2.分布式体系结构

1.体系结构的样式

1.1 基本概念

软件体系结构（ Software Architecture）

- 软件的组件， 以及组件之间的相互关系

软件体系结构的要素

- 组件(component)： 模块单元， 能提供良好的接口
- 连接器（ connector） ： 实现组件间通信的机制

软件体系结构的风格（ style）

如何表示一个体系结构，常用的有 4 种

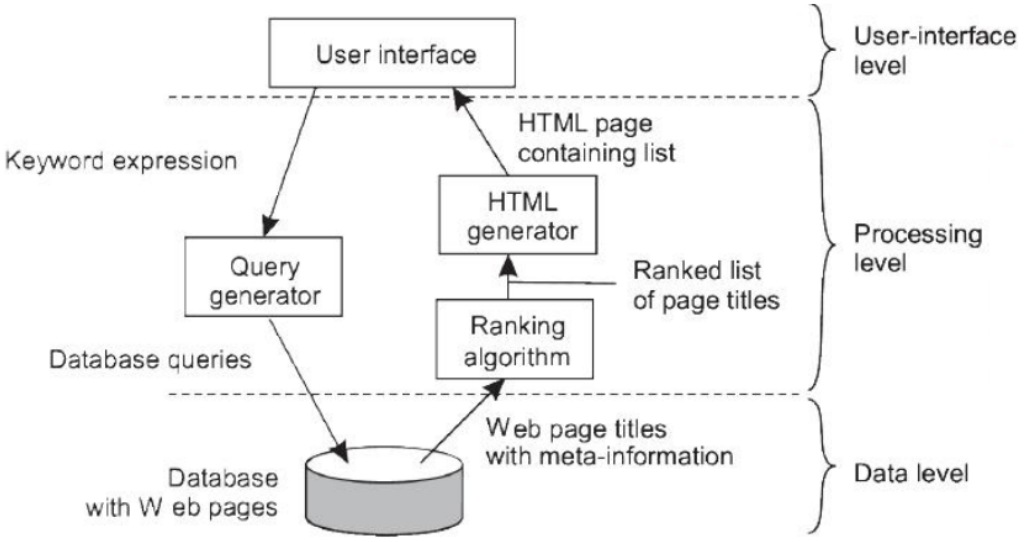
1. 层次体系结构
2. 面向对象体系结构
3. 资源为中心的体系结构
4. 面向事件的体系结构

1.2 层次型体系结构

系统由自上而下的不同层次的组件组成

只有相邻的层次可以通信

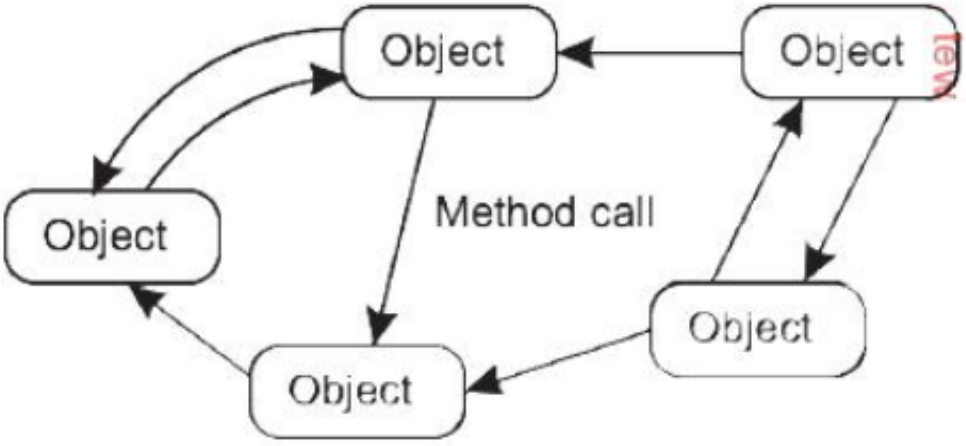
请求消息自上而下，响应自下而上



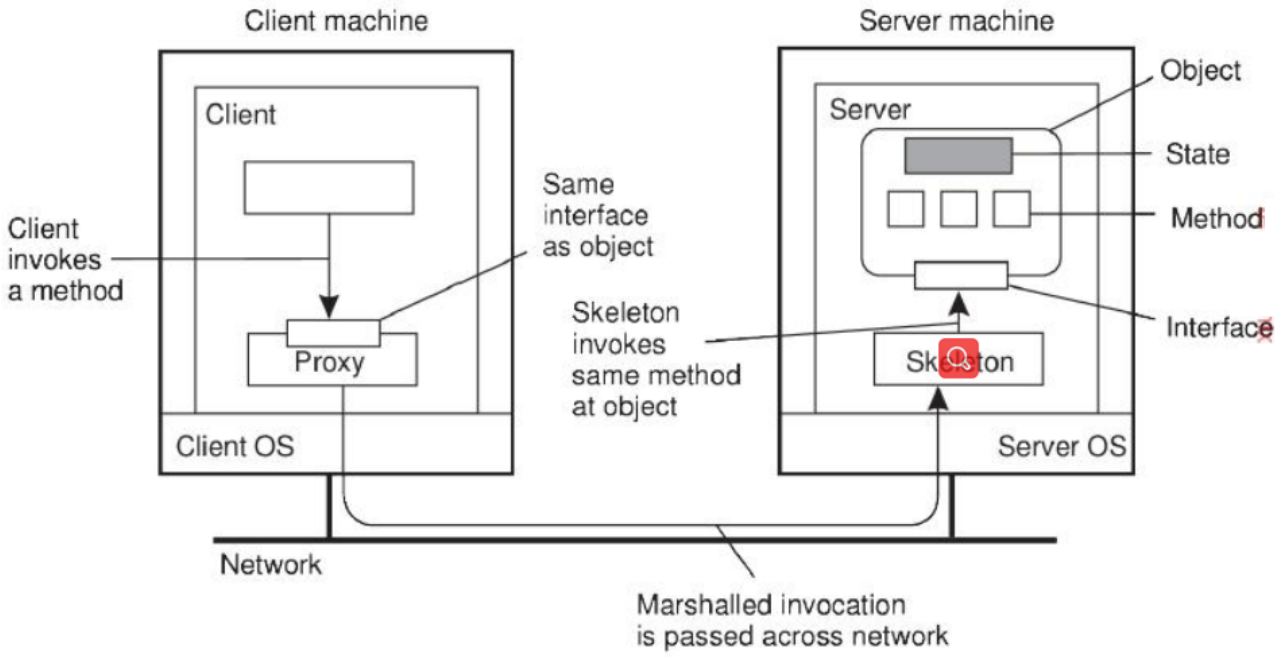
1.3 面向对象的体系结构

每个组件对应一个对象

对象间调用使用 RMI（远程方法调用）实现



远程对象的一般组织方式



1.4 以数据/资源为中心的体系结构

SOA 的集成噩梦，从 SOA 到 REST

- SOA 全英文是 `Service-Oriented Architecture`，中文意思是中文面向服务编程，是一种思想，一种方法论，一种分布式的服务架构

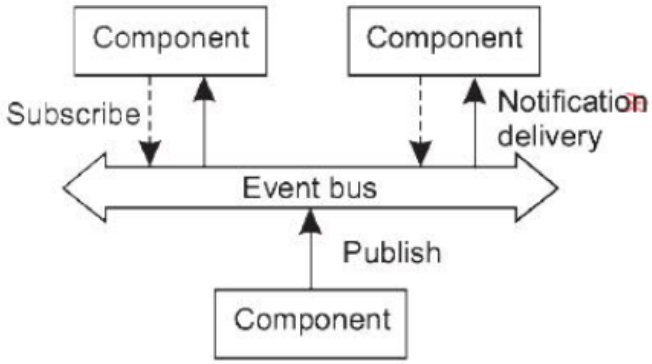
RESTful：Representation State Transfer

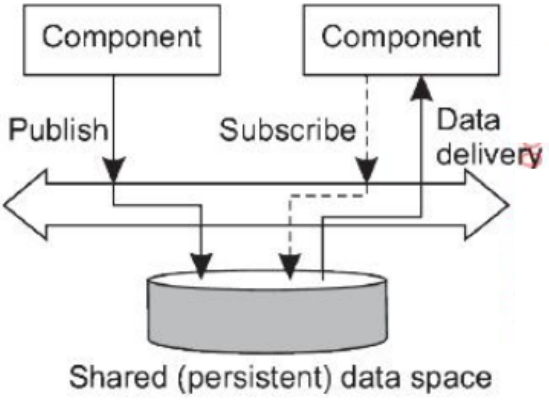
将分布式系统视为一个资源的集合，这些资源由组件们管理

- 命名方式：URI（统一资源标识符）
- 访问方式：HTTP（GET，POST，DELETE，PUT）
- 消息自描述
- 不保留上下文：轻量级

1.5 以事件为中心的体系结构

发布/订阅（publish/subscribe）系统





2.体系结构与中间件

分布式系统是由组件组成，这些组件可以按照某些 style 组织到一起

组件们能否被组织到一起？有两种常用的方法：**适配器**和**拦截器**

2.1 适配器

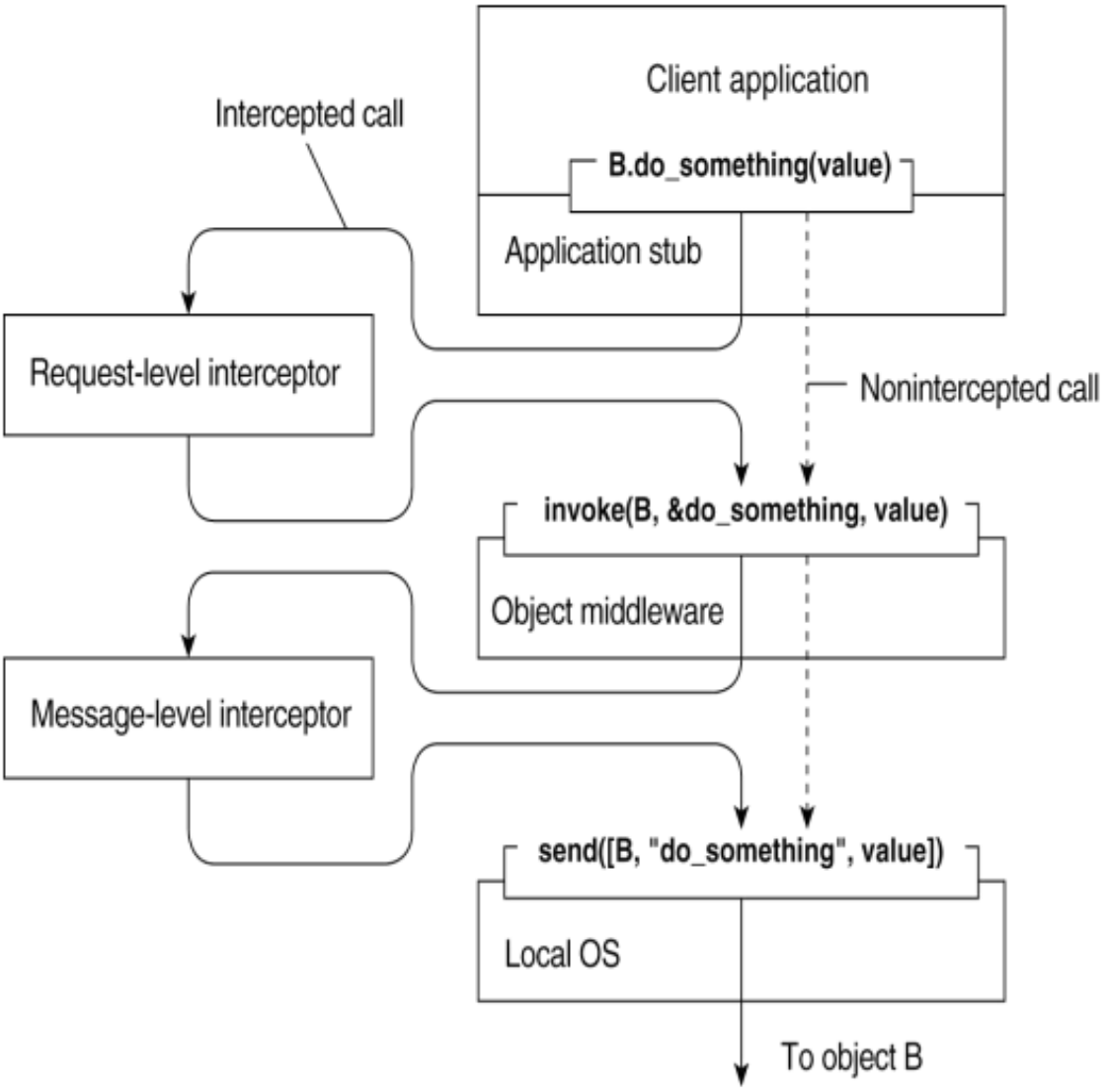
适配器是一种特殊的组件，专门用来解决接口不匹配的问题

2.2 拦截器

拦截器是在 servlet 执行之前执行的程序（这里就是 controller 代码执行之前），它主要是用于拦截用户请求并作相应的处理，比如说可以判断用户是否登录，做相关的日志记录，也可以做权限管理。

可中断正常执行的控制流，插入执行其他代码

A 和 B 在不同的网络，A 想调用 B 的方法。拦截器将 A 的调用请求拦截，封装为一个数据包，发送给远程对象 B。



2.3 自适应软件

（1）概念

软件运行环境是动态的：移动、QoS、故障、能耗都可能变化。

组件也是动态变化的：上线、掉线、备份

自适应软件：可以自动适应环境变化，随着环境变化而变化

（2）场景

- 多网卡服务器上，某物理链路故障时，自动切换线路
- 云服务后端服务器，根据负载调整虚拟机数量
- 移动终端电量下降，任务模块在云端和客户端迁移，降低消耗

（3）实现三种方法

1. **分离关注点**（Separation of concerns）：把主要功能与附加功能分离开；面向方面的软件开发
2. **计算反射**（Computational reflection）：自我检查，并调整自身行为
3. **基于组件的设计**：运行时，进行动态配置；迟后绑定（late binding）

3.系统体系结构 Architecture

- 软件体系结构的具体实例（实现）
- 根据各软件组件的安放位置，确定不同类型的体系结构

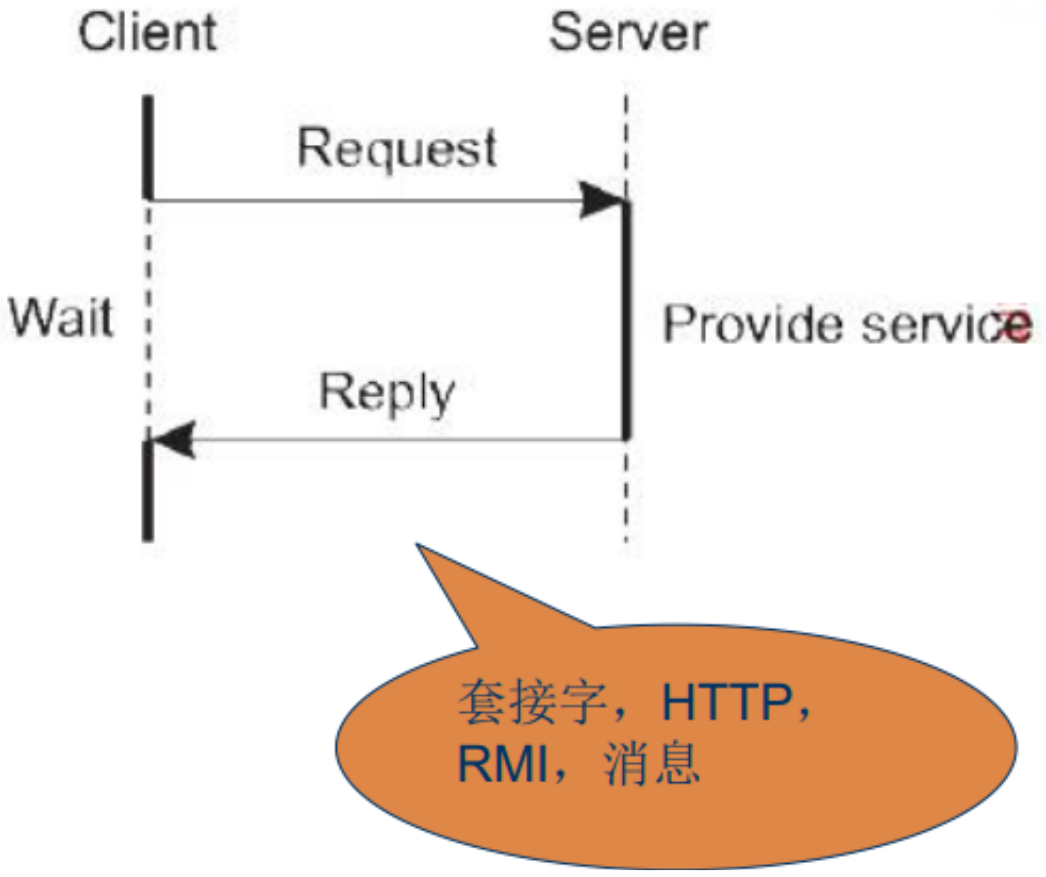
共分为三种体系结构

1. 集中式体系结构
2. 非集中式体系结构
3. 混合方式

3.1 集中式体系结构

客户/服务器模型（C/S）

- 服务器：实现特定服务的进程
- 客户：向服务器提出请求，等待进程的答复
- 请求/答复模式



存在问题：

- 中心节点的单点失效
- 中心节点的维护成本
- 可伸缩性

3.2 非集中式体系结构

- 垂直分布：不同功能的分布
- 水平分布：相同功能的复制（负载均衡）
- 对等型（peer-peer）分布：（点对点系统）

（1）P2P 技术

优点

非中心化、可扩展性、健壮性、高性价比性、隐私保护、负载均衡

应用

- 文件内容共享和下载：BT 等
- 计算能力和存储共享
- 协同与服务共享平台
- 即时通讯工具
- P2P 通讯与信息共享
- 网络电视

拓扑结构

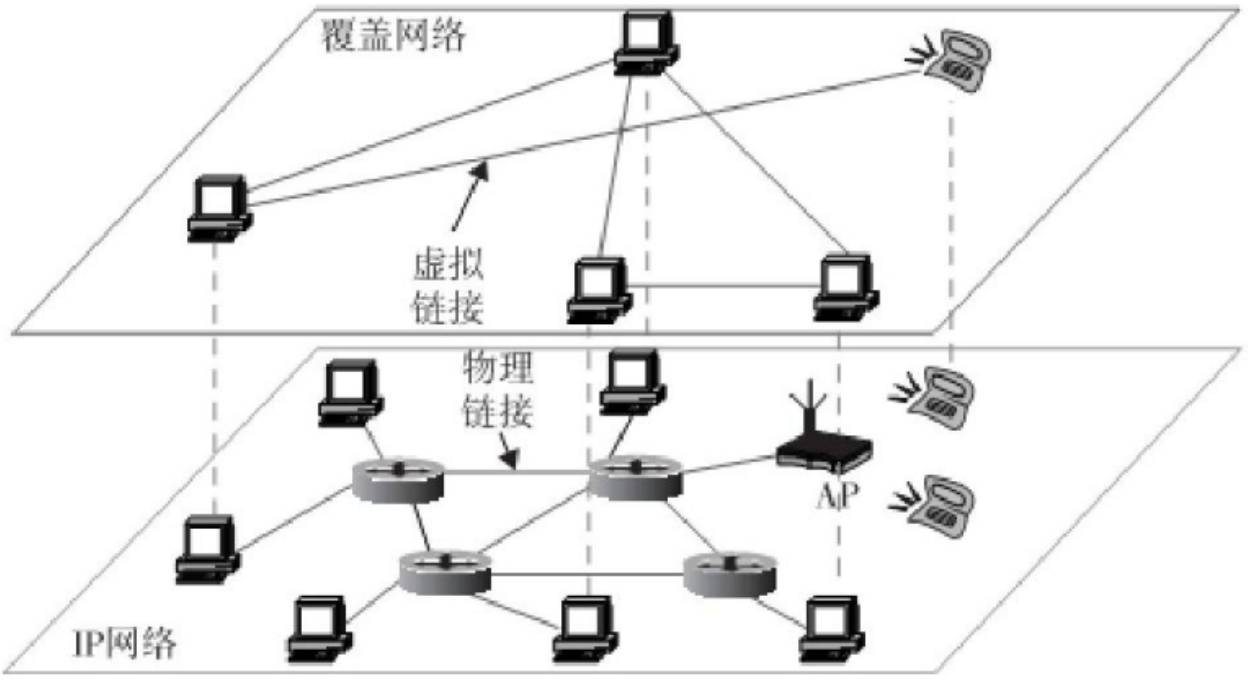
P2P 网络是建立在 Internet 之上的一种覆盖网络

P2P 网络的拓扑结构

- **集中型**（Centralized Topology）
- **分散型结构化拓扑**（Decentralized Structured Topology，最典型的 DHT 网络）
- **分散型无结构拓扑**（Decentralized Unstructured Topology）
- **半分散型拓扑**（Partially Decentralized Topology）

（2）覆盖网络（组织点对点系统）

建立在另一个网络上的网络，属于应用层网络，面向应用层的，不考虑或很少考虑网络层、物理层的问题。

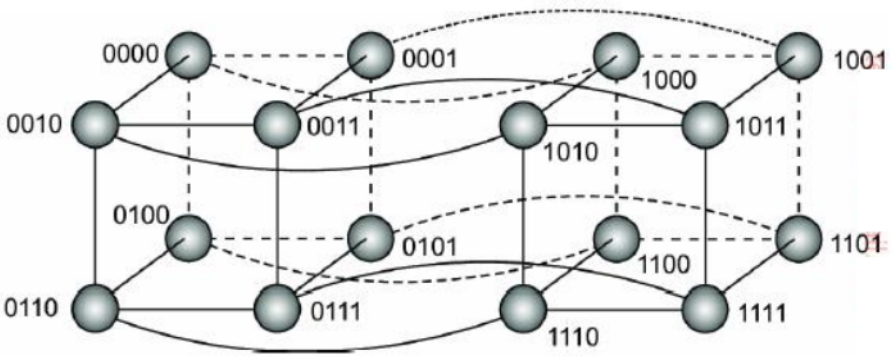


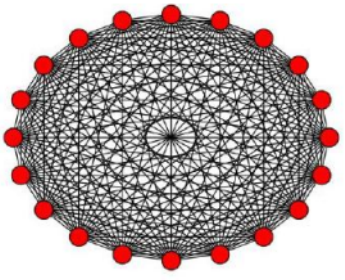
（3）结构化拓扑

在一个结构化的对等系统中，节点被组织在一个覆盖网络中，这个覆盖网络遵循一个特定的、确定的拓扑：一个环、一个二叉树、一个网络。

具体有：

- 全连接
- 超立方
- DHT
- Can





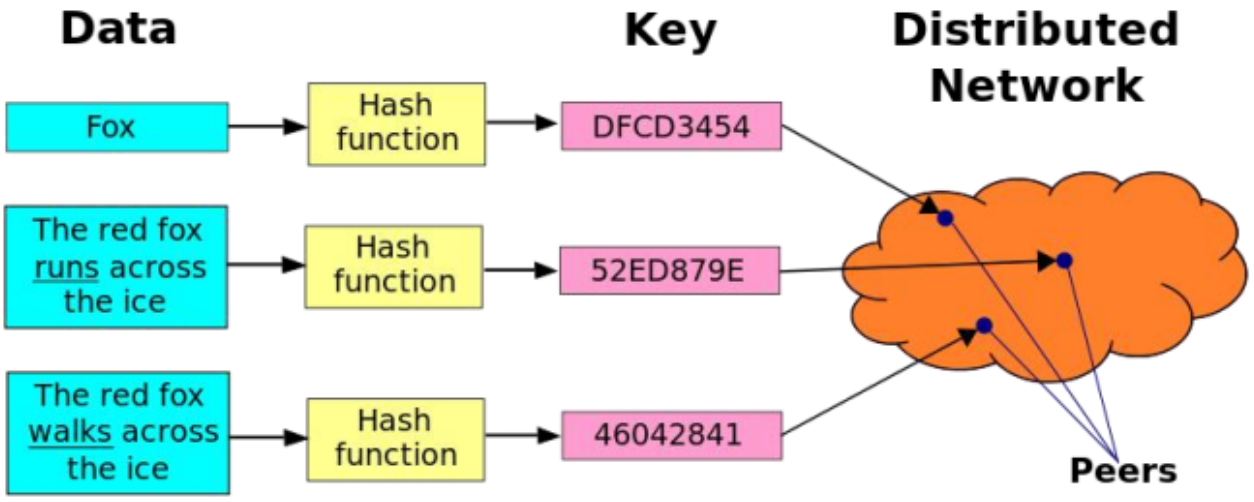
DHT：Distributed Hash Table

给定假设：系统中有大量的结点；结点是动态变化的，有人加入，有人退出

目标：从系统中定位资源

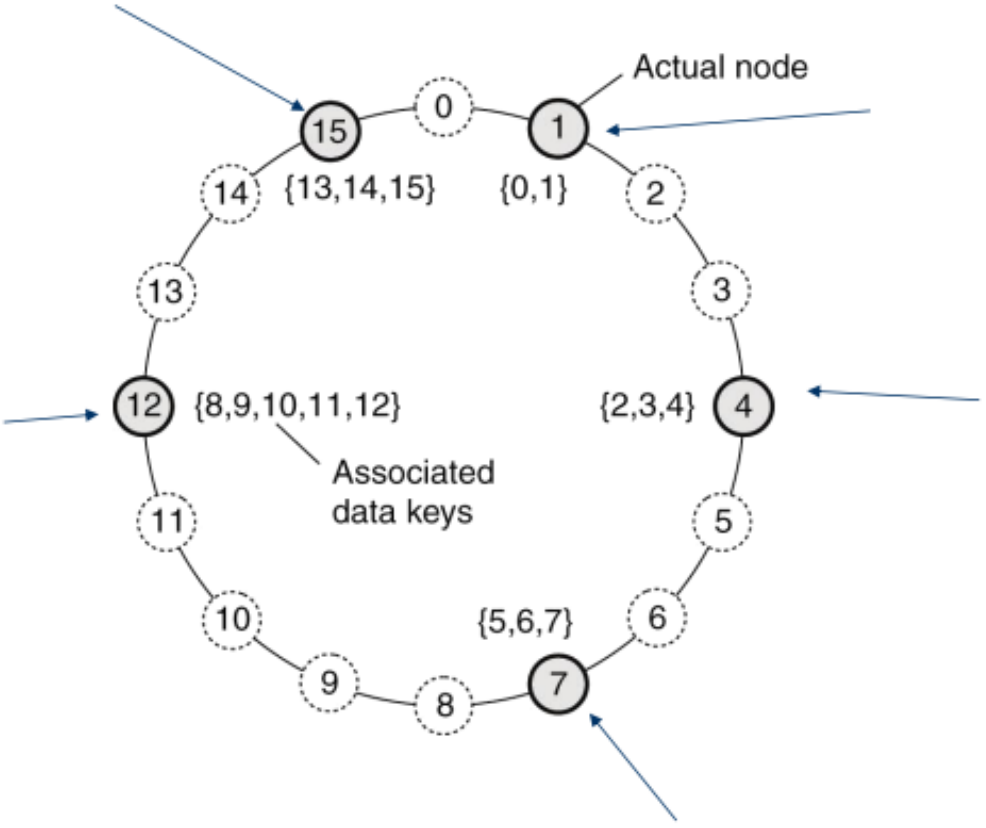
加密散列函数，将一个任意类型的对象（比如字符串）映射到 128 位或 160 位的散列值

SHA–1 可以生成一个被 160 位（20 字节）散列值，即 40 个十六进制数



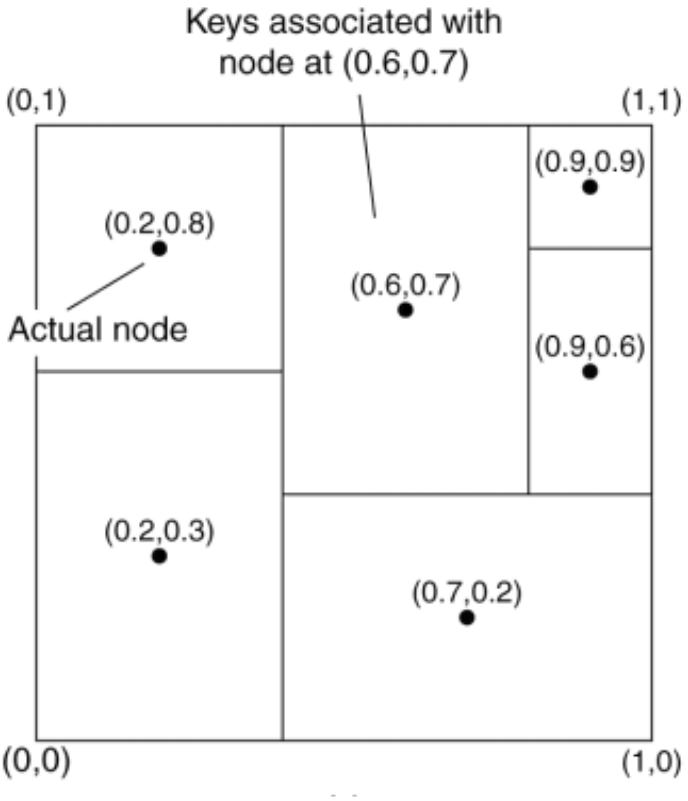
DHT–chord

1. 将哈希空间表示成一个环
2. 将服务器映射到环的结点上
3. 对文件进行 put、get 操作



Can 系统

1. 将（key，value）对存储在拥有该点所在区域的结点内
2. 将请求传给当前结点四邻中距离最接近目标点的结点
3. 时间复杂性 O(n/d)，d 为系统维度
4. 每结点维护的路由表信息和网络规模无关为 O(d)



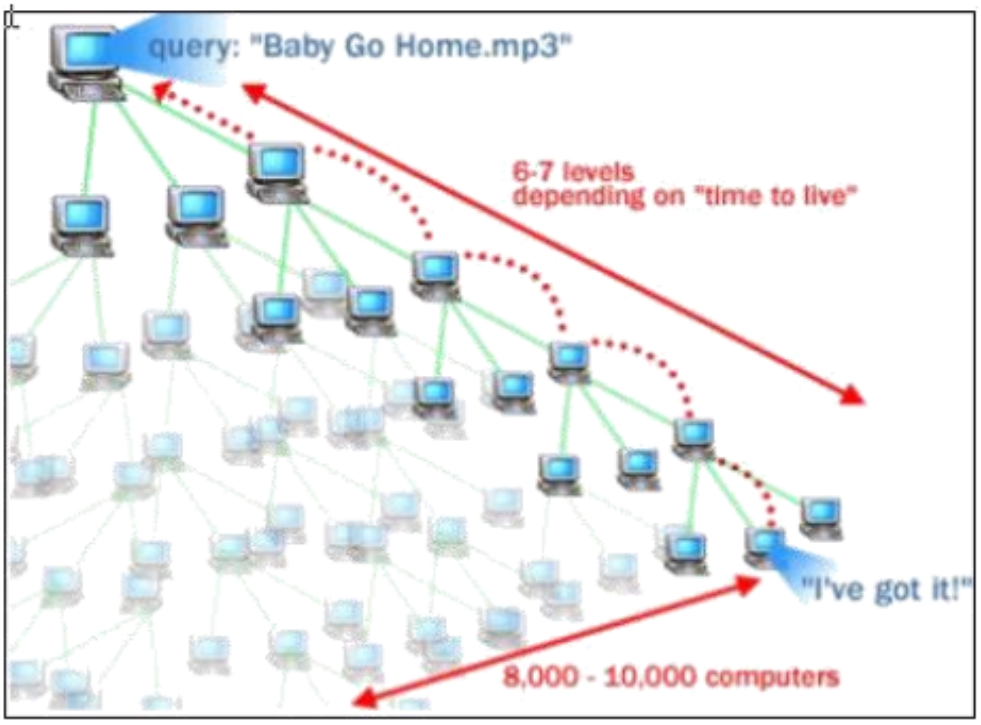
（4）非结构化拓扑

任意两个结点间存在边（虚拟链路）由**概率**决定。或者说非结构化的覆盖网络是一个随机图。

每个结点都无法得知全局拓扑网络结构，只知道自己的几个邻居

Flooding 泛洪

不能保证性能，网络带宽的消耗非常大，可伸缩性差



Gossip

基于流言的协议（Gossip-based protocols）

- 关于所有节点的表，称为全部视图（total view）
- 每个节点维护一个部分视图（partial view），含有 c 个邻居结点的列表，表项：=<IP, age>
- 节点之间定期交换表项，由主动线程（可主动发起通信）和被动线程完成

节点的加入：

- 通过与任意一个已知的节点进行试图交换

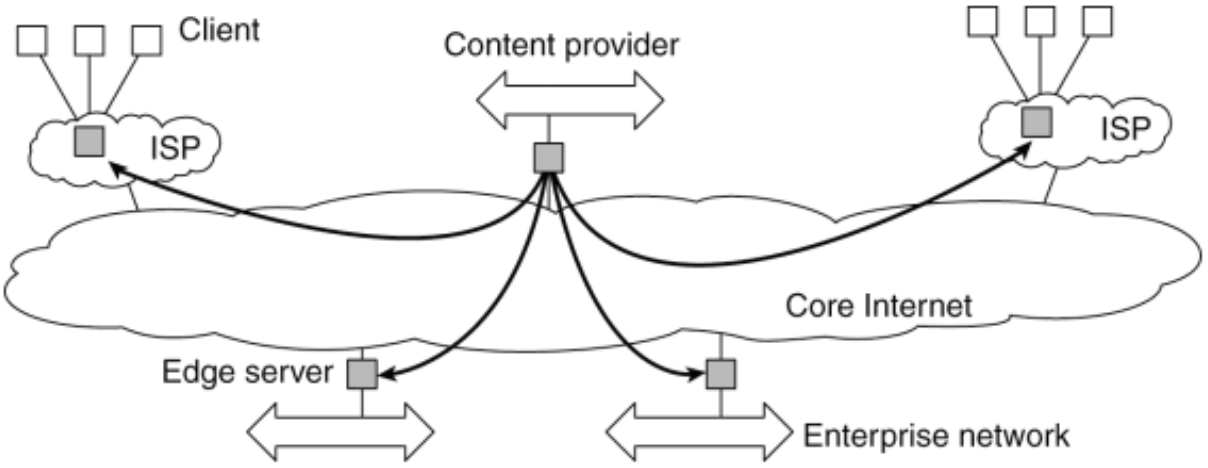
节点的删除：

- 可自行离开，无需通知其他节点
- 当其他节点发现某节点 P 不再响应，将其从表中删除

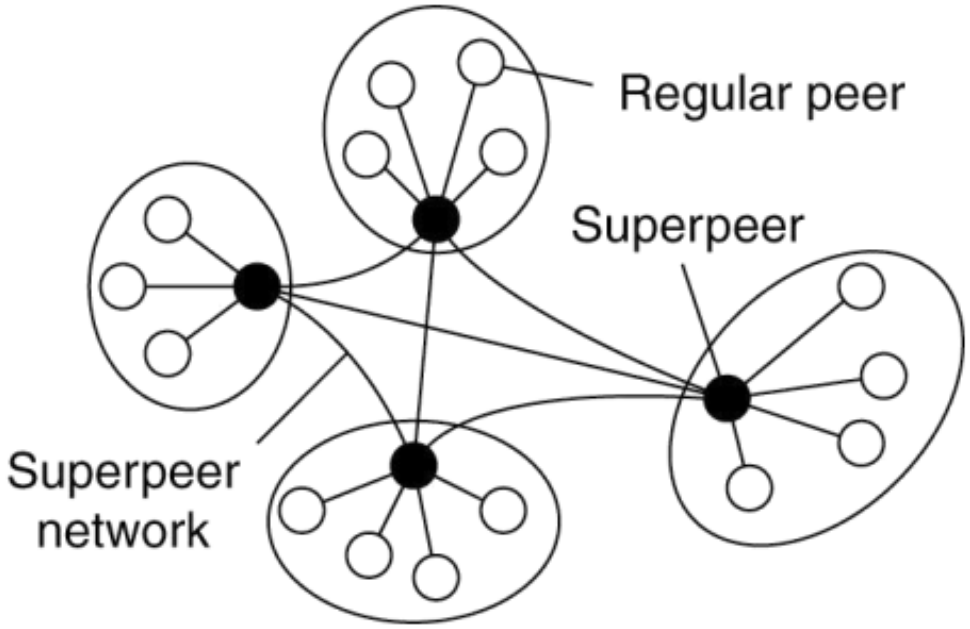
3.3 混合方式

将客户/服务器结构与非集中式结构相结合

例：边界服务器系统（Edge Server）



超级节点：能够维护索引或充当代理的节点

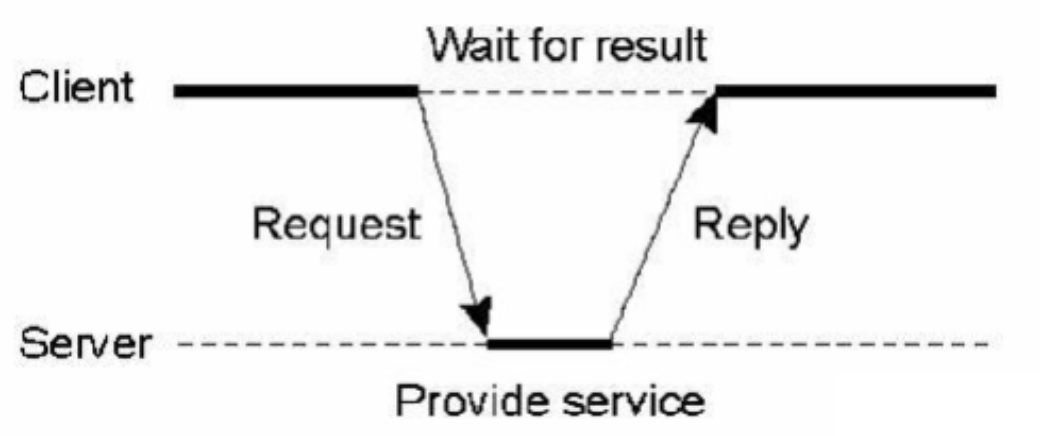


4.客户–服务器模型

服务器（Server）：服务方，实现特性服务的进程

客户（Client）：委托方，请求服务的进程

交互方式：请求–回答（Request–Reply）


Wait for result
Client
Request
Reply
Server
Provide service