atitit.通用基础软件 与有哪些

[1. 数据库 1](#_Toc31221)

[2. 开发平台，业务运行平台 1](#_Toc3267)

[3. Ｏｓ 1](#_Toc10930)

[4. oa 1](#_Toc12427)

[5. 中间件 1](#_Toc6145)

[6. 语言处理系统（包括编译程序、解释程序和汇编程序） 1](#_Toc28366)

[7. 中间件编辑 2](#_Toc6729)

[7.1. 相关概念编辑 3](#_Toc16865)

[7.2. 分类编辑 4](#_Toc24560)

# 数据库

# Ｏｓ

# oa

# 中间件

# 开发平台，业务运行平台

# [语言处理系统](http://baike.baidu.com/view/2961271.htm" \t "http://baike.baidu.com/_blank)（包括[编译程序](http://baike.baidu.com/view/454895.htm" \t "http://baike.baidu.com/_blank)、[解释程序](http://baike.baidu.com/view/47200.htm" \t "http://baike.baidu.com/_blank)和[汇编程序](http://baike.baidu.com/view/14850.htm" \t "http://baike.baidu.com/_blank)）

**流行的说法**

口头上可以说，基础软件是操作系统、数据库、办公软件和[中间件](http://baike.baidu.com/view/23710.htm" \t "http://baike.baidu.com/_blank)的统称。

**比较全面的说法**

基础软件包括操作系统、[数据库系统](http://baike.baidu.com/view/7809.htm" \t "http://baike.baidu.com/_blank)、[中间件](http://baike.baidu.com/view/23710.htm" \t "http://baike.baidu.com/_blank)、[语言处理系统](http://baike.baidu.com/view/2961271.htm" \t "http://baike.baidu.com/_blank)（包括[编译程序](http://baike.baidu.com/view/454895.htm" \t "http://baike.baidu.com/_blank)、[解释程序](http://baike.baidu.com/view/47200.htm" \t "http://baike.baidu.com/_blank)和[汇编程序](http://baike.baidu.com/view/14850.htm" \t "http://baike.baidu.com/_blank)）和[办公软件](http://baike.baidu.com/view/979640.htm" \t "http://baike.baidu.com/_blank)（包括文字处理、[电子表格](http://baike.baidu.com/view/2057578.htm" \t "http://baike.baidu.com/_blank)、幻灯片以及一些初级[图片处理](http://baike.baidu.com/view/539807.htm" \t "http://baike.baidu.com/_blank)程序）。

**软件划分的历史**

早些时候，人们把[软件系统](http://baike.baidu.com/view/8343.htm" \t "http://baike.baidu.com/_blank)分为系统软件与应用软件两大部分。后来提出[支撑软件](http://baike.baidu.com/view/1371360.htm" \t "http://baike.baidu.com/_blank)。现在又大讲基础软件。这些划分的变迁反映出人们对软件的认识正在逐步深入。

# 中间件[编辑](http://baike.baidu.com/javascript:;)

作为基础软件的重要组成部分，中间件的作用不言而喻，其真正在世界范围内进入产业化阶段是 20世纪90年代，目前已经迅速发展成为开发分布式应用系统不可缺少的关键基础设施，与操作系统、数据库系统共同构成基础软件体系的三大支柱。

中间件是一类连接软件组件和应用的计算机软件，它包括一组服务。以便于运行在一台或多台机器上的多个软件通过网络进行交互。该技术所提供的互操作性，推动了一致[分布式](http://baike.baidu.com/view/402382.htm" \t "http://baike.baidu.com/_blank)体系架构的演进，该架构通常用于支持并简化那些复杂的[分布式应用程序](http://baike.baidu.com/view/553502.htm" \t "http://baike.baidu.com/_blank)，它包括web服务器、[事务](http://baike.baidu.com/view/121511.htm" \t "http://baike.baidu.com/_blank)[监控器](http://baike.baidu.com/view/1938865.htm" \t "http://baike.baidu.com/_blank)和[消息队列](http://baike.baidu.com/view/262473.htm" \t "http://baike.baidu.com/_blank)[软件](http://baike.baidu.com/view/37.htm" \t "http://baike.baidu.com/_blank)。

中间件(middleware)是[基础软件](http://baike.baidu.com/view/1299596.htm" \t "http://baike.baidu.com/_blank)的一大类，属于可复用[软件](http://baike.baidu.com/view/37.htm" \t "http://baike.baidu.com/_blank)的范畴。顾名思义，中间件处于[操作系统](http://baike.baidu.com/view/880.htm" \t "http://baike.baidu.com/_blank)[软件](http://baike.baidu.com/view/37.htm" \t "http://baike.baidu.com/_blank)与用户的[应用软件](http://baike.baidu.com/view/7886.htm" \t "http://baike.baidu.com/_blank)的中间。

中间件在操作系统、网络和[数据库](http://baike.baidu.com/view/1088.htm" \t "http://baike.baidu.com/_blank)之上，[应用软件](http://baike.baidu.com/view/7886.htm" \t "http://baike.baidu.com/_blank)的下层，总的作用是为处于自己上层的[应用软件](http://baike.baidu.com/view/7886.htm" \t "http://baike.baidu.com/_blank)提供运行与开发的环境，帮助用户灵活、高效地开发和集成复杂的应用软件。在众多关于中间件的定义中，比较普遍被接受的是IDC表述的：中间件是一种独立的[系统软件](http://baike.baidu.com/view/7860.htm" \t "http://baike.baidu.com/_blank)或服务程序，[分布式](http://baike.baidu.com/view/402382.htm" \t "http://baike.baidu.com/_blank)应用软件借助这种软件在不同的技术之间共享资源，中间件位于客户机服务器的操作系统之上，管理计算资源和[网络通信](http://baike.baidu.com/view/538641.htm" \t "http://baike.baidu.com/_blank)。

中间件是位于平台（硬件和操作系统）和应用之间的通用服务，如图1所示，这些服务具有标准的程序接口和协议。针对不同的操作系统和硬件平台，它们可以有符合接口和协议规范的多种实现。

图1 中间件概念模型图

也许很难给中间件一个严格的定义，但中间件应具有如下一些特点：

①满足大量应用的需要；

②运行于多种硬件和OS平台；

③支持分布计算，提供跨网络、硬件和OS平台的透明性的应用或服务的交互；

④支持标准的协议；

⑤支持标准的接口。

由于标准接口对于可移植性、标准协议对于互操作性的重要性，中间件已成为许多标准化工作的主要部分。对于应用软件开发，中间件远比操作系统和网络服务更为重要，中间件提供的程序接口定义了一个相对稳定的高层应用环境，不管底层的计算机硬件和系统软件怎样更新换代，只要将中间件升级更新，并保持中间件对外的接口定义不变，应用软件就几乎不需任何修改，从而保护了企业在应用软件开发和维护中的重大投资。

## 相关概念[编辑](http://baike.baidu.com/javascript:;)

·企业服务[总线](http://baike.baidu.com/view/1389.htm" \t "http://baike.baidu.com/_blank)（ESB ： Enterprise Service Bus）：ESB 是一种开放的、基于标准的[分布式](http://baike.baidu.com/view/402382.htm" \t "http://baike.baidu.com/_blank)同步或异步信息传递中间件。通过 XML、Web Service接口以及标准化基于规则的[路由选择](http://baike.baidu.com/view/632880.htm" \t "http://baike.baidu.com/_blank)文档等支持，ESB 为企业应用程序提供安全互用性。

·事务处理（TP：Transaction Processing）[监控器](http://baike.baidu.com/view/1938865.htm" \t "http://baike.baidu.com/_blank)：为发生在[对象](http://baike.baidu.com/view/2387.htm" \t "http://baike.baidu.com/_blank)间的事务处理提供监控功能，以确保操作成功实现。

·[分布式](http://baike.baidu.com/view/402382.htm" \t "http://baike.baidu.com/_blank)计算环境（DCE：Distributed Computing Environment）：指创建运行在不同平台上的[分布式](http://baike.baidu.com/view/402382.htm" \t "http://baike.baidu.com/_blank)应用程序所需的一组技术服务。

·远程过程调用（RPC：Remote Procedure Call）：指客户机向服务器发送关于运行某程序的请求时所需的标准。

·对象请求代理（ORB：Object Request Broker）：为用户提供与其他[分布式](http://baike.baidu.com/view/402382.htm" \t "http://baike.baidu.com/_blank)网络环境中对象通信的接口。

·[数据库](http://baike.baidu.com/view/1088.htm" \t "http://baike.baidu.com/_blank)访问中间件（Database Access Middleware）：支持用户访问各种操作系统或应用程序中的数据库。SQL 是该类中间件的其中一种。

·信息传递（Message Passing）：[电子邮件系统](http://baike.baidu.com/view/2296101.htm" \t "http://baike.baidu.com/_blank)是该类中间件的其中一种。

·基于[XML](http://baike.baidu.com/view/63.htm" \t "http://baike.baidu.com/_blank)的中间件（XML-Based Middleware）：XML 允许开发人员为实现在[Internet](http://baike.baidu.com/view/11165.htm" \t "http://baike.baidu.com/_blank)中交换[结构化信息](http://baike.baidu.com/view/1677752.htm" \t "http://baike.baidu.com/_blank)而创建文档。

　　因此，在系统软件之中，操作系统、数据库、中间件的三驾马车，中间件是最神秘的。因为，好歹大家通过Windows基本上会了解操作系统是个什么东西，尽管不会很全面，很专业，毕竟是有感觉的。数据库，虽然没有直接见过，但基本上明白数据是要一个仓库来储存的，因此，也大致知道数据库管理系统是干什么的。

　　长期以来，中间件是一个专业化非常强的细分产业。因为中间件的技术门槛比较高，玩家也不多，无论是国外还是国内都是如此。因此，行业内对什么是中间件并不特别在意。而公司名称直接叫中间件的就更少了，金蝶中间件应该是国内外直接在公司名称中冠以中间件字眼最早，也是很少的公司之一。另一方面，因为中间件软件还处于发展阶段，还没有完全成熟，因此对中间件的定义也就没有深究，或者权威的说法。

　　但现在情况有点变化，其中一个原因在于2008年底，国家启动了核高基重大科技专项，在基础软件领域明确提出重点支持操作系统、数据库、中间件、文字处理等基础软件产业的自主创新，几乎一夜之间大大小小的软件公司都宣称是做中间件的了，只要不是做最终应用

## 分类[编辑](http://baike.baidu.com/javascript:;)

中间件分类（IDC的分类）：大致可分为六类:[终端仿真](http://baike.baidu.com/view/190233.htm" \t "http://baike.baidu.com/_blank)/屏幕转换中间件、数据访问中间件、[远程过程调用](http://baike.baidu.com/view/431455.htm" \t "http://baike.baidu.com/_blank)中间件、[消息中间件](http://baike.baidu.com/view/3118541.htm" \t "http://baike.baidu.com/_blank)、交易中间件、[对象](http://baike.baidu.com/view/2387.htm" \t "http://baike.baidu.com/_blank)中间件。

中间件所包括的范围十分广泛，针对不同的应用需求涌现出多种各具特色的中间件产品。但至今中间件还没有一个比较精确的定义，因此，在不同的角度或不同的层次上，对中间件的分类也会有所不同。由于中间件需要屏蔽分布环境中异构的操作系统和网络协议，它必须能够提供分布环境下的通讯服务，我们将这种通讯服务称之为平台。基于目的和实现机制的不同，我们将平台分为以下主要几类：

远程过程调用中间件（Remote Procedure Call）

面向消息的中间件（MesSAge-Oriented Middleware）

[对象请求代理](http://baike.baidu.com/view/4452465.htm" \t "http://baike.baidu.com/_blank)中间件（object RequeST Brokers）

它们可向上提供不同形式的通讯服务，包括同步、排队、订阅发布、广播

金融行业中间件

常见的中间件　ｗｅｂ　ｓｅｒｖｅｒ　　ｒｅｓｉｎ，jms，mq

还没有玩 ，参考 中间件\_百度百科.htm

****2. 中间件的起源****

****2.1 中间件发展的历史****

事情从1946年说起，世界上第一台电子计算机埃尼阿克诞生，人类进入信息时代。1955年，约翰巴克斯发明了最早的程序语言Fortran，现代意义上的软件就诞生了。

　　1964年，IBM发布OS/360操作系统，软件与硬件分离，同时，软件成为一个独立的产业正式登上产业界的舞台。中间件就是软件产业不断发展过程中自然产生的。

　　90年代，文顿·瑟夫这位互联网之父的发明成为改变IT业的重大革命性创新。互联网促使分布式系统和网络应用的诞生，中间件就是伴随网络技术的产生、发展而兴起的，可以说没有网络就没有现代意义上的中间件。因为，网络环境需要解决异构分布网络环境下软件系统的通信、互操作、协同、事务、安全等共性问题，提高异构分布网络环境下软件系统的互操作性、可移植性、适应性、可靠性等问题。

　　1968年IBM发布CICS交易事务控制系统，使得应用软件与系统服务分离，这是中间件技术萌芽的标志，因为CICS还不是分布式环境的产物，因此我们往往还不将CICS作为正式的中间件系统。

　　一般来说，我们将1990年诞生于ATT公司的BELL实验室的Tuxedo系统（后来被NOVELL从ATT公司随着UNIX系统一起买走，后来又卖给了BEA公司，现在归于ORACLE公司旗下了）作为中间件的诞生标志。Tuxedo解决了分布式交易事务控制问题，中间件开始成为网络应用的基础设施，中间件正式成型，这是最早的交易中间件。

　　1994年IBM发布消息队列服务MQ系列产品，解决分布式系统异步、可靠、传输的通讯服务问题，消息中间件诞生。

　　1995年，JAVA之父James Gosling发明JAVA语言，JAVA提供了跨平台的通用的网络应用服务，成为今天中间件的核心技术。JAVA是第一个天生的网络应用平台，特别是J2EE发布以来，JAVA从一个编程语言，演变为网络应用架构，成为应用服务平台的事实标准。应用服务器中间件，成为中间件技术的集大成者，也成为事实上的中间件的核心。

　　2001年，微软发布.NET，中间件演变为.NET和JAVA两大技术阵营。但由于.NET还不是一个完全开放的技术体系，只有一个玩家，因此，虽然.NET也是一种中间件，但由于IBM/ORACLE/SUN/SAP等巨头都无一例外成了JAVA阵营的支持者，因此，我们习惯上提到中间件时，往往不包括.NET中间件体系。

　　以上谈的是历史，但透过历史事实背后，为什么会出现中间件，这其中的本质因素是什么？

****2.2 中间件发展的驱动力****

　　中间件出现的驱动力主要来自软件研发过程碰到的种种问题。从软件出现最早是用于科学计算，然后是计算机辅助设计、辅助制造等等工业应用。在企业管理领域大规模应用后，业务需求不断的变化、系统不断增加、流程更复杂、系统越来越不堪重负，出现了需求交付方面的重大挑战，以至于人们用软件危机来描述软件工业所面临的困境。

　　总结起来，软件工业面临的主要问题是四个方面：质量问题、效率问题、互操作问题、灵活应变问题。这些问题今天依然困扰着这个行业。

　　造成这个局面的原因是异构性和标准规范的滞后。

* ****屏蔽异构性****

　　异构性表现在计算机的软硬件之间的异构性，包括硬件（CPU和指令集、硬件结构、驱动程序等），操作系统（不同操作系统的API和开发环境）、数据库（不同的存储和访问格式）等等。长期以来，高级语言依赖于特定的编译器和操作系统API来编程，而他们是不兼容的，因此软件必须依赖于开发和运行的环境。

　　造成异构的原因源自市场竞争、技术升级以及保护投资等因素。希望屏蔽异构平台的差异性问题是促成中间件发展的驱动力之一。

* ****实现互操作****

　　因为异构性，产生的结果是软件依赖于计算环境，使得各种不同软件之间在不同平台之间不能移植，或者移植非常困难。而且，因为网络协议和通信机制的不同，这些系统之间还不能有效地相互集成。

　　造成互操作性不好的原因，主要是标准的滞后。解决软件之间的互操作性问题也是促成中间件发展的驱动力之一。

* ****共性凝练和复用****

　　软件应用领域越来越多，相同领域的应用系统之间许多基础功能和结构是有相似性的，每次开发系统都从零开始绝对不是一种好的方法，也是对质量和效率的很大的伤害。

　　尽可能多地凝练共性并复用以提高软件开发效率和质量，通过中间件通过提供简单、一致、集成的开发和运行环境，简化分布式系统的设计、编程和管理，这也是中间件发展的重要驱动力。

　　在长期的探索过程中，解决软件的四个问题的办法总结起来两个方面：工程方法、平台与技术。

　　工程方法就是用工业工程、系统工程的理论、方法和体系来解决软件研发过程中的管理问题，包括团队管理、项目管理、质量控制等等，这就是软件工程。除了软件工程方法之外，我们发明了更多的架构规划、设计和实施的方法，不断累积领域的知识与经验等等。

　　更好的技术手段，包括更好的程序设计语言、更好的平台和软件开发技术，如面向对象、组件开发、面向服务等等。而这方面，在技术上逐渐发展的成果大部分都凝聚在今天的中间件平台之中。

　　而这些更好的技术手段，从本质上是通过复用、松耦合、互操作（标准）等机制来提高软件质量、加快软件研发效率、使研发出来的产品能够相互集成并灵活适应变化。  
这些因素逐渐促成了中间件软件的形成和发展。

****3. 中间件的概念****

　　讲了这么多，究竟什么是中间件，也就是中间件的定义是什么？

　　针对这个问题，应该说还没有一个标准的定义，或者说还没有完全取得学术界和产业界的共识。

　　顾名思义，中间件就是处于中间的软件。但这种不是从功能，或者特性来定义的概念，而是用位置来定义的名字，就容易被不同的人从不同角度赋予其不同的含义。

　　IDC曾经给中间件下的定义是中间件是一种独立的系统软件或服务程序，分布式应用软件借助这种软件在不同的技术之间共享资源，中间件位于客户机服务器的操作系统之上，管理计算资源和网络通信。

　　我国学术界一般认可的定义是中间件是指网络环境下处于操作系统、数据库等系统软件和应用软件之间的一种起连接作用的分布式软件，主要解决异构网络环境下分布式应用软件的互连与互操作问题，提供标准接口、协议，屏蔽实现细节，提高应用系统易移植性（北京大学梅宏）。

　　中科院软件所研究员仲萃豪形象地把中间件定义为平台＋通信。这个定义限定了只有用于分布式系统中的此类软件才能被称为中间件，同时此定义还可以把中间件与支撑软件和实用软件区分开来。

　　中间件处于操作系统软件与用户的应用软件的中间。中间件在操作系统、网络和数据库之上，应用软件的下层，总的作用是为处于自己上层的应用软件提供运行与开发的环境，帮助用户灵活、高效地开发和集成复杂的应用软件。形象地说就是上下之间的中间。

　　此外，中间件主要为网络分布式计算环境提供通信服务、交换服务、语义互操作服务等系统之间的协同集成服务，解决系统之间的互连互通问题。形象地说就是所谓左右之间的中间。

　　要深入理解什么是中间件，形式化的定义固然重要，我们还得从概念本身去深入理解其核心特征才是最重要的。要理解一个概念，从内涵和外延两个方面去描述是哲学上非常重要的一套方法体系。

****3.1 中间件的特征（内涵）****

总结分析，中间件有几个非常重要的特征是必须具备的：

* ****平台化****

　　所谓平台就是能够独立运行并自主存在，为其所支撑的上层系统和应用提供运行所依赖的环境。显然，不是所有的系统或者应用都可以称之为平台的。中间件是一个平台，因此中间件是必须独立存在，是运行时刻的系统软件，它为上层的网络应用系统提供一个运行环境，并通过标准的接口和API来隔离其支撑的系统，实现其独立性，也就是平台性。

　　因此，目前许多的开发语言、组件库和各种报表设计之类的软件，很难满足平台性，将这类软件叫中间件，是很不合适的。例如，JAVA是一种语言，这种语言的开发工具和开发框架，如Eclipse、JBuilder、Struts，Hibernate等等就不能称为中间件，充其量叫中间件开发工具，而不能叫中间件本身，就如同各种建筑工程设备和机械，如吊臂、搅拌机等不能叫建筑，而只能成为建筑工具一样。而J2EE应用服务器提供JAVA应用的运行环境，就是经典的中间件。

* ****应用支撑****

　　中间件的最终目的是解决上层应用系统的问题，而且也是软件技术发展到今天对应用软件提供最完善彻底的解决方案。

　　高级程序设计语言的发明，使得软件开发变成一个独立的科学和技术体系，而操作系统平台的出现，使得应用软件通过标准的API接口，实现了软件与硬件的分离。

　　现代面向服务的中间件在软件的模型、结构、互操作以及开发方法等四个方面提供了更强的应用支撑能力：

　　模型：构件模型弹性粒度化，即通过抽象层度更高的构件模型，实现具备更高结构独立性、内容自包含性和业务完整性的可复用构件，即服务。并且在细粒度服务基础上，提供了更粗粒度的服务封装方式，即业务层面的封装，形成业务组件，就可以实现从组件模型到业务模型的全生命周期企业建模的能力。

　　结构：结构松散化，即，将完整分离服务描述和服务功能实现以及服务的使用者和提供者，从而避免分布式应用系统构建和集成时常见的技术、组织、时间等不良约束。

　　互操作：交互过程标准化，即，将与互操作相关的内容进行标准化定义，如服务封装、描述、发布、发现、调用等契约，通信协议以及数据交换格式等等。最终实现访问互操作、连接互操作和语义互操作。

　　开发集成方法：应用系统的构建方式由代码编写转为主要通过服务间的快捷组合及编排，完成更为复杂的业务逻辑的按需提供和改善，从而大大简化和加速应用系统的搭建及重构过程。

　　而要最终解决软件的质量问题、效率问题、互操作问题、灵活应变问题这四大问题，需要在软件技术的内在结构(Structure)、架构(Architecture)层面进行思考。

　　解决这些问题，技术的本质是复用、松耦合、互操作(标准)等软件技术的内在机制。这也是中间件技术和产品的本质特征。

* ****软件复用****

　　软件复用，即软件的重用，也叫再用，是指同一事物不作修改或稍加改动就多次重复使用。从软件复用技术的发展来看，就是不断提升抽象级别，扩大复用范围。最早的复用技术是子程序，人们发明子程序，就可以在不同系统之间进行复用了。但是，子程序是最原始的复用，因为这种复用范围是一个可执行程序内复用，静态开发期复用，如果子程序修改，意味着所有调用这个子程序的程序必须重新编译、测试和发布。

****复用对象复用范围****

|  |  |
| --- | --- |
| **复用对象** | **复用范围** |
| 子程序 | 一个可执行程序内复用，静态开发期 复用 |
| 组件(DLL,Com等) | 系统内复用，动态运行期复用 |
| 企业对象组件（Com+,.NET,EJB等) | 企业网络内复用，不同系统之间复用 |
| 服务 (如WebService,SCA/SDO) | 不同企业之间，全球复用，动态可配置 |

　　为了解决这个问题，发明了组件（或者叫控件），如MS操作系统下的DLL组件。组件将复用提升了一个层次，因为组件可以在一个系统内复用（同一种操作系统），而且是动态、运行期复用。这样组件可以单独发展，组件与组件调用者之间的耦合度降低。

　　为解决分布式网络计算之间的组件复用，人们发明了企业对象组件，如（COM+, .NET, EJB等)，或者叫分布式组件。通过远程对象代理，来实现企业网络内复用，不同系统之间复用。

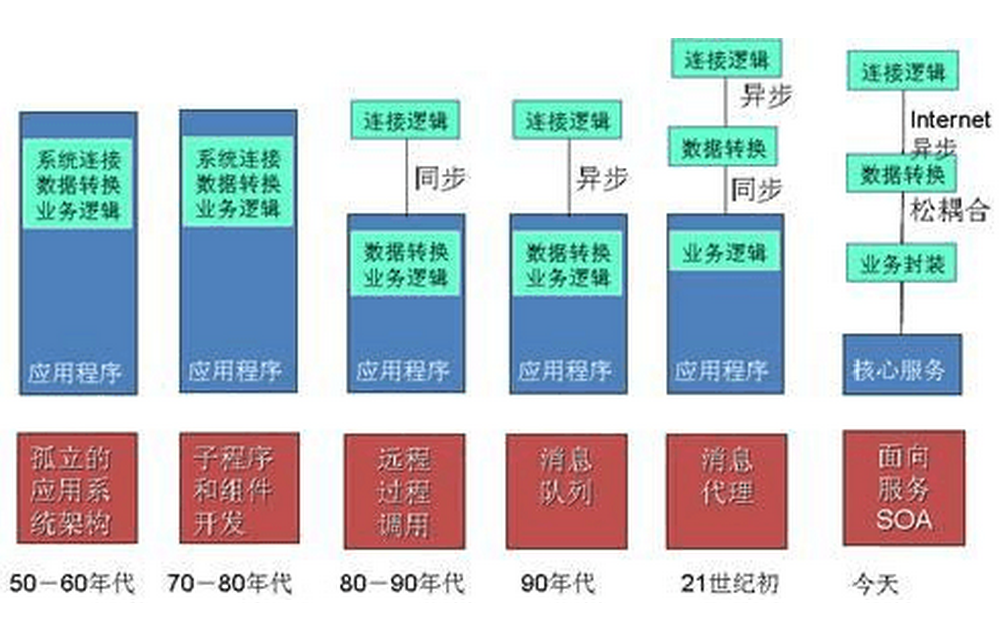
　　传统中间件的核心是组件对象的管理。但分布式组件也是严重依赖其受控环境，由于构件实现和运行支撑技术之间存在着较大的异构性，不同技术设计和实现的构件之间无法直接组装式复用。

　　而现代中间件的发展重要趋势就是以服务为核心，如WebService, SCA/SDO等。通过服务，或者服务组件来实现更高层次的复用、解耦和互操作，即SOA架构中间件。

　　因为服务是通过标准封装，服务组件之间的组装、编排和重组，来实现服务的复用。而且这种复用，可以在不同企业之间，全球复用，达到复用的最高级别，并且是动态可配置的复用。

* ****耦合关系****

基于SOA架构的中间件，在松耦合解耦过程也发展到了最后的境界。传统软件将软件之中核心三部分——网络连接、数据转换、业务逻辑全部耦合在一个整体之中，形成铁板一块的软件，牵一发而动全身，软件就难以适应变化。分布式对象技术将“连接逻辑”进行分离，消息中间件将“连接逻辑”进行异步处理，增加了更大的灵活性。消息代理和一些分布式对象中间件将数据转换也进行了分离。而SOA架构，通过服务的封装，实现了业务逻辑与网络连接、数据转换等进行完全的解耦。



软件技术的不断解耦的过程

# Other

## mq

# Ref

浅析深究什么是中间件\_知识库\_博客园.htm