文章编号:1671-4598(2020)12-0086-04 DOI:10.16526/j.cnki.11—4762/tp.2020.12.018 中图分类号:TM721 文献标识码:A

基于数据转发服务器的车辆监控管理系统升级方法

周 霞[1],张 驰[2],曹鑫亮[2],于大为[1]
(1.苏州信息职业技术学院 大数据与互联网学院,江苏 苏州 215200;
2.南通大学 地理科学学院,江苏 南通 226007)

摘要:针对车辆管理系统在新旧系统衔接过程中出现的问题,以数据中转服务器为基础,构建了用于新旧系统衔接的框架;在该框架中,采用高性能 Netty 架构实现了大规模并发车辆数据的实时接收与转发;采用协议转换的技术解决了新旧系统通讯协议不一致的问题;采用数据缓存配合延迟入库的方法对大规模车载传感信息的实时接收、动态表达和批量入库进行了实现;实例表明,该方法能够很好地实现新旧系统的衔接,并有效地解决了衔接过程中数据并发、协议转换、批量入库等问题。

关键词:车辆管理;数据中转;协议转换;数据并发

Method for Upgrading Vehicle Monitoring Management System Based on Data Transfer Server

Zhou Xia[1], Zhang Chi[2], Cao Xinliang[2], Yu Dawei[1]
(1. Suzhou College of Information Technology, College of big data and Internet, Suzhou 215200, China;
2. Nantong University, College of geographical sciences, Nantong 226007, China)

Abstract: Aims at the problems in the process of connecting the new and old systems of the vehicle management system, based on the data transfer server, a framework for connecting the old and new systems is constructed. In this framework, the high-performance Netty architecture is used to realize the real-time collection and transferring of large-amount concurrent vehicle data; the protocol conversion technology solves the problem of inconsistent communication protocols between the new and the old systems; the method of data caching and delayed storage is used to achieve large-amount concurrent vehicle sensor data real-time collection, real-time expression and batch storage. Application examples show that the system can well support the connection between the old and the new system, and effectively solve the problems of data concurrency, protocol conversion, and batch storage in the connection process.

Keywords: vehicle management; data transfer; protocol conversion; data concurrency

0 引言

车辆监控管理平台[1-3]是以因特网、大数据[4-5]及云计算为基础,结合 4G 通信技术、地理信息技术、泛在技术,将不断运动的人、车和静态的路、桥等对象结合[6]。在运转过程中,平台采用车辆传感设备,实时获取车辆行驶过程中的位置、速度、状态等信息,将这些数据进行存储,并进行相关的智能分析[7-11]。对车辆管理系统来说,所需要解决的一个核心问题是大量汽车同时在线所带来的高并发情况。针对该问题,对数据的采集、处理等相关的问题进行了优化[12-16]处理,使得系统能够对高并发数据实现管理。

目前的车辆管理系统大多采用交通部部标 808 协议,该协议能够很好地对车辆的状态进行跟踪并与车辆进行有效交互。然而,目前也存在着大量不采用 808 协议的旧系统。为了将旧系统中的车辆进行统一管理,有必要将这些系统升级到 808 协议平台。但在升级过程中,会遇到如下问题:在新系统开发完成之后,往往不能直接替换,而是需要试运行一段时间。在此期间,旧系统仍然需要继续运行,如何让新、旧系统同时接入实时数据进行运行,是需要解决的核心问题[17]。

针对上述问题,本文提出一种服务器抓包转发的方式来构建一个中转服务器,以该中转服务器为核心,将所接收到的数据分别转发给新旧系统,实现新旧系统的同时运行。

要实现传感器数据的中转,需要解决以下几个问题:

1)大规模数据并发问题。车辆管理系统要实现数据中

收稿日期:2020-09-18; **修回日期**:2020-10-21。
基金项目:国家自然科学基金项目(41501422);苏州高职高专院校优秀教学团队(2019—5);苏州高职高专院校优秀科技服务团队(201509);2020 年苏州高职高专第二批产学研合作基地(2020—5)。

作者简介:周 霞(1982-),女,江苏宜兴人,硕士,讲师,主要从事信息通信技术等方向的研究。
张 驰(1982-),男,江苏南通人,博士,副教授,主要从事地理信息系统方向的研究。
于大为(1970-),男,黑龙江阿城人,硕士,副教授,主要从事信息通信技术方向的研究。

转，大规模并发数据的压力都会集中在中转服务器上。服务器需要同时承担数据的汇聚和转发，很容易形成数据瓶颈。

2）协议转换问题。新系统统一采用部标 808 协议，而旧系统所采用的数据传输协议与新系统不同。因此中转服务器还需要解决的一个问题，即不同协议之间的兼容问题。

3）数据持久化问题。车辆管理系统运行中数据量较大，若每收到一帧数据就立即连接数据库进行入库处理，将会造成数据库的频繁开、关操作，从而造成数据库崩溃的后果。但如果不能对数据实时地进行处理，客户端的数据显示则会产生延迟。

本文接下来将对数据并发、数据入库、协议转换等问题进行解决，最终实现新旧车辆管理系统的有效衔接。

1　数据中转服务器

1.1　并发数据的处理框架

传统 BIO 在接受远程用户的连接请求之后，就会为远程用户建立一个线程并进行处理。服务器在处理过程中会一直保留该线程直至处理完毕后销毁。该架构中，系统为每个单独的连接创建线程的处理方式，在面对大规模并发数据时，系统开销将急剧增长。传统 BIO 模型的处理方式，导致系统伸缩性较差，从而严重制约了整个系统的效率。在极端情况下，大量用户在同一时间申请连接，服务器将会触发线程创建失败、堆栈溢出等情况，甚至可能出现系统崩溃等灾难性后果。

针对上述问题，本项目选择中间件 Netty 作为高性能通信框架，它采用一种基于 NIO 的异步非阻塞通信框架。NIO 框架与传统的 BIO 框架存在较大的区别，它有很强的可扩展性与健壮性。该框架中相关核心组件如下：

1）Buffer。Buffer（缓冲区）是包含一个数组的容器。通道（Channel）中的数据无论是读取还是写入都必须以缓冲区为基础。

2）Channel。Channel（通道）与一般流数据最显著的区别在于，传统流数据是单向的，但对于通道来说，它可以进行双向的读取和写入操作。

3）Selector。Selector（选择器）作为 NIO 框架的核心部件，它能够通过轮询的模式检测到注册的通道是否发生了相关事件，如果有事件发生，则调用事件处理程序进行相应处理。

本文设计的 NIO 的异步非阻塞通信架构中，存在一个线程池，其中包含大量线程。这种情况下，就不需要为每个连接请求重新创建线程。采用线程池之后，用少量的线程就能为大量的远程终端服务了。远程终端建立连接之后，首先需要到选择器上完成注册（见图 1），选择器将对注册在其上的通道轮询，查询是否触发了相应事件。发现查询的通道有事件发生，便可截获该事件并进行处理。当涉及读或写的事件触发时，线程池会调用线程来对其进行处理。该操作模式将原有的同步阻塞式通信方式变为异步非阻塞式通信方式，系统维护成本得以大幅度降低，系统效率也得以大幅度提高。通过该框架，数据中转服务器就可以对大规模并发数据实现很好的接收。

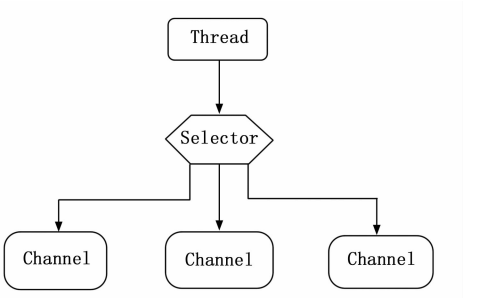

图 1　NIO 内部执行图

1.2　数据中转服务器框架

基于 Netty 框架，本文设计了数据中转服务器，数据中转服务器主要解决大规模数据的并发接收、中转以及协议转换问题。其工作原理如图 2 所示，远程传感器终端发送大量并发数据到达中转服务器的网卡之后，中转服务器对数据包进行抓取，并基于 NIO 框架的线程组向目标主机实现转发。在此期间，中转主机可对所抓取的数据不经过处理，直接转发；也可以在进行简要处理后再向目标服务器进行转发。由于本文的中转服务器需要实现新旧系统之间的协议转换，因此采取上述第二种模式，在接收到远程终端发来的数据后，首先对其进行解析，在此基础上，根据所对接的新、旧系统所使用的协议进行协议转换，最后将转换后的数据通过线程组发送给相应的模块。

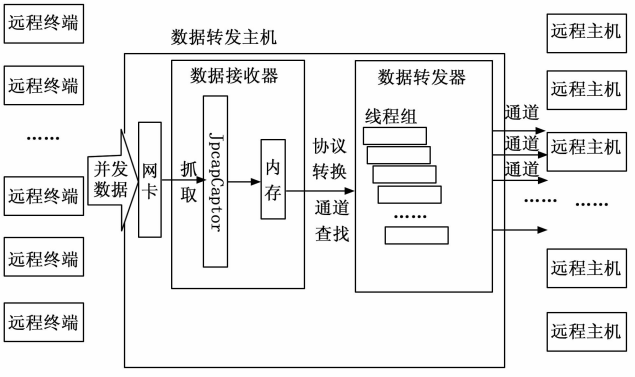

图 2　数据中转模块示意图

通过高性能 NIO 框架以及数据中转服务器，就能很好地解决数据的高并发问题以及新旧系统之间的协议转换问题，使得在系统更替、试运行时，能够实现双系统的同步运行，有效降低系统升级所带来的不确定性。

2　数据持久化

2.1　数据持久化框架

作为一个大型服务系统，在系统运行过程中，经常出现大量传感器同时发送数据的情况，在此影响下，后台数据库服务器在特定时刻或时段将会收到大规模并发数据。

在接收到数据之后，需要综合考虑两个问题。首先数据能够实时地在客户端进行显示；其次数据能够完整地进行持久化，为以后的历史数据反演（如轨迹回放）做准备。

这就带来一个矛盾，对于规模巨大的数据，如果每收到一个数据就进行处理的话，由于数据并发量太大，频繁地开、关后台数据库必然会使得数据库崩溃；与此同时，频繁的数据库开、关操作也将导致数据库的开、关时间要高于原始的数据处理时间。因此倘若接到一帧数据就进行数据库入库操作，不管从性能上还是技术上都不具备可操作性。但如果等收集到一定数据之后集中进行处理的话，又会面临一个问题，即数据不能够实时地在客户端进行显示。考虑上述问题，本文采用图 3 所示的框架实现数据的持久化。

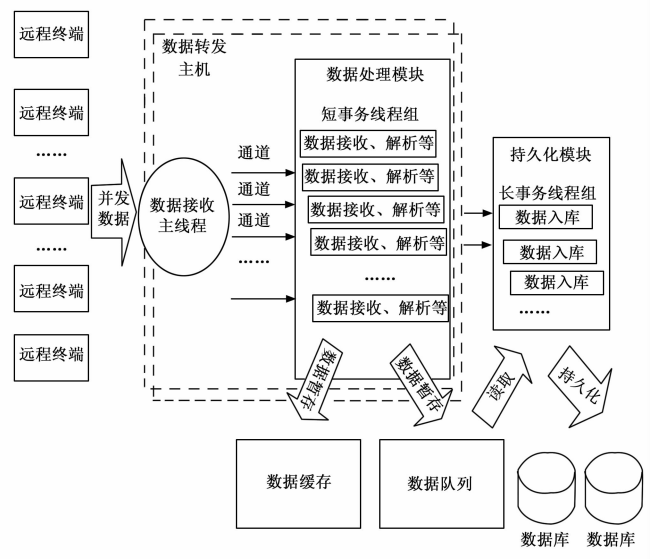

图 3 数据持久化处理框架

该模块同样采用支持 NIO 框架的 Netty 来进行实现。该框架由数据接收主线程、数据处理模块、数据缓存、数据队列及持久化模块等组成。在数据接入之后，首先由主线程进行接收，并分派给子线程。子线程中，短事务线程先进行接收。短事务线程所做的操作主要包括数据接收、协议解析等。如遇到需要处理的长事务，如文件读写、数据库读写等，则将其包装成一个“任务”，交给长事务线程进行调用。

2.2 延迟入库与数据缓存策略

要想使得数据库能够容纳大规模并发数据的入库，则必须采用集中入库的方式，即服务器再接收到一定的数据之后，统一入库。集中入库带来一个问题即数据延迟，考虑到该问题，本文采用的核心思路是数据缓存与延迟入库。该框架在收到远程终端发来的一条数据后，首先将其存储在数据缓存中，不直接进行入库操作。等待一定的时间后，再将之前存储在缓存中的数据进行批量入库，通过这种方式，来避免数据库的频繁断、连操作，从而提高平台的运行效率。对于缓存的容量以及延迟入库的时间，也必须要根据具体情况进行合理设置。一般情况下为 5 s 或 8 s，即每隔 5 s 或 8 s 将把缓存中的数据批量写入数据库。但是假如在一个时间点接收到大量数据，那么即使入库时间并未到达，由于数据缓存已经用完，也需要立刻对暂存在缓存中的数据进行批量入库操作。

在这种方式下，短事务线程在接收到数据之后，同时对数据缓存服务器与数据队列进行写入，并以相应的策略对缓存服务器进行维护。以保证延迟入库不对数据的实时性产生影响。

该框架在清空原有数据缓存空间以及数据批量入库的过程中，仍有可能会有大规模传感器数据接入。如果不进行相应处理，将会丢失大量数据。针对该问题，本文重新设置一个数据缓存来与现有的数据缓存进行配合。即系统中同时使用两个数据缓存——缓存 X 和缓存 Y。当缓存 X 容量已满，系统将暂时不再对缓存 X 进行写入，而是转而对缓存 Y 进行写入；相应地，当缓存 Y 容量已满，而缓存 X 被清空之后，系统将重新对缓存 X 进行写入。当系统批量入库时间到后，缓存中的数据被写入数据库，缓存也将被清空。在某些特殊的应用场景下，需要对某传感器进行单独跟踪，这就需要对原始入库的时间进行调整，加大入库频率（如设置 3 s 的入库间隔时间）。由于该要求比较特殊，不同于其他一般的入库信息，所以需要设置一个单独的线程来独立负责接收该车载传感器的数据，从而确保对该车载传感器的持续跟踪。

为了提高车辆管理系统的整体运行效率，在数据入库过程中，需要将接收数据和写数据库的功能分开。在本文所述框架中，若将数据接收和数据批量入库功能整合在同一个线程，不仅会使当前系统的运行效率大幅度下降，而且很可能会造成服务器的崩溃。所以本文设定接收数据的线程只负责数据接收，与此同时，单独启动一个线程，专门负责数据库的写入操作。这种操作模式不仅可以更好地提升整个系统的运转效率，还能在极大程度上加强系统的可维护性与稳定性。

3 实验结果与分析

3.1 实验思路

为了验证本文基于上述提出的一系列方法的有效性。为了保证数据的安全性，在新系统上线之后，旧系统将和新系统同步运行一段时间，等新系统完全稳定之后，再将旧系统停止。考虑到该需求，本系统采用数据转发的方式对新旧车辆管理系统进行衔接。在数据到达转发服务器后，一路数据由转发服务器转入旧系统，另一路数据则转入新系统。旧系统为如东市出租车管理信息系统，采用非部标 808 协议。新系统为江苏太平洋通讯科技有限公司开发的支持部标 808 协议的车辆通讯管理系统，如图 4 所示。

主要从两个方面来验证本文方法：1）数据转接后，新系统对协议转接与大规模并发车辆数据的支持。这就要求新系统能够顺利的实现数据转接，从旧系统协议转换到部标 808 协议，且支持 10 000 以上级别的大规模并发车辆同

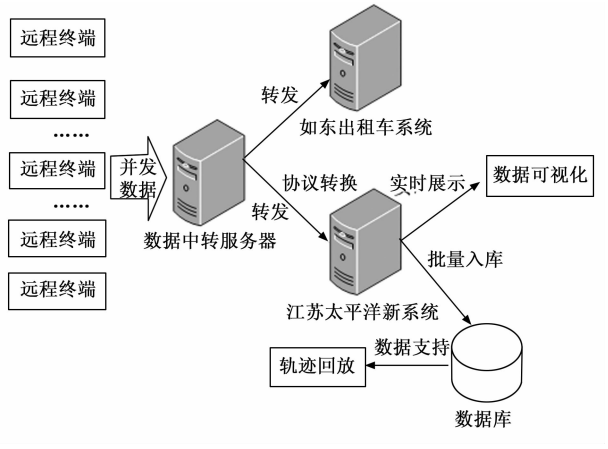

图 4　实验思路

时在线；2）新系统对数据批量入库的支持。这就要求新系统能够支持历史数据的存储和访问，支持类似历史轨迹回放等功能的使用。

3.2　实验步骤

实验步骤如下：第一步，将如东出租车系统数据接入数据转发服务器；第二步，将数据转发服务器所获取的数据划分为两组数据转出。一组为如东市出租车系统，另一组为江苏太平洋通讯科技有限公司的新系统。将接入新系统的数据做预处理，即数据协议转换，将旧系统的协议转化为部标 808 协议；第三步，将新系统采用数据持久化框架进行处理，实现大规模并发数据的批量入库与实时可视化。最后，访问历史数据库，实现车辆的历史轨迹回放功能，验证数据入库的有效性。

3.3　实验结果

经过上述步骤，本文实验结果如下：

1）新系统对协议转换及大规模并发数据的支持。图 5 展示了通过衔接系统中转平台并进行协议转换之后新系统的运行界面。该系统可以支持 50 000 级别的车辆同时在线，上述车辆每隔 5 s 向服务器发送 1 kB 大小的数据帧，该系统能够比较好地实现并发数据的接收，不出现异常情况。

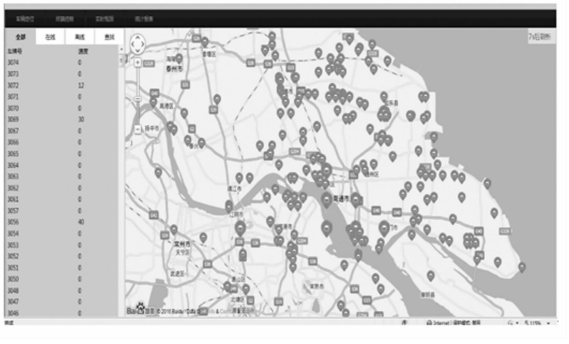

图 5　车辆监控系统界面图

2）该系统很好地处理了数据实时显示与数据入库的问题。在接收到一条数据之后，系统能够很好地对其进行响应，并在界面上进行显示。在积累一定的数据之后，数据会批量存入数据库。图 6 展示了轨迹回放功能，该功能需要基于历史数据对车辆的位置进行查询，并形成轨迹。该功能的正常运行表明了系统能够将历史数据完好的存入数据库，为后期的数据分析提供支持。

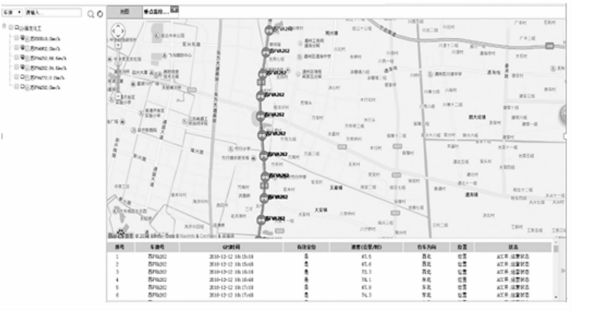

图 6　车辆历史轨迹回放

3.4　存在的问题

实验表明，新系统能支持 50 000 级别的大规模并发数据的接入，实时表达与批量入库。但依然存在一定的问题，主要体现在：一旦数据量突破 50 000 级别时，系统会出现 CPU 和内存使用率上升的问题，并导致服务器数据丢失。这是因为单台服务器的处理能力存在一定的上限，一旦达到这个上限，数据服务器将会成为整个系统的瓶颈。随着 CPU 和内存的占用率上升，系统接入大规模并发传感器数据的能力逐步减弱，在达到一个阈值之后，将有可能产生服务器整体崩溃的情况。

在今后，将考虑对系统进一步升级。以数据中转服务器为基础，实现多台服务器的负载均衡，进一步提高系统的吞吐量。

4　结束语

本文基于数据中转框架，对新旧车辆管理系统进行了衔接，并有效解决了衔接过程中的相关问题，并得出以下结论：

1）基于 Netty 架构构建的数据转发服务器，能够很好地实现大规模并发车辆传感器数据的转发。在转发服务器中进行协议转换，就可以有效地实现新旧系统的衔接。

2）基于延迟入库与数据缓存策略，本系统能够有效实现大规模并发车载传感数据批量入库与实时显示。

3）基于本文所提出的数据中转框架，可以有效地实现基于不同协议的车辆管理系统的衔接，实现新旧系统的无缝过渡。

参考文献：

[1] 杨党旗，崔玉萍，张　锐，等. 基于 GIS 的城市道路桥梁信息管理系统 [J]. 市政技术，2004，22 (5)：323－326.

[2] 王生昌，杨再研. 一个车辆管理计算机系统的设计与实现 [J]. 交通与计算机，2003，21 (2)：27－29.

[3] Loganathan G. GPS and GIS technology trends [J]. Electronics and Communication Engineering Journal，2002，14 (6)：292－294.

(下转第 94 页)