面向服务的架构（SOA）

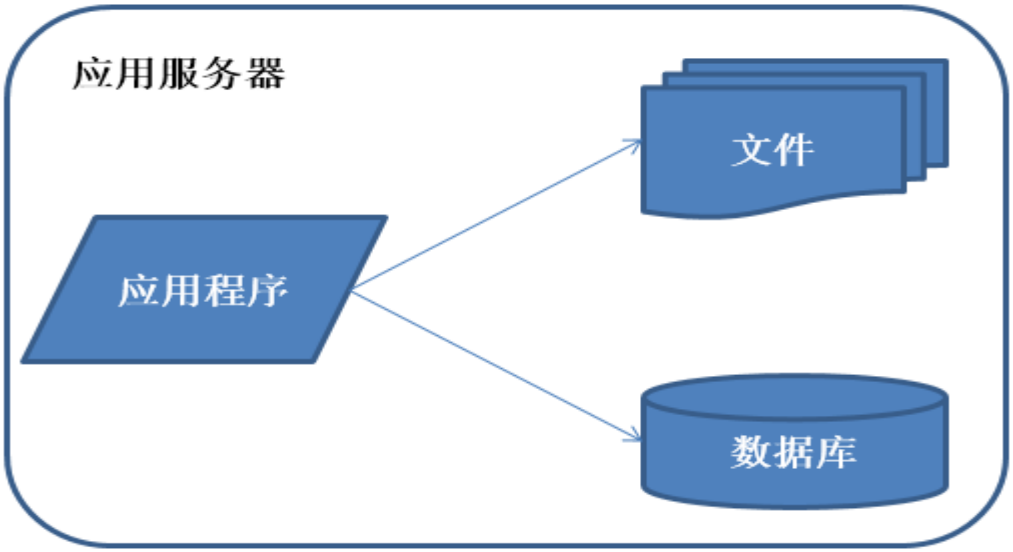
lliao@seu.edu.cn

目录

- 发展历史及背景
- SOA基本概念
- SOA核心思想
- 构建SOA
- SOA的应用

发展历史及背景

- 系统架构演化历程-初始阶段架构

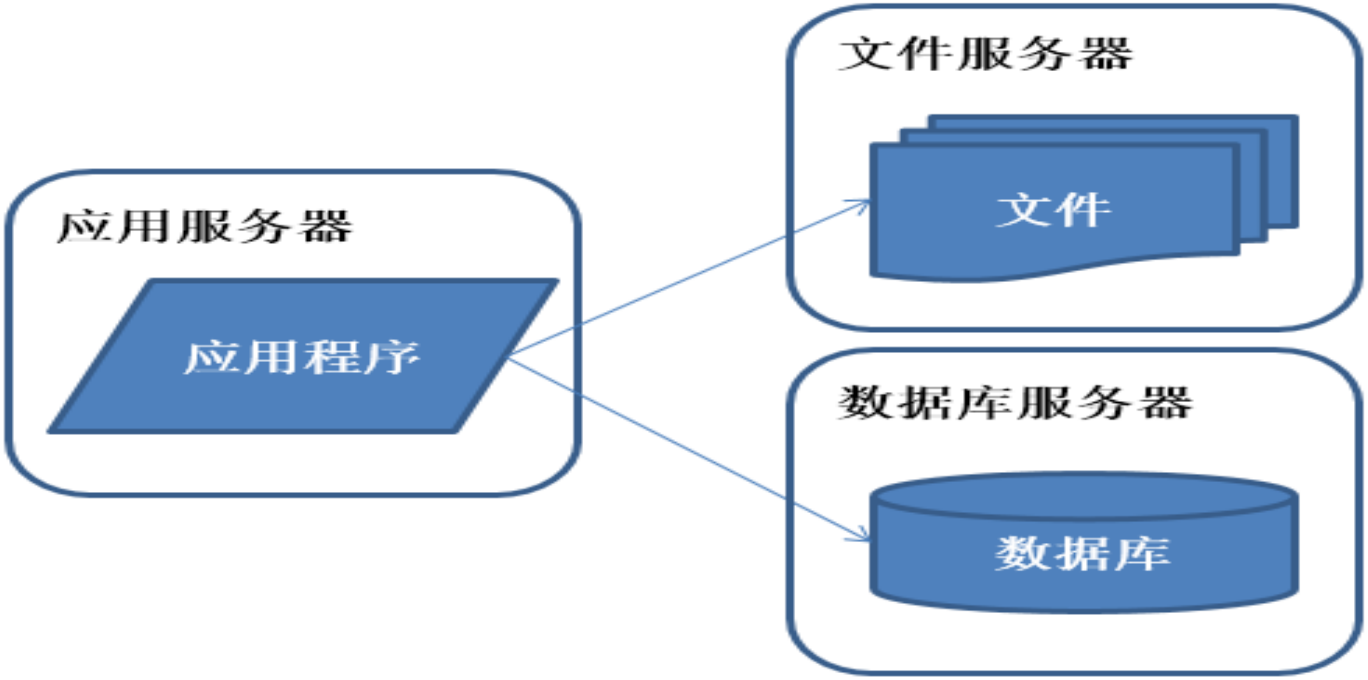


应用程序、数据库、文件等所有的资源都在一台服务器上。

通常服务器操作系统使用linux，应用程序使用PHP开发，然后部署在Apache上，数据库使用Mysql，汇集各种免费开源软件以及一台廉价服务器就可以开始系统的发展之路了。LAMP、J2EE、.Net等。

发展历史及背景

- 系统架构演化历程-应用服务和数据服务分离

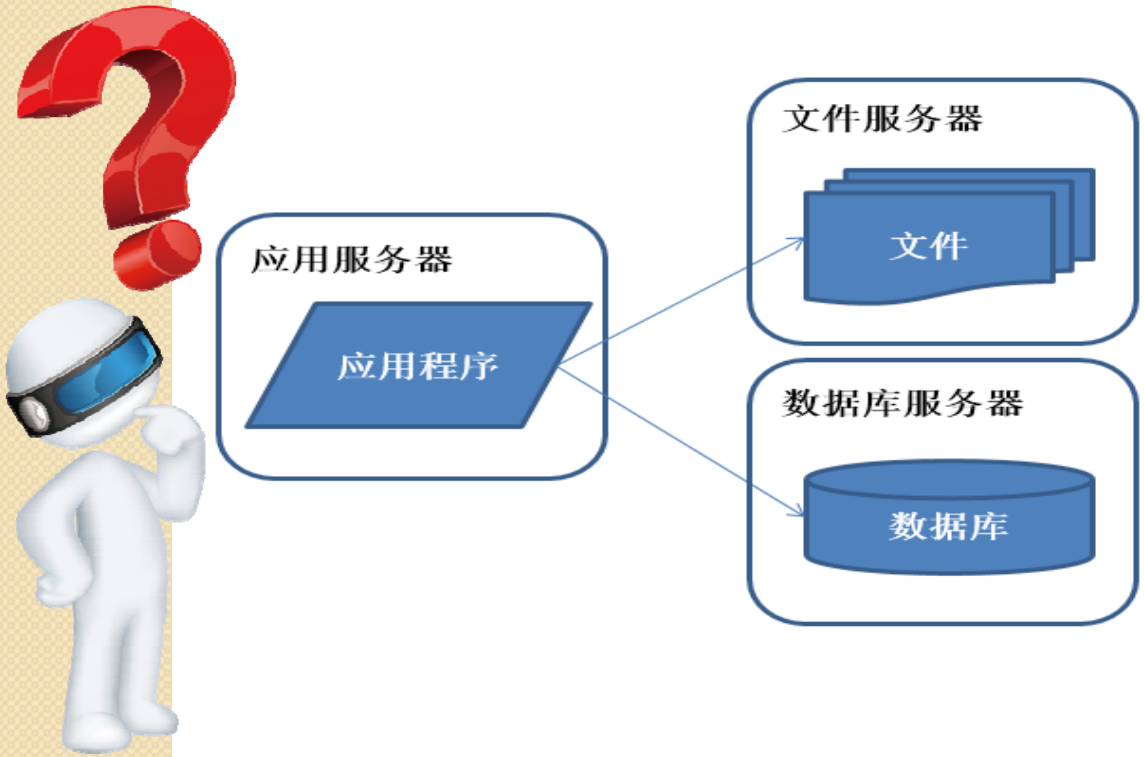


应用程序、数据库、文件分别部署在独立的资源上。

数据量增加，单台服务器性能及存储空间不足，需要将应用和数据分离，并发处理能力和数据存储空间得到了很大改善。

发展历史及背景

- 系统架构演化历程-使用缓存改善性能

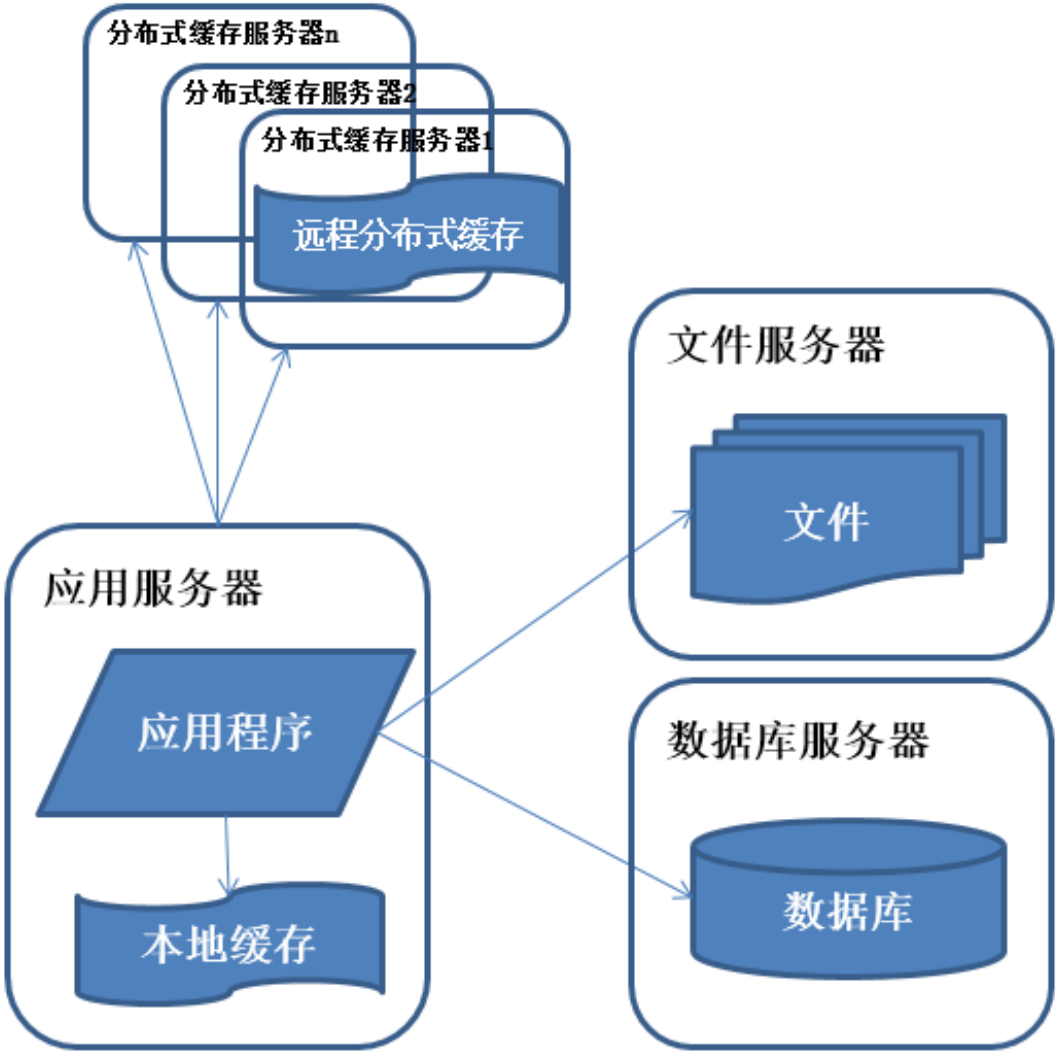


数据库中访问较集中的一小部分数据存储在缓存服务器中，减少数据库的访问次数，降低数据库的访问压力。

系统访问特点遵循二八定律，即80%的业务访问集中在20%的数据上。缓存分为本地缓存和远程分布式缓存，本地缓存访问速度更快但缓存数据量有限，同时存在与应用程序争用内存的情况。

发展历史及背景

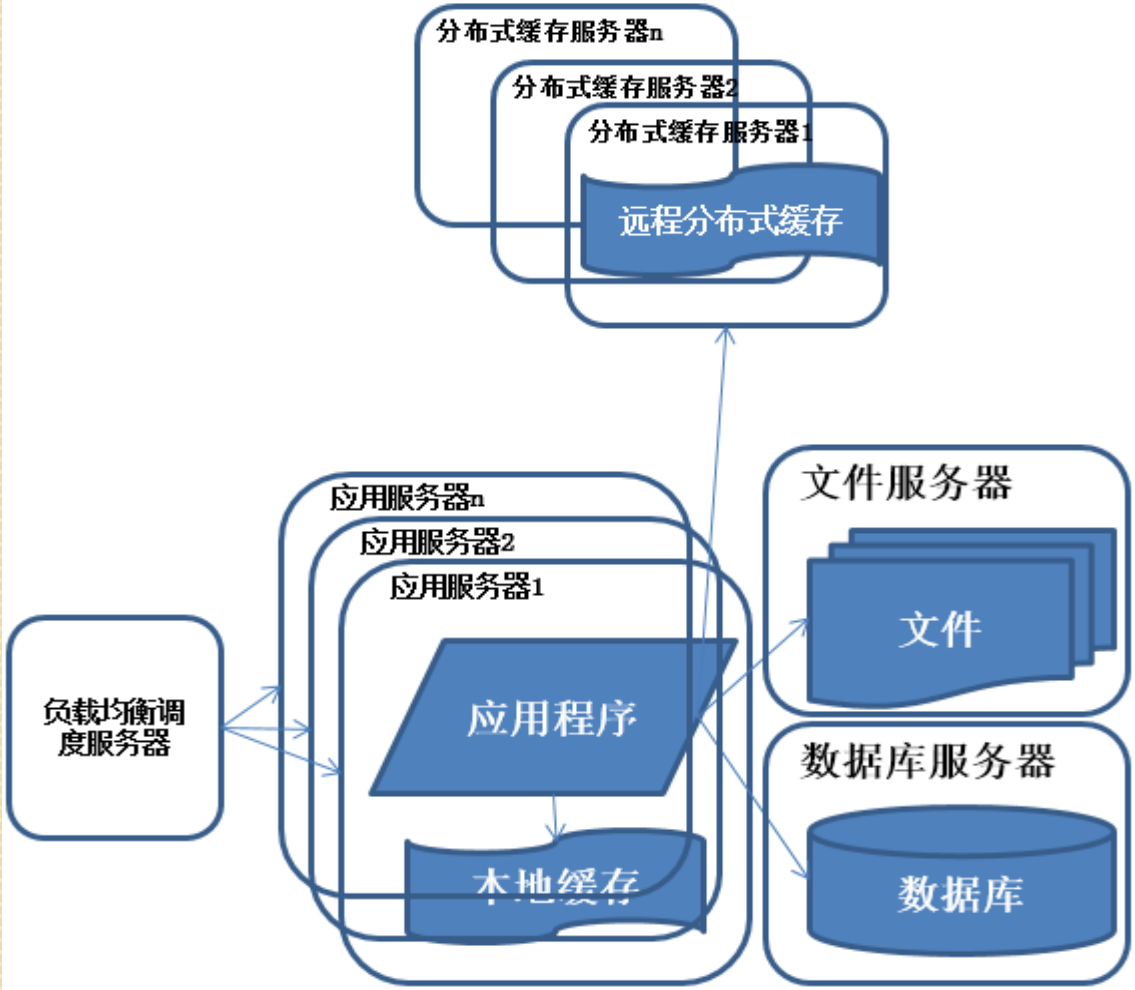
- 系统架构演化历程-使用缓存改善性能



在做完分库分表这些工作后，数据库上的压力已经降到比较低了，又开始过着每天看着访问量暴增的幸福生活了，突然有一天，发现系统的访问又开始有变慢的趋势了，这个时候首先查看数据库，压力一切正常，之后查看webserver，发现apache阻塞了很多的请求，而应用服务器对每个请求也是比较快的，看来 是请求数太高导致需要排队等待，响应速度变慢

发展历史及背景

- 系统架构演化历程-使用应用服务器集群



多台服务器通过负载均衡同时向外部提供服务，解决单台服务器处理能力和存储空间上限的问题。

使用集群是系统解决高并发、海量数据问题的常用手段。通过向集群中追加资源，提升系统的并发处理能力，使得服务器的负载压力不再成为整个系统的瓶颈。

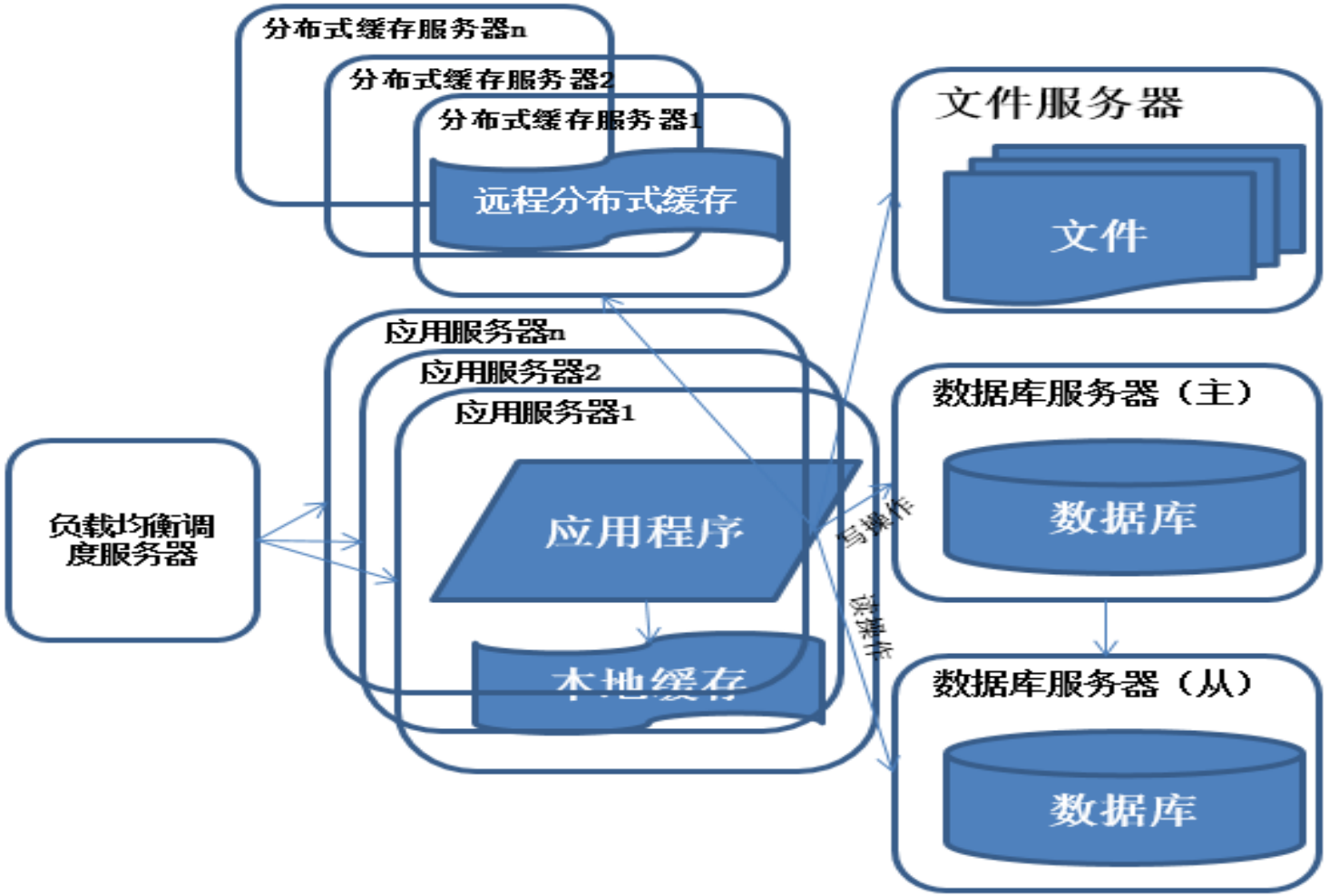
发展历史及背景

系统架构演化历程-使用应用服务器集群

享受了一段时间的系统访问量高速增长的幸福后，发现系统又开始变慢了，这次又是什么状况呢，经过查找，发现数据库写入、更新的这些操作的部分数据库连接的资源竞争非常激烈，导致了系统变慢

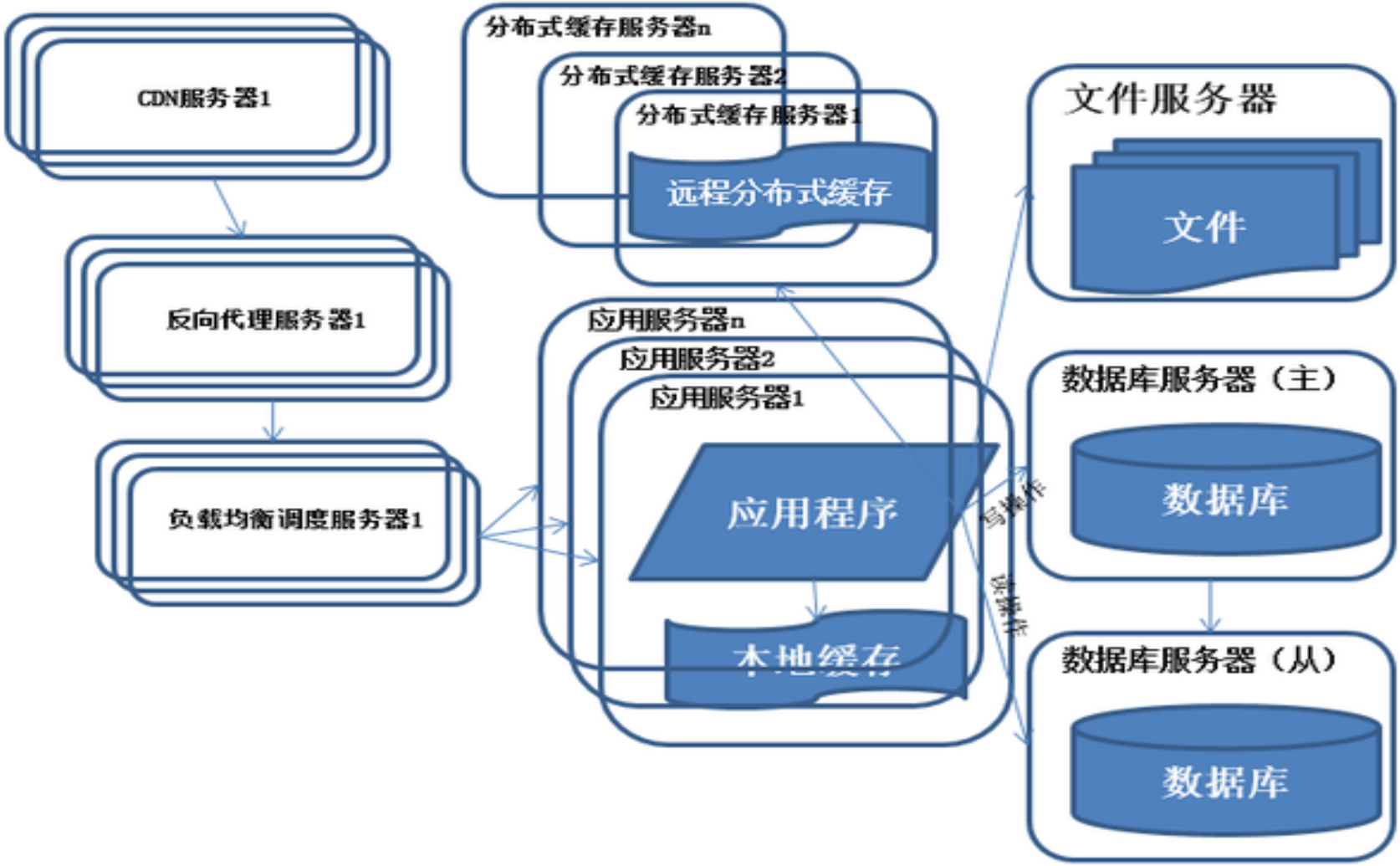
发展历史及背景

- 系统架构演化历程-数据库读写分离



发展历史及背景

- 系统架构演化历程-反向代理和CDN加速



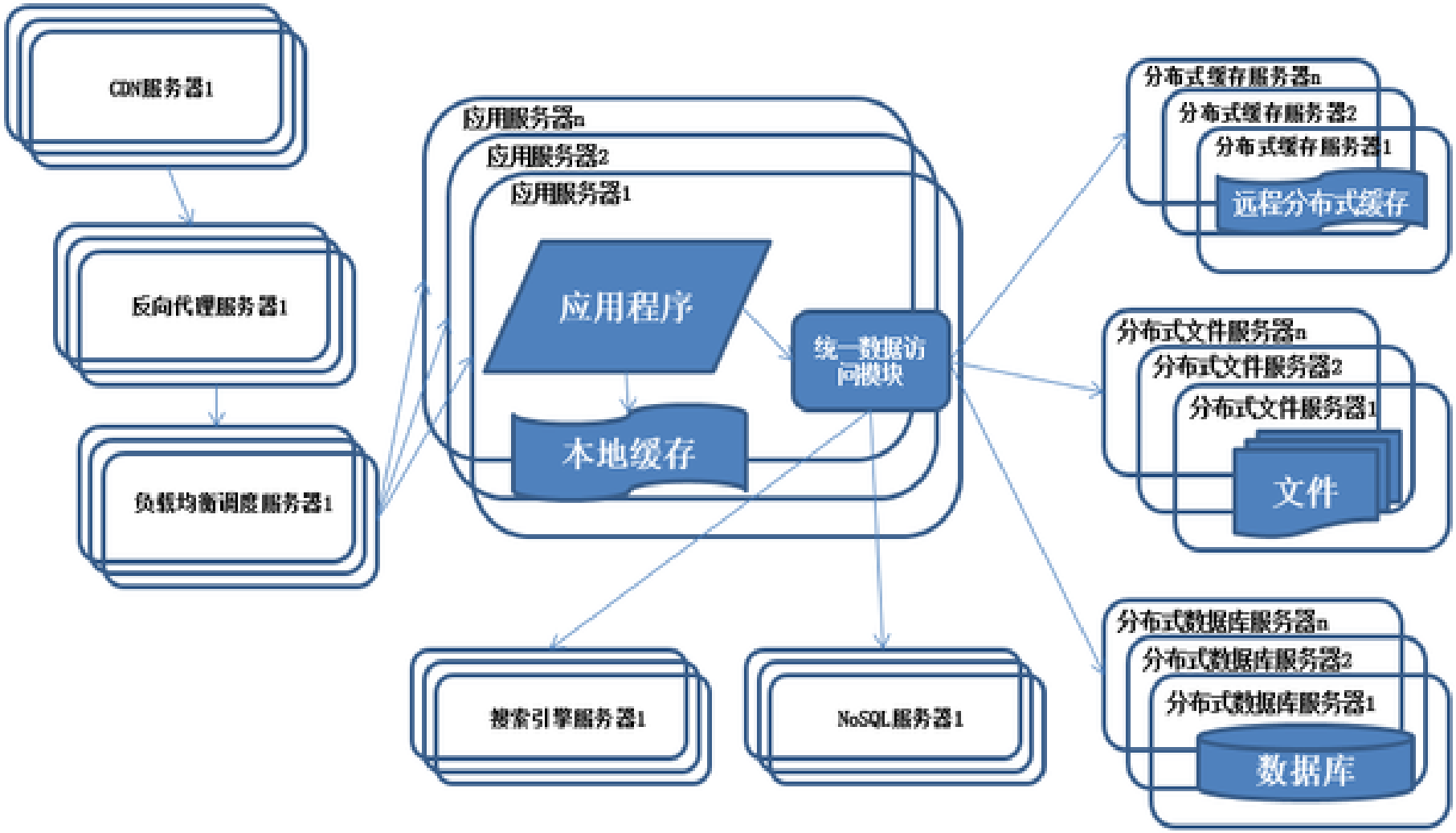
发展历史及背景

- 系统架构演化历程-分布式文件系统和分布式数据库

任何强大的单一服务器都满足不了大型系统持续增长的业务需求，数据库读写分离随着业务的发展最终也将无法满足需求，需要使用分布式数据库及分布式文件系统来支撑。分布式数据库是关系数据库拆分的最后方法，只有在单表数据规模非常庞大的时候才使用，更常用的数据库拆分手段是业务分库，将不同的业务数据库部署在不同的物理服务器上。

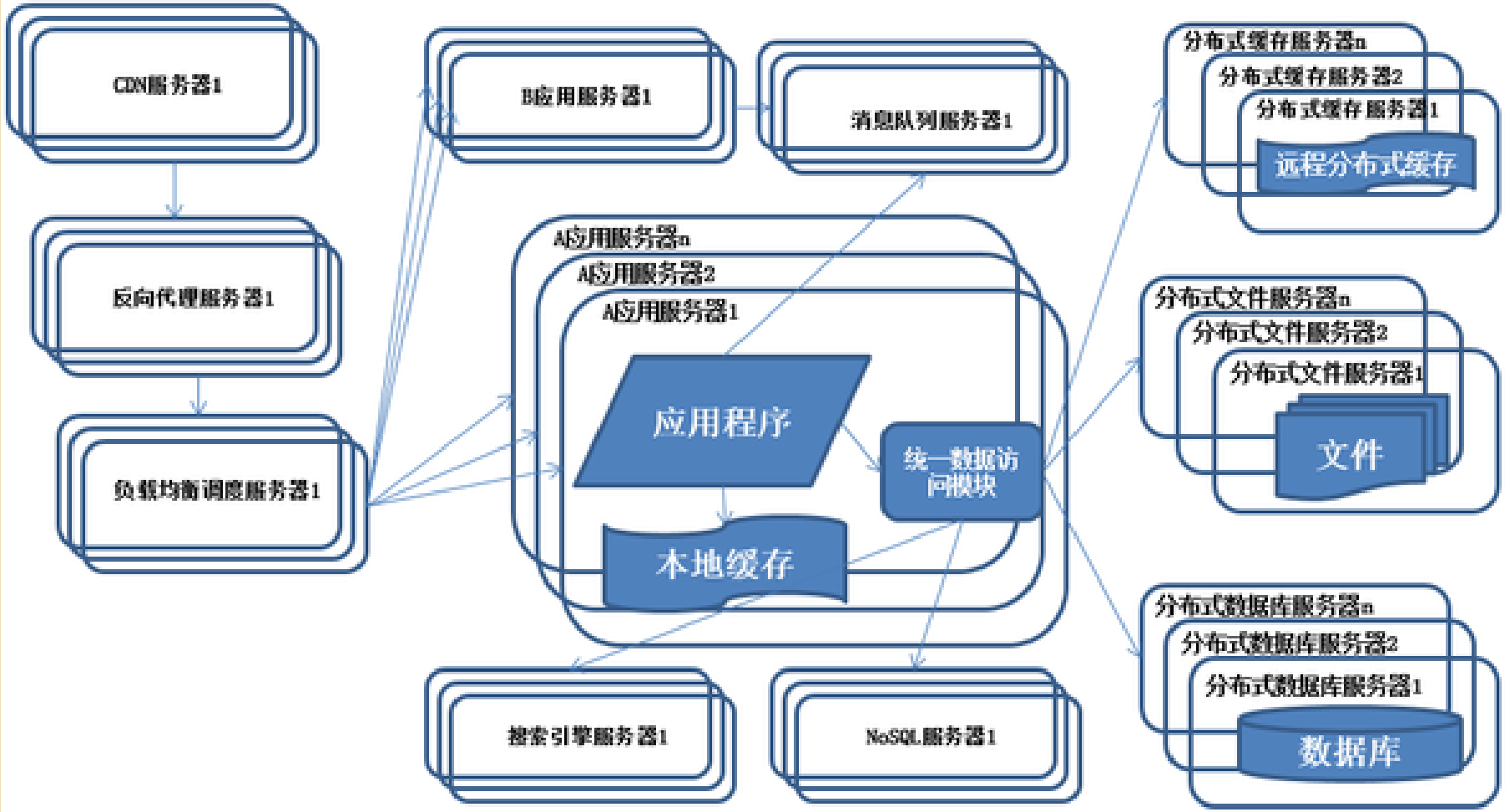
发展历史及背景

- 系统架构演化历程-使用**NoSQL**和搜索引擎



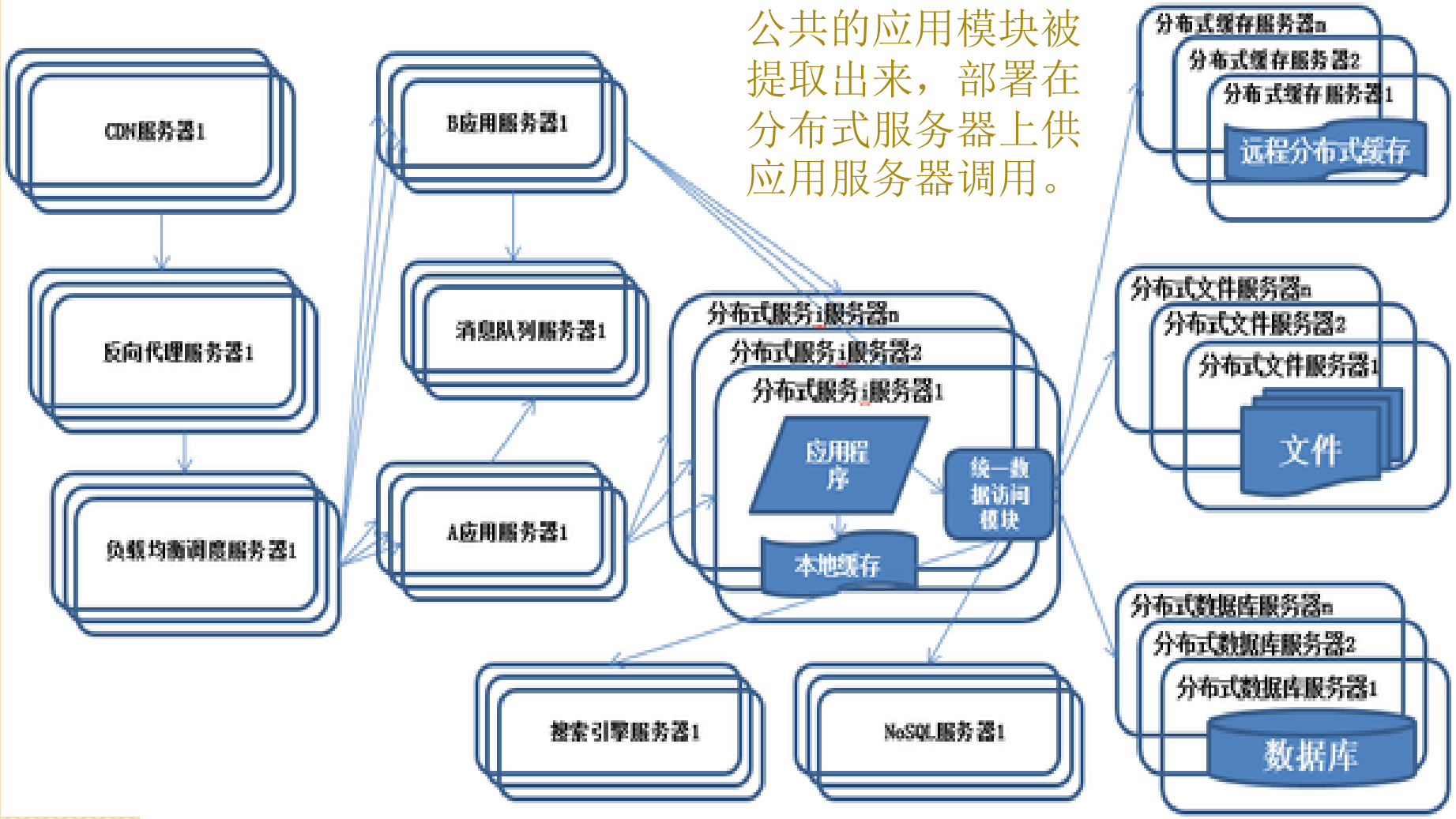
发展历史及背景

- 系统架构演化历程-业务拆分（横向拆分）



发展历史及背景

- 系统架构演化历程-分布式服务（横向拆分）



发展历史及背景

- 为什么需要SOA？

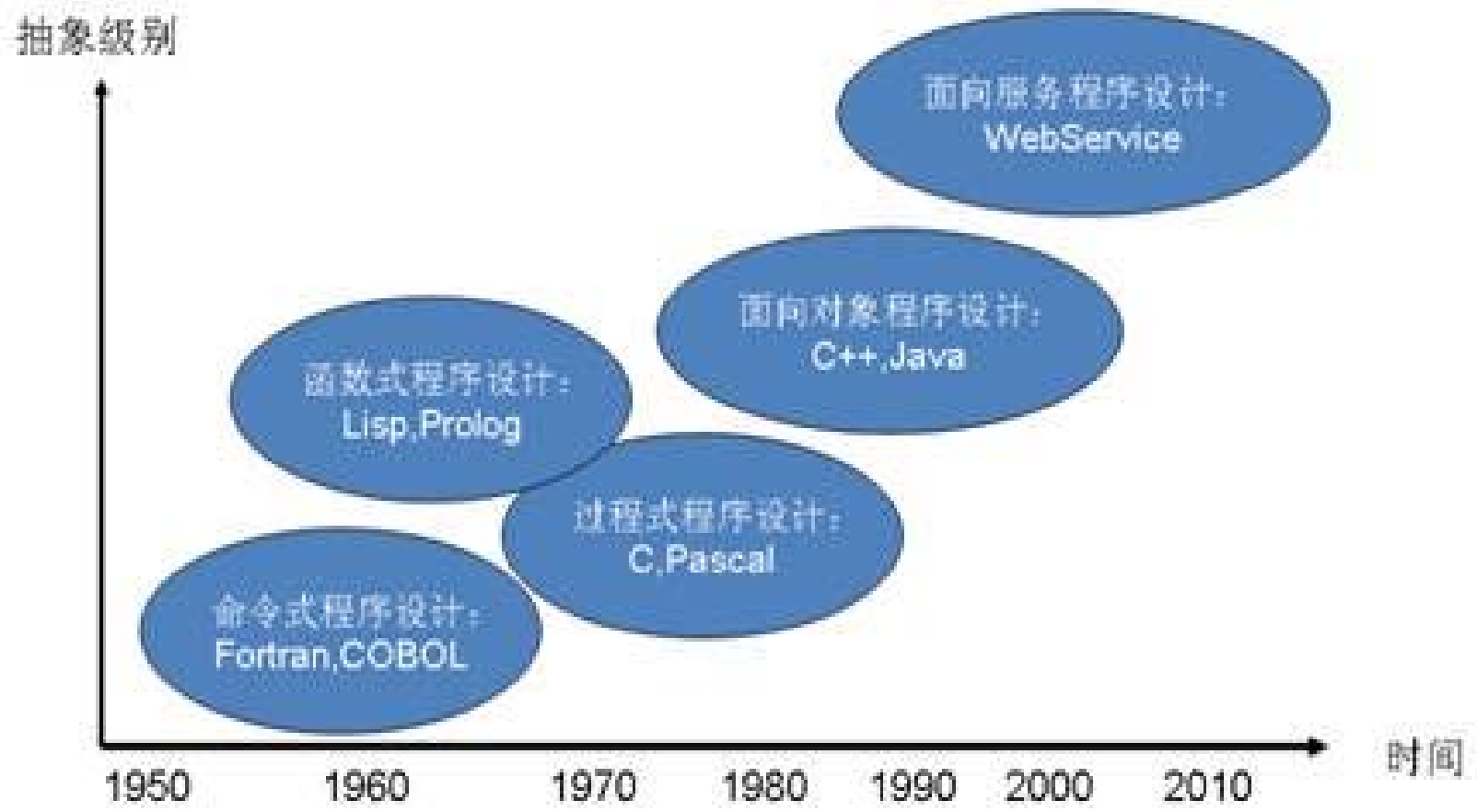
软件应用领域越来越多，相同领域的应用系统之间许多基础功能和结构是有相似性的，每次开发系统都从零开始绝对不是一种好的方法，也是对质量和效率的很大的伤害。

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SOA发展的重要推动力。

软件技术发展内容，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SOA解决方案之中。

发展历史及背景

- 为什么需要**SOA**？



基本概念

- 如何准确理解**SOA**？

OASIS（一个SOA标准组织）给出的SOA定义“SOA是一个范式，用于组织和利用可能处于不同所有权范围控制下的分布式系统。”

维基百科给出的SOA定义“面向服务的体系结构（Service-oriented architecture）是构造分布式系统的应用程序的方法。它将应用程序功能作为服务发送给最终用户或者其他服务。它采用开放标准、与软件资源进行交互并采用表示的标准方式”。

SOA是包含运行环境、编程模型、架构风格和相关方法论等在内的一整套新的分布式软件系统构造方法和环境，涵盖服务的整个生命周期：建模-开发-整合-部署-运行-管理。

基本概念

- **WebService**

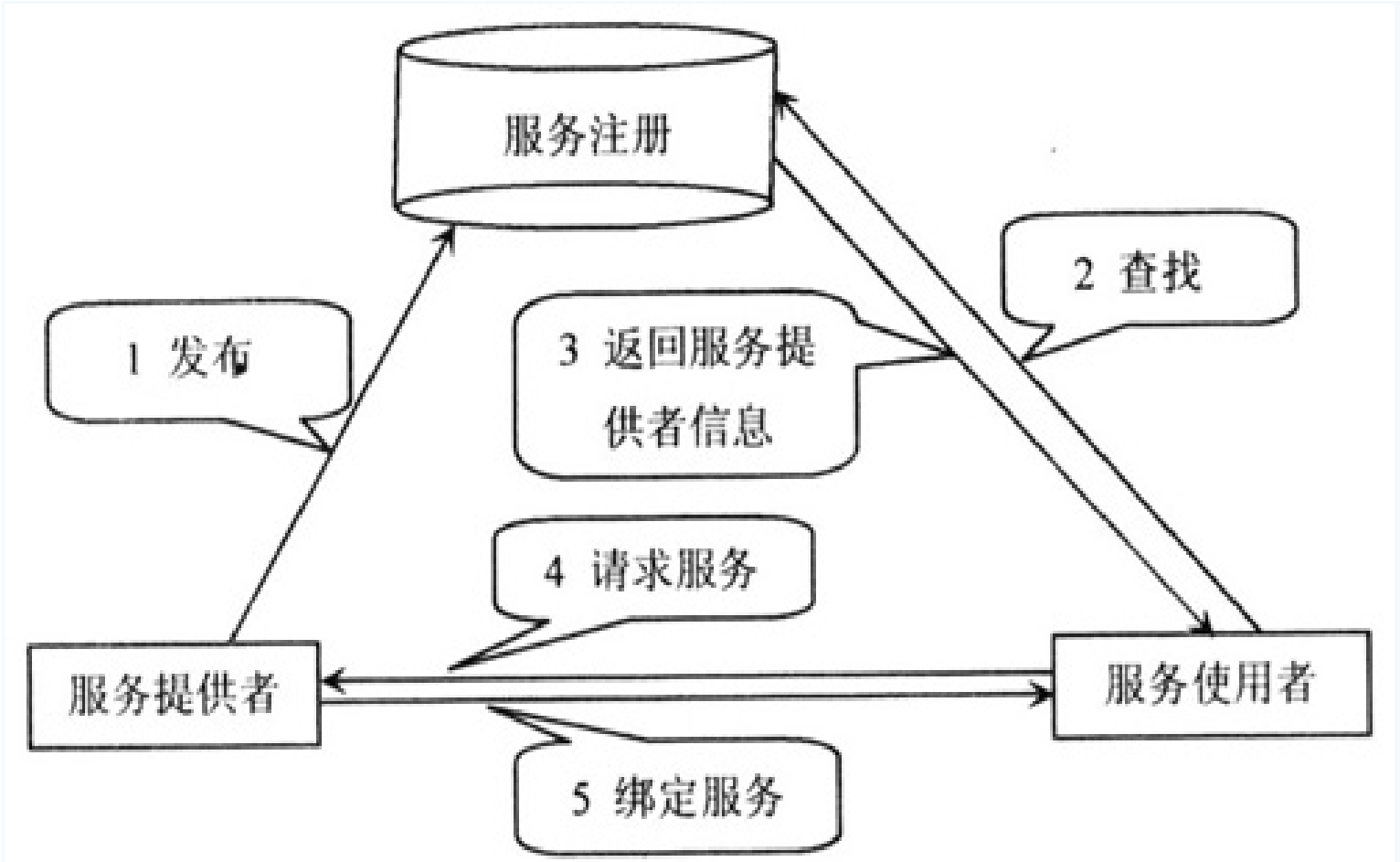
Web Services，从字面上理解就是通过Web提供的服务。我们可以理解Web Services是自包含的、模块化的应用程序，它可以在网络(通常为Web)中被描述、发布、查找以及调用；

也可以理解Web Services是基于网络的、分布式的模块化组件，它执行特定的任务，遵守具体的技术规范，这些规范使得Web Sevices能与其他兼容的组件进行互操作；

也可以这样理解，所谓Web服务，它是指由企业发布的完成其特别商务需求的在线应用服务，其他公司或应用软件能够通过Internet来访问并使用这项应用服务Web 。

service 简单来说就是一个向外界暴露出的能够通过internet进行调用的api和应用程序， 是基于SOA松耦合等思想开发出来的一套技术,但是它并一定完全符合SOA的架构。

SOA工作流程



SOA 中的不同组件及其工作流程

SOA角色

SOA架构中有三种角色：

- 服务提供者：发布自己的服务，并且对服务请求进行响应。
- 服务注册中心：注册已经发布的web service，对其进行分类，并提供搜索服务。
- 服务请求者：利用服务中心查找所需要的服务，然后使用该服务。

SOA操作

SOA的三种操作：

- 发布操作：为了使服务可访问，需要发布服务描述以使服务使用者可以发现它。

- 查找操作：服务请求者定位服务，方法是查询服务注册中心来找到满足其标准的服务。

- 绑定操作：在检索到服务描述之后，服务使用者继续根据服务描述中的信息来调用服务。

SOA的相关标准——WSDL、UUDI、SOAP

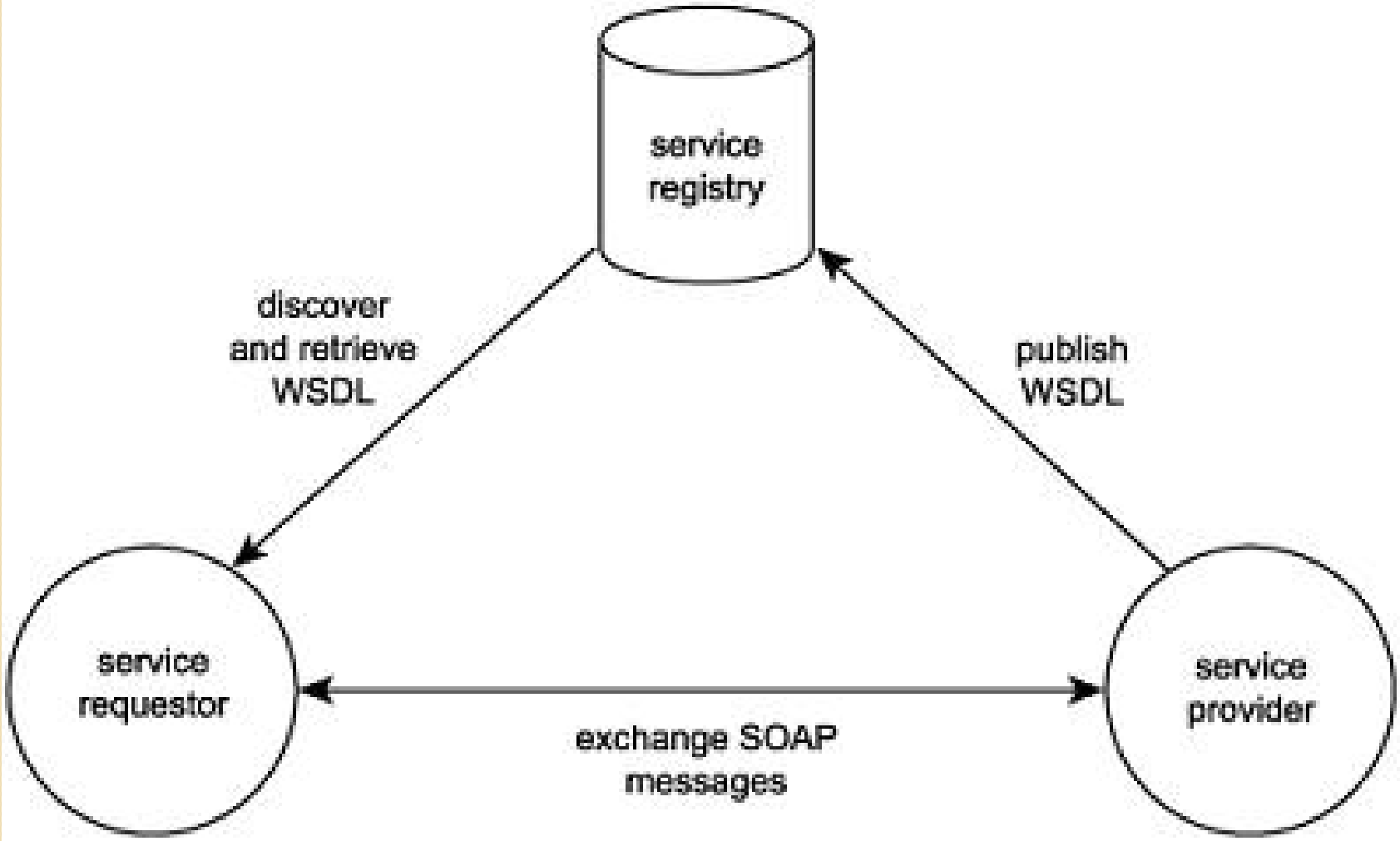
SOAP: 简单对象访问协议 (Simple Object Access Protocol)

WSDL: Web服务描述语言 WSDL (Web Services Description Language)

UDDI: 统一描述、发现和集成 (Universal Description,Discovery and Integration)

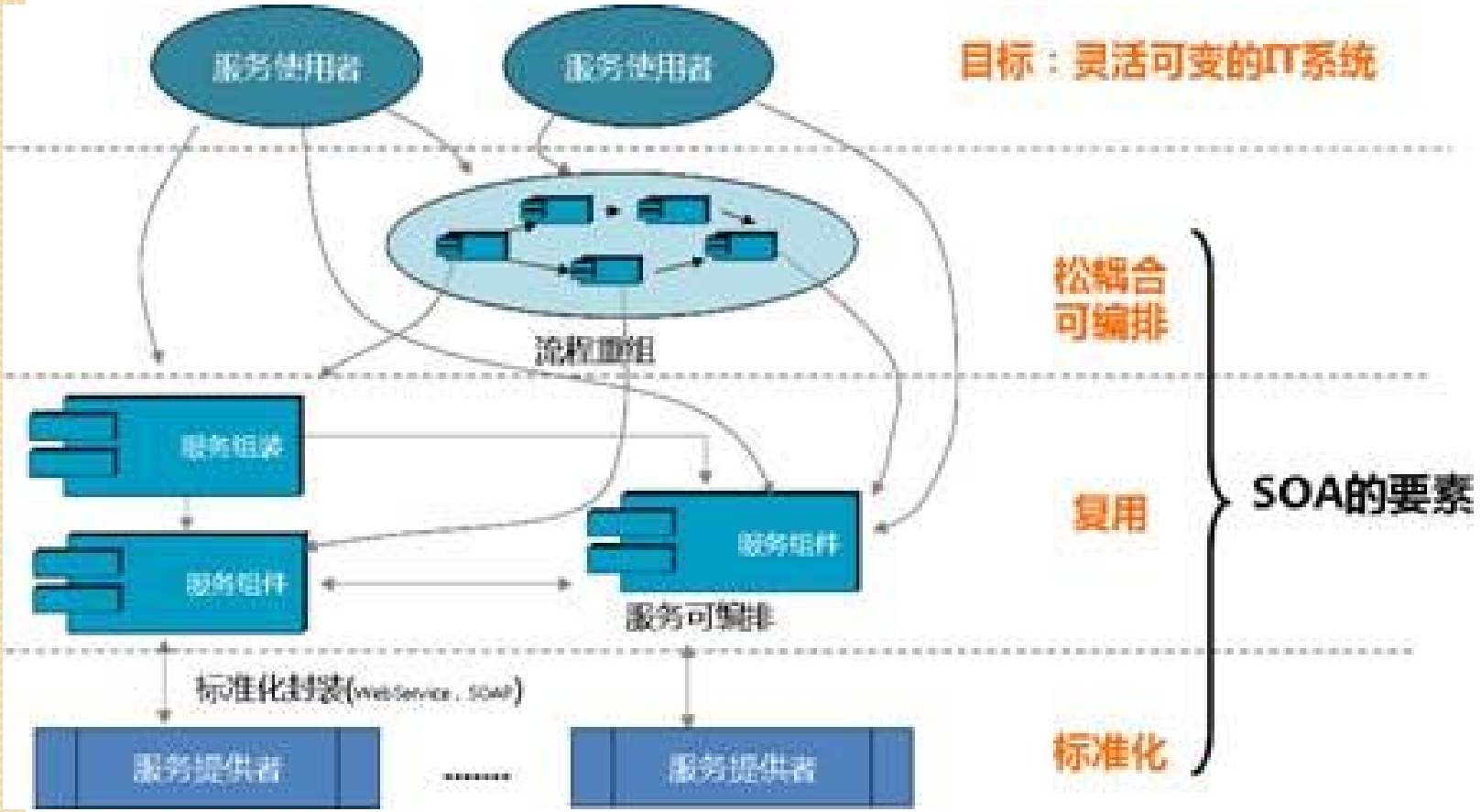
WSDL用来描述服务；UDDI用来注册和查找服务；

SOAP作为传输层，用来在消费者和服务提供者之间传送消息。一个消费者可以在UDDI注册表（registry）查找服务，取得服务的WSDL描述，然后通过SOAP来调用服务。



soap用来描述传递信息的格式， WSDL 用来描述如何访问具体的接口， uddi用来管理，分发，查询webService 。具体实现可以搜索Web Services简单实例

核心思想


服务使用者
服务使用者
目标：灵活可变的IT系统
流程重组
松耦合
可编排
服务组装
服务组件
服务组件
复用
SOA的要素
服务可编排
标准化封装(Web Service，SOAP)
服务提供者
.....
服务提供者
标准化

核心思想

- 标准化封装（满足互操作性）

在软件的互操作方面，传统中间件只是实现了访问互操作，即通过标准化的API实现了同类系统之间的调用互操作，而连接互操作还是依赖于特定的访问协议，如JAVA使用RMI等。而SOA通过标准的、支持Internet、与操作系统无关的**SOAP协议**实现了连接互操作。而且，服务的封装是采用XML协议，具有自解析和自定义的特性，这样，基于SOA的中间件还可以实现语义互操作。

SOA要实现互操作，就是通过一系列的标准族，来实现访问、连接和语义等各种层面的互操作。

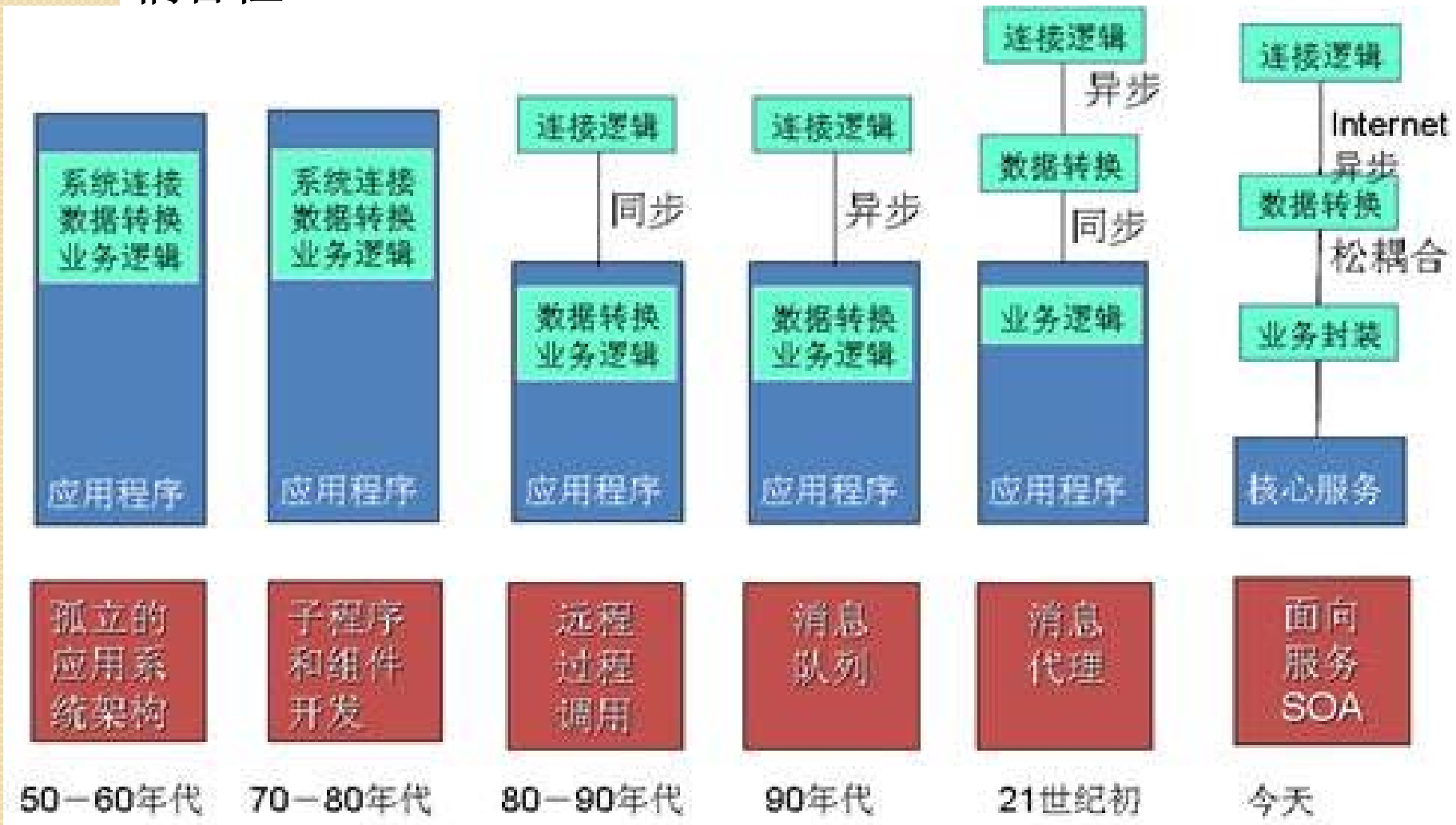
核心思想

- 复用

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

核心思想

- 耦合性


系统连接
数据转换
业务逻辑
应用程序
系统连接
数据转换
业务逻辑
应用程序
连接逻辑
同步
数据转换
业务逻辑
应用程序
连接逻辑
异步
数据转换
业务逻辑
应用程序
连接逻辑
异步
数据转换
同步
业务逻辑
应用程序
连接逻辑
Internet
异步
数据转换
松耦合
业务封装
核心服务
孤立的
应用系
统架构
子程序
和组件
开发
远程
过程
调用
消息
队列
消息
代理
面向
服务
SOA
50—60年代
70—80年代
80—90年代
90年代
21世纪初
今天

如何构建SOA

一个SOA系统要具有以下六大关键要素——基础设施、已有资源、企业服务、流程模型、服务展现和系统工具（包括开发、测试和管理工具等）。因此，在基础设施和已有资源都已具备的基础上，开发和构建一个SOA系统要包括以下几方面的工作：

- 首先需要设计开发出符合标准的服务，这是整个SOA系统最核心的要素。
- 基于标准服务，借助流程编排工具和建模工具，组织构造流程，生成流程模型，更好地满足业务需求。
- 实际构建和开发SOA系统，具体包括服务和应用程序的开发，数据的访问、处理和管理，及对服务各种形式的展现等。

使用SOA进行服务组合实例

假设股票行业存在以下6个服务：

- **Country**（）

输入参数：国家编码。输出项：国家名称和其他信息。

- **YellowPages**（）

输入参数：公司名称；输出项：企业代码，所在国家等其他信息。

- **NewYorkStock**（）

输入参数：公司代码，时间；输出项：该公司在纽约的股票价格（美元）。

- **LondonStock**（）

输入参数：公司代码，时间；输出项：该公司在伦敦的股票价格。

- **USToRMB**（）

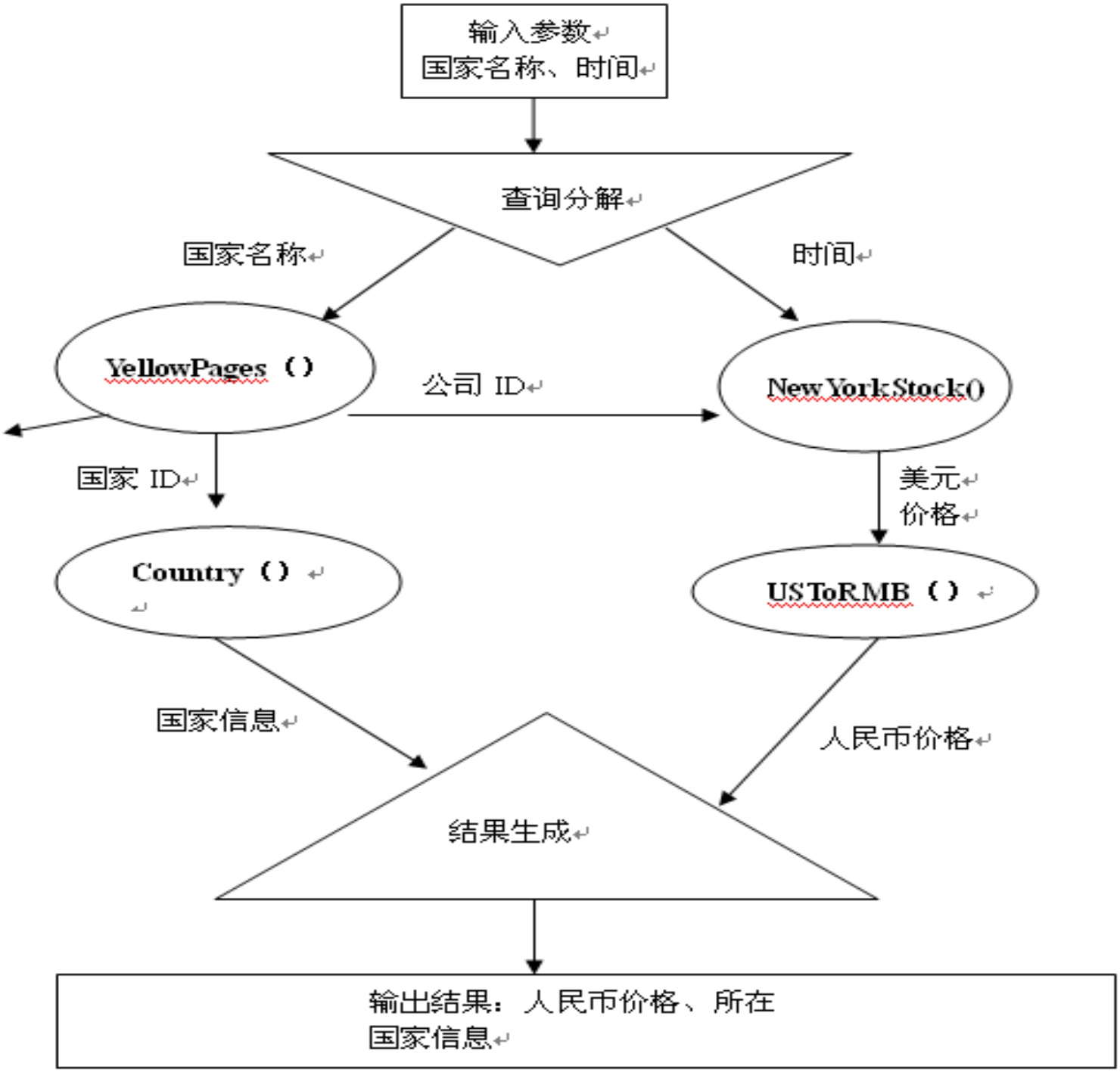
输入参数：美元价格，时间；输出项：对应的人民币价格。

- **UKToRMB**（）

输入参数：英镑价格，时间；输出项：对应的人民币价格。

使用SOA进行服务组合实例

- 用户想通过“跨国公司名称”和“时间”找出该跨国公司在纽约的股票折合成人民币的价格以及该公司所在国家的信息。
- 分析：
  - 输入参数：跨国公司的名称、时间
  - 输出：该公司的股票价格（RMB）以及该公司所在国家
  - 如何实现对给定服务的组合，找出满足用户的信息？

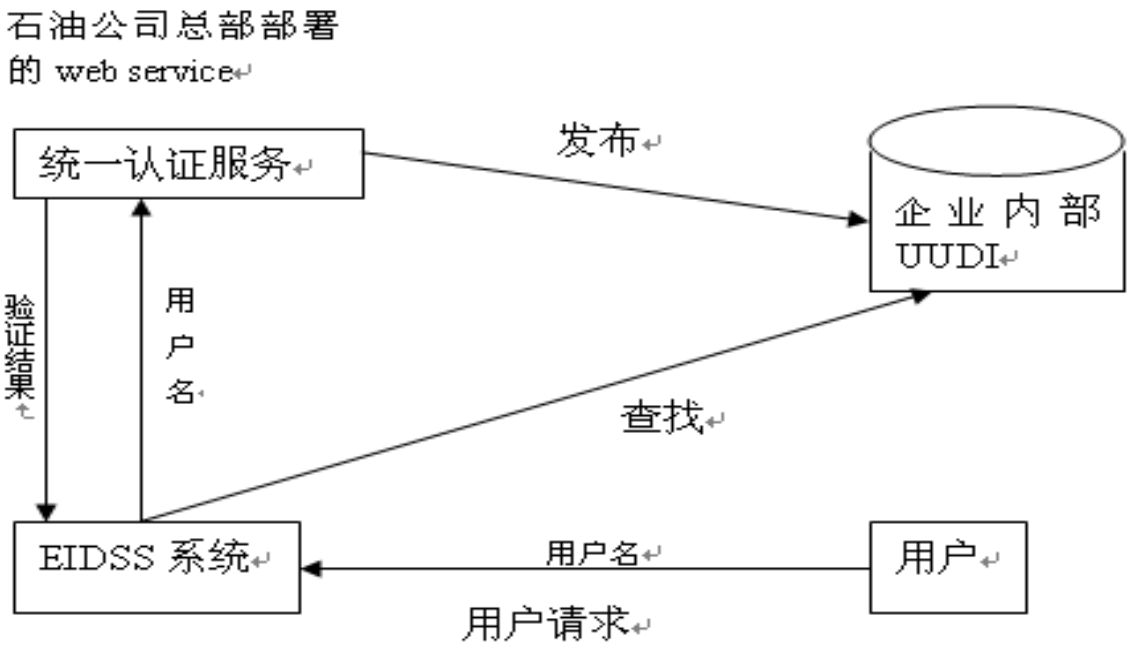

输入参数
国家名称、时间
查询分解
国家名称
时间
YellowPages（）
公司 ID
NewYorkStock()
国家 ID
美元
价格
Country（）
USToRMB（）
国家信息
人民币价格
结果生成
输出结果：人民币价格、所在
国家信息

SOA应用——统一认证

在石油企业内部，有许多不同的网站，进入每个网站，都需要身份验证，不仅浪费时间而且容易遗忘代码，另外，网站维护人员对各种服务需要建立相应的用户认证与信息管理系统，分布于个服务器中的用户数据不仅浪费维护人员的时间，而且过于分散的用户数据不利于统计和管理。用户的需求和管理要求促使用户趋于统一，产生了统一者认证。

统一认证的实现是基于SOA的架构。

SOA应用——统一认证



从中可以看出使用SOA的优点：将身份验证这一功能模块发布成一种服务，其他的软件可以通过UUDI查找该服务，然后将该服务与服务的实现进行绑定。