Introducing Spring Batch

Dave Syer, SpringSource

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Overall presentation goal

An introduction to Spring Batch and how it is being used in projects today.

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Agenda

- **Introduction: batch patterns and typical use cases**
- Spring Batch overview: modeling the batch domain
- Case Studies
- Roadmap and product development process

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Batch Processing Patterns

- Close of business processing
  - Order processing
  - Business reporting
  - Account reconciliation
- Import/export handling
  - Instrument/position import
  - Trade/allocation export
- Large-scale output jobs
  - Loyalty scheme emails
  - Financial statements

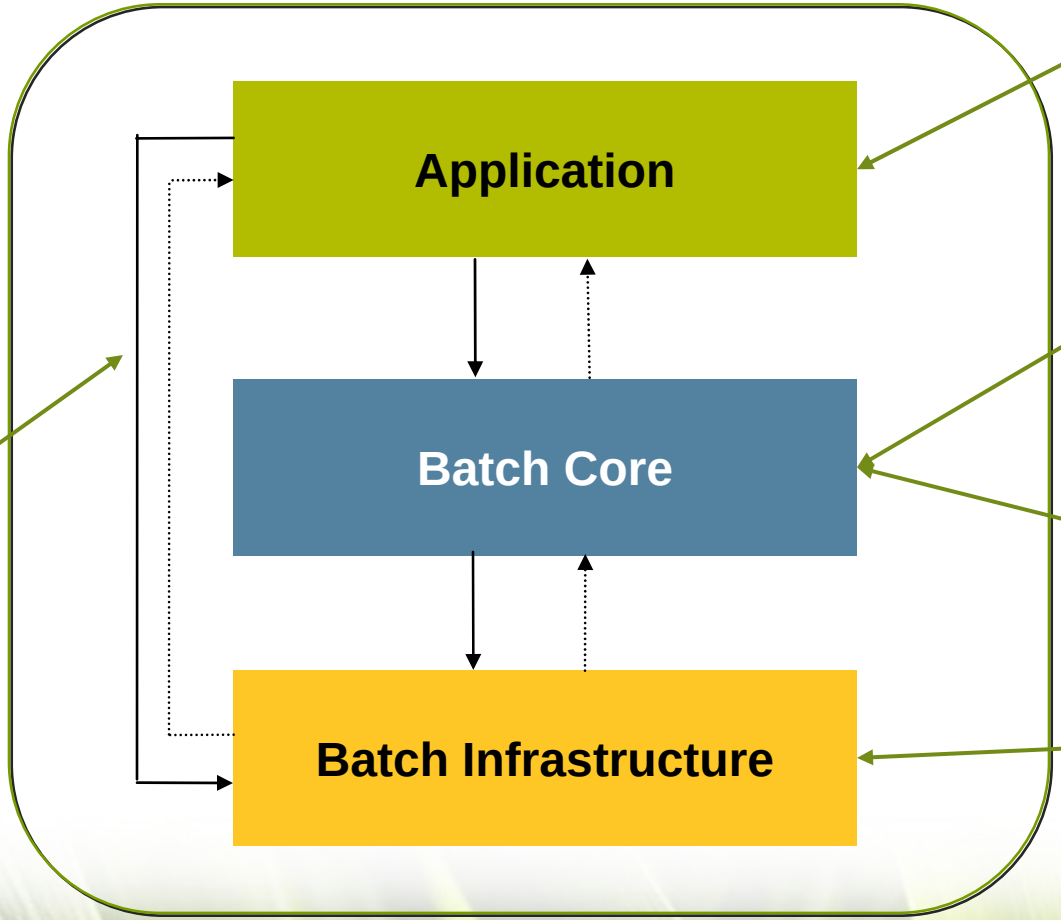
Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Agenda

- Introduction: batch patterns and typical use cases
- **Spring Batch overview: modeling the batch domain**
- Case Studies
- Roadmap and product development process

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Spring Batch: Layered Architecture



Business Domain – Record-level data (e.g. Trade)

Job, Step, JobLauncher

JobLauncher, Implementations of Core API

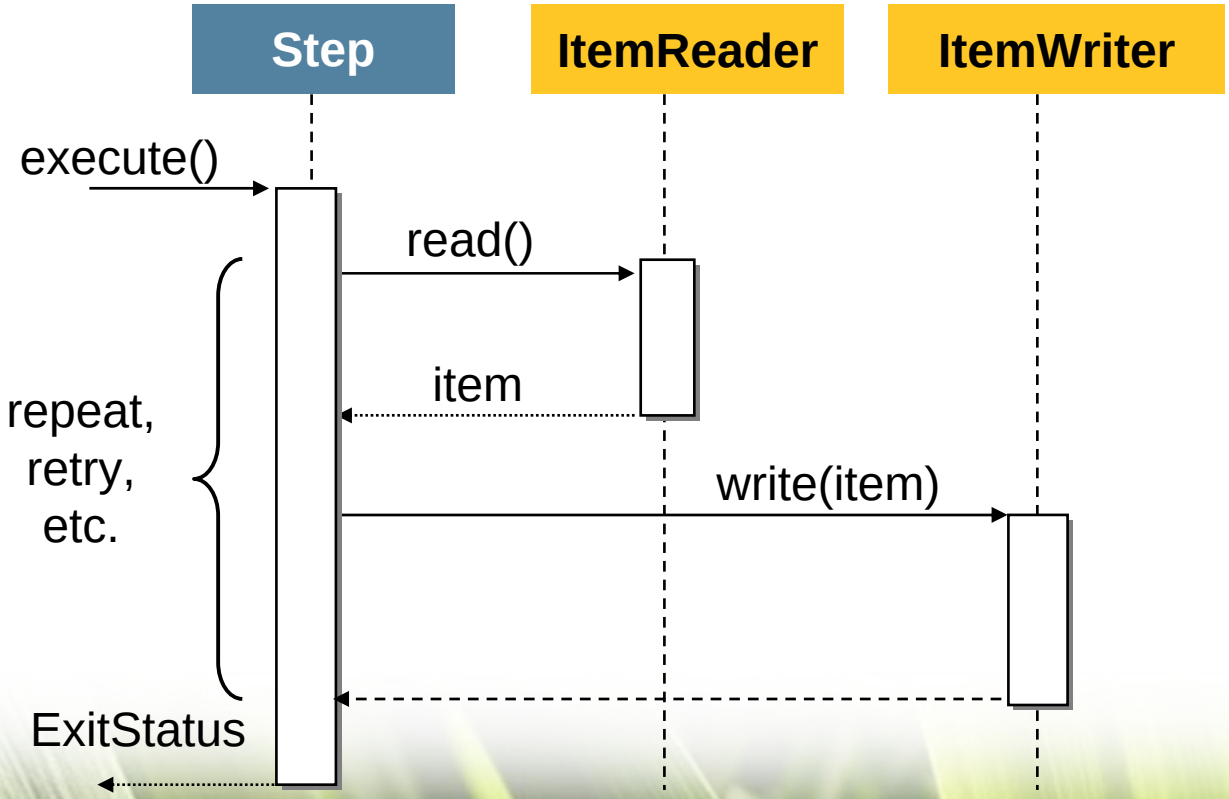
Repeat, Retry, Transaction, Input, Output

Declarative, plus Abstractions and Extension Points

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

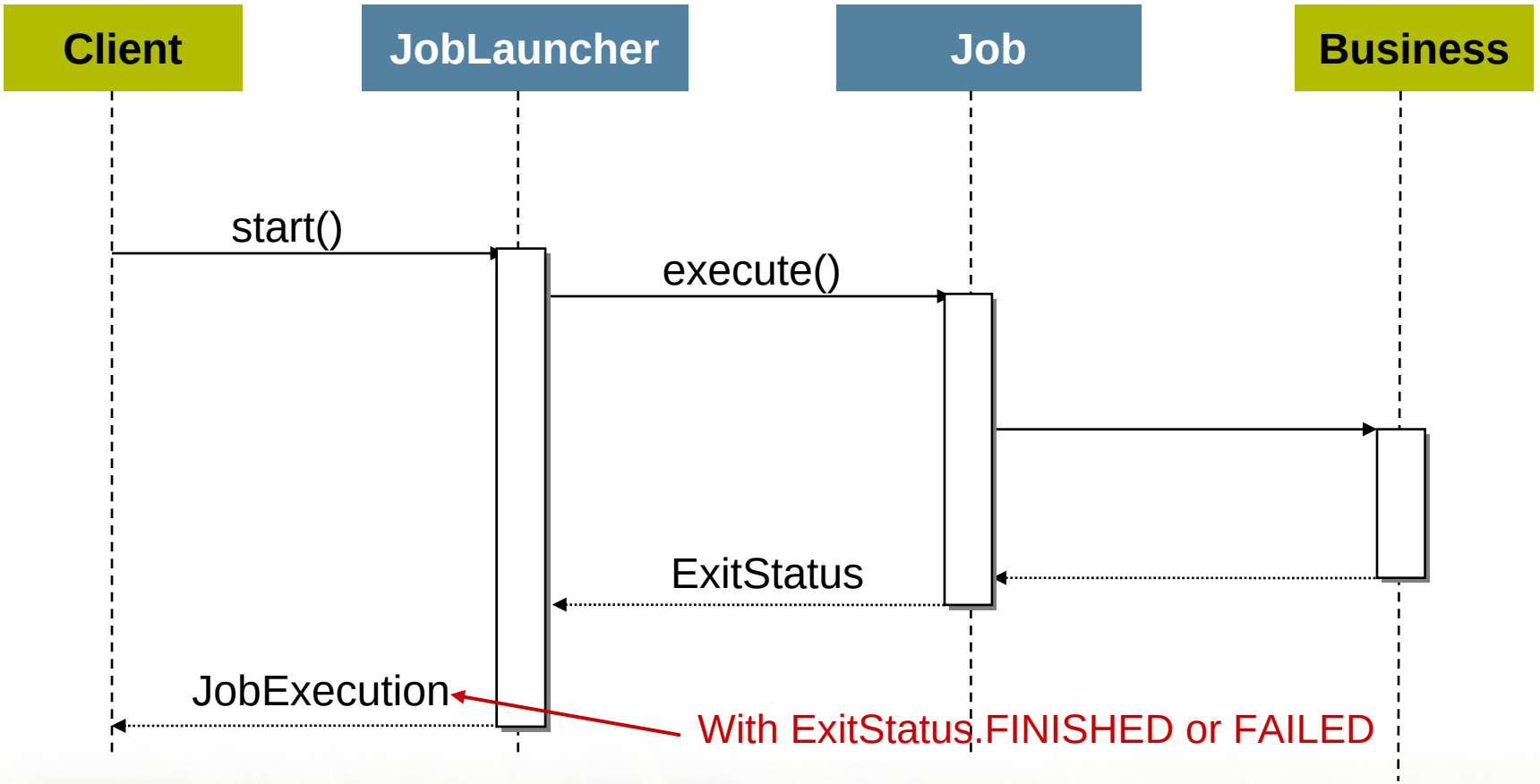
Item-Oriented Processing

- Input-output can be grouped together = Item-Oriented Processing



Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

JobLauncher



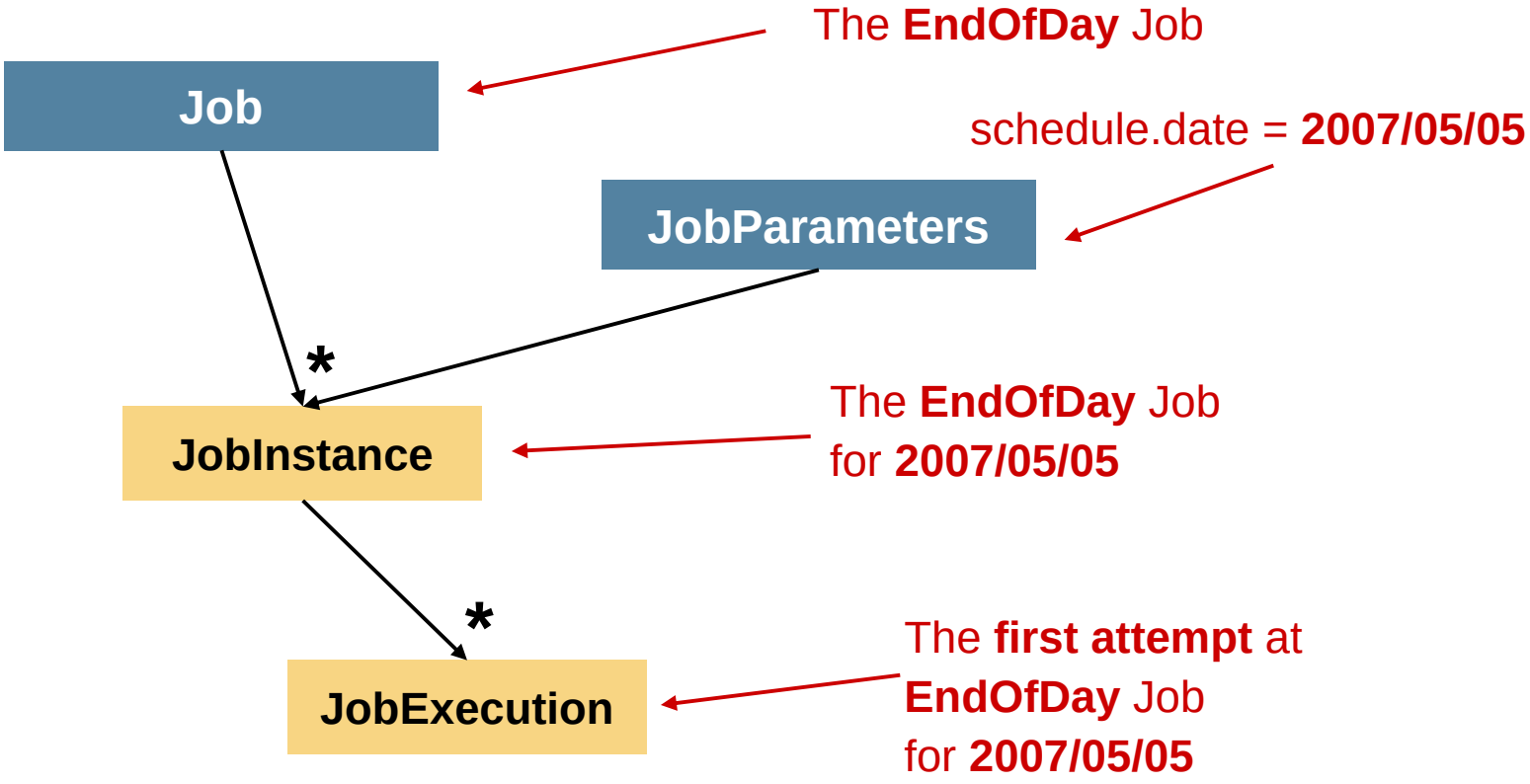
Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Batch Infrastructure and Batch Domain

- Batch infrastructure adds optimisation and low level abstractions
- The batch domain adds some value to a plain business process by introducing new concepts:
  - A **job** has an identity – not just a stream of bytes
  - A **job** has **steps**
  - A **job** can be restarted after a failure – a new **execution**
  - Each **execution** has a start time, stop time, status
  - The **job** has a status
  - Each **execution** can tell us how many **items** were processed, how many commits, rollbacks, skips

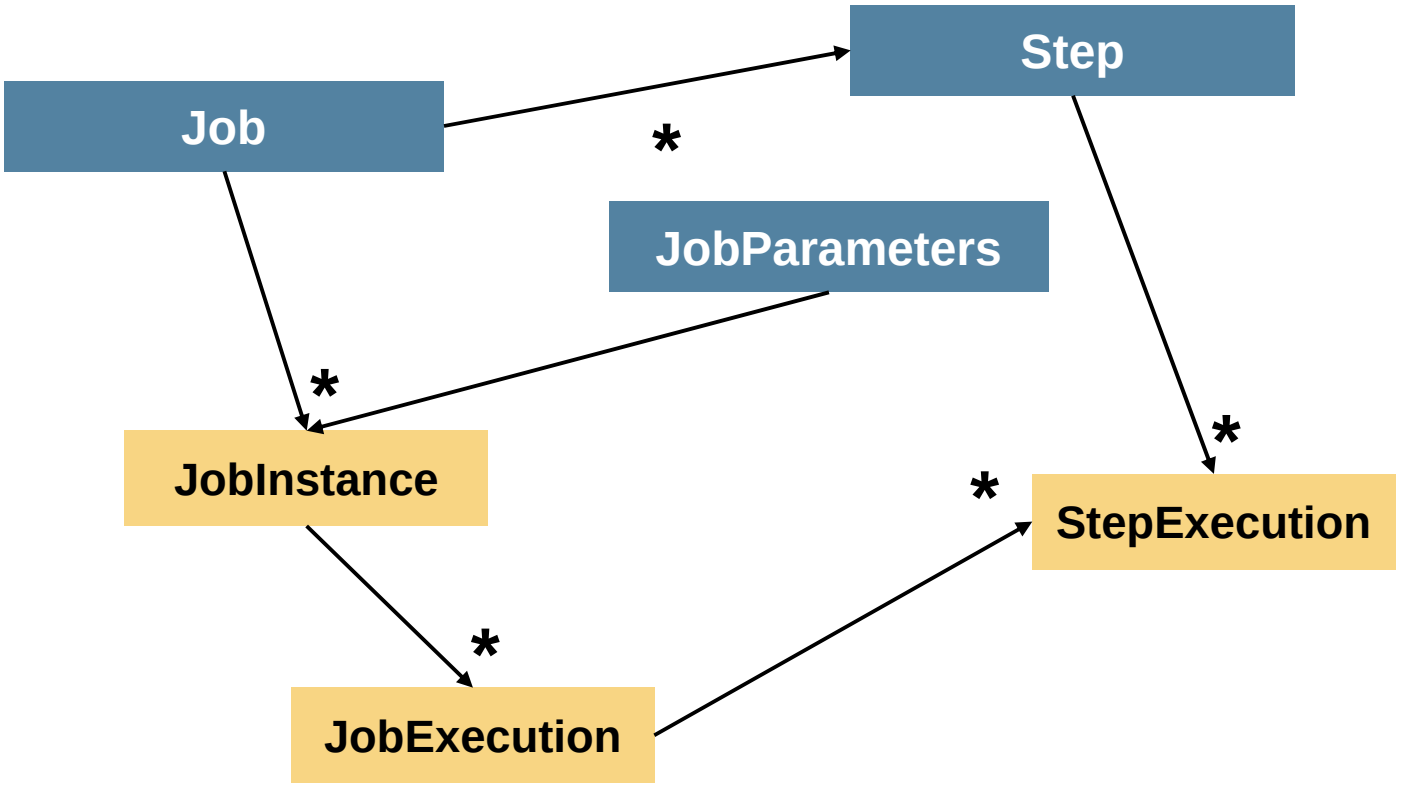
Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Job Configuration and Execution



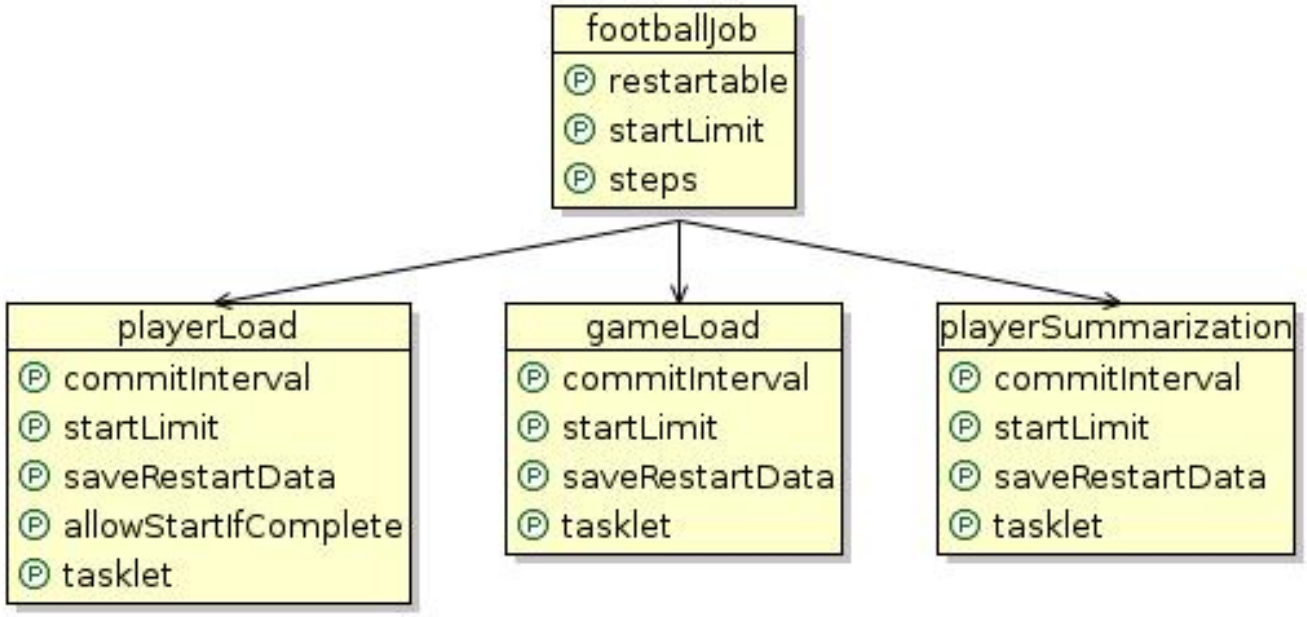
Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Job and Step



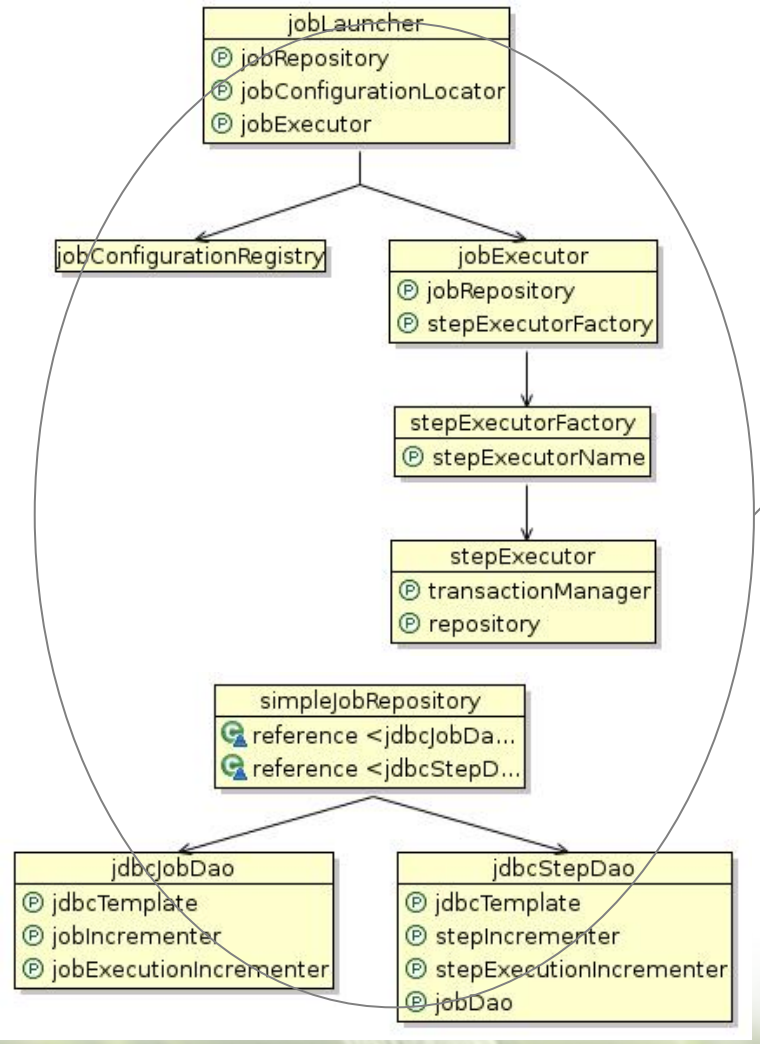
Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

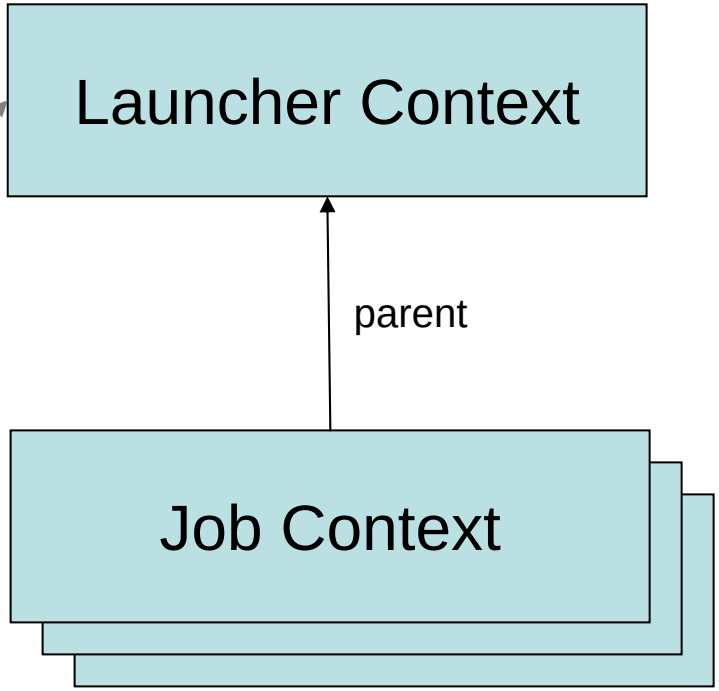
Job Configuration – football job sample



Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Job Launcher Configuration





Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Sample Jobs

Job/Feature	delimited input	fixed-length input	xml input	multiline input	db driving query input	db curosr input	delimited output	fixed length output	xml output	multiline output	db output	skip	restart	quartz
fixedLengthImportJob		X									X			
multilineJob		X		X										
multilineOrderJob	X			X				X		X				
quartzBatch														X
simpleModuleJob		X									X			
simpleSkipSample												X		
skipWithRestartSample												X	X	
sqlCursorTradeJob						X								
tradeJob	X				X		X							
xmlJob			X						X					

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Agenda

- Introduction: batch patterns and typical use cases
- Spring Batch overview: modeling the batch domain
- **Case Studies**
- Roadmap and product development process

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Case Studies

- Large European Healthcare
- State Government
- Leading Investment Bank

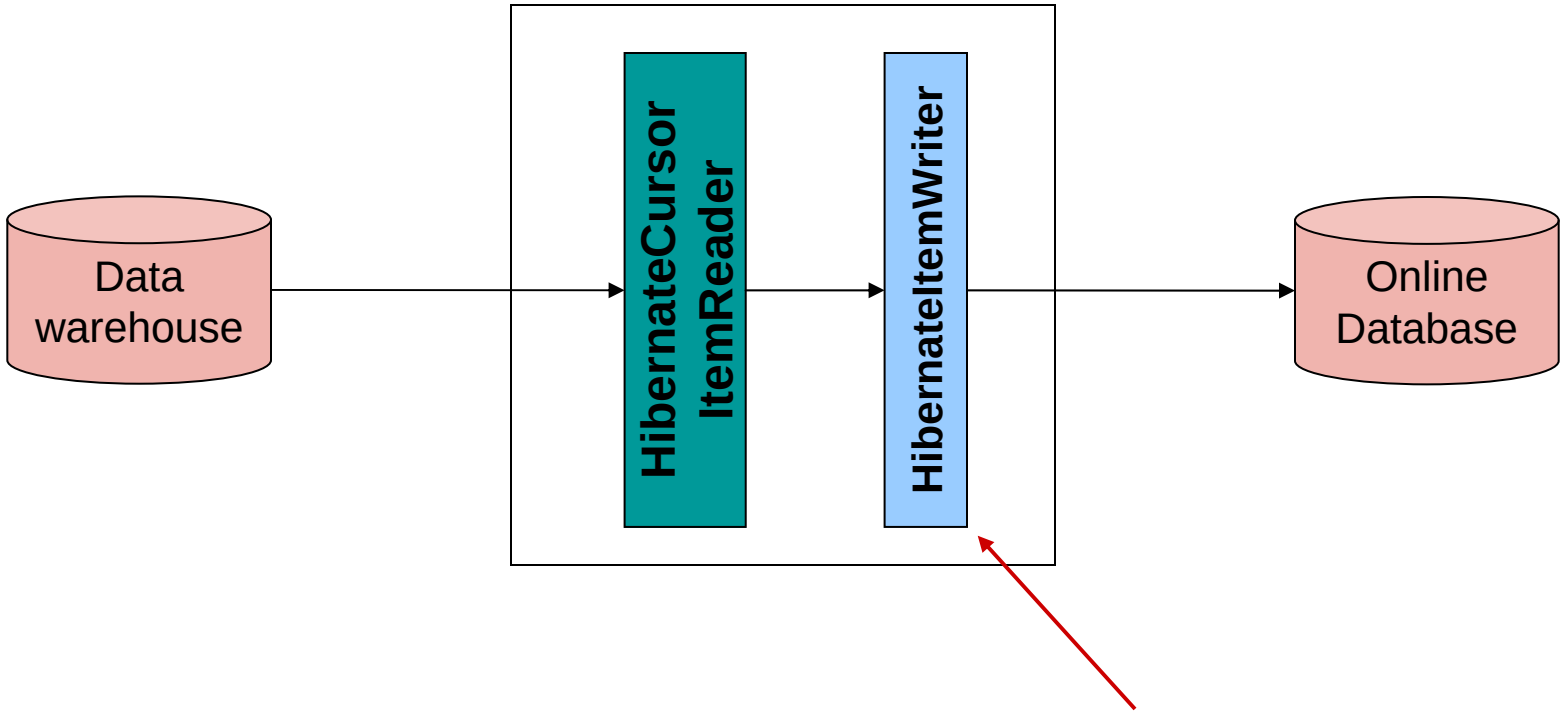
Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Large European Healthcare

- Background: In order to align with the corporate strategy of using Java with Spring. COBOL mainframe batch processing partially migrated to Spring Batch.
- Solution:
  - Used XML Input Sources, allowing for a batch orientated approach to XML input.
  - Both input and output from the database used Hibernate, allowing online DAOs to be reused.
- Benefit
  - Allowed for a homogeneous development environment with online developers using Spring and Web Flow.

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Typical Job: Updates for Online Database



Re-using business logic implementation from online system

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Hibernate Input

- Cursor Driven
- Stateless or standard Session.

```
<bean id="hibernateInputSource“
    class="....item.database.HibernateCursorInputSource"
    >
    <property name="queryString" value="from
    CustomerCredit" />
    <property name="sessionFactory"
    ref="sessionFactory" />
</bean>
```

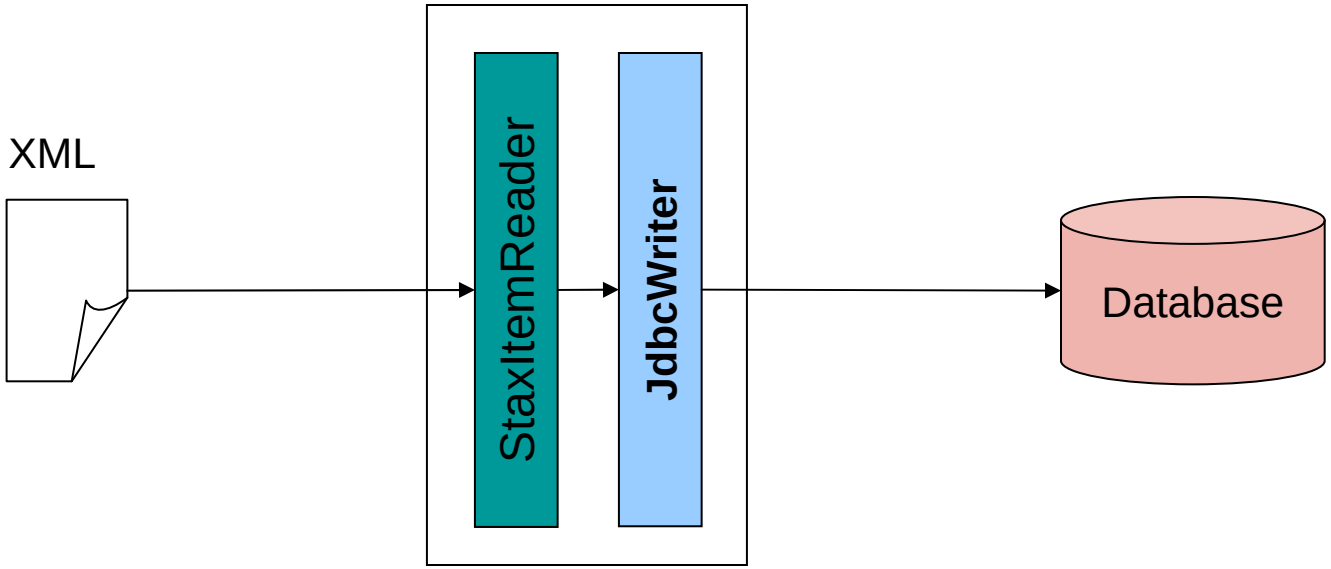
Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Hibernate Output

- Allows for Online DAOs to be reused by item processor
  - Flush per item
  - Inefficient
  - Allows for easy identification of failed items.
- Flush per chunk
  - Greatest efficiency
  - Failed records cannot be determined (requires search, e.g. binary)

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Integration Layer: Support Job



Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

XML I/O

- Built on StAX
  - Root element representing an ‘Item’ must be defined.
  - Each root element is read in via a StAX parser, then deserialized to an Item
- Uses Spring-OXM to bind event fragments to ‘Items’

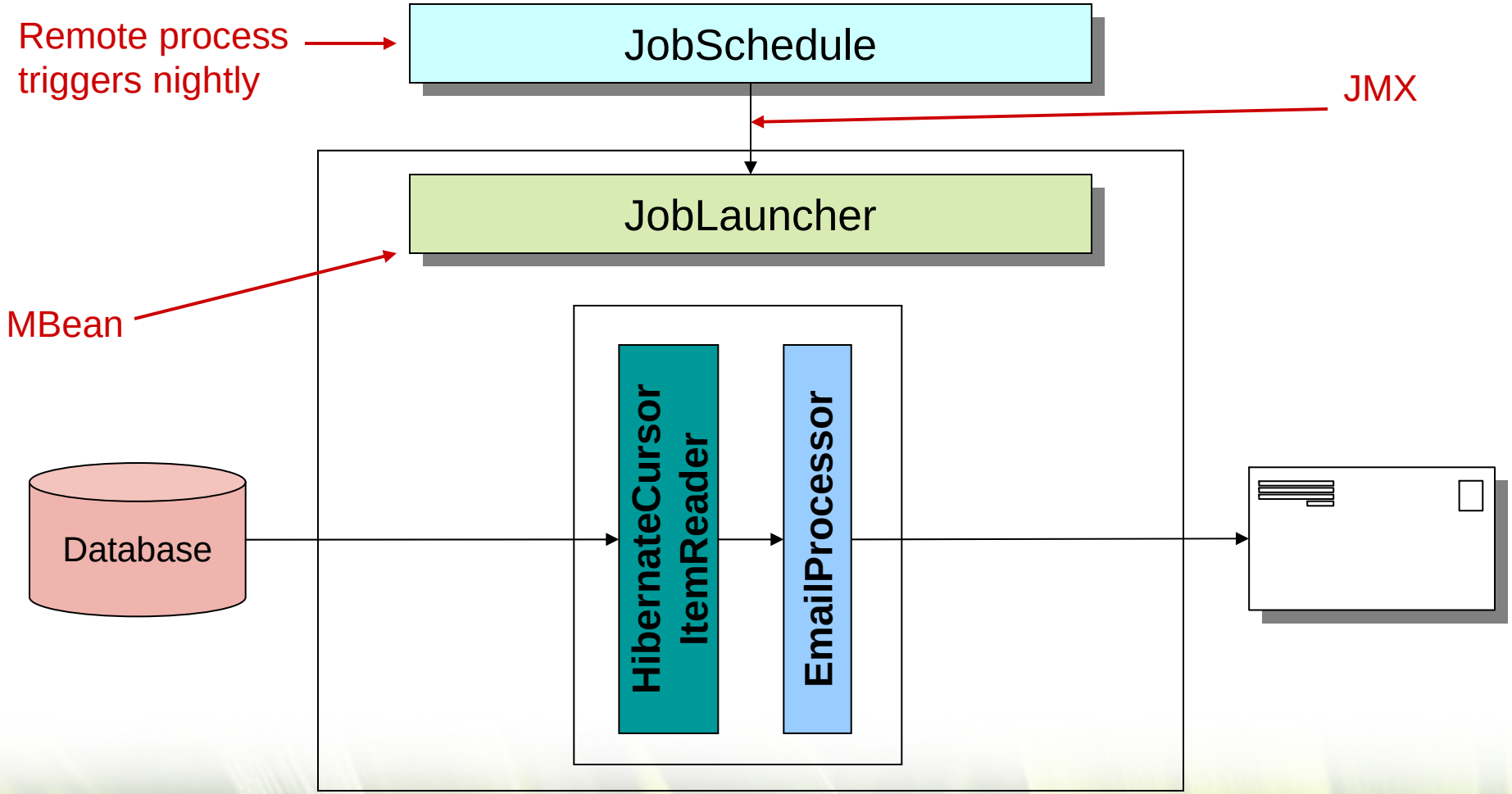
Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

XML I/O

```
<bean class="...item.file.support.StaxEventReaderInputSource">
   <property name="fragmentRootElementName" value="trade" />
   <property name="fragmentDeserializer">
    <bean ...>
        <constructor-arg>
            <bean
   class="org.sfw.oxm.xstream.XStreamMarshaller">
                <property name="aliases" ref="aliases" />
            </bean>
        </constructor-arg>
    </bean>
   </property>
</bean>
```

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Notification E-Mails



Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

State Government

- Background: In-Progress IT Renewal project replacing mainframe batch jobs with Java in order to process unemployment claims.
- Solution:
  - Custom Item Readers for processing CopyBook and EBCDIC input
  - Custom LineTokenizer for specific Government flat file formats.
  - DrivingQuery ItemReader provided a better approach for using DB2 with online DAOs.
- Benefits
  - Batch job development is only piece ahead of schedule.

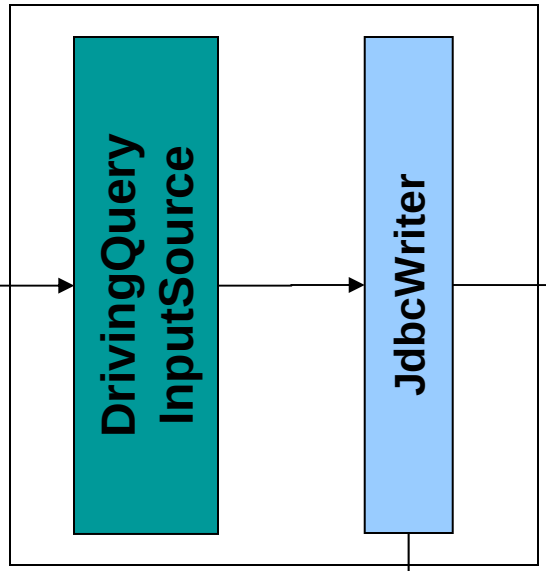
Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

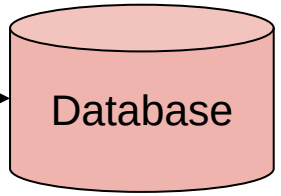
Driving Query

- Requires an initial ‘driving’ SQL statement that will return a list of all the unique keys to be processed.
- Each call to read() returns one key
- Allows for reuse of other DAOs to return ‘details’.
- Smaller database footprint (than cursor) allowing for better concurrency with online systems, especially in database systems with pessimistic locking strategies.
- All DAOs using provided keys in a batch scenario should use PreparedStatements, to allow for caching that greatly improves performance.

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Driving Query





Keys obtained at step initialization

Existing DAO

Query for details using key as parameter

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Driving Query

```
<bean id="drivingQuery"
    class="...item.database.DrivingQueryInputSource"
    <property name="keyGenerator" ref="keyGenerator" />
</bean>
<bean id="keyGenerator"
    class="...item.database.support.SingleColumnJdbcKey
    Generator">
    <constructor-arg ref="jdbcTemplate" />
    <constructor-arg ref="select ID from TRADES" />
</bean>
```

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Restartability and Parallel Processing

- Challenge for restartability:
  - Chunks can be processed in any order
  - Chunk 1 can fail after Chunk 100 has completed
- Single VM, multi-threaded (Spring Batch 1.0)
- Batch Pattern: **Process Indicator**
  - Add column to input records
  - Mark “finished” on chunk boundary
  - Restart: initialise to only process unfinished records

SELECT * FROM T_INPUT WHERE PROCESSED=‘N’

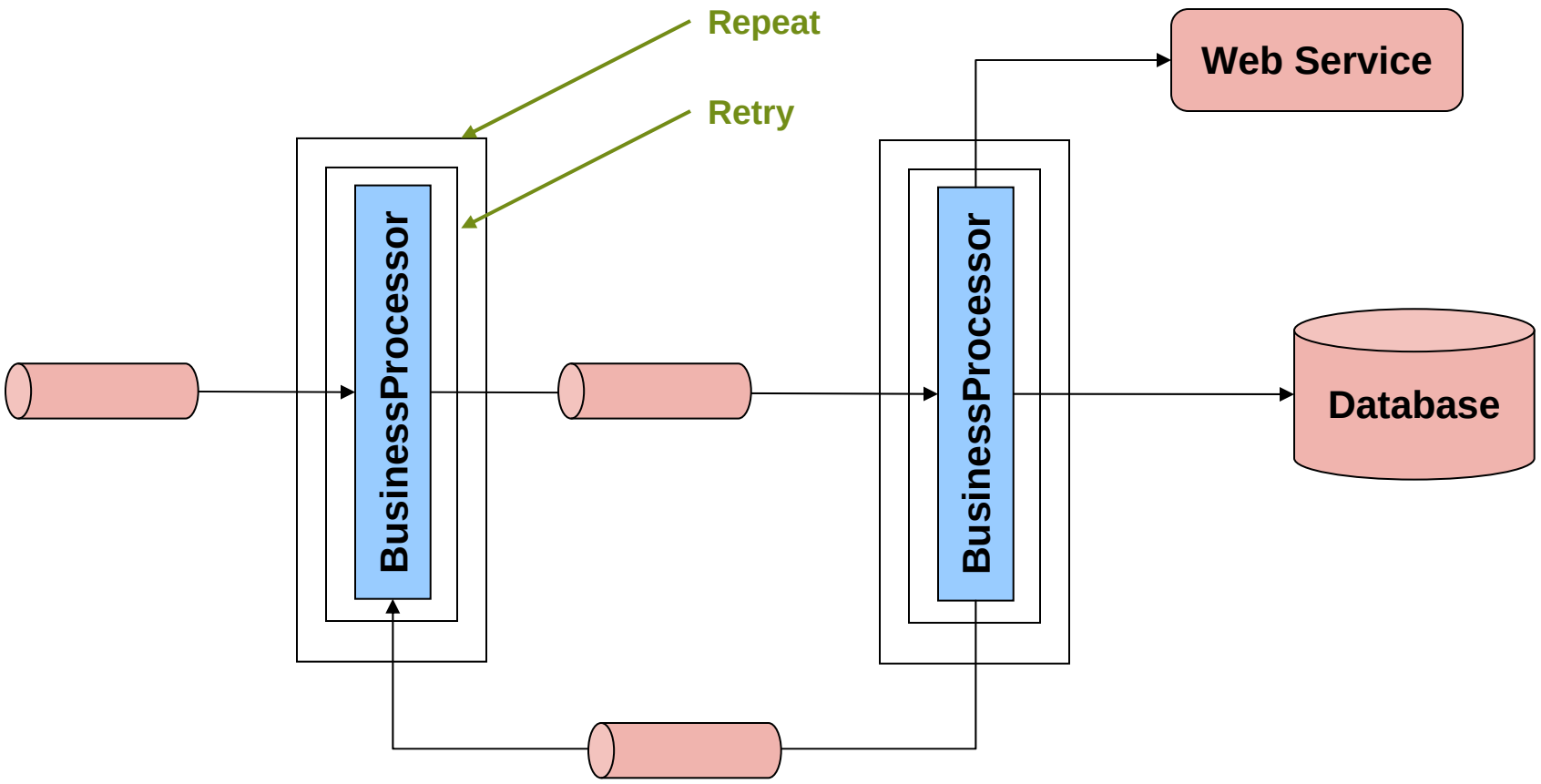
Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Leading Investment Bank

- Background: Large networks of message-driven pipelines processing orders, payments and other transactions.
- Solution:
  - Use RepeatTemplate to batch messages together transparently.
  - Use RetryTemplate to retry either
    - Whole transaction (batch)
    - Or an external webservice call
- Benefits:
  - Greatly increased throughput in high volume process.
  - Reduce burden on developers to understand optimisations.

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Message-driven Pipeline Optimisation



Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Agenda

- Introduction: batch patterns and typical use cases
- Spring Batch overview: modeling the batch domain
- Case Studies
- **Roadmap and product development process**

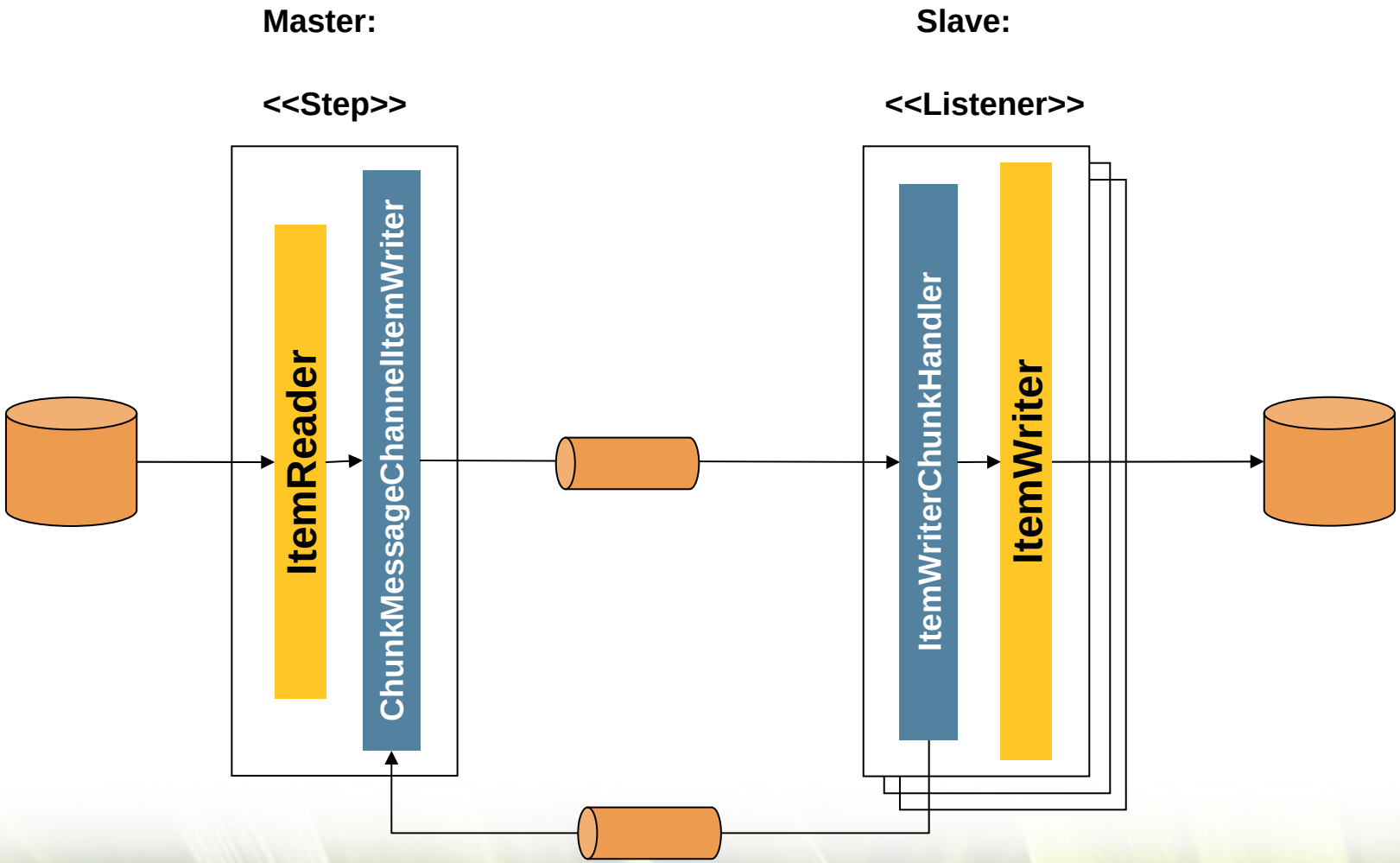
Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Spring Batch—Road Map

- Release 1.0.1.FINAL released May 2008; themes:
  - Infrastructure support for
    - Flat files, XML, Database (JDBC + Hibernate)
    - Repeat, retry
  - Single-JVM, multi-threaded
- In use at roughly 40+ projects with at least 3 in production
- 1.1 July 2008; themes: shared state between Steps
- 2.0 target 2008Q4; themes:
    - Java 5
    - Scalability
    - Non-linear item processing
    - Dependencies between jobs or steps

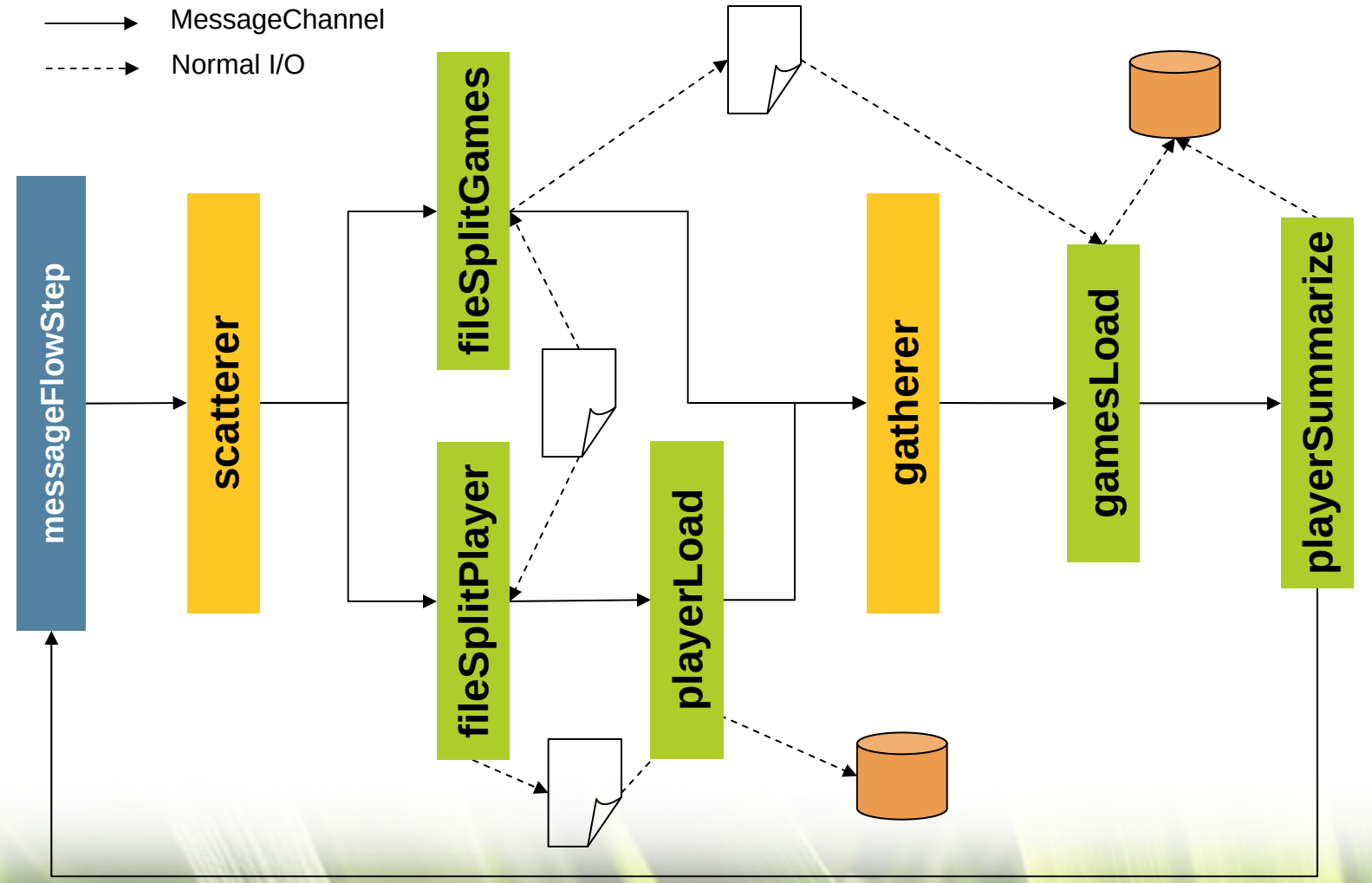
Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Scaling a Step



Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Non-linear Job



Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Accenture and SpringSource Partnership

- Why is Accenture contributing to open source?
  - Consolidating decades worth of experience in building high-performance batch solutions
  - Driving standardization in batch processing
- Why Spring?
  - Established, active, and leading community with significant momentum
  - Logical home for a batch architecture framework as part of the Spring Portfolio
- End Goal
  - Provide a highly scalable, easy-to-use, customizable, industry-accepted Batch architecture framework

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Summary

- Lack of a standard enterprise batch architecture is resulting in higher costs associated with the quality and delivery of solutions.
- Spring Batch provides a highly scalable, easy-to-use, customizable, industry-accepted batch framework collaboratively developed by Accenture and SpringSource
- Spring patterns and practices have been leveraged allowing developers to focus on business logic, while enterprise architects can customize and extend architecture concerns

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.

Q&A

Copyright 2007 SpringSource. Copying, publishing or distributing without express written permission is prohibited.