JSR 352 Expert Group Meeting
February 15th, 2012
Minutes

Meeting Time: 9-10:00AM ET

Attendees:

Kevin Connor
Mahesh Kannan
Wayne Lund
Simon Martinelli
Michael Minella
Joe Pullen
Chris Vignola

References: EGAgenda-15Feb2012 (slides)

Discussion Summary:

Group walked through the referenced slides. The following points were made:

- Regarding @GetReturnCode , Chris remarked there may be a defined range of well-known return code values. We need to discuss that in a future meeting .

- Regarding specifying @Step properties , Chris remarked there should be a way to specify properties at the Step level in the annotation, that these would constitute default properties, and that there must be some mechanism to override those defaults dynamically at job initiation time. Wayne remarked there is a way to do that in Spring Batch and he would make that known.

- Regarding @Step, we spent some time discussing the 3 closure Spring Batch model: reader, processor, writer. We discussed for a time whether a generic @Step orientation was the way to go or whether a more structurally specific "item processor" approach should be adopted. This requires further discussion.

- Joe remarked on the utility of the Spring Batch "tasklet" approach. Chris confessed he did not quite understand and asked if someone could explain how the tasklet concept could be projected into a Java SE environment. Wayne agreed to make a proposal.

- The group discussed CDI briefly. Chris remarked the specification is targeted to Java SE and EE and should be careful to not require other JSRs that are not part of Java SE. Presently, CDI is not part of Java SE.   So Chris proposed that CDI not be required for JSR 352. However, he also remarked it should be possible for a developer to optionally use CDI to construct a batch application. So the JSR should allow for, but utilize CDI.

- The group discussed whether to use both annotations and JCL to specify jobs. Many members have already made the point in the public mailing list that it's a good idea to offer two ways. Mike offered a good admonition, with which the group generally agreed: there should be parity between annotations and JCL where there is overlap. I.e. it should never be the case that you need to switch from annotations to JCL because of some missing feature in the annotations or vice versa. It appears we will define both annotations an a JCL.

- The group discussed the distinction between application JCL and job JCL: the idea that application JCL (e.g. Spring Batch DSL) is part of the application and created by the application developer; the idea that job JCL exists outside the application and is created by a "job developer".   A job developer may or may not be the same person as the application developer. Chris asserted a job JCL is more strongly associated with a job scheduler - a scheduling component separate and apart from the batch container; he reminded the group ,as a matter of JSR 352 staging, scheduling is proposed as a revision 2 item.   Wayne agreed both JCL models are valid and useful. No disagreement among the group.

- A member (Wayne?) asked about job dependencies - i.e. the completion of one job (or jobs) causing the launch of another.   Chris called this "job orchestration" and said he believes that is more strongly associated with a scheduler function than with the batch application definition or the batch container; he referenced schedulers like Control-M as exemplars.

- A member (Mike?) asked about jobs submitting jobs.   Chris remarked a batch application is free to use batch container APIs to submit/monitor jobs; additionally, that the programming model could be used to implement a reusable "job submitter" batch step that could be used in any batch job.

- The group meets again on Friday, 17 February 2012.   The main topic of discussion will be readers/writers and checkpointing.

Submitted,

Chris Vignola
February 15th, 2012