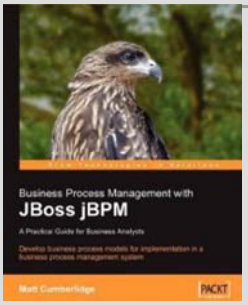

Business Process Management with JBoss jBPM: A Practical Guide for Business Analysts

Matt Cumberlidge
PACKT Publishing, 2007
$49.99 206 pages

Reviewed by Paul Harmon

Matt Cumberlidge is a Business Analyst who works for a leading provider of information services in the UK. JBoss is a division of Red Hat and is the leading open source enterprise Java application server. JBoss jBPM engine and JBoss JBPM developer are utilities that let Java developers create BPMS applications. *Business Process Management with JBoss jBPM* is a guide designed to help Business Analysts and other Java developers learn the basics of BPMS development using the JBoss open source products.

Mr. Cumberlidge begins with a brief but accurate discussion of the history of business process and the current role of BPMS development. He contrasts the BPMS approach with an Agile approach to software development, but doesn't compare jBPM with any of the more sophisticated BPMS products available. He proposed the following product lifecycle:

- Understand the target process
- Develop the process in jBPM
- Prototype the process workflow user interface
- Iterate the workflow prototype
- Pilot and implement the workflow
- Monitor process execution and improve

The entire book is organized around a case study that focuses on a record company, Bland Records, that wants to improve the way it does business. It has four divisions: an Acquisition & Record Group, a Production Group, a Sales & Marketing Group and a Finance Group. The case focuses on the A&R Group and specifically on the Product Music Products, whose subprocesses are: Form band, Find musicians, Write song, Arrange music, and Shoot video. It's a good case study. Notice, for example, that all of the activities are performed by employees. There may be automation elements – databases may be used, but, overall, we are looking at a process that involves lots of people and lots of human decisions. Put a different way, this is a workflow application, as long as we understand that workflow, in this instance, doesn't mean document processing, but means the management of the flow of a process from one computer terminal to another.

Cumberlidge suggests that analysts model the business process in Visio, and illustrates with swimlane diagrams with Visio workflow symbols. (It's a shame he didn't use BPMN, but as jBPM doesn't support BPMN and its unlikely that anyone who works in jBPM will want to generate BEPL code, so perhaps it's OK.) Cumberlidge doesn't use a customer lane on his swimlane diagram. In other words, he lets the process begin when a Talent Scout holds auditions and lets it end when the record producer completes the album and DVD. By doing this he avoids any consideration of how one might scope the effort. How did the Talent Scout decide who to invite to the audition? What type of album do we want to make? Stepping back, who is the customer for

this album and what is likely to make that customer buy a DVD. It's no wonder Bland Records is struggling if their acquisition process doesn't consider what it's doing or why it's doing it. I point this out to say that this is not a book on process analysis, and that even the "high level" process that Cumberlidge is discussing is, in fact, rigged to be a process,that will interface with a database. In other words, we are considering a process that generates decisions that we will then store in a database as we work our way through the process.

Moving on, Cumberlidge works readers through a nice exercise in which they identify all the stakeholders of the process and create a matrix that maps each stakeholder to a step in the process. At this point we realize that Cumberlidge is using a very narrow definition of a "stakeholder." A stakeholder, in this case, refers to an employee who takes part in the process. This analysis does not consider other possible stakeholders like Customers, or Managers or Shareholders in the company. In essence, it doesn't have criteria by which to judge the ultimate success of the process. This approach is very much a software development effort. We are simply assuming that software automation will be a good thing – it will save money and time – and will thereby justify the effort. (Anyone who took the case study seriously would realize that this effort involves rearranging deck chairs on the Titanic. This company is failing because it is not producing interesting records. Producing bland records faster or for slightly less money isn't going to do much good!)

At about this point we move to JBoss jBPM and enter the process description into the software environment. If you look at a jBPM diagram, on screen, you see two types of things: boxes and arrows. A box can be either a transition (start, join, fork) or a task node (a person undertakes some action). Arrows indicate the transition paths. Actions are code that tells the BPMS application to perform some software activity (read DB file, update DB file) and these are not shown on the diagram. This works for the application that Cumberlidge has chosen, but imagine we created a diagram of a process that had many fully automated activities. All the automated activities would disappear from the diagram displayed in jBPM. This might be a bit confusing to a business manager who thinks a process has 5 major steps, but only sees three steps on the screen. In fact we could create virtual tasks to generate the appearance we wanted to, but what this really says is that jBPM is best used with workflow applications and not with any process that has lots of software integration activities.

Cumberlidge explains that jBPM supports a Graph Oriented Programming paradigm and goes on to say that jBPM is built around the concept of waiting. In essence, we send a request to an employee to perform a task – Create a Band – and we wait till we are told that the Band is formed. That's probably the main idea a programmer will take away from this book and its important.

Any calls to software applications are written in jPDL – the specialized programming language of jBPM. The tool generates a basic form for each user's interface screen, but the developer needs to do quite a bit of work to actually create a useful screen. The whole second half of the book is full of code examples. Readers will need Java, JBoss jBPM engine, JBoss BPM designer, JBoss application server, Ant and MySQL to do these exercises.

Once the application is complete Cumberlidge moves on to a short discussion of monitoring and BI and introduces SeeWhy, a BI tool that can be used in conjunction with JBoss. Using this tool, the reader is stepped through developing a manager dashboard that would allow a business manger to monitor the process during execution.

I actually liked this book and would recommend it to a Java developer that wanted a straight forward introduction to developing an application in JBoss jBPM. The book is well written, the overviews clean, and the exercises nicely paced.

On the other hand, I would want to warn anyone who bought this book to discount its claim to be an introduction to Business Process Management. Broadly, BPM is about solving business process problems. This book focuses so narrowly on an automation problem that it completely loses sight of what would be involved in helping Bland Records survive. SAP has been promoting the idea that we need to move beyond the Business Analysts role and create a new role that SAP terms a BPx – a Business Process eXpert. Clearly this book targets someone who is just going to get the requirements for an automation task and then undertake it without any concern for what kind of process change might be useful to make Bland Records a success.

Ignoring the fact that the book has a very narrow idea of the task of a Business Analyst, it also defines BPMS in a narrow way. Those who hope that BPMS applications will lead to a new era in which business managers understand their business processes better and can intervene quickly to alter processes that need to be changed will certainly be disappointed in any BPMS application built in jBPM. To begin with the modeling is so limited that the business person would hardly recognize a complex process that he or she had diagrammed in Visio. Second, so much of the application is written in code that I suspect that a senior Java programmer could change straight Java code almost as fast as a Business Analysis could change an application in jBPM. This product certainly wouldn't support the new relationship between business and IT that the leading BPM gurus urge and the best BPMS products support.

If you are a Java developer and you want a simple introduction to programming a BPMS system in jBPM, this book will step you through the process. If you are a business manager, or Business Analyst who is trying to evolve yourself into a Business Process eXpert, and want some help in understanding what people who are excited about BPM are excited about, this book won't help.

Paul Harmon is the Executive Editor of Business Process Trends