开发的技术人员与业务人员提供了直接沟通的可能性。三是正式性。服务基于接口描述来发布、发现和使用，而接口本身具备明确的行为名称、输入和输出。四是明确性。服务的名称往往需要与具体的业务目标相一致，正式性与明确性也构成了服务与业务直接沟通的更多可能性。

SOA的思想就是面向服务的思想，应用面向服务的思想进行系统规划、设计和构建就是

SOA 实践。

那么，在规划系统的时候如何应用 SOA 思想呢？首先了解非 SOA 思想是如何指导系统规划的。传统的企业应用系统在规划时往往以业务产品驱动。以银行为例，银行业务部门各自为满足本部门业务的发展需要，自己独立规划、建设符合自身业务需求的独立应用系统。各个业务部门之间很少使用可共用的系统或系统模块，即使大家都知道有些东西完全是可以共用的。但基于现有框架下的协调、管理成本，各部门还是愿意建设自己的系统。有时甚至不同部门涉及的业务实际上是一样的，但即使把产品包装改名，也还是要建设自己部门的系统，这也许就是所谓的利益攸关吧。

应用 SOA 思想进行系统规划是以业务流程驱动为导向的。在分析业务流程的基础上进行业务服务的识别，进而复用或开发新的服务，集成、组装和构建出符合业务目标的应用系统，尽管这些服务可能存在于不同的业务部门。假设某银行新建供应链融资业务部门，准备配合新推出的专为中小企业服务的融资产品建设一个系统。在详细分析业务流程的基础上，发现所有的业务服务都已经存在，那么，系统建设的过程就变成了流程分析、服务确认、服务编排、系统集成、流程实现和测试上线的过程。这就是 SOA 思想驱动下的系统规划建设的一种情况。

如果应用 SOA 思想进行系统规划相对比较直接和宏观，那么应用 SOA 思想设计和构建应用系统就属于相对微观层面了。如果业务分析、产品设计和系统构建都是基于 SOA 的思想和技术，则系统开发过程就是构建基于 SOA 架构的应用系统的实践过程。应用系统业务目标的实现是通过分析业务服务而设计和实现系统服务，通过分析业务流程而设计、编排内部若干系统服务，进而串接和集成各个流程服务、产品服务和功能，以实现业务需求目标。SOA 思想在系统规划和应用构建过程中的体现可用以下两图区别表示。

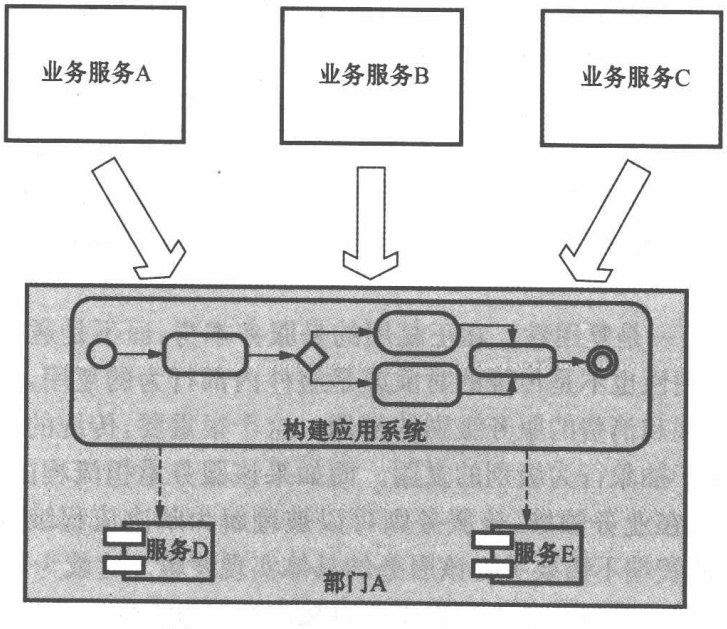

图 1-1　基于面向需求的应用系统

图 1-1 为构建一个部门 A 需要的应用系统，需要首先分析部门 A 的业务需求，识别业务服务，进行而考察部门所在集团或组织内、业务关联方等所有系统服务资产，发现服务 A、服务 B 和服务 C 都是已有的服务，它们可能作为 IT 服务资产分布在不同的集团内或关联方组织

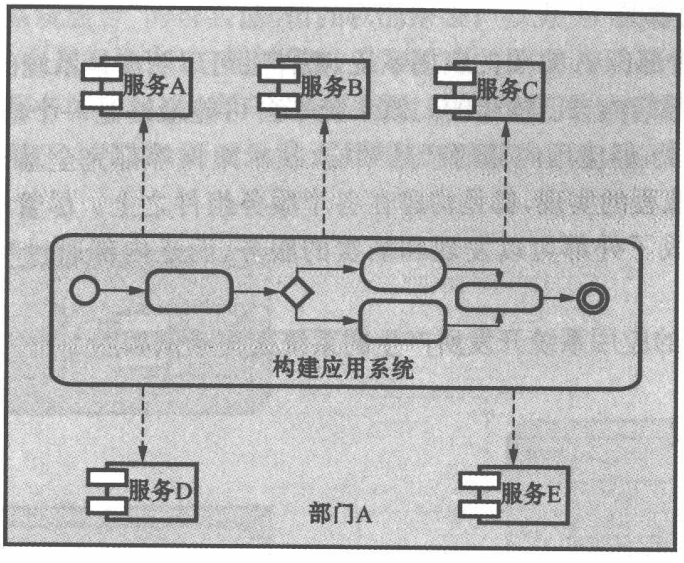

图 1-2　基于面向服务的应用系统

内。另外一种情况是，服务 A、服务 B 和服务 C 还不存在，那么为构建部门 A 的应用，实际上需要整体规划和设计集团内的 IT 系统服务资产，这些服务可能分布于不同的内部组织或系

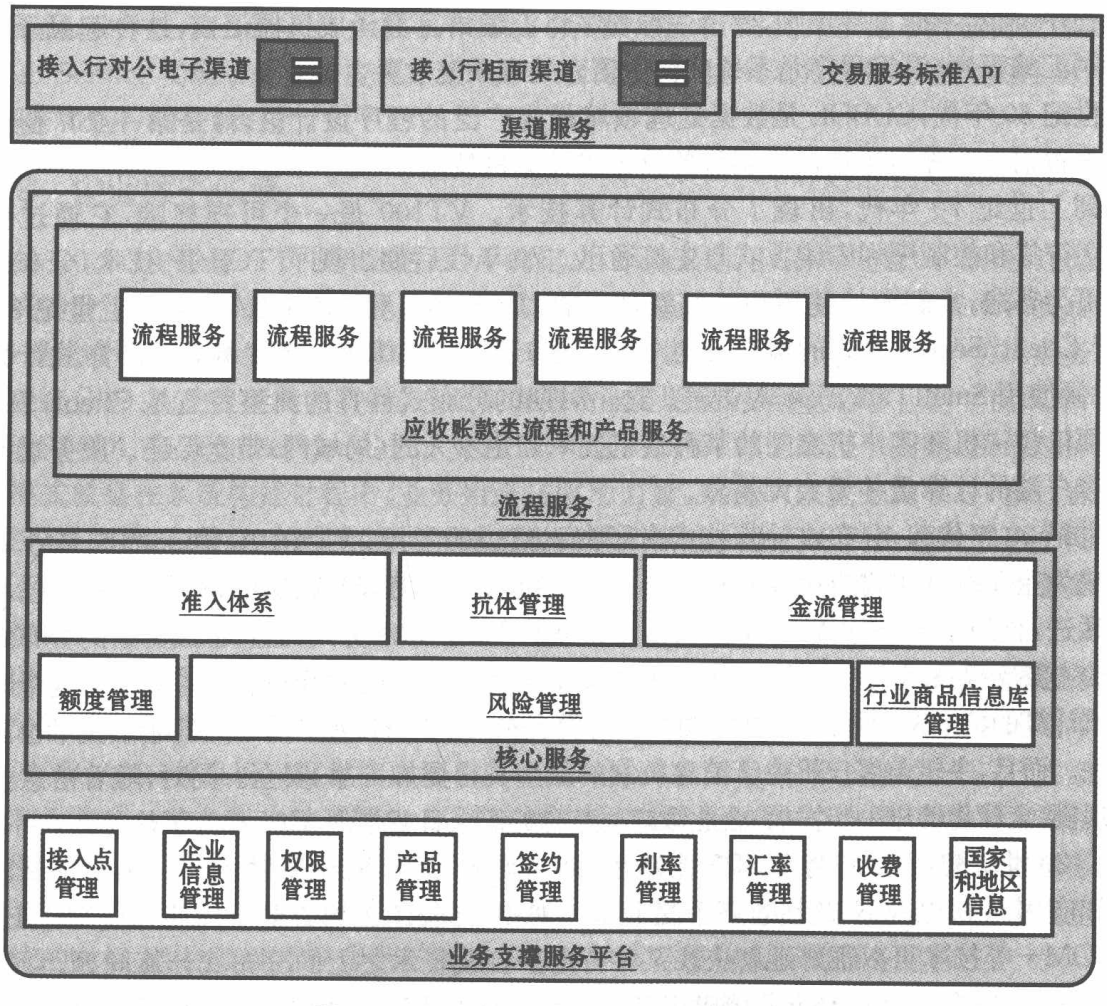

图 1-3　SOA 思想主导下的应用系统架构

统。显然，这两种情况都是 IT 规划所必须面对的情况。

图 1-2 为构建一个部门 A 需要的应用系统，整体上可以明确该系统的业务范围，而且大部分功能服务都是需要部门内自己构建的。该系统平台可能最终是一个应用部署，服务间可能都是运行在一个 VM 上，但应用内部的产品架构、技术架构等都完全基于面向服务的思想和技术。整个系统业务流程的实现，都是构建在各个服务组件之上。尽管这些服务可能不一定或者没有必要都发布成了外部可以发现和消费的服务，而是内部通过“注入反转”的方式被消费。

SOA 思想主导下的应用系统开发所产生的系统架构示例如图 1-3。

1.2 SOA 技术

1.2.1 软件技术发展历程

如果说 SOA 的落地需要有别于传统技术支撑的话，那么 SOA 也是技术。从软件技术的发展历程，我们可以看到：众所周知，软件技术的发展程序最初用汇编语言编写程序，也就是上世纪 50 年代，那时还没有分布式计算。汇编语言是面向机器的程序设计语言，使用汇编语言编写的程序，机器不能直接识别，要由一种程序将汇编语言翻译成机器语言，这种起翻译作用的程序叫汇编程序，汇编程序是系统软件中语言处理系统软件。

上世纪 60 年代，COBOL 是数据处理领域最为广泛的程序设计语言，是第一个广泛使用的高级编程语言。那时也还没有明显的分布式计算机技术。

直到上世纪 70 年代，出现了分布式计算技术。VT100 是一个可视终端，它通过使用 ASCII 字符集和控制序列串线方式与主机通讯。70 年代后期出现了 TCP/IP 技术，分布式计算技术更为普遍，人们开始使用更为高级的语言，PASCAL 和 SIMULA。到了上世纪 80 年代，基于 Client/Server 方式的架构变得十分普遍，还有诸如 RPC、NFS 等其他分布式计算技术，这时候使用 SmallTalk、Ada 等语言。这个时期的分布式计算的典型特点是 Client/Server 模式必须依赖主机和客户机之间的某种通讯技术和足够大的(局域网)带宽保证。服务端的计算量与客户端的计算量并无太大差异。

上世纪 90 年代起，分布式计算技术有了突飞猛进的发展。CORBA、MQ、EJB、EAI 等技术日益成为企业级分布式应用开发和运行的主流技术。伴随着互联网的发展，WWW 技术更是突飞猛进，直到今天，WWW 不但成为信息共享的主要技术和方式，而且也成为新的分布式应用构建的主要支撑技术之一。这时，Java 和 C++编程语言开始流行并变为主流了。这个时期的分布式计算的特点是，服务端分担了更多的计算负荷，服务端和客户端之间的通讯技术也出现了多样性。而且，主机与客户机由于消息机制的诞生变得更加可靠、安全。同时，随着信息技术和企业规模、全球化的进一步发展，企业级应用整合的需求日益强烈，EAI 成为了热门。

到了 20 世纪初，SOAP 与 WSDL 出现了，即出现了 Web Service 技术。SOAP 与 WSDL 的出现和应用是软件技术史上的一个里程碑。那它强在什么地方呢？CORBA、MQ、EJB、COM/COM+等技术可以很好地解决在某种特定平台或技术之上的分布式计算问题，它们都很强大。然而，现今又有几家企业级应用是搭建在一种技术平台或者特定组件技术之上的呢？如今，业务全球化和企业国际化是导致信息现代化所必须面临“不同系统平台、不同组件技术

和不同技术下的遗留系统整合”的现实情况。而 Web Service 技术提供了一种技术，即不管什么平台、什么技术和什么开发语言，它能够通过 WSDL 技术和标准将不同平台、不同技术和不同开发语言下的业务服务发布出去，客户端可以通过基于 HTTP 访问之上的 SOAP 协议来远程调用。由于访问是基于 HTTP，因而远程调用可以突破防火墙，实现互联网级别的远程调用，如图 1-4 所示。

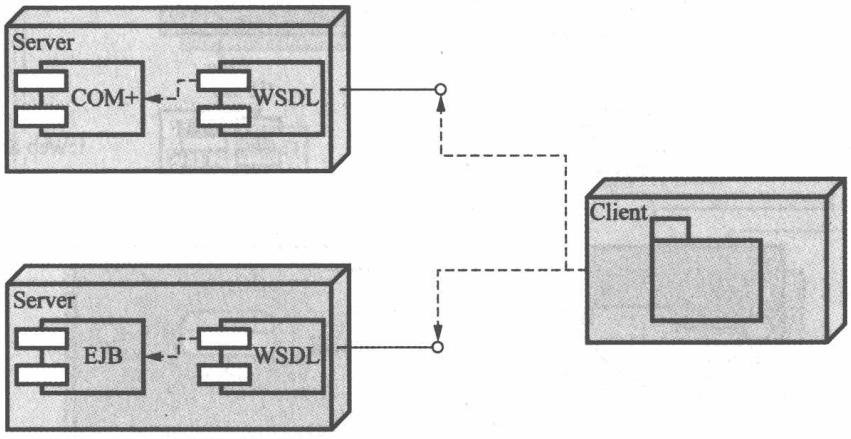

图 1-4　SOA 协议可实现互联网级别的远程调用

目前，软件技术已走向了“无技术”时代。所谓“无技术”时代并不是不要任何技术，而是通过 Web Service 实现了企业级应用系统基于平台无关性、技术无关性和语言无关性的开发、整合、部署和运行的全新时代。

1.2.2　企业服务总线

在 1.1 SOA 思想一节中给出的 SOA 的定义，不管是运用 SOA 规划 IT 系统，还是进行应用系统的设计，似乎都有点理想化。尤其是这三个问题：一是“不关心内部技术和实现”，二是“服务提供者与服务消费者到底如何协同？基于何种技术协同操作?”，三是“实现业务需求目标与系统分析、构建之间的直接性，进而实现所谓业务驱动的架构模式”。对于前两个问题，其实基于 WSDL 和 Web Service 的技术分析，即可迎刃而解；对于第三个问题，所谓业务驱动的架构模式就是在系统构建过程中，业务始终是驱动力量，或者叫以业务为中心，该过程包括业务分析、系统分析、设计、构建和测试等整个生命周期活动。我们可以由此推理这个问题隐含表达了这样一个意思，即基于 SOA 架构下具体技术不是问题，包括服务本身以及服务间如何协同操作。

以上叙述似乎让人感觉 SOA 就是 Web Service，其实不然。Web Service 只是服务端与客户端分布式远程调用(RPC)的一种实现方法，而且这种方法是基于 HTTP 之上的简单对象访问协议(SOAP)，并且可以穿过防火墙。所以这种方式由于是基于已有的 HTTP 之上，简单、安全和无障碍。这一点也被理解为 SOA 实现“技术无关性”的一个方面，即服务消费层面。

SOA 除了具备服务端与客户端通讯层面的服务消费技术无关性之外，还具备服务提供技术无关性。这不仅包括新增服务的提供，还包括企业已有服务的复用。我们在构建企业级应用系统的时候，往往要复用很多原有的基于各种不同传统技术下的服务组件。

图 1-5 所示，SOA 应用可以通过 Web Service 集成和复用服务 A 和服务 F。但显然也需要解决使用不同技术的服务组件如 EJB，COM+，FTP 等服务提供发布的问题。

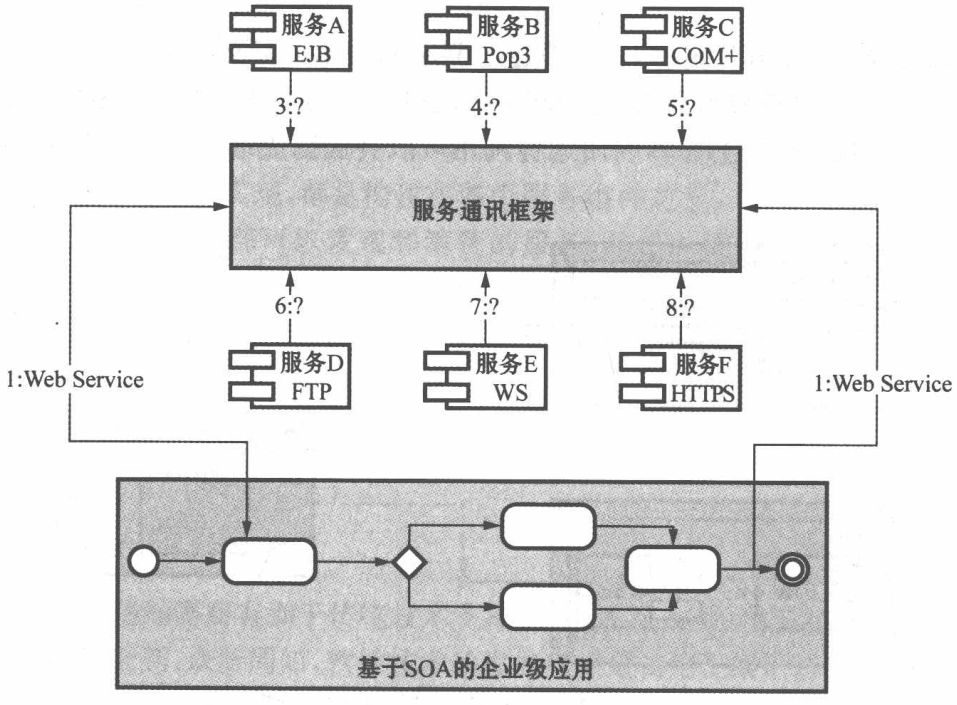

图 1-5 SOA 应用

由此看来，要实现 SOA 的远大目标，还需要一个强大的公共平台，它能够通过简单工作即可发布各类技术组件而成为服务，或者说，这个公共平台可以使得 SOA 应用构建者不需要担心不同技术组件发布、使用服务的问题，而是只关心如何分析业务，分解服务，引用服务，集成服务等。至此，我们引出一个 SOA 技术落地的核心内容，即企业服务总线(Enterprise Service Bug，简称 ESB)，如图 1-6 所示。

何为 ESB？在 SOA 架构下，服务方与服务消费者通过一个公共的通讯框架进行通讯，这个公共的通讯框架就是企业服务总线 ESB。在计算机里面，有个总线负责连接 CPU、RAM 和 I/O 设备等，ESB 总线的概念即出自于此。如图 1-6 所示，每个服务组件提供给 ESB 接口，这些接口可以间接地通过 ESB 与无数个其他类型的服务通讯交换。例如，客户端 8 发一个消息给 ESB，ESB 接收并传递给服务组件 3，服务组件 3 完成计算后再返回给 ESB，ESB 再处理并返回给客户端 8，从而完成一次在 SOA 框架下、通过 ESB 设施控制并实现的、在不同技术、不同类型组件之间的一次调用。在这个过程中，客户端 8 并不知道服务组件 3 内部是使用何种技术、何种编码语言实现，只知道其暴露到 ESB 上的业务接口即可。这就是 SOA 应用构建过程中业务驱动、技术无关性的具体体现。

ESB 作为 SOA 架构的基础设施，一般要为服务提供者和服务消费者提供诸多功能，譬如，服务命名和查找，以便在运行时发现服务和消费；要有服务注册表，以储存服务协议；当然还要有安全、事务控制、消息机制、服务管理和监控等功能，具体本章不再细述。

综上所述，SOA、WSDL、SOAP、ESB 之间的关系如下：SOA 架构是新型的 Client/Server 架构，它是实现真正意义上的业务驱动型架构。WSDL 是 SOA 架构中服务定义和发布的一种方法，SOA 不限于这一种方法，例如还有 IDL 等。SOAP 是服务客户端与服务端实现远程调用(RPC)的一种基于 HTTP 之上的一种访问协议。ESB 是 SOA 技术落地的一种核心基础设施，以实现服务间跨技术互操作。

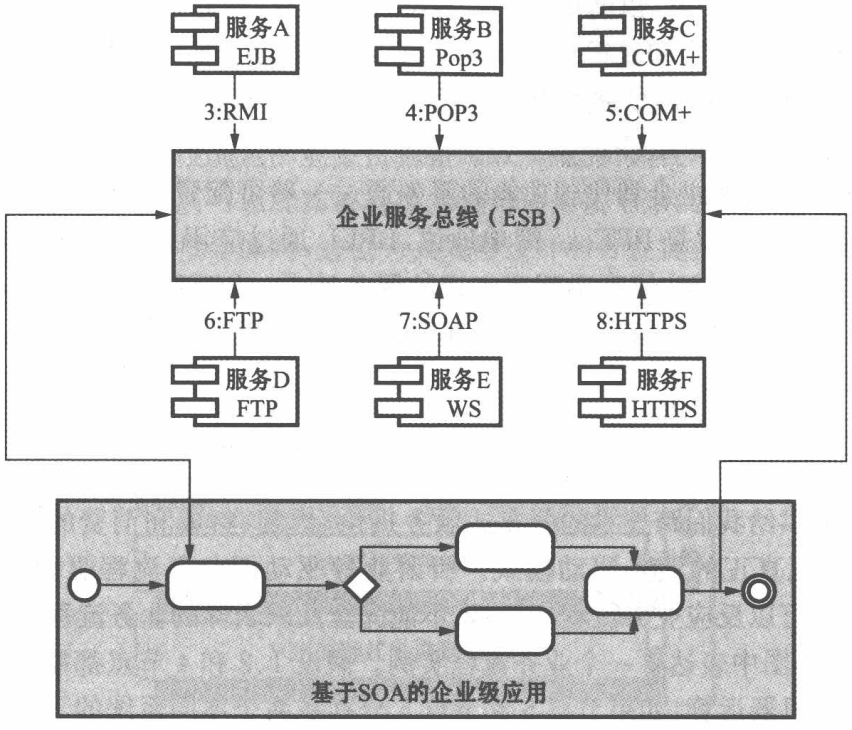

图 1-6　ESB 功能

1.2.3　服务编排技术

SOA 架构本质是服务/客户端模式，客户端消费服务可以通过 WSDL 来实现，即服务通过 WSDL 发布并对外提供服务，客户端通过基于 HTTP 之上的 SOAP 协议来调用服务。设想以下情形，有两个服务，一个是服务 A，另外一个是服务 B。客户端基于某个业务需求，需要根据客户端的输入参数，决定是调用服务 A 或调用服务 B。如果把客户端按照服务来设计，也被设计成可以提供其他客户端消费的服务组件，可以将此新的服务定义为服务 C，那么新的客户端可以调用服务 C，服务 C 根据新的客户端的输入参数判断来调用服务 B 或 C 以完成运算，并把结果返回给客户端。如图 1-7 所示。

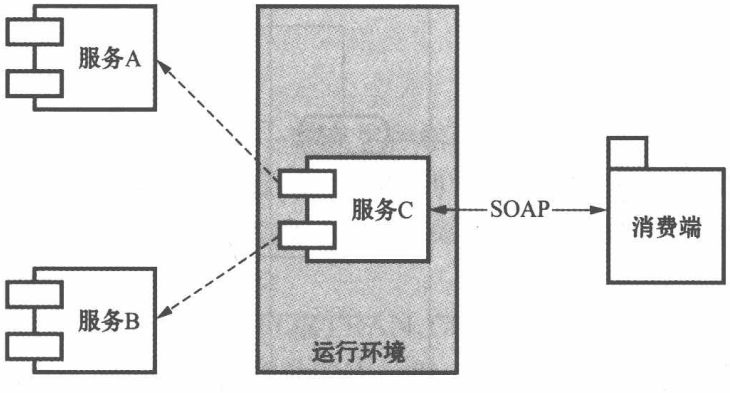

图 1-7　通过 SOAP 协议调用服务模型

服务 C 将运行于某运行环境。服务 A 和服务 B 可能是遗留系统的服务组件，通过 WSDL 发布并提供服务。服务 C 实际上是一个组合代理服务，如果服务 C 的实现是采用某种特定的语言，譬如 Java，那么 Java 语言下的某个类会实现代理调用的逻辑。这样的结果是，即使服务

C还是可以发布为Web服务，供消费端直接调用，但服务C实际上本身的实现技术决定了消费端的请求逻辑的处理方式和过程。而服务C实际上仅仅是做了输入参数的判断后请求背后的服务A或B。并且，服务C被建模并实现为服务的ROI和其他KPI评估时，并不一定适合被构建为一个独立的服务组件。

能否将这种担当服务组合和代理功能的服务通过一种可配置的，技术无关性的方式来实现呢？答案是肯定的，这就是BPEL。简单地说，BPEL通过应用XML技术，定义各种命令，将各种服务接口组合在一起，代理实现客户端的服务请求。BPEL实现了服务组装逻辑的技术无关性，继承了SOA的本质特征，为构建真正的SOA应用提供了核心支持。过多的BPEL的知识不在这里描述。

1.2.4 流程引擎

BPEL能够提供给我们跨技术的分布式服务提供、组装、集成和消费的能力，但我们还需要流程引擎，以实现真正的业务驱动模式。所谓业务驱动即业务流程驱动，在一定程度上，BPEL的执行路线可以反应业务流程，但终究不能完全反映实际的业务流程需求。

以图1-8为例：图中表达了一个业务流程实例。假设1、2和4节点都需要人工岗位操作，3、5和6节点直接机器运算，如果节点5和节点6分别是两个不同系统的不同服务，那么就可依据BPEL的服务组装和编排原理，通过BPEL构造一个新的组合服务，来技术无关地实现两个服务的单点调用和代理。同时，我们也知道至少在目前的BPEL规范里面，人工任务支持是不足的，即便后来又有了BPEL4People可以支持，但运用成熟、强大的厂家或开源工作流引擎统一实现业务流程仍较使用BPEL具有优势（同样，此处不详细描述流程引擎技术）。

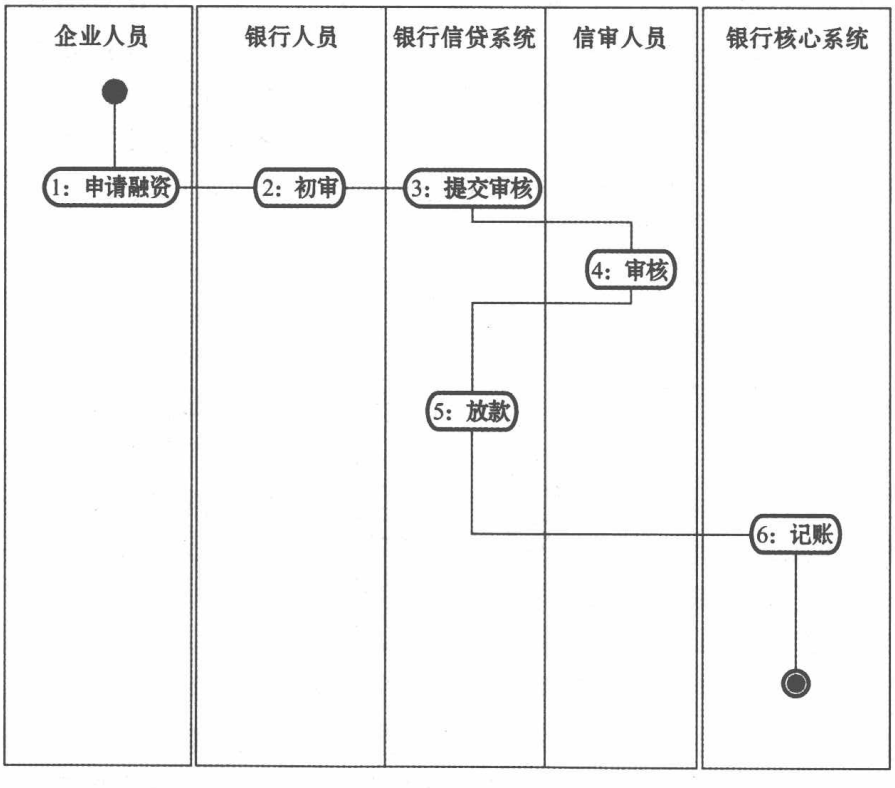

图1-8 业务流程实例