基于 Rest Web Services 的移动终端数据采集系统研究*

花 奇 刘幺和
(湖北工业大学 武汉 430068)

摘 要 随着移动设备的普及,二维条码得到了广泛的应用,其主要是用于文本信息的存储,而对图像、声音和视频的存储是少之又少。用二维条码存储图像、声音和视频等信息会比文本存储的内容更加丰富、更加直观。但随之而来的是海量数据的产生,而怎样对数据进行采集,然后在后端对数据进行高效的存储和操作是当今的热点问题。论文在对数据采集的基础上,研究如何高效地进行存储和交互数据。

关键词 二维条码; MongoDB; 轻量级; 数据存储交互
中图分类号 TP311 **DOI**:10.3969/j.issn1672-9722.2014.11.038

Date Acquisition System for Mobile Terminal Based on Rest Web Services

HUA Qi LIU Yaohe
(Hubei University of Technology, Wuhan 430068)

Abstract With the popularity of mobile devices, QR bar code has been widely used, which is mainly used for text information storage, but very limitedly for images, audios and videos. Using QR bar code store images, audios and videos, which will be much more plentiful and intuitive than the text, will produce massive amounts of date. So, the current hot issue is how to collect data and then to handle and store them efficiently in the back-end. This article is a study of efficient data storage and interaction based on the data collection.

Key Words QR barcode, MongoDB, lightweight, date storage interactions
Class Number TP311

1 引言

物联网是以互联网为基础,建立一个物物相连的网络。它是基于互联网的进一步扩展,并扩展到了任何物和物之间进行数据交互。

当前数据的爆炸式的增长推动了移动设备的发展,诸如智能手机,平板电脑等的出现使得移动终端成为了热门的平台,而二位编码技术与移动设备的结合,则使得信息交流更加便捷。

通过移动终端采集所需数据,然后经互联网传输到数据处理中心,再由数据处理中心完成对信息的相关操作,并反馈给用户需要的数据。这就需要进行移动数据的采集和后台数据的高效处理。在拥有海量数据的物联网中,如何快速高效地存储、处理和提取这些海量数据,从中挖掘出有价值的信息,成为当下的热点问题。

数据通过移动终端采集后在云平台上进行交换,当数据量比较大时,就会影响数据的传输性能。而一种新生的称为 REST 风格的 Web Services 可以通过缓存来提高反应速度,其在性能和简易性上也大大优于 SOAP 协议。

数据传输到后台数据库,要对这些海量数据进行存储、处理和提取。但在如此巨大的、不规则的数据面前,传统的关系型数据库是很难处理的,它面临着对海量数据的管理和存储、灵活性等一系列的问题。互联网的高速发展,一种非关系型数据存储(NoSQL)解决方案被提出。

本文利用非关系型数据库来研究 REST 风格

* **收稿日期**:2014 年 5 月 5 日,**修回日期**:2014 年 6 月 27 日
作者简介:花奇,硕士研究生,研究方向:机械工程。

的 Web Services 的移动终端数据采集系统。

2 Restful Web Services

由于一般的 Web Services 连接数据库的相关性、SQL 语言及 SOAP 协议的使用,使得数据量越大,系统处理数据的能力就越差。为解决这一问题,Rest 风格的 Web 服务被提出。REST 式架构是面向资源的[1],它将实际问题转换成 REST 风格的 Web Services,使 URI、HTTP 和 XML 的工作方式与其他 Web 应用一样[2]。它的四个标志特性:可寻址性、无状态性、连通性以及统一接口。时下,越来越多的网站使用 Rest Web Services 来设计。

REST 是一种轻量级的 Web Services 架构风格,它基于 HTTP 协议[3]。因此,没有复杂的 SOAP 协议的引入,可以通过 HTTP 协议来实现的。它比 SOAP 更简洁。

例如,从 Web Servicse 中显示一个 phonebook 图像,用 SOAP 代码形式表达为

```
〈? xml version="1.0"?〉
〈soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding"〉
〈soap:bodypb="http://www.acme.com/phonebook"〉
〈pb:GetUserDetails〉
〈pb:UserID〉12345〈/pb:UserID〉
〈/pb:GetUserDetails〉
〈/soap:Body〉
〈/soap:Envelope〉
```

用 Restful Web Service 代码形式表达为

```
http://www.acme.com/phonebook/UserDetails/12345
```

显然,SOAP 要建立大量的 XML 消息,而 REST 则不同,它是建立一个网址字符串来访服务,并直接连接到网络,在浏览器中显示,既简单又实用。

REST 约束条件可以生成一个轻量的、高效率的架构。由于它的轻量以及高效等特性,REST 成为 SOAP 服务最有前途的替代者。REST 充分利用有缓存支持的 HTTP 协议,让大数据从服务器的内存中解放[4]。当客户端首次通过 HTTP GET 请求从服务器获得数据后,该数据就被缓存服务器缓存。下一次客户端发出相同请求时,缓存就会直接给予响应,而不必向远程服务器发出请求。而这一切对客户端来说都是透明的。因此,REST 相对于 SOAP 和 XML-RPC 更加的轻量[5],它能够通过 HTTP 协议彻底完成,还能够使用缓存来提高响应速度,其性能、效率和易用性比 SOAP 协议更好。

3 NoSQL

随着信息时代的到来,网络用户量快速增长。互联网应用内容从最初的如航班预定、股票交易,发展到如今的涉及到购物,娱乐,物流,社交,通信等全方位领域。数据量从以前的 TB 级升至 PB 级,并继续爆炸性增长,互联网已经进入了大数据时代。面对海量数据,传统的关系型数据库显得不再那么适用[6]。

1) 高并发读写

根据用户对 Web2.0 网站的操作,提供实时的动态信息,很难用动态页面静态技术来实现,数据库的并发控制和负载很高,经常读/写请求可以达到每秒几万次。

2) 对大数据的高效存储和操作

如新浪微博、人人网等 SNS 网站,用户动态信息每时每刻都会大量的产生。在关系型数据库中查询效率比较低下。此外,例如京东、淘宝等这样的电商以亿为单位的用户账号,用传统的关系型数据库来管理查询也是很难应付的。

3) 高可用和高可扩展的数据库

基于 Web 构架,关系型数据库很难像 Web 服务和 APP 服务,简单地通过横向扩展增加更多的硬件和服务器节点。现在大多数网站都需要对其进行实时更新和服务,可是数据库的升级和扩展需要停机维护和数据迁移,这样就会对公司效益造成比较大的损失。

因此,越来越多的应用在关系型数据库也显得不那么方便适用,为了进一步的发展而应运而生了 NoSQL 数据库。

删除传统关系型数据库关系的特性,内部使用的是非关系型数据组织方式,这就是 NoSQL 数据库。NoSQL 数据库并非取代关系型数据库,而是用非关系型的方式,来解决关系型数据库所遇到的问题。

3.1 NoSQL 在移动领域的应用模型

如图 1 所示的应用模型由三个重要部分组成[7]。

3.1.1 数据库服务器(NoSQL 类型)

因为热门的社交网站有庞大的用户群,产生了传统的关系型数据库无法应对的数据量,所以应当

选用专门的数据库来解决这些问题。这里选用非关系型数据库。

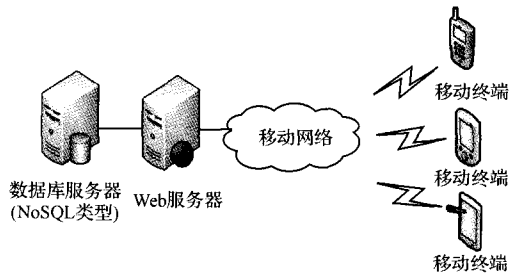

图 1　NoSQL 数据库与无线通信相结合的应用模型

3.1.2　Web 服务器

主要功能是对发出请求的一方，提供所需信息，供其浏览。移动终端通过移动网络和 Web 服务器连接数据库服务器，以获得所需的数据。

3.1.3　*移动终端*

一般移动设备由于大小和硬件等条件的限制，其处理能力和存储能力和 PC 相比都受到了限制，所以移动终端用户通过设备以移动网络为媒介来访问应用服务器中的数据。

3.2　NoSQL 数据库分类与选择

NoSQL 数据库按照数据的存储类型和特征能够分为很多种[8]，如文档式存储的 MongoDB、列式存储的 Hbase、键值式存储的 Redis、对象式存储的 db4o 等多种类型。它们的共同的特点是都没有关系型数据库的关系型特征。现在，NoSQL 中最热门当属 MongoDB 了。MongoDB 是非关系型数据库，它是面向文档的，一条记录就能够使层次关系表示的非常复杂，它适合于面向对象的开发应用。它的存储不需要固定的模式，所以很适用于当下移动终端上海量的、不规则的数。同时，用 MongoDB 传输协议与服务器进行交互，其开销远低于用 HTTP/REST。

MongoDB 数据库之间没有关系性，数据结构非常简单，所以其读/写性能非常高，并且非常容易扩展。按照官方文件，当数据量到达 50GB 或更多，MongoDB 数据库访问速度 10 倍于传统关系型数据库。

MongoDB 数据库存储结构非常简单，不需要表结构固定，而是以文档形式存储，文档是 MongoDB 的核心[9]，一组文档就构成了一个集合，MongoDB 中每个数据就就是由一个或者多个集合组成。如果将 MongoDB 和关系型数据库比较，那么文档就类似于行，而集合就如同表一样。它能很好地存储对象类型数据，因此就不需要进行表与表之间的关联查询了。

例如：在关系型数据库中创建一个表用来存储学生个人信息。存储的信息包括：姓名(name)、年龄(age)、性别(sex)。

首先需要创建这个表，将这个表命名：studentinfo，该表的创建代码如下：

```
Create table studentinfo
{
        name varchar2(50),
        age number(10),
        sex varchar2(10)
}
```

然后将学生信息值插入表中，代码如下：

```
insert into studentinfo values("小周",19,"男")
```

当需要往学生信息中添加一个字段来存储学生的地址信息(地址信息包括门牌号、详细地址信息)，由于地址信息较多，一个字段不足以放下，那么就需要再创建一个表来存储这些信息，然后学生信息表再引用地址信息表的一个编号。

创建表的代码如下：

```
create tablestudentinfo
{
        name varchar2(50),
        age number(10),
        sex varchar2(20),
address_no number(5)
}
```

创建地址信息表，代码如下：

```
create table address
{
      address_no number(5),
      house_number varchar2(20),
      address_info varchar2(1000)
}
```

首先向地址信息表中插入地址信息，代码如下：

```
insert into address values(1,"1","湖北省、武汉市、
洪山区、南湖花园")
```

接下来再插入学生的基本信息，代码如下：

```
insert intostudentinfo values("小周",19,"男",1)
```

查询学生的详细信息时就必须对这两张表进行关联查询，代码如下：

```
Select a. name,
        a. age,
        a. sex,
        b. house_number,
        b. address_info
from studentinfo a,
      address b
```

```
where a. address_no=b. address_no
```

当运用 MongoDB 数据库存储上例中的学生信息和学生地址信息时,可以这样做,代码如下:

```
>db. studentinfo. insert({"name":"小周","age":19,
"sex":"男","address":{"house_number":"1","ad-
dress_info":"湖北省、武汉市、洪山区、南湖花园"}})
>
```

直接将地址信息作为一个文档对象存入到 address 字段中。查询时直接使用简单的单表查询,代码如下:

```
>db. studentinfo. find()
{"name":"小周","age":19,"sex":"男","address":
{"house_number":"1","address_info":"湖北省、武
汉市、洪山区、南湖花园"}}
>
```

执行完查询之后,使用 MongoDB 的游标直接可以输出详细的学生个人信息。

目前数据仓库技术直到最近才跟得上数据库容量的增长趋势。MongoDB 的出现很好地解决这个问题。例如,关系型数据库存在的并发负载高的问题、高效存储和访问大量数据的问题、高可用性和高扩展性的问题。这些都是关系型数据库无法解决的。而 MongoDB 可以很好的解决这些问题,所以 MongoDB 是今后最为合适的选择。

4 MongoDB 在信息采集系统中的应用

以二维条码对信息管理作为研究对象。信息采集系统对查询本身的要求并不复杂,但需要在面对海量数据时保证查询的速度。常见的一种查询是,根据某种查询条件,如关键字或者时间点,来返回符合条件的信息,并可以按时间顺序排序结果。数据采集系统要采集各种信息,数据量非常大,而且二维条码用户的活跃度很高,每天都有成百上千的新二维条码生成。传统的关系型数据库很难达适用。

针对信息采集系统的数据库需求,选用 MongoDB 数据库进行数据的存储和查询。

4.1 存储方案

因为 MongoDB 中的一个集合由多个文档构成,每个文档的数据结构,可以不一样,可以达成数组类型和嵌套子文档,而信息采集系统的数据结构在 MongoDB 中只要设计一个集合就可以实现。

针对本系统,首先创建一个数据库 webinfo,在其中创建一个集合来存储信息,具体实现代码如下[10]:

```
MongoServer server = MongoServer. Create("mon-
godb://127. 0. 0. 1:27017");//指定数据库服务器
MongoDatabasedb = server. GetDatabase("webin-
fo");//获取 webinfo 数据库,不存在则创建
MongoCollection<BsonDocument> posts = db. Get-
Collection("posts");//创建 Collection,名为 posts
```

创建一个文档的代码如下:

1) 使用 BsonDocument 类创建文档,添加简单数据类型的属性:

```
BsonDocument doc = new BsonDocument ();//创
建文档
doc. put("url","http://bbs. hupu. com/");//添加
URL 属性的键值对
doc. put("title","NBA 战报");//添加帖子标题属
性的键值对
```

2) 使用 BasicDBObject 类创建子文档。

```
BsonDocument reply = new BsonDocument ();//创
建子文档 floor
reply. put("author", author);//添加子文档的发帖
人属性键值对
reply. put("date", date);//添加子文档的发帖时间
属性键值对
reply. put("content", content);//添加子文档的发
帖内容属性键值对
```

3) 循环读取所有的回复,重复步骤 2)的语句,将每一个回复的子文档 reply i 添加入文档;

4) 将创建好的文档插入集合中

```
posts. insert(doc);
{
  {author:作者字符串}
  {title:标题字符串}
  {date:发布日期整型}
  {url:网址字符串}
  {content:内容字符串}
  {reply:回复嵌套子文档
    {
      {author:作者字符串}
      {date:发布日期整型}
      {content:内容字符串}
    }
}
```

4.2 查询方案

在信息采集系统中,常见的是以关键字(例如地点、姓名等)进行查询,查询的结果按照指定的顺序排列。以"MongoDB"关键字为例,在所有帖子中查找,返回帖子信息及其所有回复信息,并按时间降序排序。MongoDB 的 shell 查询语句如下:

```
db. posts. find({"title":{ $exists : "MongoDB"}}).
sort({date:-1})
```

该语句会返回所有包含关键字“MongoDB”的文档。

另一种常见查询，查询具体时间段的帖子数量，其查询语句如下：

```
db. posts. find({"date": { $gt : 201402100000, $lt: 201403100000 }}). count()
```

通过 C# 客户端实现数据库查询工作，主要代码如下：

```
MongoServer server = MongoServer. Create("mongodb://127. 0. 0. 1:27017");//指定数据库服务器
MongoDatabasedb = server. GetDatabase ("webinfo");//获取 webinfo 数据库，不存在则创建
MongoCollection<BsonDocument> posts = db. GetCollection("posts");//指定集合 posts
var query = new QueryDocument("author", "zhuba");//定义查询条件
MongoCursor<BsonDocument> cur = posts. Find(query);//执行查询，返回 MongoCursor 对象存储查询结果文档
foreach (BsonDocumentemp in cur)//循环遍历 MongoCursor，读取查询结果显示
{
lab. Text += "zhuba's post:" + emp["date"]. ToStTing() + emp["title"]. ToString() + "\n";
}
```

5　结语

在海量数据的查询、数据处理中，以 MongoDB 为代表的 NoSQL 数据库，与以 SQLSever 为代表的传统关系型数据库相比，具有明显的优势。数据容量越大，使用 NoSQL 数据库对数据的处理速度与传统关系型数据库相比，其处理速度就越快。因此 NoSQL 更适用于海量数据的处理。这在移动计算领域具备广阔的前景。同时面对不规则的大数据时，使用 REST Web Service 架构比传统的 Web Service 架构的处理速度更加快捷有效。

参 考 文 献

[1] 唐旭华，皱峥嵘. 基于 RESTful Web Services 的空间数据共享[J]. 测绘科学，2010，35(4)：122-124.
TANG Xuhua, ZHOU Zhengrong. Spatial Date Share Based On RESTful Web Services[J]. Science Of Surveying And Mapping, 2010, 35(4): 122-124.

[2] Leonard Richardson, Sam Ruby. RESTful Web Services[M]. 北京：电子工业出版社，2008：79-80.
Leonard Richardson, Sam Ruby. RESTful Web Services[M]. Beijing: Publishing House Of Electronics Industry, 2008: 79-80.

[3] 王非，蔡勇，贺志军. RESTful Web Services 在信息系统中的应用[J]. 计算机系统应用，2013，22(2)：222-226.
WANG Fei, CAI Yong, HE Zhijun. RESTful Web Services Applied In Information System[J]. Computer Systems & Applications, 2013, 22(2): 222-226.

[4] 谢瑞莲，耿国华. 分布式搜索引擎中缓存系统的研究与实现[D]. 西安：西北大学，2009：22-23.
XIE Ruilian, GEN Guohua. Research and Implementation of the Cache System in Distributed Search Engine [D]. Xi'an: Northwest University, 2009: 22-23.

[5] Subbu Allamaraju. RESTful Web Services Cookbook [M]. 北京：电子工业出版社，2011：149-160.
Subbu Allamaraju. RESTful Web Services Cookbook [M]. Beijing: Publishing House Of Electronics Industry, 2011: 149-160.

[6] 刘一梦，王移芝. 基于 MongoDB 的云数据管理技术的研究与应用[D]. 北京：北京交通大学，2012：1-4.
LIU Yimeng, WANG Yizhi. Research And Application Of Cloud Date Management Technology Based On MongoDB[D]. Beijing: Beijing Jiao Tong University, 2012: 1-4.

[7] 马豫青，李晓宇. NoSQL 数据库技术在移动互联网中的应用[J]. 平顶山学院学报，2013，28(2)：60-61.
MA Yuqing, LI Xiaoyu. NoSQL Datebase Technology In The Mobile Internet[J]. Journal of Pingdingshan University, 2013, 28(2): 60-61.

[8] 陈敏敏，李芳，张韧弦. 基于 MongoDB 云存储平台的论坛信息抽取与存储研究[D]. 上海：上海交通大学，2012：1-6.
CHEN Minmin, LI Fang, ZHANG Renxian. Research On Forum Information Extraction And Storage Based On Cloud-Based MongoDB[D]. Shanghai: Shanghai Jiao Tong University, 2012: 1-6.

[9] 邹贵金. MongoDB 管理与开发实战详解[M]. 北京：中国铁道出版社，2013：1-10.
ZHOU Guijin. MongoDB Mangement And Development Of Practical Details[M]. Beijing: China Railway Publishing House, 2013: 1-10.

[10] 沈姝，顾韵华. NoSQL 数据库技术及其应用研究[D]. 南京：南京信息工程大学，2012：49-56.
SHEN Mei, GU Yunhua. Research On NoSQLDatebase Technology And Application[D]. Nanjing: Nanjing University Of Information Science & Technology, 2012: 49-56.