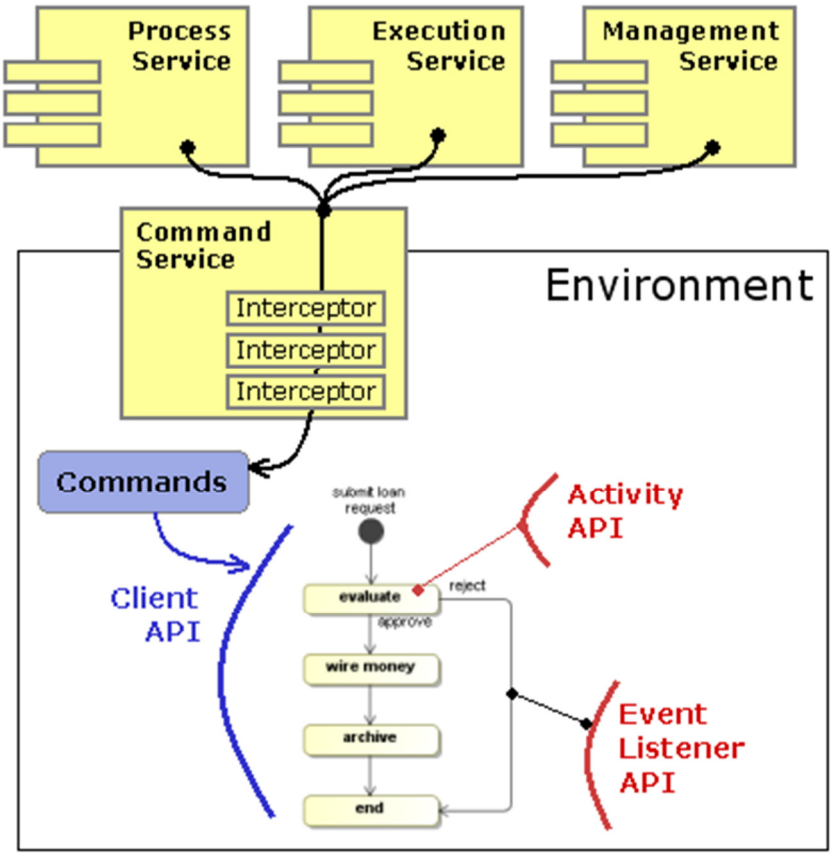
Proposal for jBPM

What is jBPM?

JBoss jBPM is a JBoss Enterprise Framework that delivers workflow, basic business process management (BPM) capabilities, and process orchestration in a scalable and flexible product footprint. The project's home is www.jboss.org/jbpm

What is the Architecture?



Basically, the architecture is a set of services that manage processes, and process instances. It takes a XML definition file written in either jPDL or BPMN and deploys it to the system. Then instances of the process definition are created for each unique flow.

Two services are not shown in this picture. One is a history service, which is used to get information about previous process instances. The other is an Identity service which is used to pull users and groups for task assignment and management.

It uses Hibernate to persist the states of the processes into a database.

How long has the project been around?

The project was started in 2003. Currently they are on version 4.3 of the system. There is a large community of users and companies that are using the product. This means that many people have presented, or written about their experiences with the product. jBPM is a mature, well supported open source tool for us to use.

How does it compare to what currently have implemented?

Use Case	jBPM Supports	Homegrown Supports
System moves workflow to next step.	X	
External event (human interaction, etc) moves workflow to next step.	X	
Engine assigns step to user group.	X	
Engine assigns step to specific user.	X	
Engine delegates assigning to custom handler class.	X	
Supports instance of process per business entity.	X	
Parallel steps with a process.	X	
Have others create processes.	X	X
Support for a decision based task.	X	
Built in task management system	X	
Supports custom actions on transitions.	X	?
Supports time based actions (after x time do transition)	X	
Monitoring service to monitor process states	X	
Historical duration of entities in certain states.	X	
Historical log of each transition per entity	X	?
Use exisiting Spring Beans within workflow.	X	?
Use SystemBeans	?	X
Assign a workflow process instance to a table.	X	X
Supports versioning of process definitions	X	
Can pass variables to a workflow process instance	X	
Different role can only do certain parts of workflow (i.e. swimlanes)	X	
Supports sub-processes within a process	X	

How would we integrate it with our system?

To start the process engine, we use a helper factory class and hook it into Spring.

```
<bean id="springHelper" class="com.clifton.workflow.JbpmSpringHelper" >
        <property name="jbpmCfg"><value>META-INF/jbpm.cfg.xml</value></property>
</bean>

 <bean id="processEngine" factory-bean="springHelper" factory-method="createProcessEngine" />
```

Then we inject it into our service beans to wrap the process engine.

```
private ProcessEngine processEngine;
```

All services can be accessed from this ProcessEngine. It is Thread-safe and can be shared amongst all of our services.

Why should we use it?

Given the maturity of the product, its simple integration, and its rich feature set, using this tool will help us by saving time implementing the features ourselves. We can leverage this tool and start implementing complex business processes. There is work necessary to fully integrate it, but it is less work then would be necessary to implement all the use cases ourselves.