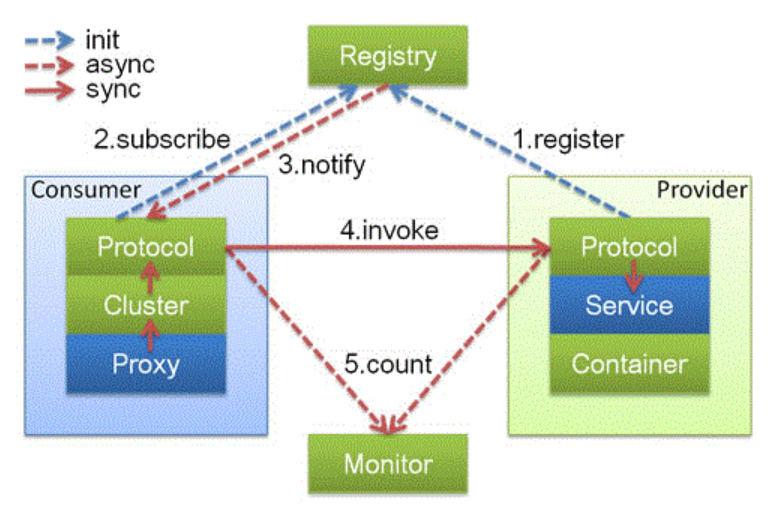
中间件

1. DUBBO

Dubbo是alibaba公司开源的一个高性能优秀的分布式服务治理框架，致力于提供高性能和透明化的RPC远程服务调用方案，使得应用轻松实现服务的输入和输出的功能，而且可以和Spring框架无缝集成，故Dubbo可以全Spring的配置方式，对本地应用没有任何API侵入，只要在Spring的配置文件进行Dubbo的相关服务配置即可。Dubbo的主要核心部件有网络通信框架、支持负载均衡、容灾和集群功能的RPC、一个主要用于服务注册和事件发布、订阅的目录框架、暴露服务的“服务提供者”、调用远程服务方的“服务消费者”、用于服务注册与发现的“服务注册中心”、统计服务调用次数、时间的“服务监控中心”。



从服务模型来看，Dubbo使用了一个十分简单、容易理解的模型，对应用程序来说要么是服务的提供方（Provider），要么是服务消费方（Consumer）。服务注册中心可以将服务统一管理起来，并且能够优化所有的服务发布和使用的流程。因为对于服务提供方来说，不管接口实现有多复杂、服务的数量和类型有多庞大，它只需知道提供哪些服务接口给别的系统调用；而对于服务消费方也只关心在哪方便的获取到自己所需要的服务接口。因此需要使用特定协议来完成统一服务的注册中心。Dubbo提供了Multicast、Zookeeper、Redis、Simple等几种类型的注册中心，从官方文档介绍来看，其推荐使用Zookeeper作为注册中心。Monitor也是上面模型的重要的一部分，虽然监控中心的安装、可用与否并不影响整个Dubbo框架的运行，但monitor能够对所有服务调用的实际状态进行监控，因此开发人员可以根据监控中心改进服务的质量。

在接口调用中，如果服务消费方向提供方发送了一次请求，因为网络等问题造成了调用失败，我们可以通过重试策略再次发起同样的请求就可以使调用成功。但如果服务提供方发布的服务所在单节点机器发生故障，那消费方是怎么也不会再调用成功了，所以此时我们就要使用集群容错模式，所谓集群就是由多台提供相同服务的机器组成的一个整体，每台机器都可以提供。Dubbo提供了六种集群模式供开发人员依据实际的需求进行选择去适应特定的应用场景，Failover Cluster模式自动切换调用可用服务的节点机器，对于幂等性操作，因每次调用副作用相同可以使用此模式，Dubbo默认使用此种模式，此外还有失败立即报错的快速失败模式（Failfast Cluster）、调用失败直接忽略的失败安全模式（Failsafe Cluster）、失败后定时重发的Failback Cluster模式、并行调用多台服务机器的Forking Cluster模式、逐个调用所以服务提供者的广播（Broadcast Cluster）模式。除了这六种模式外，Dubbo框架提供的扩展接口可以使技术架构师私人定制满足自己特定需求的集群容错模式。

1. Zookeeper

Zookeeper是Apache Hadoop的子项目，强度好，同时也是Dubbo分布式服务框架官方推荐的注册中心客户端，因为Dubbo未对Zookeeper的服务器端做任何侵入修改，可以直接安装原生Zookeeper，dubbo在调用Zookeeper客户端时完成所有注册中心的逻辑适配工作。

下面简单介绍一下Zookeeper提供主要功能。

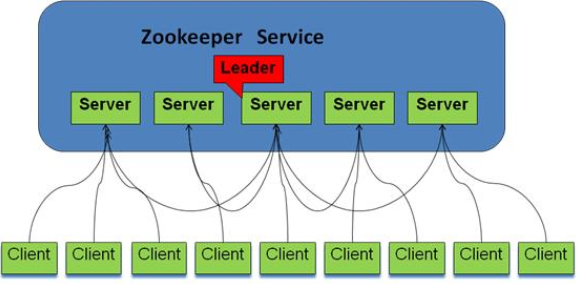
第一：配置管理，在分布式系统中肯定有很多台部署了同一个项目的机器并组成集群协调工作，如果在一个主机器修改了配置文件时，那么这台主机器所在集群的其他对这个配置感兴趣的机器都应该及时获得相应变更。Zookeeper就提供了这样一种集中管理配置的办法，这样不仅不用再手工拷贝配置，还保证了集群的可靠和一致性。

第二：集群管理，机器的集群管理其实说白了就三点，新机器的加入、不能工作的机器的退出、集群master的选举。Zookeeper实现了类似心跳机制的功能，当有新节点加入或已有节点的退出时，集群的master节点能够感知到这种变化并做出对应的动作。当master节点出现故障或失去大多数follower机器时，Zookeeper就进入了恢复模式，基于一定的算法重新选举出master。Zookeeper的选举算法有两种：basic paxos、fast paxos，默认选举算法为fast paxos。

第三：命名服务，Zookeeper会在其文件系统中创建一个唯一path的服务目录，通过查找名称来搜索服务提供者地址，便于识别不同服务，类似于域名和ip地址的关系。

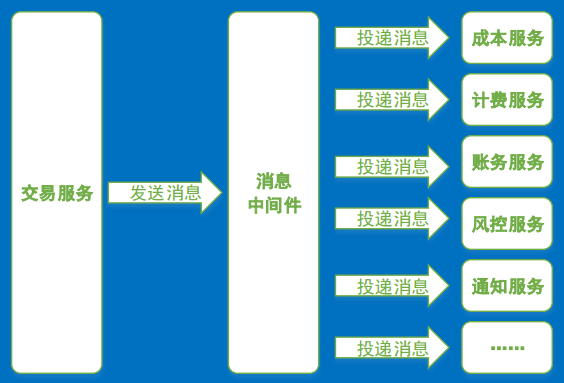
第四：分布式锁，在单机程序中，对某个共享资源来说，为避免多个进程或线程同时访问，需要加互斥锁，这很好理解。那在分布式情况下，也会出现这种情况，不过这是在服务器之间的层面。同一时间只永许一台机器执行对公共资源的操作。

Zookeeper集群系统模型如下：



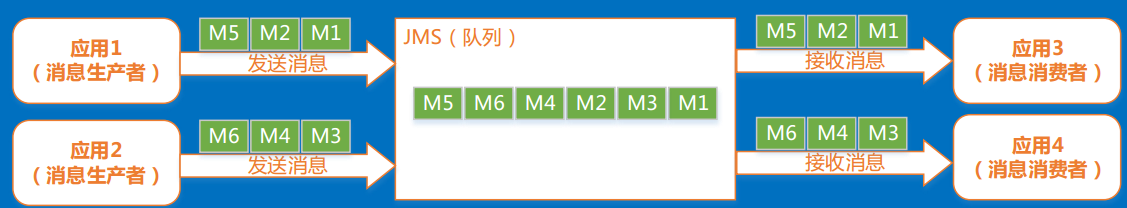
1. ActiveMQ

ActiveMQ是Apache的一个开源项目，属于消息中间件的一种。因此介绍activemq之前应该先理解消息中间件。消息中间件是指面向消息，就像java面向对象一样，消息既是一个信息单元，也是消息中间件处理的对象，在第三方支付平台中，几十个分布式系统同时工作又相互调用配合，如何保证应用程序间可靠的通信，与传统的应用程序交互相比较，消息只提供者只需要把消息发送给运行着消息中间件集群的服务器即可，消息消费者只需要从该服务器获取所需要的消息。消费者、生产者只管发送和接收消息，既不用知道彼此，也不需要关心消息是通过什么协议、什么路由实现消息的传输的。消息中间的通信模型如下所示：

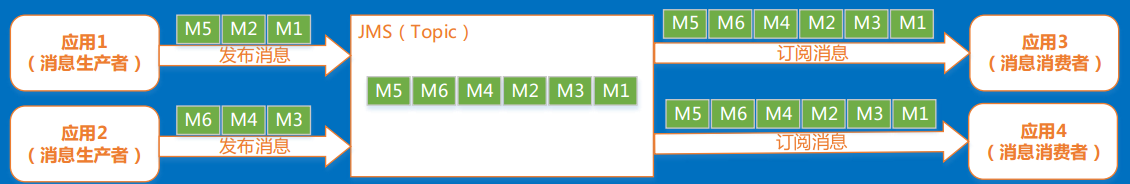
 

消息中间件的这种特性是分布式计算领域是一个重要的理念，它不仅为分布式系统提供了异步、解耦、稳定、可扩展和安全等优点，而且能够通过预制的不同连接协议（HTTP、multicast、SSL、TCP/IP、UDP）提供多种编程语言支持，大大降低了不同环境下实现消息传递的难度。ActiveMQ就是JMS的一个实现，JMS是JavaEE中一个关于消息中间件的技术规范，JMS实现了一套与平台无关的API，它类似于JDBC，一组可以访问不同关系型数据库的API。ActiveMQ通过支持JMS实现了java应用程序的消息传递。

JMS消息的两种模型：队列模型、发布者/订阅者模型



在队列模型中消息生产者将消息发送到队列中，消息消费者从队列中取出消息，如何可见队列支持多个消费者，但一个消息只能被一个消息者消费，而且消息一旦被消费者拿走后就不再被保存到队列中。



在发布者/订阅者模型中，消息生产者将消息发布到一个topic中，只要消费者订阅了这个topic就可以拿走topic中的所有信息。Topic中的消息可以人为的设定保存多长时间。

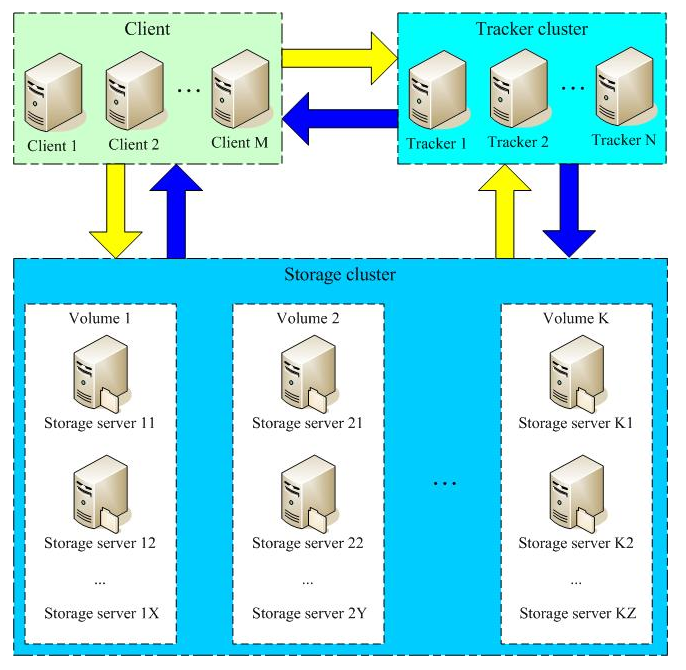
1. Reids

Redis是一个高性能的键值对存储系统，也可以叫做缓存数据库。由于Redis数据库的所有数据都存储在内存中，因此其读写速度相对于普通的关系型数据库更快。对于全流程相应时间要求很好的第三方支付系统来说，数据库的频繁读写所带来的延迟往往是性能的瓶颈所在。故用Redis来保存频繁访问的数据，程序直接从Redis中读取数据，可以加快程序运行的速度，缩短响应时间。例如，当商户发来转账请求时，程序需要先去查询商户信息，包括资金账号、开通相关产品信息，此时先去Redis缓存中读取信息，如果缓存中有，直接读取，如果Redis中没有，就去商户系统CIF中查询，查到后返回，然后存入到Redis中，这样当下次再查询这个商户时，就可以直接读取了，大大加快了响应速度。

1. FastDFS

对于一个大型网站或者高并发请求大数据量存储的系统来说，文件资源的存储与访问一定要集中管理，便于快速的读取和存储，FastDFS就实现了整合一个服务器集群共同运行提供这样的服务。FastDFS全称快速分布式文件系统，它的主要功能为提供文件存储、同步、访问（上传、下载），对于大容量存储和高并发访问，FastDFS实现了负载均衡。对于程序开发者而言，FastDFS可以大大降低开发的精力。开发人员只需要使用Client API通过访问路径进行文件的上传和下载，不需要关心文件上传、下载、同步的细节，都由FastDFS去实现。

FastDFS服务器分两类：跟踪器（tracker）和存储节点（storage）。跟踪器负责调度，包括负载均衡的工作，存储节点完成所有上述的文件功能。其系统结构如下图所示：



Tracker和Storage都可以根据需求由一台或集群构成，并且都可以随时增加和减少节点都不会影响线上服务，因此可以很方便的根据实际请求压力动态的增加或减少服务器。当增加服务器的时候已有的文件会自动同步，当然，多台服务器可以更好的完成备份和负载均衡的工作。