分析zookeeper到底能做什么？

Zookeeper是hadoop的一个子项目，虽然源自hadoop，但是我发现zookeeper脱离hadoop的范畴开发分布式框架的运用越来越多。今天我想谈谈zookeeper，本文不谈如何使用zookeeper，而是zookeeper到底有哪些实际的运用，哪些类型的应用能发挥zookeeper的优势，最后谈谈zookeeper对分布式网站架构能产生怎样的作用。

Zookeeper是针对大型分布式系统的高可靠的协调系统。由这个定义我们知道zookeeper是个协调系统，作用的对象是分布式系统。为什么分布式系统需要一个协调系统了？理由如下：

开发分布式系统是件很困难的事情，其中的困难主要体现在分布式系统的“部分失败”。“部分失败”是指信息在网络的两个节点之间传送时候，如果网络出了故障，发送者无法知道接收者是否收到了这个信息，而且这种故障的原因很复杂，接收者可能在出现网络错误之前已经收到了信息，也可能没有收到，又或接收者的进程死掉了。发送者能够获得真实情况的唯一办法就是重新连接到接收者，询问接收者错误的原因，这就是分布式系统开发里的“部分失败”问题。

Zookeeper就是解决分布式系统“部分失败”的框架。Zookeeper不是让分布式系统避免“部分失败”问题，而是让分布式系统当碰到部分失败时候，可以正确的处理此类的问题，让分布式系统能正常的运行。

下面我要讲讲zookeeper的实际运用场景：

场景一：有一组服务器向客户端提供某种服务（例如：我前面做的分布式网站的服务端，就是由四台服务器组成的集群，向前端集群提供服务），我们希望客户端每次请求服务端都可以找到服务端集群中某一台服务器，这样服务端就可以向客户端提供客户端所需的服务。对于这种场景，我们的程序中一定有一份这组服务器的列表，每次客户端请求时候，都是从这份列表里读取这份服务器列表。那么这分列表显然不能存储在一台单节点的服务器上，否则这个节点挂掉了，整个集群都会发生故障，我们希望这份列表时高可用的。高可用的解决方案是：这份列表是分布式存储的，它是由存储这份列表的服务器共同管理的，如果存储列表里的某台服务器坏掉了，其他服务器马上可以替代坏掉的服务器，并且可以把坏掉的服务器从列表里删除掉，让故障服务器退出整个集群的运行，而这一切的操作又不会由故障的服务器来操作，而是集群里正常的服务器来完成。这是一种主动的分布式数据结构，能够在外部情况发生变化时候主动修改数据项状态的数据机构。Zookeeper框架提供了这种服务。这种服务名字就是：统一命名服务，它和javaEE里的JNDI服务很像。

场景二：分布式锁服务。当分布式系统操作数据，例如：读取数据、分析数据、最后修改数据。在分布式系统里这些操作可能会分散到集群里不同的节点上，那么这时候就存在数据操作过程中一致性的问题，如果不一致，我们将会得到一个错误的运算结果，在单一进程的程序里，一致性的问题很好解决，但是到了分布式系统就比较困难，因为分布式系统里不同服务器的运算都是在独立的进程里，运算的中间结果和过程还要通过网络进行传递，那么想做到数据操作一致性要困难的多。Zookeeper提供了一个锁服务解决了这样的问题，能让我们在做分布式数据运算时候，保证数据操作的一致性。

场景三：配置管理。在分布式系统里，我们会把一个服务应用分别部署到n台服务器上，这些服务器的配置文件是相同的（例如：我设计的分布式网站框架里，服务端就有4台服务器，4台服务器上的程序都是一样，配置文件都是一样），如果配置文件的配置选项发生变化，那么我们就得一个个去改这些配置文件，如果我们需要改的服务器比较少，这些操作还不是太麻烦，如果我们分布式的服务器特别多，比如某些大型互联网公司的hadoop集群有数千台服务器，那么更改配置选项就是一件麻烦而且危险的事情。这时候zookeeper就可以派上用场了，我们可以把zookeeper当成一个高可用的配置存储器，把这样的事情交给zookeeper进行管理，我们将集群的配置文件拷贝到zookeeper的文件系统的某个节点上，然后用zookeeper监控所有分布式系统里配置文件的状态，一旦发现有配置文件发生了变

化，每台服务器都会收到zookeeper的通知，让每台服务器同步zookeeper里的配置文件，zookeeper服务也会保证同步操作原子性，确保每个服务器的配置文件都能被正确的更新。

场景四：为分布式系统提供故障修复的功能。集群管理是很困难的，在分布式系统里加入了zookeeper服务，能让我们很容易的对集群进行管理。集群管理最麻烦的事情就是节点故障管理，zookeeper可以让集群选出一个健康的节点作为master，master节点会知道当前集群的每台服务器的运行状况，一旦某个节点发生故障，master会把这个情况通知给集群其他服务器，从而重新分配不同节点的计算任务。Zookeeper不仅可以发现故障，也会对有故障的服务器进行甄别，看故障服务器是什么样的故障，如果该故障可以修复，zookeeper可以自动修复或者告诉系统管理员错误的原因让管理员迅速定位问题，修复节点的故障。大家也许还会有个疑问，master故障了，那怎么办了？zookeeper也考虑到了这点，zookeeper内部有一个“选举领导者的算法”，master可以动态选择，当master故障时候，zookeeper能马上选出新的master对集群进行管理。

下面我要讲讲zookeeper的特点：

1. zookeeper是一个精简的文件系统。这点它和hadoop有点像，但是zookeeper这个文件系统是管理小文件的，而hadoop是管理超大文件的。

2. zookeeper提供了丰富的“构件”，这些构件可以实现很多协调数据结构和协议的操作。例如：分布式队列、分布式锁以及一组同级节点的“领导者选举”算法。

3. zookeeper是高可用的，它本身的稳定性是相当之好，分布式集群完全可以依赖zookeeper集群的管理，利用zookeeper避免分布式系统的单点故障的问题。

4. zookeeper采用了松耦合的交互模式。这点在zookeeper提供分布式锁上表现最为明显，zookeeper可以被用作一个约会机制，让参入的进程不在了解其他进程的（或网络）的情况下能够彼此发现并进行交互，参入的各方甚至不必同时存在，只要在zookeeper留下一条消息，在该进程结束后，另外一个进程还可以读取这条信息，从而解耦了各个节点之间的关系。

5. zookeeper为集群提供了一个共享存储库，集群可以从这里集中读写共享的信息，避免了每个节点的共享操作编程，减轻了分布式系统的开发难度。

6. zookeeper的设计采用的是观察者的设计模式，zookeeper主要是负责存储和管理大家关心的数据，然后接受观察者的注册，一旦这些数据的状态发生变化，Zookeeper 就将负责通知已经在 Zookeeper 上注册的那些观察者做出相应的反应，从而实现集群中类似 Master/Slave 管理模式。

由此可见zookeeper很利于分布式系统开发，它能让分布式系统更加健壮和高效。

前不久我参加了部门的hadoop兴趣小组，测试环境的hadoop、mapreduce、hive及hbase都是我来安装的，安装hbase时候安装要预先安装zookeeper，最早我是在四台服务器上都安装了zookeeper，但是同事说安装四台和安装三台是一回事，这是因为zookeeper要求半数以上的机器可用，zookeeper才能提供服务，所以3台的半数以上就是2台了，4台的半数以上也是两台，因此装了三台服务器完全可以达到4台服务器的效果，这个问题说明zookeeper进行安装的时候通常选择奇数台服务器。在学习hadoop的过程中，我感觉zookeeper是最难理解的一个子项目，原因倒不是它技术负责，而是它的应用方向很让我困惑，所以我有关hadoop技术第一篇文章就从zookeeper开始，也不讲具体技术实现，而从zookeeper的应用场景讲起，理解了zookeeper应用的领域，我想再学习zookeeper就会更加事半功倍。

之所以今天要谈谈zookeeper，也是为我上一篇文章分布式网站框架的补充。虽然我设计网站架构是分布式结构，也做了简单的故障处理机制，比如：心跳机制，但是对集群的单点故障还是没有办法的，如果某一台服务器坏掉了，客户端任然会尝试连接这个服务器，导致部分请求的阻塞，也会导致服务器资源的浪费。不过我目前也不想去修改自己的框架，因为我总觉得在现有的服务上添加zookeeper服务会影响网站的效率，如果有独立的服务器集群部署

zookeeper还是值得考虑的，但是服务器资源太宝贵了，这个可能性不大。幸好我们部门也发现了这样的问题，我们部门将开发一个强大的远程调用框架，将集群管理和通讯管理这块剥离出来，集中式提供高效可用的服务，等部门的远程框架开发完毕，我们的网站加入新的服务，我想我们的网站将会更加稳定和高效。