文章编号：1008-0775(2014)-07-09-04

基于Java的通用批处理作业系统的设计与实现

李代伟，李 藁
（成都信息工程学院，四川 成都 610225）

摘 要：为了解决当前业务数据处理中客户群庞大、业务数据大、人工操作繁杂、重复性高等问题，针对现有核心系统构架并非针对完全连线化的设计，提出了同一Job内批次作业平行处理的通用批处理作业系统框架模型，采用断点续接、参数化和模块化设计等实现了该系统。研究结果表明：通过调节参数，Batch循环流程控制，以满足气象、电信等不同业务的需要，系统能为其提供业务数据的快捷处理手段，提高了操作效率，减少了人工误操作率，达到了通用批处理作业的效果。

关键词：批处理作业；Batch循环流程控制；断点续接
中图分类号：TP311.1 文献标识码：A

Design and Implementation of Java-based Universal Batch Job System

LI Daiwei,LI Qu
(*Chengdu University of Information Technology*,*Chengdu* 610225,*China*)

Abstract: In order to solve the current business data processing,with a huge customer base and a mass of business data,complicated manual operation,higher repeatability problems.For existing core system architecture is not directed entirely connection design,the paper proposes a batch job system framework model,which has batch operations in the same general purpose parallel processing job,using breakpoint access mechanisms,parametric and modular design.The results show that: by adjusting parameter,batch loop flow control,to meet the needs of meteorology and telecommunications,etc.The system can provide means of fast processing business data,improve the operational efficiency,reduce labor misuse rate,reaching a common batch job effects.

Keywords:batch job;batch process control loop;resume broken process

1 引言(Introduction)

在当前计算机和网络高速发展的时代，现代企事业单位规模不断壮大，众多企事业都拥有数以万计的客户群，涉及庞大的业务数据，尤其是气象、电信、银行、保险等，人们开始充分利用现有资源进行信息的高度集成处理，使用批处理模式使整个流程作业变得简单化、规律化、集成化，逐渐脱离原先人工处理效率低下的局限性，从而大大增强了系统数据处理的能力。

传统的线程和资源锁并发编程，复杂、容易出错、无法横向扩展。目前，虽然Spring Batch是一款优秀的、开源的大数据量并行处理框架，通过它也可构建出轻量级的健壮的并行处理应用[1]；但却没有一个关于Java批处理架构的工业标准，商业化的批处理似乎处在一个严峻的状态：错误的架构风格和能力。尽管SOA日益增长，但仍需一种高强度的批处理架构来最有效地自动处理大容量的数据或事务却无需人工干预；批处理，作为绝大多数IT项目的组成部分，当前却处在一个没有商业或开源Java框架来为其提供健壮的企业解决方案的尴尬境地；在企业应用里，批处理通常用来处理每天数以亿计的事务处理，且这些处理任务非常苛刻。

因此，尽管缺乏批处理标准，但构建一个基于Java的通用批处理作业系统处理海量的业务数据，提高业务工作效率是非常有必要的，也具有很大推广应用价值。

2 系统总体架构设计(Overall system architecture design)

Batch处理系统是由服务器统一管理，所有业务逻辑都集中在此Batch处理系统中，各类用户负责数据的录入和查询，用户分布广泛，数据集中处理，因此在设计时充分考虑多种体系结构的优缺点，系统采用三层C/S体系结构进行架构设计。

模块化就是把一个复杂的系统分解为若干个规模较小、功能较简单的、相对独立、更易于建立和修改的模块，分别加以设计实现，各模块在一定关系的约束下共同构成一个统一的整体，完成系统的功能[2]。采用模块化设计思想，系统功能分为七大模块：用户交互界面、系统流程控制管理模块、系统配置文件管理模块、Batch流程处理模块、日志管理模块、数据库管理模块、文件控制模块。

由于关系数据库中包含多个数据表信息，含有数据定义、查询、更新、控制等。Oracle是关系数据库中使用最为广

泛的一种，采用标准SQL语言，数据类型支持大至4GB的二进制数据，为数据库的面向存储提供了强大的数据支持、安全措施、数据互操作性、动态的数据存储机制。考虑到Batch处理的客观条件，系统开发采用了Oracle。当然，此框架系统是一个系统运行框架，可根据需要简单、方便地设置数据库类型，比如MySQL、MS SQL等。系统总体架构设计如图1所示。

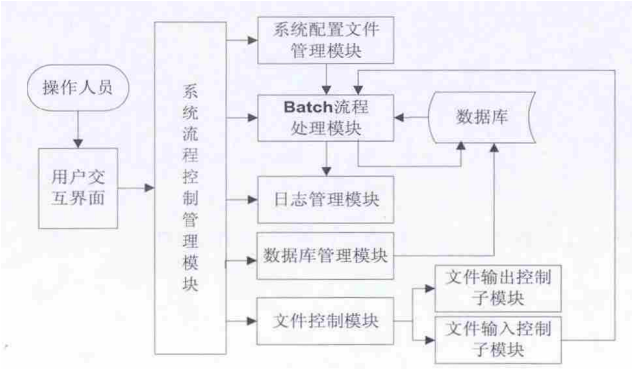

图1 系统总体架构

Fig.1 Overall system architecture

3 核心模块软件设计(Core Module Software Design)

根据面向对象设计和模块化的设计思想，将系统划分的七大模块中：用户交互界面模块依赖于系统流程控制管理模块，为用户提供可视化交互界面；系统配置文件管理模块、Batch流程处理模块、日志管理模块、数据库管理模块、文件控制模块作为独立模块存在，相互之间不存在直接依赖关系，其业务逻辑关系由系统流程控制管理模块负责维护。其中，系统配置文件管理模块、Batch流程处理模块、文件控制模块(输入和输出处理)为系统三大核心模块。

3.1 系统配置文件管理模块

通过用户交互界面，选择需要运行的Job，系统驱动模块将读取系统输入模块中所有的配置文件，并传入系统执行模块，在系统执行过程中会产生相应的系统活动日志和系统控制日志等日志文件，并且同时将其处理结果输出到系统定义的输出文件或数据库。系统配置文件管理模块结构如图2所示。

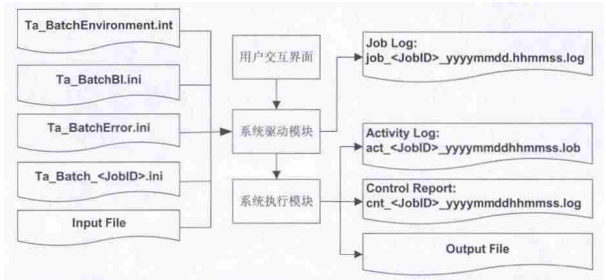

图2 系统配置文件管理模块结构图

Fig.2 System configuration file management module structure diagram

3.2 Batch流程处理模块

3.2.1 单次循环Batch流程处理框架模型

当所有系统配置文件都准备好并运行一个指定的Batch处理程序后，系统框架将按照如图3所示的流程加载系统配置文件。通过Loadini()读取系统配置参数，然后传入CheckFiles()检查所读取参数的合法性，接着CheckOutFiles()检查系统输出文件类型及其合法性，当检查都合法后，调用CheckDB()连接数据库，之后，SetupCheckPoing()检查系统断点续接的属性，然后系统正式进入系统执行模块Start()，当所有执行完成后，通过WriteTraier()输入系统完成信息，最后调用Shutdown()结束整个流程。

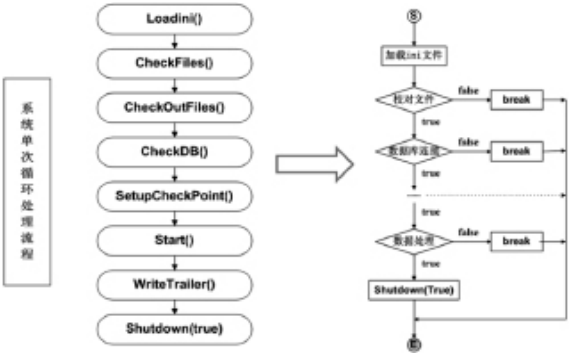

图3 单次循环Batch流程处理框架

Fig.3 A single cycle process of batch processing framework

3.2.2 循环Batch流程处理框架模型

通用批处理作业系统是对用户数据进行大批量的一次性处理，以完成其人工操作的繁琐性和复杂性，并且提高其人工操作的低效率。其主要功能是对程序循环流程的控制和对系统断点续接的控制机制，其中核心功能是对系统循环流程的控制和处理。系统循环Batch流程处理框架模型如图4所示。

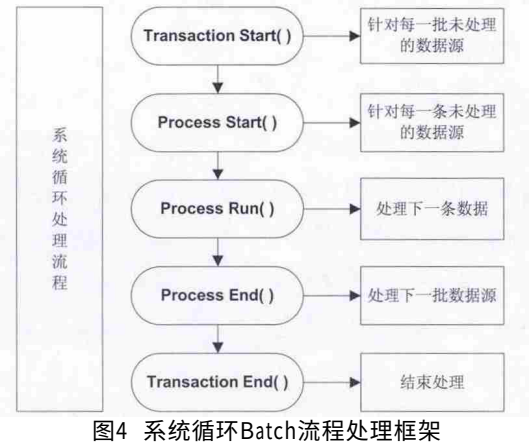

图4 系统循环Batch流程处理框架

Fig.4 System circulation process of batch processing framework

为提高批处理效率，通过多线程并行执行多个相互独

立的Step[3]，设计了同一Job内批次作业平行处理模型，如图5所示。

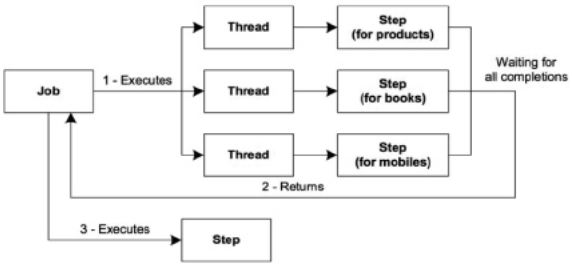

图5 同一Job内批次作业平行处理模型

Fig.5 The batch operation parallel processing model within the same Job

3.3 文件控制模块

3.3.1 文件输入处理

文件输入处理均为ini文件，包括：系统环境控制文件、系统配置文件、单个Batch处理配置文件。

(1)系统环境控制文件(Ta_BatchEnvironment.ini)

系统环境控制文件是系统的核心文件，主要是连接数据库的内容，其格式如下：

OPOMDB=[UID=orscm],[PWD=orscmdd],[DB=thin:@10.5.7.108:1521:ORUT23]

OPOMDB=[UID=root],[PWD=],[DB=localhost:3306/ComBatchJob]

其中：数据库名称OPOMDB必须与单个Batch处理输入文件中数据库名称一致。

(2)系统配置文件(Ta_BatchBI.ini)

系统配置文件包括系统的输入输出文件的物理路径，其主要内容如下所示：

系统输出数据文件夹，若系统输出为文件类型输出，则输出在此文件夹。

SYS_OUTPUT_DIR=D:\\CommBatchJob\\output

系统输入数据文件夹，若系统输入为文件输入，则输入文件存放在此文件夹。

SYS_INPUT_DIR=D:\\CommBatchJob\\input

系统参数控制文件夹：根据输入配置文件中参数来判断系统下次启动后数据的处理起始位置。

SYS_RESTART_DIR=D:\\CommBatchJob\\control\\restart

系统错误文件夹：程序中产生的错误信息存放在此文件夹。

SYS_ERROR_DIR=D:\\CommBatchJob\\error

系统数据统计结果集：在程序最后统计此次batch处理共处理了多少数据。

SYS_ACTREP_CENT_DIR=D:\\CommBatchJob\\cnt_rpt

(3)单个Batch处理配置文件(Ta_Batch_<JobID>.ini)

单个Batch处理配置文件是核心配置文件，程序所有控制参数都在此配置文件中。例如：

文件输入控制属性。若输入数据采用文件输入，则此处为文件的路径和名称；否则为空。

SYS_IN_DIR="[a=${SYS_INPUT_DIR}/OPMCAR101D/*]"

SYS_IN_FILE=""

SYS_IN_ARBDB=""

文件输出控制属性。若输出方式采用文件输出，则此处为文件的路径和名称；否则为空。

SYS_OUT_ARBDB=""

SYS_OUT_ORADB=""

SYS_DEST_DIR="[a=${SYS_OUTPUT_DIR}/OPMIOP101D{o}]"

单次数据库提交处理数据最大量设置。Batch处理会涉及大量的数据处理，若内存中存放太多没有提交的数据，则数据库会出现内存溢出现象，这里设置提交的最大量，例如设为100，则每处理完成100条数据，则向数据库commit一次。

SYS_COMMIT_FREQ="100"

数据库提交超时时间设置，单位：秒。

SYS_COMMIT_TIMEOUT="10"

重新运行处理标志设置。这是一个相当重要的程序控制属性，缺省为N。若设置为Y，则程序在下次重新运行程序时，会从上次中断处接着处理。

SYS_RESTART_FLAG="N"

3.3.2 文件输出处理

(1)处理结果输出

用户需要得到的处理输出包括文件输出或存入数据库。如果系统输入方式为文件输出，则输出文件的命名方式为：当前运行程序的Job Id加上输入配置的seq文件。在通常情况下，需要得到的数据文件输出为文本文件，但在某些特殊需求下，需把文本文件手工地提取为excel文件。为此，系统提供了相应的接口供用户使用，以便用户直接得到处理完好的excel文档。

(2)系统日志

系统日志输出部分主要包括：系统控制log文件、系统错误或异常log文件等。

(3)程序参数设定

程序在此框架下运行，需设置两个重要的系统参数：Program参数和VM参数。若程序需要调用服务器端的EJB，

则需在VM参数中设置服务器端参数，如IP地址、端口等。

4 关键技术及其实现(Key technology and its implementation)

4.1 支持多类型数据库操作

在执行数据库操作时，在系统ini配置文件的application query中配置数据库类型，这样可以在不修改代码情况下，直接配置设置文件，提高系统的灵活性。实现关键技术代码如下：

```
paraSqlOpGetSI = "SELECT A.ACCT_ID,B.MASTER_ACCT_ID FROM OP_SVC_INST A, OP_ACCT B WHERE A.ACCT_ID = B.ACCT_ID AND A.SVC_INST_ID = ?"
opGetSI = opomCon.prepareStatement(apc.getAppsParam(paraSqlOpGetSI));
PreparedStatement sqlSelectnwinst = opomCon.prepareStatement(opGetSI);
sqlSelectnwinst.setString(1,SVC_INST_ID);
ResultSet rs = sqlSelectnwinst.executeQuery( );
```

4.2 Batch循环流程控制

系统启动后是否接着上次未处理完处继续处理还是从头开始处理，并循环处理所有数据，直至全部处理完成，这就需要对批处理作业进行循环流程控制。实现关键技术代码如下：

```
if(restarted && !restarted_ok){
dbRow = chkpnt.getCurrentDbRow( );
if(dbRow == null) dbRow = chkpnt.getNextDbRow( );
restarted_ok = true;
} else  dbRow = chkpnt.getNextDbRow( );
if (dbRow  ! = null) {
apps.startSource( );
while((rsRow = dbRow.getNextRSRow()) != null){
if(dbRow.keyBased)  apps.process(rsRow);
else  apps.process(map);
dbRow.cnt++;
chkpnt.checkPoint( );
}
apps.endSource( );
}
```

5 实验结果分析(Experimental results)

利用该系统先后对某气象局、电信营业厅的一些历史业务数据进行了实验分析，并与这些历史数据以前的实际处理情况进行对比，结果如表1所示。

表1 实验分析结果数据

Tab.1 Experimental data analysis results

数据所属行业	数据处理方式	数据量(条)	处理时间(秒)	断点续接能力(%)	异常(错误)中断数据(条)
气象	通用批处理作业系统	251345	683	99	28
	原有处理系统	251345	915	90	37
电信	通用批处理作业系统	85420	296	98	12
	原有处理系统	85420	410	92	19

从表1可以看出，该系统可明显缩短业务数据处理时间，处理速度大概提高了1.34倍，断点续接能力也有所提高，减少了因异常(或错误)而中断数据处理的几率。从而表明：该系统在批处理方面要比原有系统更准确，也提高了效率。

6 结论(Conclusion)

Batch处理系统由服务器进行统一管理，集中了所有业务逻辑处理，各类用户负责数据的录入和查询，用户分布广泛，数据集中处理，因此在设计时选择了三层模式进行设计实现。和类似系统比较，本系统具有如下优势：第一，具有断点续接机制：当批处理遇到一些可恢复性错误时，比如输入数据不平等，批处理中断运行，并给出详细的错误提示和运行日志，便于维护人员查找，同时提供断点续接功能，使批处理能在断点处继续运行。第二，具备完善的控制机制：应用平台在各项操作之间提供顺序控制，能有效避免人工误操作带来的影响。通过调节参数，对系统运行参数进行设置，以满足不同业务的需要。第三，采用参数化和模块化的设计思想：系统的各种控制都以参数形式实现，通过管理和配置可以满足不同业务需要；系统分成一些基本处理单元，每一处理单元形成一个模块，这种模块化的设计使得应用系统机构清晰、维护方便、具有良好的可扩展性。

因此，本文阐述的通用批处理作业系统在快速处理海量的业务数据、提高业务工作效率方面具有较广意义的普适性和通用性。

参考文献(References)

[1] 池建强.基于Spring Batch的大数据量并行处理[EB/OL].http://wenku.it168.com/d_000687949.shtml,2012-12-13.

[2] 张海藩.软件工程导论(第6版)[M].北京:清华大学出版社,2013.

[3] 陈亚.基于中间件技术的数据批处理系统的设计与实现[J].电脑知识与技术,2008,4(34):1567-1568;1570.

作者简介:

李代伟(1976－)，男，硕士，讲师.研究领域：软件工程，计算机应用技术，云计算与大数据处理.

李　藻(1971－)，女，硕士，副教授.研究领域：软件工程，计算机应用技术.