彩铃铃音管理系统的设计与实现①

Design and Implementation of Color Ring Back Tone Service Ring Management System

徐 磊 廖建新 王 纯 朱晓民
(北京邮电大学网络与交换技术国家重点实验室 北京 100876)
(东信北邮信息技术有限公司 北京 100083)

摘 要: 彩铃业务的市场规模日益扩大。在彩铃平台中,彩铃铃音占据十分重要的地位。本文深入分析了彩铃业务对彩铃铃音的要求和彩铃业务运营过程中彩铃铃音出现的问题,提出了一种新的彩铃铃音管理系统。该系统的铃音列表生成模块以后台脚本的形式定时生成铃音列表文件,相比一般从数据库直接获取数据的方式,大大减少了忙时占用的数据库服务器资源数,提高了数据获取的速度。铃音文件处理模块,降低了彩铃平台与铃音相关故障的发生率。该铃音管理系统在整个彩铃平台中具有相对独立性,因此,对其他类似实体的管理系统具有很强的借鉴意义和参考价值。

关键词: 彩铃业务 铃音管理 排队模型 定时脚本

1 引言

作为彩铃平台的关键要素之一,铃音贯穿于彩铃平台的各个模块。从底层资源节点 RN(Resource Node)到业务逻辑,最后到用户终端,始终离不开铃音这一要素。狭义的铃音是指取代电话"嘟嘟"回铃音的音乐回铃音,广义的铃音则泛指各种具有信息量的回铃音,除包括各种类型的歌曲外,还包含笑话,相声等有声艺术形式的录音,以及新闻,天气预报,股票信息等其他各种录音。

铃音信息在彩铃平台中既包括数据库中的铃音记录,又包括存放在 RN 中的铃音文件。提高彩铃平台中铃音信息的读取速度,减少铃音信息获取过程中占用的资源,对于提升彩铃平台中管理的性能起到十分重要的作用;在彩铃业务的运营过程中,有关铃音的故障占据彩铃平台故障的大多数,加强铃音的管理会大大降低彩铃平台的故障发生率,并提高彩铃平台的稳定性,基于上述分析,在彩铃平台中构建铃音管理系统具有十分重要的意义。

2 存在的问题

一套中等规模的彩铃平台大约包含 6 万多首铃音,500 多万彩铃用户数,彩铃平台需要同时处理各个用户获取铃音信息的请求,忙时需要频繁读取数据库,占用服务器大量的资源。

彩铃平台铃音的铃音文件存放在 RN 上,一个中等规模的彩铃平台一般具备 20 台 RN,每条铃音记录在 20 台 RN 各对应一个铃音文件。在彩铃平台的运营过程中,出现如下问题:首先,通过彩铃管理平台上传铃音时,需要在数据库增加一条铃音记录,并在 20 台 RN 各上传一份铃音文件,当彩铃管理平台需要批量上传铃音时,会占用大量的操作时间;上传过程中由于网络或人为的原因,会出现 RN 铃音格式不符合 RN 规定的要求或各个 RN 的铃音文件不同步,导致给用户播放铃音时出现杂音或听不到铃音的故障[1]。

① 基金项目:国家杰出青年科学基金(No. 60525110);国家 973 计划项目(No. 2007CB307100, 2007CB307103);新世纪优秀人才支持计划(No. NCET-04-0111);电子信息产业发展基金项目(基于 3G 的移动业务应用系统);国家高技术产业化信息化装备专项项目(支持数据增值业务的移动智能网系统)

本文将构建彩铃铃音管理系统，该系统包括铃音列表生成模块和铃音文件处理模块，针对以上问题提出解决方案，并对方案在提高彩铃平台性能和稳定性方面的作用进行分析。

```
<listDef>
  <listDef-Name>ListByType_1</listDef-Name>
  <listDef-Condition>type=1</listDef-Condition>
  <orderByDef>
    <orderByDef-OrderBy>ringid</orderByDef-OrderBy>
    <fileDef>
      <fileDef-FileNamePrefix>ringid</fileDef-FileNamePrefix>
      <fileDef-NumPerPage>50</fileDef-NumPerPage>
    </fileDef>
    <fileDef>
      <fileDef-FileNamePrefix>ringid_wap</fileDef-FileNamePrefix>
      <fileDef-NumPerPage>5</fileDef-NumPerPage>
    </fileDef>
  </orderByDef>
  <orderByDef>
    <orderByDef-OrderBy>ringname</orderByDef-OrderBy>
    <fileDef>
      <fileDef-FileNamePrefix>ringname</fileDef-FileNamePrefix>
      <fileDef-NumPerPage>50</fileDef-NumPerPage>
    </fileDef>
  </orderByDef>
</listDef>
```

图 1　铃音信息文件生成模块 XML 配置文件的基本配置单元

3　系统设计

针对第 2 节中提到频繁读取数据库占用大量资源的问题，铃音管理系统的铃音列表生成模块将用户铃音信息在闲时一次性读出，并存入文件系统，对于忙时用户的请求，直接从文件系统中读出，一方面大大降低了数据库服务器的负荷，另一方面提高了用户获取铃音信息的速度。

铃音管理系统的铃音文件处理模块主要包括三部分：铃音拷贝脚本，铃音转换脚本和铃音同步脚本。

铃音拷贝脚本将以前彩铃平台上传铃音的同步方式改为异步方式，CP（Content Provider）上传铃音文件时只要通过初步的格式检查，即收到成功反馈，对铃音文件后续向各个 RN 分发拷贝的操作放在后台由铃音拷贝脚本完成。

各个 RN 对铃音文件的格式要求不尽相同，铃音文件上传以后，铃音转换脚本将按照 XML（Extensible Markup Language）配置文件的要求，对各台 RN 的铃音文件进行转换。

在向每个 RN 拷贝铃音的过程中以及某些故障发生时，会出现部分 RN 铃音文件大小异常或缺失的情况，铃音同步脚本可以对彩铃平台的所有铃音进行全量扫描，得到 RN 铃音文件的缺失情况，并根据一定规则定位 RN 铃音文件的大小异常，最后对缺失和大小异常的铃音分别进行相应处理。

铃音拷贝脚本，铃音转换脚本和铃音同步脚本均在后台各自以一个进程的形式运行，需要考虑各自启动的时间间隔，运行期三者之间的冲突以及如何避免冲突等问题，使三者协调工作。

3.1　铃音列表生成模块

3.1.1　设计原理和实现机制

彩铃平台向用户展现的铃音记录分类的途径包括按类型，按 CP 等，在各种分类途径之上又可以按不同的排序方式进行排列，排序方式包括按铃音编号，按铃音名称，按铃音歌手，按提供商名称，按价格等。

铃音列表生成模块实现了通过配置 XML 文件将铃音记录按照各种分类途径，以各种排序方式，采用不同写入方式自动生成各种类型铃音信息文件的功能，图 1 是 XML 配置文件的一个基本配置单元。

图 1 中的基本配置单元表示将类型为 1 的铃音记录，按照铃音编号和铃音名称排序，形成两种类型铃音记录，将按照铃音编号排序的铃音记录生成两类铃音信息文件，一类每个文件写入 50 条铃音记录，另一类每个文件写入 5 条铃音记录，将按照铃音名称排序的铃音记录生成每个文件写入 50 条铃音记录的铃音信息文件。

铃音信息文件生成模块包含 XML 配置文件解析程序，将层次结构的配置项逐一对应到层次结构的数据结构中。

图 2 为配置文件解析程序的基本类，其中 ListDef

类为分类途径类，OrderByDef 类为排序方式类，FileDef 类为文件类型类。使用 org. w3c. dom 包提供的支持[2]，编写程序层次遍历 XML 配置文件各个结点的同时，使用 ListDef 类，OrderByDef 类和 FileDef 类的 setter 方法对各自的属性进行赋值。

在模块的主程序中，使用 ListDef 类的 getter 方法取出条件属性值和 OrderByDef 类的 List 值，在主程序的第一层循环里对该 List 的各个 OrderByDef 元素逐个进行处理，使用 getter 方法从 OrderByDef 类取出排序方式属性值和 FileDef 类的 List 值以后，从数据库中获取符合 ListDef 类的条件属性值且按 OrderByDef 类的排序方式属性值排序的铃音信息记录，在主程序的第二层循环中对 FileDef 元素逐个处理，使用 getter 方法从 FileDef 类取出文件前缀属性值以及单个文件行数属性值，将每条铃音信息记录按照文件前缀值和单个文件行数值的要求生成铃音信息文件。

```
class ListDef {
    String name ;
    String condition ;
    List orderByDefList;
    public void setName ();
    public String getName ();
    public void setCondition ();
    public String getCondition ();
    public void setOrderByDefList ();
    public List getOrderByDefList ();
}
```

```
class OrderByDef {
    String orderBy;
    List fileDefList;
    public void setOrderBy ();
    public String getOrderBy ();
    public void setFileDefList ();
    public List getFileDefList ();
}
```

```
class FileDef {
    String fileNamePrefix;
    int numPerPage;
    public void setFileNamePrefix ();
    public String getFileNamePrefix ();
    public void setNumPerPage ();
    public int getNumPerPage ();
}
```

图 2　配置文件解析程序类设计

3.1.2 性能分析

以下将建立一个适当的模型，基于该模型对彩铃平台数据库连接部分进行分析，以得出铃音列表生成模块对减少彩铃平台数据库负荷，提高彩铃平台数据库性能方面的量化指标。

彩铃平台的应用通过数据库连接池获得到数据库的连接，连接池的连接数存在一个上限值。当接收到一个应用的连接请求时，如果当前连接池内存在可用的连接则将该连接分配给该应用，如果连接数达到上限，则该应用的连接请求返回失败。

以上的实际运行机制可以抽象成一个多服务窗损失制排队模型，设应用对数据库请求相继到达的时间间隔服从负指数分布，数据库处理请求的时间间隔也服从负指数分布。该模型可以用 M/M/n/n 来表示，其中前两个 M 分别代表请求和服务时间间隔服从负指数分布，第一个 n 代表服务窗的个数，即连接池的上限值为 n，第二个 n 代表系统容量，即连接池同时容纳的连接数为 n。

设应用的数据库连接请求平均到达率为 λ，数据库处理应用请求的平均服务率为 μ，则根据多服务窗损失制排队模型的 K 氏代数方程可以求出如下目标参量[3]：

损失概率 $p_n = \lambda^n/\mu^n n! \sum_{i=0}^{n}(\lambda^i/\mu^i i!)$　(1)

单位时间占用服务窗的均值 $L_{服} = \lambda(1-p_n)/\mu$　(2)

增加铃音列表生成模块前后的 λ 值分别为 30 次/s 和 15 次/s，增加铃音列表生成模块前数据库处理一个请求的平均时间为 200ms，μ 值为 5 次/s，增加铃音列表生成模块后数据库处理一个请求的平均时间为 150ms，μ 值为 6.7 次/s，连接池的上限值即 n 值设为 10。将数值代入公式(1)和(2)得：

表 1　增加铃音列表生成模块前后数据对照表

	增加铃音列表生成模块前	增加铃音列表生成模块后
λ/μ	6	2.25
n	10	10
p_n	0.043	9.659×10^{-5}
$L_{服}$	5.741	2.250

从表 1 中可以看出，由于铃音列表生成模块的增加使损失概率即彩铃平台应用数据库连接请求失败的

概率由 0.043 降低到 9.659×10^{-5}，每秒占用服务窗的均值即每秒彩铃平台应用数据库连接请求占用数据库连接池的连接数由 5.741 降低到 2.250。

3.2 铃音文件的处理

3.2.1 铃音拷贝脚本

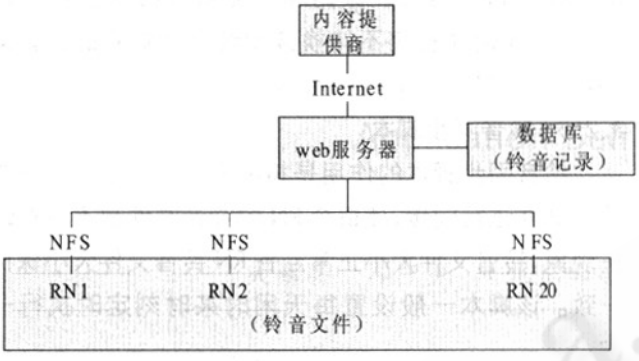

图 3　彩铃平台铃音上传相关实体

如图 3，彩铃平台通过 web 服务器与 Internet 相连，CP 将铃音文件上传到 web 服务器，web 服务器通过 NFS（Network File System）与各个 RN 相连，NFS 是一个 RPC（Remote Procedure Calls）服务，使 web 服务器和各台 RN 能够共享文件，web 服务器使用“mount”命令把各个远端 RN 文件系统挂接在自己的文件系统之下，使各个 RN 的文件使用上和 web 服务器本机上的文件没有不同。

传统的铃音上传实现机制为：铃音文件先上传到第一个 RN，假定为 RN1，然后再从 RN1 依次拷贝到其他 RN 上，由于 RN 一般在物理上是分散的，此种方式下，拷贝的速度比较慢，CP 上传铃音尤其批量上传铃音会等待较长时间，影响用户体验，而且在通过 NFS 挂载到 web 服务器的各个 RN 之间同时读写文件时，由于网络原因会出现文件传输不完整，造成某些 RN 铃音文件数据不完整的现象。

为了解决这种问题，新的机制在上传铃音时，只将铃音文件上传到 web 服务器本地文件系统，同时向数据库里插入铃音记录，铃音的分发由铃音拷贝脚本来完成，以下详细介绍该脚本的实现机制。

铃音拷贝脚本每隔 60 分钟定时执行一次，初始化时，读取 XML 配置文件的数据库连接信息，文件路径信息以及数据库表结构信息，然后扫描所有的铃音记录，找出状态为待拷贝的铃音记录，根据读取的信息将文件分发到各个 RN。

遍历铃音表待拷贝状态的铃音记录，根据同步类型判断铃音文件在本地文件系统的位置，规定同步类型为“1”表示拷贝首次上传的铃音文件，同步类型为“4”时为拷贝重传铃音文件，首次上传的铃音原文件存放在“CPID”命名的文件夹下，而重传的铃音原文件存放在“CPID”+“1”命名的文件夹下。

检测 web 服务器与 RN1 的 NFS 连接是否正常，检测 RN1 正常连接的机制为：在 RN1 共享给 web 服务器的 RingFile 目录下放一个 exist.txt 标记文件，脚本判断该文件是否存在，如果可以检测到该文件存在则认为 RN1 正常 mount 到 web 服务器。

检测本地待拷贝的铃音文件是否存在，如果不存在，则记录重要异常日志然后退出程序，否则将原文件逐个拷贝到各个 RN 上，此脚本只负责铃音拷贝，不需要铃音转换，铃音转换由铃音转换脚本实现，具体参见 3.2.2 节。

拷贝中错误规避机制为：增加异常计数器，初始值为 0，如果铃音文件拷贝到各个 RN 操作完成，则异常计数器为 0，拷贝铃音表记录状态字段置为 2，表示拷贝成功完成并删除本地原文件；

如果异常计数器大于 0 小于 RN 总数，且正常上传到 RN1，则拷贝铃音表记录状态字段置为 1，表示拷贝失败或部分失败，但 RN1 拷贝成功，所以可以删除本地原文件，因为铃音同步脚本可以从 RN1 同步到其他 RN，详见 3.3.3 节。

如果“异常计数器大于 0 小于 RN 总数，且 RN1 上传不成功”或者“异常计数器等于 RN 数”则拷贝铃音表记录状态字段仍然为 0，表示仍然为待处理状态，下次定时执行时需要再次处理[4]。

该脚本除了可以分发铃音文件外，还可以通过修改配置文件向其他机器分发试听文件，语音文件，CP 活动文件等。

3.2.2 铃音转换脚本

铃音文件上传成功以后，由铃音转换脚本按照 XML 配置文件的要求对铃音文件进行 wav 格式到 vox 格式的转换。该脚本在转换铃音时调用底层可执行文件对某一铃音文件进行格式转换。

对一个铃音一次完整的处理过程中需要进行两次拷贝操作，两次删除操作和一次铃音格式转换操作，如

果有多个转换脚本的进程同时运行，会占用服务器大量 CPU 资源，但又必须保证上传的铃音文件得到及时转换，否则会出现播放杂音等问题。因此，在及时性和资源的占用两者之间进行折中考虑，将该脚本设置为每隔 40 分钟执行一次，且任意时刻只有一个转换脚本的进程在运行；另外为减少扫描时间和占用的资源，铃音转换脚本只扫描并转换当天新上传的铃音的记录。

需要建立一个 timestamp 文件，脚本每次执行转换操作前读取 timestamp 文件中的三行信息，第一行为"0"或"1"，"0"代表当前没有铃音转换脚本进程正在运行，"1"代表当前某一铃音转换脚本正在运行；第二行记录上一铃音转换脚本进程的时间戳；第三行记录上一铃音转换脚本进程的 PID。

第一行为"1"，且当前时间戳与第二行时间戳的差值小于 18000000(ms)时，表示有另一铃音转换脚本进程正在执行且未超时，本脚本此时终止执行。

第一行为"0"，或者第一行为"1"且当前时间戳与第二行时间戳的差值大于 18000000(ms)时，表示当前没有另一铃音转换脚本进程在运行或有另一铃音转换脚本进程在运行但超时，则当前脚本继续执行下面的操作。

遍历铃音记录，选出铃音表里当日上传且尚未转换的铃音记录，从铃音记录中取得铃音文件路径信息，调用转换脚本逐个进行转换。

转换的机制为：在 web 服务器本地文件系统建立一个临时目录，将 wav 文件转换成 vox 文件后，放到该临时目录里，然后删除原 wav 文件，再从临时目录里将 vox 文件拷贝到原目录，最后删除掉临时目录里的铃音文件。

3.2.1 节中提到，RN 使用 NFS 机制 mount 到 web 服务器，如果转换过程中原文件和目标文件都存放在 RN 目录下，NFS 自身传输机制的缺陷会使铃音数据文件拷贝不全，导致各个 RN 铃音文件大小不一致。因此，该脚本采用如上建立本地中间临时目录的机制。

本地临时文件夹里生成 vox 文件以后，由于通过 NFS 机制 mount 到 web 服务器的 RN 目录下的文件，默认对本地非组内用户没有写权限，因此，如果用 vox 格式的铃音文件直接覆盖 RN 目录下同名 wav 格式的铃音文件，将会出现"权限拒绝"的异常。该脚本采用的机制为调用 java 程序，使用 delete() 方法将 RN 目录下的铃音文件先删除掉，然后将 vox 文件拷贝到 RN 目录下。

遍历完毕，更新 timestamp 文件，向第一行写入"0"，第二行和第三行分别写入当前的时间和脚本启动进程的 PID，然后在数据库中插入该已转换铃音记录，下次定时执行将不转换该铃音记录对应的铃音文件。

3.2.3 铃音同步脚本

铃音同步脚本的作用是将所有铃音在各个 RN 的铃音文件进行同步，使每个 RN 的铃音文件都不存在缺失现象，铃音文件大小正常且各 RN 铃音文件大小保持一致。该脚本一般设置每天闲时某时刻定时执行一次；

遍历所有铃音记录，从铃音记录中取出路径信息，逐个检查各个 RN 目录的铃音缺失和各个 RN 目录铃音的大小情况。遇到某个 RN 铃音缺失或某 RN 铃音大小异常，则用 RN1 上的铃音文件进行同步。3.2.1 节中提到，上传铃音先传到 RN1 上，如果 RN1 上缺失铃音文件，其他 RN 必然同样缺失，RN1 上铃音文件缺失在日志中记录异常，脚本不做自动处理。

RN 的板卡有两类，分别为支持 wav 铃音文件格式的 D 卡和支持 vox 文件格式的 N 卡，一般将 D 卡 RN 放在前面，N 卡 RN 放在后面。

铃音大小异常的标准是：当 RN1 为 N 卡时，即所有 RN 均为 N 卡，其他 RN 对应铃音的大小应与 RN1 大小一致。当 RN1 为 D 卡时，则其他 D 卡铃音大小应与 RN 对应铃音大小相同，其他 N 卡铃音大小应为 RN1 上 wav 格式铃音大小的 1/4，偏移值在 2kB 以内；

缺失铃音的拷贝机制为：对于 D 卡 RN，建立一个临时目录，将 RN1 目录下的 wav 文件拷贝到临时目录，再从临时目录拷贝到缺失铃音文件的 RN 目录下；对于 N 卡 RN，同样建立一个临时目录，如果 RN1 为 N 卡，直接将 RN1 铃音拷贝到临时目录里，如果 RN1 为 D 卡，将 RN1 上的 wav 文件转换成 vox 文件后，放到该临时目录里，最后从临时目录里将 vox 文件拷贝到缺失铃音文件的 RN 目录下；

铃音大小异常的拷贝机制为：对于 D 卡 RN，建立

(下转第 18 页)

(上接第 14 页)

一个临时目录,将 RN1 目录下的 wav 文件拷贝到临时目录,删除原 RN 目录中大小异常的铃音,再将铃音文件从临时目录拷贝到铃音大小异常的 RN 目录下;对于 N 卡 RN,同样建立一个临时目录,如果 RN1 为 N 卡,直接将 RN1 铃音拷贝到临时目录里,如果 RN1 为 D 卡,将 RN1 上的 wav 文件转换成 vox 文件后,放到该临时目录里,删除原目录下大小异常的铃音,最后从临时目录里将 vox 文件拷贝到铃音大小异常的 RN 目录;

建立中间临时目录以及先删除原铃音文件再拷贝的目的同 3.2.2 节。

3.2.4 各脚本的配合

CP 上传铃音时将铃音记录插进数据库,并将铃音文件上传在 web 服务器本地目录,如果铃音拷贝脚本还没有将铃音文件拷贝到各台 RN,铃音转换脚本就开始执行,发现各台 RN 铃音记录对应的路径下无铃音文件,此时铃音转换脚本不会将该铃音记录插入已转换铃音记录中,但如果在一天之内铃音拷贝脚本由于 RN 目录文件夹权限问题或其他意外原因,没有将铃音文件从本地拷贝到 RN 目录下,铃音转换脚本第二天就不对该铃音记录进行检查,此时每天定时执行的铃音同步脚本可以在铃音拷贝脚本拷贝完成后,进行转换和同步操作,从而保证了各台 RN 铃音格式的正确性。

4 总结

为满足彩铃平台铃音这一关键要素的新需求,构建了一种彩铃铃音管理系统。该系统的铃音列表生成模块,采用闲时将铃音信息一次性读出,生成各种类型铃音信息文件,忙时从铃音信息文件直接获取相应信息满足用户请求的机制,与一般直接从数据库获取数据的方式相比,降低了彩铃平台数据库忙时的负荷,并提高了获取铃音信息的速度。

参考文献

1 沈奇威、廖建新、王纯、朱晓民,彩铃业务的研究和设计,第九届全国青年通信学术会议论文集,重庆,中国,2004.5:484-489.

2 [美]Didier Martin 等著,李哲、严春莹、马琳等译,XML 高级编程,机械工业出版社,2001.1.

3 陆传赉编著,排队论,北京邮电大学出版社,1994.5.

4 [美]William Crawford, Jonatban Kaplan 著,刘绍华、毛天露译,J2EE Design Patterns, 中国电力出版社,2005.10.