JSR 352 Expert Group

Working Session

24 February 2012

Agenda

- Discussion: Job Parameters Redux
- Discussion: Job Launch, Operations, Explorer – Continued
- Discussion: Runtime Configuration Wrapup?
- Discussion: Readers/Writers
- List for Next Meeting

Discussion: Job Parameters

- Substitutions Values

Optional substitution symbol

```
@Job(name="Job1")
@SubstitutionDefaults({"substitution1=defaultValue1")
@Properties({"key3=value3","key4=${substitution1}",…})
public class MyJob {

        @Step(name="Step1")
        @Properties({"key1=${substitution2}","key2=override2",…})
                MyStep _step;

}
```

Required substitution symbol

Substitution syntax:

$("symbol") – regular symbol

$("system:symbol") – resolved from System properties

$("env:symbol") – resolved from environment variables

Discussion: Job Parameters

- Substitution Symbols When Submitting (Launching) Jobs

```
BatchContainer container= new BatchContainerImpl(“config.xml”);
String[] properties= {“substitution1=value1”,”substitution2=value2”,…}
container.submit(“Job1”, properties);
```

or

```
Properties props= new Properties();
props.setProperty(“substitution1”,”value1”);
props.setProperty(“substitution2”,”value2”);
constainer.submit(“Job1”,props);
```

Discussion: Launch, Operator, Explorer

Idea:

- one stop shopping for Repository CRUD methods
- everything you need to interact with batch container (e.g. build operations console

```
public interface BatchManager {

    JobId submit(String jobName,String[] substitutions);
    void stop(JobId jobInstance);
    void restart(JobId jobInstance, String[] substitutions);
    void remove(JobId jobInstance);
    JobResults getResults(JobId jobInstance); // exitstatus/job RC
    void getJobLog(JobId jobInstance,Writer jobLogWriter);
    List<JobInfo> getJob(JobFilter filter);
}
```

Discussion: Runtime Configuration

- BatchContainer config file (xml)
  - List of plugin implementations:
    - TransactionManager **<<< More discussion?**
    - JobRepositoryManager
    - JobLogManager
    - ExecutionManager (i.e. task executor)
    - CommandLineHandler
  - Properties bags to configure each plugin

Discussion: Readers/Writers

Idea: combine ItemReader(Writer) and ItemStream semantics

- Annotations

```
@Reader (or @Writer)
public  class MyReader {
    @Open void open(CheckpointInfo chkpt) {…}
    @Close void close() {…}
    @ItemReader {Type} read() {…}
    @GetCheckpointInfo CheckpointInfo getCheckpt() {…}
}

@ItemWriter void write({Type} record) {…}
```

Discussion: Readers/Writers

- Usage

```
@Step(…)
@Reader(ref="MyFileReader")
@Writer(ref="MyFileWriter")
public class MyStep {

@ItemProcessor {Type1} process({Type2} rec) {…}

}
```

List for Next Meeting

- Listeners
- Concurrency
- Future
  - Repeat
  - Exit codes
  - Step conditions
  - Execution Context
  - Metrics
  - Java EE