文章编号：1005-8451（2018）10-0026-04

基于Zookeeper的配置管理中心设计与实现

苗　凡，阎志远，戴琳琳
（中国铁道科学研究院集团有限公司　电子计算技术研究所，北京　100081）

摘　要：针对现有客票交易中间件缺乏有效机制管理配置文件的问题，在研究Zookeeper的基础上，提出了一种基于Zookeeper的统一配置管理方案，将配置文件内容存储在Zookeeper的节点中，利用Zookeeper的发布/订阅机制，实现了配置文件在管理中心与客票交易中间件同步，结合日常维护需求，提供了配置文件的版本管理与B/S管理功能。测试数据表明该方案简化了配置文件的管理过程，提高了升级维护的效率。

关键词：同步；Zookeeper；版本管理
中图分类号：U293.22：TP39　**文献标识码：**A

Configuration management center based on Zookeeper

MIAO Fan, YAN Zhiyuan, DAI Linlin
(Institute of Computing Technologies Corporation Limited, China Academy of Railway Sciences, Beijing　100081, China)

Abstract: Aiming at the problem of lacking effective mechanism to manage configuration files in the existing ticketing transaction middleware, this paper proposed a unified configuration management scheme based on Zookeeper, which stored the configuration file contents in Zookeeper nodes, used the publish/subscribe mechanism of Zookeeper to implement the synchronization of the configuration file in the management center and the ticketing transaction middleware, combined with daily maintenance needs, provided version management and B/S management functions of configuration files. The test data shows that this scheme simplifies the management of configuration files and improves the efficiency of upgrade and maintenance.

Keywords: synchronization; Zookeeper; version management

客票系统核心交易中间件线上包括CTMS、INETIS、CTMSX等，主要为铁路12306网站、12306 APP提供接口服务。线下包括CTMS、AFCIS、TVMIS等，主要为窗口、代售点、自动售票机、闸机等提供接口服务。其中，CTMSX、INETIS与CTMS部署在铁路总公司的两个数据中心，共同承担各渠道的交易请求。客票系统日均售票800万张，节假日高峰期超过1 200万张[1]，中间件服务作为交易系统中的重要一环，在保证系统的稳定性中起着重要的作用。

交易中间件的应用均含有多个配置文件，根据业务场景的不同，替换的配置文件也不同。当前中间件服务更换配置文件后，需要重启服务才能生效，若服务与其它应用保持长连接，还需要通知其它应用进行重连，整个升级可能需要10 min才能恢复，极大地影响了生产环境的可用性[2]。为解决上述问题，本文提出了一种基于Zookeeper进行配置管理的方案，Zookeeper是一个高可用的分布式数据管理与系统协调框架，能够保证分布式环境中数据的一致性。

1　Zookeeper的数据结构

Zookeeper采用了类似文件系统目录树型结构的数据模型，结构中的每一个节点称为znode。与文件系统不同的是，每个节点具有与之对应的数据内容，可以扩展任意的节点和叶子节点，每个节点都可以存储KB级别的数据[3]。节点维护一个stat数据结构（包括数据变化的版本号、ACL变化、时间戳），以允许缓存验证与协调更新。以INETIS下的test节点为例，通过get命令获取节点内容，如图1所示。

每当节点数据内容改变，版本号dataVersion均会增长，客户端获取数据的同时也会获取数据版本

收稿日期：2018-01-01
基金项目：中国铁道科学研究院科研开发课题（2017YJ096）。
作者简介：苗　凡，助理研究员；阎志远，副研究员。

```
[zk: localhost:2181(CONNECTED) 4] get /INETIS/CONF/test
Hello,test
cZxid = 0x100000c54
ctime = Thu Jue 03 13:29:53 CST 2017
mZxid = 0x100000d37
mtime = Thu Jue 08 07:17:34 CST 2017
pZxid = 0x100000c54
cversion = 0
dataVersion = 4
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 10
numChildren = 0
```

图1　znode节点信息

号。节点的数据内容以原子方式读写，读操作读取全部内容，写操作替换全部内容，节点还具有一个访问控制列表来约束某些操作。

Zookeeper 有两种类型的节点：(1) 临时节点，在创建该节点会话的存活期间存在，会话结束，临时节点自动删除；(2) 持久节点，节点创建后一直存在，直到有删除操作来主动清除这个节点 [4]。

2　实现方案

2.1　配置管理中心

配置管理中心以 B/S 方式提供给维护人员使用，可以进行增加、更改、删除、查看等操作。每一个配置文件均对应一个 Zookeeper 中的节点，节点变更会通知到注册过该节点的各应用。同时可以查看配置文件的历史版本，便于对比，配置管理中心与各应用的调用关系如图 2 所示。

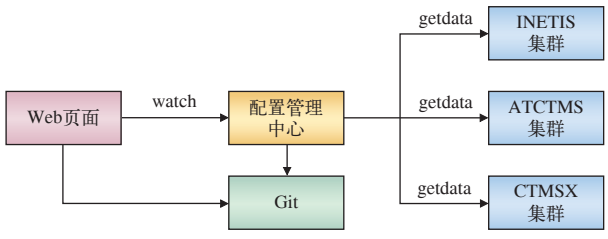

图2　配置管理中心与各应用的调用关系

配置管理中心以 Web 页面的方式供维护人员使用，并与客票系统交易中间件保持连接，配置管理中心保存着 INETIS、ATCTMS 和 CTMSX3 个应用服务的配置文件。配置中心的主要功能如下：

(1) 基于 Web 页面的配置文件管理。通过页面可以查看各个服务当前的配置文件，并检查当前运行的配置文件是否与 Zookeeper 中的配置文件一致。同时 Web 界面可以连到各个集群中的服务，通过后台服务提供的接口，查看各配置项在内存中的值与配置文件中是否一致。

(2) 同一个 APP 下的多个配置文件。同一个应用下存在多个配置文件，对配置进行分类，对于经常更改的，加入到 Zookeeper 节点中。并在客户端中对该节点注册 watcher 事件。

(3) 配置文件的版本管理。生产中每次更改的配置文件需要进行备份，如果升级不成功可以回退。选取 Git 进行配置文件的版本管理，Web 页面查看历史版本配置文件。

(4) 数据持久化保存。当交易中间件从统一配置管理中心得到数据更新后，将数据保存在配置文件中。以防 Zookeeper 异常时，无法得到配置文件，应用无法启动。

(5) 实时更新。对于新获取的配置文件，在持久化保存后，需要将更改的值重新加载入内存替换旧的值，达到无需重启应用即完成配置生效的功能。

2.2　配置文件节点设计

客票系统交易中间件的配置文件以节点的方式存于 Zookeeper 中，配置文件在 Zookeeper server 中节点的设计如图 3 所示，<> 表示目录。从图中可以看到，最上一级目录名为 middle_ware，表示客票交易中间件的目录，它的子目录为 app_name，即 CTMS、INETIS、CTMSX 等。每个中间件应用服务下有两个子目录，分别为 conf 与 version，conf 节点下的子节点为最终的配置文件节点，节点里面的内容即为配置文件里的值。以中间件应用服务 INETIS 为例，INETIS 中的配置文件 gemfire.cfg 在 Zookeeper Server 里的节点路径即为 /middle_ware/INETIS/CONF/gemfire，配置文件 eid.cfg 对应的节点为 /middle_ware/INETIS/CONF/eid。

2.3　配置文件的获取与更新

客票交易中间件如 CMTS、INETIS 等在运行之前，必须先加载本地配置文件，再进行一系列参数的校验与初始化后才能正常启动。更新配置文件时，

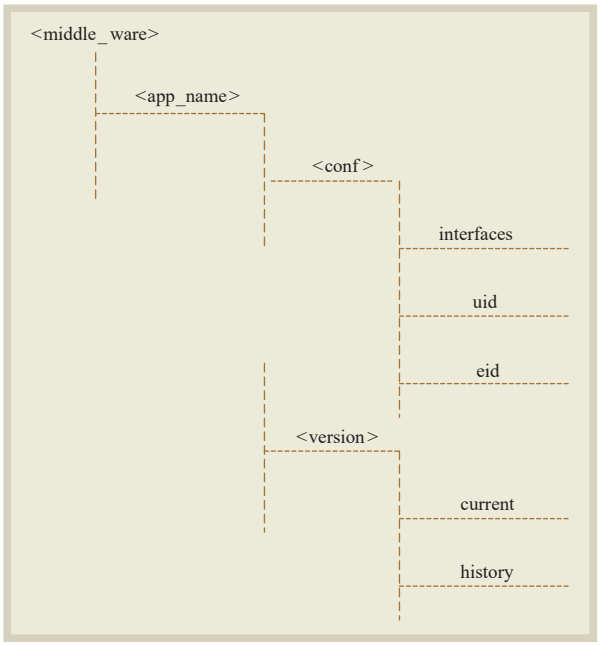

图3 配置文件节点设计

需要同步两个中心的所有服务器，确保各服务器的配置无误后才能重启服务。为了改进这种更新方式，提高应用程序维护的效率，需要解决两个问题。

2.3.1 自动获取

配置文件的自动获取即应用程序从Zookeeper中获取配置文件，Zookeeper所有读操作getdata、getchildren和exists均具有设置watch的选项，Zookeeper的watch事件是一次性触发器，当watch监视的数据发生变化时，通知设置了该watch的client，即watcher[5]。

为了完成应用程序从Zookeeper中自动获取配置文件的内容，需要改造应用程序，使应用程序启动前读取znode节点里的内容，并且将watch事件一直注册在该节点上，只要节点内容有变化，作为Zookeeper客户端的应用将获取数据变化事件。

2.3.2 重新加载

当Zookeeper节点内容发生变化时，watch该节点的回调函数便会执行。函数里实现了重新解析配置文件的内容，刷新各个配置项在内存中的值，便可实现配置文件的实时更新。同时为了防止Zookeeper服务器出现异常，需要将最新的配置文件持久化写入本地文件，以防程序重启时因无法获取配置文件而导致服务不可用，还可以进行版本管理。

2.3.3 更新

配置文件节点的创建与更新需要开发一个单独的发布程序，发布程序的接口与配置文件获取类似，区别主要在发布用的是setdata，获取用的是getdata[6]。由于每个应用均包括多个配置文件，在发布程序里肯定会包括全部的配置文件的加载，但有时生产只需要更新一个配置文件，因此在setdata之前是需要对比新的配置文件里的内容与Zookeeper里对应节点的内容，如果两者内容一致，则不需要设置这个节点，反之如果不一样，则对这个节点进行setdata。

涉及到节点改变的不同方式，Zookeeper可能维护两个watch列表：节点的数据watch和子节点git的watch。getData和exists设置了内容watch，getChildren设置了子节点watch，操作返回的数据类型不同，前者是节点的内容，后者是节点的子节点列表[7]。

2.4 配置文件版本管理

版本管理的功能在于跟踪记录整个配置文件的变更过程，在时间上全程跟踪记录工具将会自动记录开发过程中的每个更改细节和不同版本，以便对不同阶段的配置文件进行差别分析，辅助协调管理维护开发团队。

Git是当前流行的版本管理工具，Git的每次提交都会根据SHA1算法生成唯一的commitid，而不是像SVN那样对单个文件分别进行版本更改，所以跟踪以前某次提交的代码时，不用考虑到底提交了哪些文件，所有变动的配置文件都会一次性的取出来。常用的git命令有git add,git commit,git checkout等。

对于统一配置管理中心的维护人员来说，显然不希望了解版本库里的所有信息，因此，本系统的目标是对git版本库中的配置文件信息进行针对性的展示。在版本控制系统中，分支的个数，提交的次数，提交者以及提交的时间都是十分重要的信息，在配置文件首页看到的是配置文件节点的最新提交信息。

2.5 REST Web Service接口设计

REST包含3个主要内容：资源，表示和状态。资源指网络上一种体现为比特流的实物或抽象概念，可通过统一资源定位符URI定位；表示指资源呈现的方式，为构建可扩展，松耦合的Web提供准则；状态是服务器端资源状态或终端状态，资源的状态

保存在服务端，应用的状态由应用自身维护，由于REST所有交互都是无状态的，因此终端的每次请求需要携带交互所需要的全部信息[8]。

REST将整个服务端抽象成资源的集合，资源由URI标识，终端调用HTTP的主要方法包括POST、DELETE、PUT及GET，可以分别对资源进行增加、删除，更改和查询等操作。基于REST实施的上述架构约束，可轻松解决系统开发中接口可扩展性和终端异构性等问题。

统一配置管理中心给维护人员使用，根据统一配置中心的需求分析，其提供的主要功能有：

（1）集群中各服务的运行状态，包括服务连接池使用大小，错误日志的监控。（2）集群中各主机的状态，包括CPU，内存，磁盘，网络的利用率。（3）对各服务进行操作，包括对服务的启动与停止。（4）更新或更改配置文件等功能，根据需求，通过表单提交配置项的值，并将新的配置信息返回给开发人员。（5）权限管理，包括用户角色分配、用户登录、注册、添加及修改用户基本信息等。

结合上述需求分析，遵循REST规范，设计出符合REST风格的统一配置管理中心接口设计，如表1所示。

表1　配置文件管理REST接口设计

接口	名称	类型	备注
配置文件详情	Middle/{appID}	GET	
配置文件列表	Middle/{appID}/list	POST	
创建配置文件	Middle/cfgfile	POST	
删除配置文件	Middle/cfgfile	DELETE	
获取所有应用	Middle/app	GET	

3　测试数据分析

为了获得更好的可靠性服务和更快的同步速度，测试环境采用集群的方式部署Zookeeper服务，通过客票交易中间件建立到ZooKeeper实例的连接，然后在服务集群中创建节点，并将节点设置为监听状态，该节点的内容为交易中间件INETIS配置文件的内容。当维护人员需要修改INETIS配置文件内容时，只需要通过Web页面选择对应的配置文件，并对文件内容进行修改，可以看到Zookeeper中的节点内容已经修改，而此时监听了这个节点INETIS配置文件的最新修改时间已经更新为当前时间，内容与修改后的一致。新的同步方式几乎在秒级完成，而采用传统的文件传输方式完成一次同步需要近10 min。

4　结束语

服务器增长的速度已远远高于开发人员的增长速度，通过传统人工手段登录服务器实现配置文件的更新不仅效率低而且容易出错，本文提出了一种基于Zookeeper的配置文件管理方案，将客票交易中间件应用服务器众多的配置文件集中式地存放在Zookeper节点上，在Zookeeper的客户端，通过与服务端连接，可以方便快捷地获取应用服务器最新的配置文件，并在此基础上增加了过期配置文件的备份功能。目前，该方案已经应用在客票系统双中心，应用结果表明，此方案不仅能保证各配置文件的一致性而且无需重启服务，整个更新在秒级完成，极大地提高了升级与维护的效率，保证了服务的不间断运行。Zookeeper除了这个功能外，还能管理集群中的节点，下一步我们将继续深入研究，持续提高系统的可靠性。

参考文献：

[1] 戴琳琳，阎志远，苗　凡，等．OpenStack在新一代客票系统中的应用[J]．铁路计算机应用，2015，24（11）：21-23.

[2] 朱建生．新一代客票系统总体技术方案的研究[J]．铁路计算机应用，2012，21（6）：1-6.

[3] 何慧虹，王　勇，史　亮．分布式环境下基于ZooKeeper服务的数据同步研究[J]．信息安全，2015（9）：227-230.

[4] 李汝光，赵　俊．基于ZooKeeper的分布式缓存设计与实现[J]．绵阳师范学院学报，2011，30（11）：116-119.

[5] 李东辉，吴小志，朱广新，等．分布式数据库协调技术Zookeeper[J]．科技展望，2016，26（1）：13-14，16.

[6] 王润华，任化敏，周艳芳．分布式系统开发利器—Zookeeper研究[J]．中国电子商情：通信市场，2012（1）：64-67.

[7] 唐海东，武延军．分布式同步系统Zookeeper的优化[J]．计算机工程，2014，40（4）：53-56.

[8] 潘　冰．面向资源的RESTful Web应用研究[J]．网络新媒体技术，2010，31（7）：38-43.

责任编辑　陈　蓉