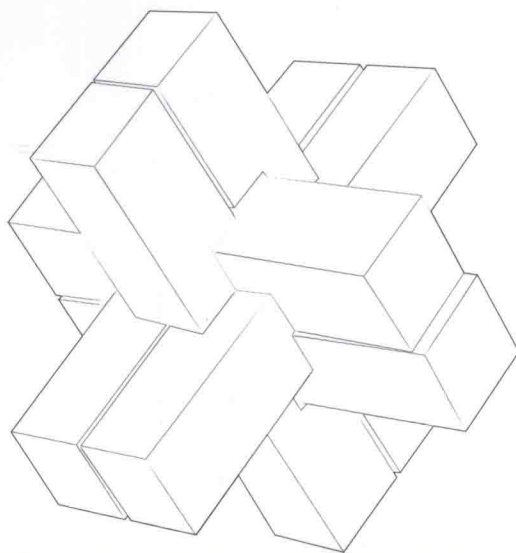


大数据时代数据批处理利器
首度原创解析Spring Batch框架

Broadview[®]
www.broadview.com.cn



Spring Batch 批处理框架

全 列
深入的架构设计 底层的源码剖析
详细解读设计原理

刘相 编著



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
<http://www.phei.com.cn>

Spring Batch

批处理框架



刘相 编著

电子工业出版社

Publishing House of Electronics Industry

北京•BEIJING

内 容 简 介

本书全面、系统地介绍了批处理框架 Spring Batch, 通过详尽的实战示例向读者展示了 Spring Batch 框架对大数据批处理的基本开发能力, 并对框架的架构设计、源码做了特定的剖析; 在帮助读者掌握 Spring Batch 框架基本功能、高级功能的同时, 深入剖析了 Spring Batch 框架的设计原理, 帮助读者可以游刃有余地掌握 Spring Batch 框架。

本书分为入门篇、基本篇和高级篇三部分。入门篇介绍了批处理、Spring Batch 的基本特性和新特性, 快速入门的 Hello World 等内容引领读者入门, 从而进入数据批处理的世界。基本篇重点讲述了数据批处理的核心概念、典型的作业配置、作业步配置, 以及 Spring Batch 框架中经典的三步走策略: 数据读、数据处理和数据写, 详尽地介绍了如何对 CVS 格式文件、JSON 格式文件、XML 文件、数据库和 JMS 消息队列中的数据进行读操作、处理和写操作, 对于数据库的操作详细介绍了使用 JDBC、Hibernate、存储过程、JPA、Ibatis 等处理。高级篇提供了高性能、高可靠性、并行处理的能力, 分别向读者展示了如何实现作业流的控制, 包括顺序流、条件流、并行流, 如何实现健壮的作业, 包括跳过、重试和重启等, 如何实现扩展作业及并行作业, 包括多线程作业、并行作业、远程作业和分区作业等, 从而实现分布式、高性能、高扩展性的数据批处理作业。

本书适合需要具体使用批处理作业、大数据处理的开发人员, 设计人员和架构师, 对于企业中存在大量作业的运维人员亦有一定的参考价值。

未经许可, 不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有, 侵权必究。

图书在版编目(CIP)数据

Spring Batch 批处理框架 / 刘相编著. —北京: 电子工业出版社, 2015.2
ISBN 978-7-121-25241-9

I. ①S… II. ①刘… III. ①数据处理 IV. ①TP274

中国版本图书馆 CIP 数据核字 (2014) 第 302744 号

策划编辑: 孙学瑛

责任编辑: 徐津平

特约编辑: 顾慧芳

印 刷: 三河市双峰印刷装订有限公司

装 订: 三河市双峰印刷装订有限公司

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本: 787×980 1/16 印张: 25.25 字数: 582 千字

版 次: 2015 年 2 月第 1 版

印 次: 2015 年 2 月第 1 次印刷

印 数: 3000 册 定价: 69.00 元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zltz@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。

批处理编程之美

这是一部论述批处理程序编程的书。在信息系统中，联机和批处理是计算机处理的两种基本模式，前者快速响应、超时中断、密集并发，后者处理时间长、异常需要支持重做、通常以顺序执行。早期的计算机系统多采用批处理的处理模式，客户机/服务器架构的产生让联机模式越来越多地被采用，但批处理模式一直在信息系统中起着重要作用，随着 OLAP、大数据等新技术的应用，批处理的处理模式又成为热点，例如在传统银行 IT 系统中每日动辄运行上万个批处理作业，在互联网应用中，腾讯、阿里每日的批处理作业可达百万量级的水平。

编程之美，美在架构，架构之美，美在抽象，只有具备充分理解复杂业务场景的格局，才能进行将复杂问题做简单化的抽象。同联机模式汗牛充栋的著作、框架相比，批处理模式的抽象不多，著名的 MapReduce 就是其中之一，MapReduce 将大批数据的处理过程进行了抽象，而 Spring Batch 则是对编写批处理程序本身的特性进行了抽象。通过将批处理程序分解为 Job 和 Job Step 两个部分，将处理环节定义为数据读、数据写和数据处理三个步骤，将异常处理机制归结为跳过、重试、重启三种类型，将作业方式区分为多线程、并行、远程、分区四大特征，正所谓增一分则肥，减一分则瘦。类似之美，Spring 系列项目还有很多，例如 Spring Framework 对依赖注入的抽象，Spring integration 中利用消息、队列、处理器三个概念的组合对集成模式的抽象，都让我叹为观止。

当相相^①递给我他的新作时，我吃了一惊，惊在他“悄无声息”地完成了这样一个大部头作品，迫不及待地有一种先睹为快的冲动。我发现，书中通过对 Spring Batch 本身的论述，让我体会到了 Spring Batch 的精髓，也更加深刻地理解了批处理编程模式，还看到了相相对信息系统中如何使用这一框架的见解，毕竟信息系统中的批处理程序，不仅仅是一个框架，还需要包含更多的管理、运维方面的流程、制度与平台。近年来，我看到很多企业都在构建集中的批处理平台，管理大量出现的批处理作业，也期待相相能有更多这方面的分享。

普元 CTO 焦烈焱

2015 年 1 月

① 相相为本书作者刘相的简称。

前言

信息时代，数据是现代企业最宝贵的核心资产，是企业运用科学管理、决策分析的基础。截至目前，国内大多数企业已经完成了 OLTP（联机事务处理）的业务系统和办公自动化系统，用来记录事务处理的各种相关数据。据统计，企业的数据每年都在成倍增长，企业如能充分利用这些数据会带来巨大的商机。但目前企业通常所关注的数据仅占总数据量的 5% 左右，企业没有最大化地利用已经存在的数据资源，导致浪费了更多的时间和资金，同时也失去了制定关键商业决策的最佳契机。于是，企业如何通过各种技术手段，并把数据转换为信息、知识和商机已经成为提高其核心竞争力的主要手段。而数据批处理则是达成上述目标的一个主要技术手段，通过数据批量处理，可以完成数据的加载、抽取、转换、清洗等功能，进而支撑企业的各种数据分析。

2012 年底，笔者有幸接手某银行批处理项目，首次接触 Spring Batch 批处理框架，深入学习 Spring Batch 框架后发现：Spring Batch 框架的架构设计清晰和优雅，其功能完备，具有无所不在的扩展能力、丰富的业务组件，并且采用业务与技术分离的设计思想。笔者通过近半年多的学习，通读了官网提供的所有文档，深入学习了原版书籍 *Spring Batch In Action*。在大数据时代，批处理框架在金融、电信、大型制造业等应用非常广泛。在学习的过程中，笔者了解到同事、网络上有不少朋友苦于没有中文版的 Spring Batch 框架介绍资料，于是萌生出写一本 Spring Batch 框架中文版介绍图书的想法，希望能够帮助国内的读者快速地掌握该框架。

自 2013 年年中起笔者开始构思本书的大纲，在繁忙工作之余坚持每周完成部分章节内容，前后历时 1 年多时间完成书稿编写工作。在此感谢妻子 Phyllis 给予我充足的时间、感谢可爱女儿 Rachel 给我带来的巨大欢乐动力。本书主要的目的是全面、系统地介绍批处理框架 Spring Batch，通过详尽的实战示例向读者展示了 Spring Batch 框架的基本开发能力，并对框架的架构设计、源码做了特定的剖析；在帮助读者掌握该框架基本功能、高级功能使用的同时，深入剖析 Spring Batch 框架的设计原理，帮助读者可以游刃有余地掌握 Spring Batch 框架。

本书分为入门篇、基本篇、高级篇三个部分，从基本的入门篇讲起，通过介绍批处理、Spring Batch 基本特性、新特性、快速入门的 Hello World 等内容引领读者入门，进入批处理的世界。之后的基本篇，重点讲述了批处理的核心概念、典型的作业配置、作业步配置以及 Spring Batch 框架中经典的三步走策略：数据读、数据处理、数据写，详细介绍了如何对分隔符类型文件、定长类型文件、JSON 格式文件、复杂类型格式文件、XML 文件、数据库、JMS

消息队列中的数据进行读、处理、写操作,对于数据库的操作详细介绍了使用 JDBC、Hibernate、存储过程、JPA, Ibatis 等处理。为了能够让读者更深入地了解 Spring Batch 框架,高级篇提供了高性能、高可靠性、并行处理的能力,分别向读者展示如何实现作业流的控制包括顺序流、条件流、并行流,如何实现健壮的作业包括跳过、重试、重启等,如何实现扩展作业及并行作业包括多线程作业、并行作业、远程作业、分区作业等。

本书适合需要具体使用批处理框架 Spring Batch 的开发人员、设计人员、架构师,对于企业中存在大量作业的运维人员亦有一定的参考价值。

编 者

2014.12.23 于上海浦东

目 录

第 1 篇 入门篇

第 1 章 Spring Batch 简介..... 2	1.5 Spring Batch 2.2 新特性..... 13
1.1 什么是批处理..... 2	1.5.1 Spring Data 集成..... 13
1.2 Spring Batch..... 3	1.5.2 支持 Java 配置..... 13
1.2.1 典型场景..... 3	1.5.3 Spring Retry..... 14
1.2.2 Spring Batch 架构..... 4	1.5.4 Job Parameters..... 14
1.3 Spring Batch 优势..... 4	1.6 开发环境搭建..... 15
1.3.1 丰富的开箱即用组件..... 5	第 2 章 Spring Batch 之 Hello World..... 16
1.3.2 面向 Chunk 的处理..... 5	2.1 场景说明..... 16
1.3.3 事务管理能力..... 5	2.2 项目准备..... 16
1.3.4 元数据管理..... 5	2.2.1 项目结构..... 16
1.3.5 易监控的批处理应用..... 5	2.2.2 准备对账单文件..... 17
1.3.6 丰富的流程定义..... 5	2.2.3 定义领域对象..... 18
1.3.7 健壮的批处理应用..... 6	2.3 定义 job 基础设施..... 18
1.3.8 易扩展的批处理应用..... 6	2.4 定义对账 Job..... 19
1.3.9 复用企业现有 IT 资产..... 6	2.4.1 配置 ItemReader..... 19
1.4 Spring Batch 2.0 新特性..... 6	2.4.2 配置 ItemProcessor..... 21
1.4.1 支持 Java 5..... 7	2.4.3 配置 ItemWriter..... 22
1.4.2 支持非顺序的 Step..... 7	2.5 执行 Job..... 23
1.4.3 面向 Chunk 处理..... 7	2.5.1 Java 调用..... 23
1.4.4 元数据访问..... 11	2.5.2 JUnit 单元测试..... 24
1.4.5 扩展性..... 11	2.6 概念预览..... 26
1.4.6 可配置性..... 12	

第 2 篇 基本篇

第 3 章 Spring Batch 基本概念..... 28	3.2.3 Job Execution..... 34
3.1 命名空间..... 29	3.3 Step..... 35
3.2 Job..... 30	3.3.1 Step Execution..... 37
3.2.1 Job Instance..... 31	3.4 Execution Context..... 38
3.2.2 Job Parameters..... 33	3.5 Job Repository..... 39
	3.5.1 Job Repository Schema..... 39

3.5.2 配置 Memory Job Repository	40	5.3.5 读、处理事务	110
3.5.3 配置 DB Job Repository	41	5.4 拦截器	112
3.5.4 数据库 Schema	42	5.4.1 ChunkListener	115
3.6 Job Launcher	48	5.4.2 ItemReadListener	116
3.7 ItemReader	49	5.4.3 ItemProcessListener	116
3.8 ItemProcessor	50	5.4.4 ItemWriteListener	117
3.9 ItemWriter	50	5.4.5 SkipListener	117
第 4 章 配置作业 Job	52	5.4.6 RetryListener	118
4.1 基本配置	52	第 6 章 读数据 ItemReader	120
4.1.1 重启 Job	54	6.1 ItemReader	120
4.1.2 Job 拦截器	55	6.1.1 ItemReader	120
4.1.3 Job Parameters 校验	58	6.1.2 ItemStream	121
4.1.4 Job 抽象与继承	59	6.1.3 系统读组件	122
4.2 高级特性	61	6.2 Flat 格式文件	122
4.2.1 Step Scope	61	6.2.1 Flat 文件格式	123
4.2.2 属性 Late Binding	62	6.2.2 FlatFileItemReader	125
4.3 运行 Job	63	6.2.3 RecordSeparatorPolicy	129
4.3.1 调度作业	65	6.2.4 LineMapper	130
4.3.2 命令行执行	68	6.2.5 DefaultLineMapper	131
4.3.3 与定时任务集成	71	6.2.6 LineCallbackHandler	138
4.3.4 与 Web 应用集成	73	6.2.7 读分隔符文件	139
4.3.5 停止 Job	77	6.2.8 读定长文件	141
第 5 章 配置作业步 Step	85	6.2.9 读 JSON 文件	143
5.1 配置 Step	86	6.2.10 读记录跨多行文件	145
5.1.1 Step 抽象与继承	87	6.2.11 读混合记录文件	147
5.1.2 Step 执行拦截器	89	6.3 XML 格式文件	150
5.2 配置 Tasklet	92	6.3.1 XML 解析	150
5.2.1 重启 Step	93	6.3.2 Spring OXM	151
5.2.2 事务	94	6.3.3 StaxEventItemReader	153
5.2.3 事务回滚	96	6.4 读多文件	156
5.2.4 多线程 Step	97	6.5 读数据库	159
5.2.5 自定义 Tasklet	97	6.5.1 JdbcCursorItemReader	160
5.3 配置 Chunk	99	6.5.2 HibernateCursorItemReader	167
5.3.1 提交间隔	102	6.5.3 StoredProcedureItemReader	171
5.3.2 异常跳过	103	6.5.4 JdbcPagingItemReader	174
5.3.3 Step 重试	105	6.5.5 HibernatePagingItemReader	179
5.3.4 Chunk 完成策略	107		

6.5.6	JpaPagingItemReader	183	7.8	Item 路由 Writer	254
6.5.7	IbatisPagingItemReader	186	7.9	发送邮件	258
6.6	读 JMS 队列	190	7.9.1	SimpleMailMessageItem Writer	258
6.6.1	JmsItemReader	190	7.10	服务复用	262
6.7	服务复用	194	7.10.1	ItemWriterAdapter	262
6.8	自定义 ItemReader	197	7.10.2	PropertyExtracting DelegatingItemWriter	264
6.8.1	不可重启 ItemReader	197	7.11	自定义 ItemWrite	267
6.8.2	可重启 ItemReader	199	7.11.1	不可重启 ItemWriter	267
6.9	拦截器	202	7.11.2	可重启 ItemWriter	268
6.9.1	拦截器接口	202	7.12	拦截器	271
6.9.2	拦截器异常	203	7.12.1	拦截器接口	271
6.9.3	执行顺序	204	7.12.2	拦截器异常	273
6.9.4	Annotation	204	7.12.3	执行顺序	274
6.9.5	属性 Merge	205	7.12.4	Annotation	274
第 7 章	写数据 ItemWriter	207	7.12.5	属性 Merge	275
7.1	ItemWrite	207	第 8 章	处理数据 ItemProcessor	277
7.1.1	ItemWriter	208	8.1	ItemProcessor	277
7.1.2	ItemStream	208	8.1.1	ItemProcessor	277
7.1.3	系统写组件	209	8.1.2	系统处理组件	278
7.2	Flat 格式文件	210	8.2	数据转换	279
7.2.1	FlatFileItemWriter	210	8.2.1	部分数据转换	279
7.2.2	LineAggregator	214	8.2.2	数据类型转换	281
7.2.3	FieldExtractor	217	8.3	数据过滤	282
7.2.4	回调操作	219	8.3.1	数据 Filter	282
7.3	XML 格式文件	222	8.3.2	数据过滤统计	283
7.3.1	StaxEventItemWriter	222	8.4	数据校验	285
7.3.2	回调操作	226	8.4.1	Validator	285
7.4	写多文件	230	8.4.2	ValidatingItemProcessor	286
7.4.1	MultiResourceItemWriter	230	8.5	组合处理器	288
7.4.2	扩展 MultiResourceItem Writer	233	8.6	服务复用	291
7.5	写数据库	234	8.6.1	ItemProcessorAdapter	291
7.5.1	JdbcBatchItemWriter	235	8.7	拦截器	293
7.5.2	HibernateItemWriter	239	8.7.1	拦截器接口	293
7.5.3	IbatisBatchItemWriter	242	8.7.2	拦截器异常	295
7.5.4	JpaItemWriter	245	8.7.3	执行顺序	295
7.6	写 JMS 队列	248	8.7.4	Annotation	296
7.6.1	JmsItemWriter	248	8.7.5	属性 Merge	297
7.7	组合写	252			

第3篇 高级篇

第9章 作业流 Step Flow.....	300	10.2.3 重试拦截器	343
9.1 顺序 Flow.....	300	10.2.4 重试模板	345
9.2 条件 Flow.....	302	10.3 重启 Restart	353
9.2.1 next	303	10.3.1 重启 Job.....	353
9.2.2 ExitStatus VS BatchStatus.....	306	10.3.2 启动次数限制	355
9.2.3 decision 条件	308	10.3.3 重启已完成的任务	355
9.3 并行 Flow.....	311	第11章 扩展 Job、并行处理.....	357
9.4 外部 Flow 定义	314	11.1 可扩展性	357
9.4.1 Flow	314	11.2 多线程 Step.....	358
9.4.2 FlowStep	317	11.2.1 配置多线程 Step	359
9.4.3 JobStep	319	11.2.2 线程安全性	360
9.5 Step 数据共享	321	11.2.3 线程安全 Step	361
9.6 终止 Job	323	11.2.4 可重启的线程安全 Step.....	363
9.6.1 end	324	11.3 并行 Step.....	365
9.6.2 stop	326	11.4 远程 Step.....	366
9.6.3 fail	327	11.4.1 远程 Step 框架	366
第10章 健壮 Job	330	11.4.2 基于 SI 实现远程 Step.....	368
10.1 跳过 Skip.....	331	11.5 分区 Step.....	373
10.1.1 配置 Skip	331	11.5.1 关键接口	374
10.1.2 跳过策略 SkipPolicy	333	11.5.2 基本配置	376
10.1.3 跳过拦截器	335	11.5.3 文件分区	378
10.2 重试 Retry	338	11.5.4 数据库分区	382
10.2.1 配置 Retry	339	11.5.5 远程分区 Step	387
10.2.2 重试策略 RetryPolicy	341	后记	392

第 1 篇 入门篇

本篇从基本的入门讲起，通过介绍批处理、Spring Batch 基本特性、新特性，快速入门的 Hello World 等内容引领读者入门，进入批处理的世界。本篇包含两个章节。

第 1 章：向读者介绍什么是批处理，Spring Batch 框架适用的业务场景、技术场景，Spring Batch 的核心三层架构、Spring Batch 2.0、2.2 新特性，最后带领读者一起搭建 Spring Batch 的开发环境。

第 2 章：通过经典的 Hello World 示例向读者全面展示了 Spring Batch 的入门示例。

Spring Batch 简介

1.1 什么是批处理

现代互联网企业、金融业、电信业甚至传统行业通过 OLTP（联机事务处理）的业务系统积累了海量企业数据，需要企业应用能够在关键任务中进行批量处理来操作业务逻辑。通常情况下，此类业务并不需要人工参与就能够自动高效地进行复杂数据处理与分析。例如，定期对大批量数据进行业务处理（如银行对账和利率调整、或者跨系统的数据同步），或者是把从内部和外部系统中获取到的数据进行处理后集成到其他系统中去，这类工作被称之为“批处理”。

“批处理”工作在面对复杂的业务以及海量的数据处理时，无需人工干预，仅需定期读入批量数据，然后完成相应业务处理并进行归档操作。

典型的批处理应用有如下几个特点：

- （1）自动执行，根据系统设定的工作步骤自动完成；
- （2）数据量大，少则百万，多则千万甚至上亿；
- （3）定时执行，如每天执行、每周或每月执行。

从上面的描述中可以看出，批处理的整个流程可以明显地分为 3 个阶段：

- （1）读数据，数据可能来自文件、数据库或消息队列等；
- （2）处理数据，处理读取的数据并形成输出结果，如银行对账系统的资金对账处理；
- （3）写数据，将输出结果写入文件、数据库或消息队列等。

一个典型的批处理应用场景：系统 A 从数据库获取数据，经过业务处理后，导出系统 B 需要的数据到文件中，系统 B 读取该文件，经过业务处理后，最后存放在数据库中。通常情况下该动作在每天夜间 12:00~2:00 之间进行，此时对系统的性能影响最小。图 1-1 给出了典型批处理应用的场景。

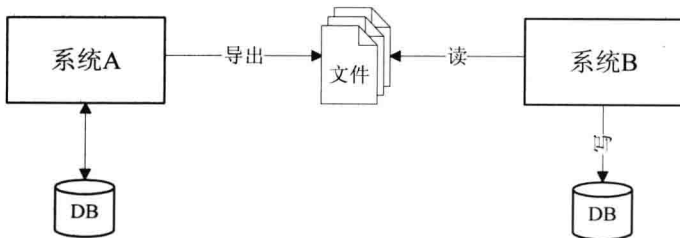


图 1-1 典型批处理应用场景

1.2 Spring Batch

Spring Batch 是一个轻量级的、完善的批处理框架，旨在帮助企业建立健壮、高效的批处理应用。Spring Batch 是 Spring 的一个子项目，使用 Java 语言并基于 Spring 框架为基础开发，使得已经使用 Spring 框架的开发者或者企业更容易访问和利用企业原有服务。

Spring Batch 提供了大量可重用的组件，包括日志、追踪、事务、任务作业统计、任务重启、跳过、重复、资源管理。对于大数据量和高性能的批处理任务，Spring Batch 同样提供了高级功能和特性来支持，比如分区功能、远程功能。总之，通过 Spring Batch 能够支持简单的、复杂的和大数据量的批处理作业。

Spring Batch 是一个批处理应用框架，不是调度框架，但需要和调度框架合作来构建完成批处理任务。它只关注批处理任务相关的问题，如事务、并发、监控、执行等，并不提供相应的调度功能。如果需要使用调度框架，在商业软件和开源软件中已经有很多优秀的企业级调度框架（如 Quartz、Tivoli、Control-M、Cron 等）可以使用。

1.2.1 典型场景

典型的批处理应用通常从数据库、文件或队列中读取数据，之后使用一些方法处理数据（抽取、分析、处理、过滤等），最终使用修改过的格式将数据写回目标系统。通常在一个无需用户交互的离线环境下，Spring Batch 能够自动进行基本的批处理迭代，也能够为一个数据集提供事务保证。批处理任务是一个大多数 IT 项目的组成部分，而 Spring Source 是唯一能够提供健壮的、企业级扩展的开源批处理框架。

Spring Batch 批处理框架支撑的业务场景：

- (1) 定期提交批处理任务；
- (2) 并行批处理，即并行处理任务；
- (3) 企业消息驱动处理；
- (4) 大规模的并行处理；
- (5) 手动或定时重启；
- (6) 按顺序处理依赖的任务（可扩展为工作流驱动的批处理）；
- (7) 部分处理，如在回滚时忽略记录；
- (8) 完整的批处理事务。

Spring Batch 批处理框架支撑的技术目标：

- (1) 利用 Spring 编程模型，使程序员专注于业务处理，让 Spring 框架管理流程；
- (2) 明确分离批处理的执行环境 and 应用；
- (3) 将通用核心的服务以接口形式提供；
- (4) 提供“开箱即用”的简单的默认的核心执行接口；
- (5) 提供 Spring 框架中配置、自定义和扩展服务；

- (6) 所有默认实现的核心服务能够容易地被扩展与替换，不会影响基础层；
- (7) 提供一个简单的部署模式，使用 **Maven** 进行编译。

1.2.2 Spring Batch 架构

Spring Batch 核心架构分为三层：应用层、核心层、基础架构层。具体参见图 1-2。

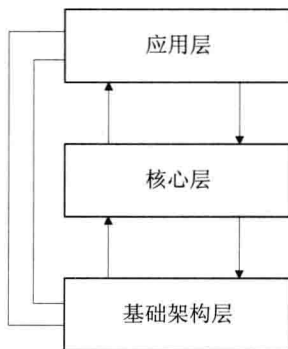


图 1-2 Spring Batch 三层核心架构

应用层包含所有的批处理作业，通过 **Spring** 框架管理程序员自定义的代码。核心层包含 **Spring Batch** 启动和控制所需要的核心类，如：**JobLauncher**、**Job** 和 **step** 等。应用层和核心层建立在基础架构层之上，基础架构层提供通用的读（**ItemReader**）、写（**ItemWriter**）和服务处理（如：**RetryTemplate**：重试模板；**RepeatTemplate**：重复模板，可以被应用层和核心层使用）。**Spring Batch** 的三层体系架构使得 **Spring Batch** 框架在不同的层级进行扩展，避免不同层级间的影响。

1.3 Spring Batch 优势

Spring Batch 是由 **SpringSource** 和 **Accenture**（埃森哲）合作开发的。**Accenture** 在批处理架构上有着丰富的工业级别的经验，贡献了之前专用的批处理体系框架（这些框架历经数十年研发和使用，为 **Spring Batch** 提供了大量的参考经验）；**SpringSource** 则有着深刻的技术认知和 **Spring** 框架编程模型。

Spring Batch 框架通过提供丰富的即开即用的组件和高可靠性、高扩展性的能力，使得开发批处理应用的人员专注于业务的处理，提升批处理应用的开发效率，通过 **Spring Batch** 可以快速地构建出轻量级的健壮的并行处理应用。

使用 **Spring Batch** 框架，你可以获得如下几小节所述的优势。

1.3.1 丰富的开箱即用组件

开箱即用组件包括对各种类型资源的读、写。

- 读：支持文本文件读、XML 文件读、数据库读，JMS 队列读。
- 写：支持写文本文件、XML 文件、数据库、JMS 队列。

该组件还提供作业仓库、作业调度器等基础设施，大大简化了批处理应用开发的复杂度。

1.3.2 面向 Chunk 的处理

面向 Chunk 的处理，支持多次读、一次写，避免了多次对资源的写入，大幅提升了批处理应用的处理效率。

1.3.3 事务管理能力

Spring Batch 框架默认采用 Spring 提供的声明式事务管理模型，面向 Chunk 的操作支持事务管理，同时支持为每个 tasklet 操作设置细粒度的事务配置：隔离级别、传播行为、超时设置等。

1.3.4 元数据管理

Spring Batch 框架自动记录 Job 的执行情况，包括 Job 的执行成功、失败、失败的异常信息，Step 的执行成功、失败、失败的异常信息，执行次数，重试次数，跳过次数，执行时间等，方便后期的维护和查看。

1.3.5 易监控的批处理应用

Spring Batch 框架提供多种监控技术，支持对批处理操作的信息进行查看和管理，通过 Spring Batch 框架为批处理应用提供了灵活的监控模式，包括：

- 直接查看数据库；
- 通过 Spring Batch 提供的 API 查看，基于 API，你可以打造自己的管理监控台；
- 通过 Spring Batch Admin 进行查看，Spring Batch Admin 是 Spring 的另外一个项目，通过该项目你可以通过 Web 控制台监控和操作 Job；
- 通过 JMX 控制台查看。

1.3.6 丰富的流程定义

Spring Batch 框架支持顺序任务、条件分支任务，基于顺序和条件任务可以组织复杂的任务流程。同时 Spring Batch 支持复用已经定义的 Job 或者 Step，同时提供 Job 和 Step 的继承

能力，方便任务的抽象。

1.3.7 健壮的批处理应用

Spring Batch 框架支持作业的跳过、重试、重启能力，避免因错误导致批处理作业的异常中断，例如如下操作。

- 跳过（Skip）：通常在发生非致命异常的情况下，应该不中断批处理应用；
- 重试（Retry）：发生瞬态异常情况下，应该能够通过重试操作避免该类异常，保证批处理应用的连续性和稳定性；
- 重启（Restart）：当批处理应用因错误发生错误后，应该能够在最后执行失败的地方重新启动 Job 实例。

1.3.8 易扩展的批处理应用

Spring Batch 框架通过并发和并行技术实现应用的横向、纵向扩展机制，满足数据处理性能的需要。

扩展机制包括：

- 多线程执行一个 Step（Multithreaded step）；
- 多线程并行执行多个 Step（Parallelizing step）；
- 远程执行作业（Remote chunking）；
- 分区执行（Partitioning Step）。

1.3.9 复用企业现有 IT 资产

Spring Batch 框架提供多种 Adapter 能力，使得企业现有的服务可以方便地集成到批处理应用中，避免了重新开发，达到复用企业遗留的服务资产。

1.4 Spring Batch 2.0 新特性

相对于 Spring Batch 1.X 系列，Spring Batch 2.X 系列提供了如下新的特性：

- （1）支持 Java 5；
- （2）非顺序的 Step 支持；
- （3）面向 Chunk 处理；
- （4）强化元数据访问；
- （5）增强扩展性；
- （6）可配置。

1.4.1 支持 Java 5

从 Spring Batch 2.X 版本开始, 使用 Java 5 进行开发, 支持 Java 5 提供的增强特性, 如泛型、参数化类型等。

1.4.2 支持非顺序的 Step

Spring Batch 2.0 支持条件判断执行 Step 的方式。在 2.0 版本之前, 仅支持顺序执行 Step。顺序执行 Step 参见图 1-3。

新的 Step 执行方式增加了条件判断功能, 参见图 1-4。

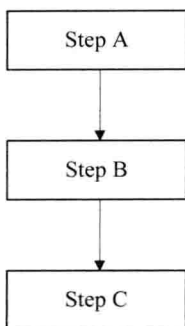


图 1-3 Spring Batch 1.X 版本仅支持顺序 Step

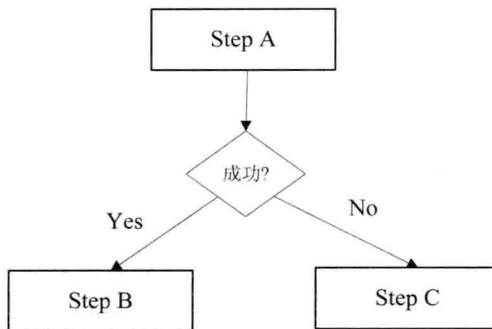


图 1-4 Spring Batch 2.X 版本支持条件 Step

代码清单 1-1 展示了如何配置图 1-4 中的条件流程。

代码清单 1-1 配置条件流程示例

```
1. <job id="job">
2.   <step id="stepA">
3.     <next on="FAILED" to="stepB" />
4.     <next on="*" to="stepC" />
5.   </step>
6.   <step id="stepB" next="stepC" />
7.   <step id="stepC" />
8. </job>
```

1.4.3 面向 Chunk 处理

Spring Batch 1.X 版本对数据处理默认提供的策略是面向 Item 处理。其序列图参见图 1-5。

在面向 Item 处理中, ItemReader 会返回一个对象 (即 Item) 给 ItemWriter 进行处理, Item 的数量为提交间隔的要求时提交计算结果。例如, 如果提交所要求的 Item 数量为 3 时, ItemReader 和 ItemWriter 分别会被调用 3 次。使用代码清单 1-2 表示图 1-5 的执行。