## 纵观jBPM：从jBPM3到jBPM5以及Activiti5

对[jBPM](http://www.jboss.org/jbpm)来说，今年最大的事件莫过于jBPM的创建者[Tom Baeyens](http://processdevelopments.blogspot.com/)离开JBoss了。Tom Baeyens离开的具体原因尚不清楚，但他的离开产生了两个结果：一是jBPM的下一个版本jBPM5完全放弃了jBPM4的基础代码，基于[Drools Flow](http://www.jboss.org/drools)重头来过；二是Tom Baeyens加入[Alfresco](http://www.alfresco.com/)后很快推出了新的基于jBPM4的开源工作流系统[Activiti](http://activiti.org/)。由此不难推测Tom Baeyens离开的部分原因：JBoss内部对jBPM未来版本的架构实现产生了严重的意见分歧。更加巧合的是12月1日Activiti5刚发布，紧接着12月2日jBPM5就发布了第一个候选发布版本，jBPM与Activiti之间的微妙关系可见一般。

在这篇文章里，我们将一起回顾jBPM从jBPM3到jBPM5以及Activiti5的发展历程，我们可以清晰的看见jBPM（包括Activiti）设计所遵循的一致原则：强调流程服务的可嵌入性和可扩展性。同时，从各个版本之间的变化我们也能看见产品设计思路的变化：更加强调面向业务人员，增加BPMS（业务流程管理系统）特性。

在回顾之前，我们首先讨论一下BPMS应该嵌入还是独立部署的问题，因为不管是jBPM还是Activiti，都强调了流程服务的可嵌入性。此外，我们还需要讨论一下什么是BPMS的特性，它们所解决的问题是什么。

相关厂商内容

[**Windows Azure在中国正式开启在线直付**](http://www.infoq.com/infoq/url.action?i=5996&t=f)

[**率先在国内落地的全球化云服务，深度了解Windows Azure**](http://www.infoq.com/infoq/url.action?i=5997&t=f)

[**利用开发测试云实现精益创业、快速迭代**](http://www.infoq.com/infoq/url.action?i=5998&t=f)

[**Windows Azure自管理服务实现近乎零停机维护**](http://www.infoq.com/infoq/url.action?i=5999&t=f)

[**Windows Azure专区上线，全面了解云服务**](http://www.infoq.com/infoq/url.action?i=6000&t=f)

相关赞助商

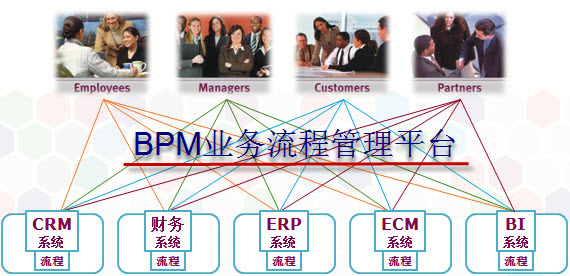
http://cdn3.infoqstatic.com/resource/sponsorship/featuredcategory/6820/100x50.jpg

Windows Azure专区上线，全面了解云服务[精彩呈现](http://www.infoq.com/infoq/url.action?i=6001&t=f" \t "_blank)！

**一、嵌入式还是独立部署？**

不管是jBPM还是Activiti，都强调了流程服务的可嵌入性。Tom Baeyens在其个人博客里称作为独立部署的BPMS已死，原因有两个：一是独立部署的BPMS需要很高的安装使用成本，需要独立部署、需要用户支出大量的培训成本和维护成本；二是独立部署的BPMS与外部系统的交互方式是分布式，这使得很多问题变得复杂，例如分布式事务。Tom Baeyens代表了相当一部分人特别是开发人员的观点。

Tom Baeyens没有完全理解BPMS。什么是BPMS？BPMS最重要的目标就是需要打破各个应用系统（CRM、ECM、ERP、SCM）之间的界线，将分散在这些系统中的流程集中管理，这是BPMS的实质。一如流程再造，打破各个部门之间的壁垒，减少浪费，建立流程驱动性的组织。如下图1所示：



**图 1：BPMS打破应用系统之间的界线**

BPMS所要解决的问题要求其必然是独立部署的。Tom Baeyens错误的根本原因在于其将BPMS与工作流系统的定义混为了一谈，他如此定义BPMS：BPMS旨在简化对组织核心流程进行支撑的软件创建。也就是BPMS面向的是软件开发人员，旨在简化他们的开发，降低他们使用流程的门槛。而这正是工作流系统需要解决的问题。

BPMS面向企业用户，工作流面向开发社区和系统集成商。

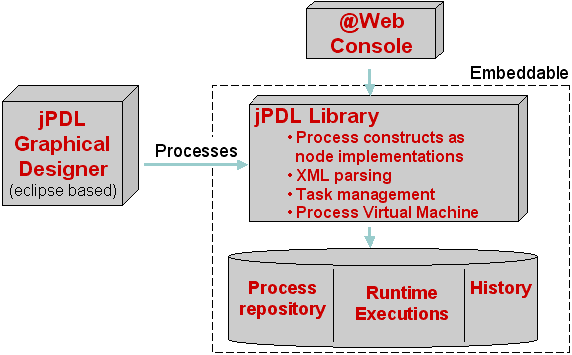
**二、BPMS特性**

jBPM4、jBPM5和Activiti5都增加了其BPMS特性，那些特性能够称为BPMS特性呢？我们先看一看BPMS需要解决的问题，为解决这些问题所增加的特性就是BPMS特性。

1. **如何设计流程，在组织中高效地对设计出的流程进行沟通，取得共识？**
   * 提供跨越组织的流程标准标记符号与术语（BPMN已经成为标准）
   * 流程及相关文档的可视化（流程/内容存储仓库）
   * 提供在组织结构内进行不同层次之间的流程导航（流程存储仓库支持组织模型）
   * 流程定义在各个层次/部门间的一致性，避免业务人员的流程建模转换到IT系统时受到损耗（流程引擎支持基于图的建模，支持扩展）
2. **如何更好地执行流程？**
   * 业务活动的实时监控，预警与控制（BAM）
   * 流程执行的仿真
   * 流程执行的统计分析与反馈（报表）
3. **如何更好地管理流程？**
   * 打破各个应用系统之间的界线，统一管理所有流程（EAI，与ESB的集成）
   * 对业务人员友好的建模工具
4. **如何在执行流程过程中遵循业内最佳实践和规则？**
   * 面向流程的知识管理
   * 规则引擎

**三、完整的工作流实现jBPM3**

jBPM3的最新版本是3.2.7，其包括了以下组件：基于Eclipse的流程设计器、用于监控案例（流程实例）和处理任务的Web控制台以及jPDL核心库。如下图2所示：



**图 2：jBPM3组件**

1. **基于Eclipse的流程设计器**

提供给开发人员绘制jPDL流程图，因为该设计器基于Eclipse，所以生成的流程文件可以与开发代码一起组织管理，非常容易进行单元测试。实现了[工作流管理系统参考模型](http://www.wfmc.org/standards/docs/tc003v11.pdf)里的接口1。

1. **Web管理控制台**

主要有两个功能：一是作为工作流客户端应用接口，给用户提供一种手段，以处理案例运行过程中需要人工处理的任务；二是对案例的状态进行监控与管理。实现了工作流管理系统参考模型里的接口2和5。

1. **jPDL核心库**

jPDL核心库是一个单独的JAR包，可以嵌入到目标应用中执行，它包括了：

* + 流程仓库：解析jPDL流程定义文件并存储读取；
  + 流程引擎：对流程定义进行初始化和调度执行，节点的运行期行为与jPDL里定义的节点类型一一绑定；
  + 任务管理：生成任务节点所对应的工作项，管理工作项的生命周期（初始化、分配执行者、执行、挂起、结束、终止）；
  + 事件管理：发布案例和任务的开始、结束事件，通过监听者模式调用相应的事件处理器；
  + 异步执行机制：通过线程实现了Job Executor，进行异步工作的处理，这些工作包括了时间处理、异步动作。
  + 身份组件模型：实现了一套简单的身份组件模型，包括了组、用户和权限。

通过调用自定义Java代码实现了对外部应用的调用，从而实现工作流管理系统参考模型里的接口3。

jBPM3是一个轻量级的嵌入式工作流系统。它在Java社区的成功得益于两个方面：一是嵌入式，这降低了使用工作流的门槛；二是对开发人员友好，这表现在易读的jPDL、流程的可测试性(Eclipse插件)以及节点行为的可扩展性，我们可以非常容易的在流程运行中加入自己定制的行为（通过事件处理器和Action）。jBPM3面向开发人员，它解决的问题是流程的自动化，它的影响力集中在Java开发社区，是一个完整的工作流系统实现。

**四、向BPMS努力的jBPM4**

与jBPM3相比，jBPM4最大的变化是引入了流程虚拟机（PVM），同时增加了BPMS的特性。jBPM4不再满足于工作流系统的定位，开始向BPMS努力。

1. **为什么引入流程虚拟机**

尽管jBPM3在Java社区取得了很大的成功，但是有一件事始终被人们诟病，那就是它不支持流程语言规范，从最开始的XPDL、BPEL到后来的BPMN，它采用了自定义的jPDL。在jBPM3中，节点的运行期行为与jPDL里定义的节点类型是一一绑定的，这造成了流程引擎与特定流程语言的绑定，要支持其他的流程语言变得困难。于是在jBPM4中，jBPM提出了流程虚拟机的概念，即流程引擎与流程语言解耦，通过一套通用的流程模型并配以可定制的节点运行期行为实现了对多流程语言的支持。

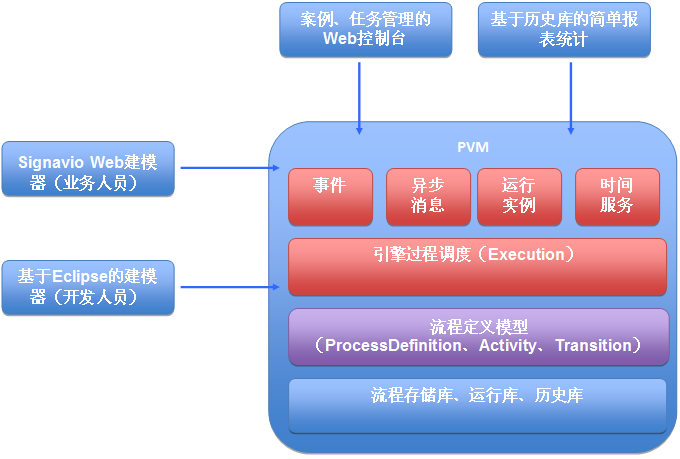
流程虚拟机带来的好处是多方面的：第一也是最重要的是jBPM4支持了BPMN。

第二是实现了基于流程组件的流程引擎，流程图(语言)与实现解耦，我们使用通用编程语言实现节点运行期行为，称之为流程组件，通过将流程图与流程组件挂接，避免了图的损耗。在这一点上，Tom Baeyens对BPMN到BPEL的转换提出了一针见血的批评：BPMN和jPDL以及XPDL都是基于图的，而BPEL是基于块的，这造成了当将业务人员使用BPMN所建立的流程模型向BPEL执行模型进行转换时，出现许多的不匹配，最初的流程模型会扭曲变形。而扭曲的后果就是业务人员与开发人员之间的协作困难，这影响了流程从业务到技术的实现。

第三个好处是我们可以定义领域特定语言（DSL），在特定的应用里，采用DSL约定并隐藏了大部分的技术细节可能做到业务人员对执行流程的直接修改，例如企业文档管理里的审批流程。

1. **BPMS特性的加入**

这表现在以下三个方面：第一是支持了BPMN，BPMN已经成为业务人员的流程建模标准；第二是引入了Signavio作为面向业务人员的Web建模器；第三是在已有的Web管理控制台加入了对案例和任务的统计功能。jBPM4的组件如下图3所示：



**图3：jBPM4组件**

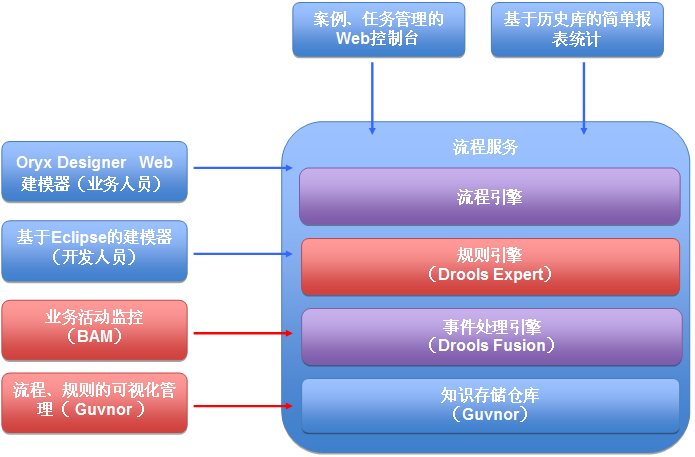
和jBPM3一样，jBPM4依然是轻量级的、可嵌入的工作流系统。相比jBPM3，它将业务人员作为最终用户之一，增加了部分BPMS特性，同时PVM的引入使得它的可扩展性得到了极大的增强，我们甚至可以定义自己的DSL。

在BPMS特性里我们提到了应该避免业务人员的流程建模转换到IT系统时受到损耗，最理想的情况是业务人员与开发人员共用一个流程模型，业务人员能够直接对流程进行调整（在特定应用中，通过DSL是可以做到的）；其次是通过BPMS将业务人员的模型与实际执行的技术模型关联起来（很多商业产品已经做到了这一点，在Activiti5中我们也会看到这一点），业务人员、开发人员以及运营团队之间能够做到很好的协调；最差是业务人员与开发人员各自为政，独立维护各自的流程模型，并且模型之间存在极大的不匹配，此时流程的迅速变化基本上是奢望。

**五、鸠占鹊巢的Drools Flow与jBPM5**

目前jBPM5刚刚发布了第一个候选发布版本，jBPM5基本上完全抛弃了jBPM4的代码，所有代码全部来自原先的Drools Flow。Drools Flow最初被用来解决规则执行顺序的问题。其实从Drools Flow开始支持BPMN时起，我们已经预感到它与jBPM的竞争关系。

jBPM5依旧定位为轻量级的可嵌入的工作流系统。在jBPM5的特性里，有这么两条引人关注：一是引入了Guvnor作为流程仓库，这解决了流程的可视化问题，流程定义作为资源被管理，我们可以对流程定义进行可视化管理以及全文检索（Guvnor使用了Jackrabbit作为了其存储实现，但我们的经验表明Jackrabbit在大数据量情况下性能存在严重问题）；第二是规则引擎(Drools Expert)、事件处理引擎(Drools Fusion)与流程引擎的合三为一，这是jBPM5最让人期待的地方。jBPM5的组件如下图4所示：



**图 4：jBPM5组件**

规则引擎在流程中的应用已经非常广泛了，我们这里说说事件处理引擎。

事件处理引擎是业务活动监控（BAM）的基础，BAM的功能及执行过程，如下：

* 捕获：BAM捕获各种事件（通过消息监听器、适配器、代理等）。这些事件来自应用、系统软件、外部交易伙伴。消息是BAM的核心——它们反应底层业务流程的状况。
* 过滤：BAM过滤掉没有直接后果的事件，在很多情况下由支持事件流处理（Event Stream Processing，简称ESP）或复杂事件处理（Complex Event Processing，简称CEP）引擎来进行过滤。
* 分析：BAM根据分析模型和规则将相关事件联系起来。
* 警告：BAM向用户提出警告，以便用户在必要时进行控制。

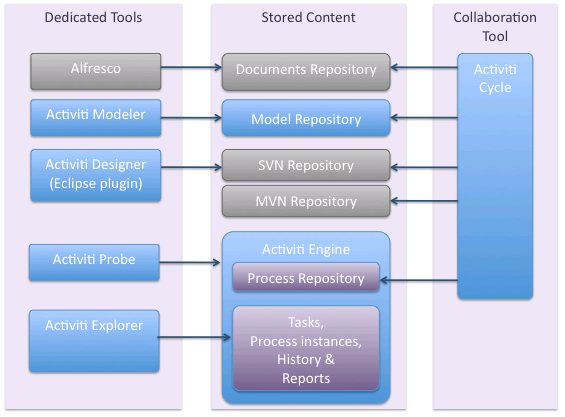
如上所示，BAM的执行过程包含四个步骤，而前三个步骤都是对事件进行相关的处理（捕获事件、过滤事件、分析事件、关联事件），因此在大多数BAM的技术实现方案中，都基于CEP和ESP的引擎来实现BAM的功能。

与jBPM4相比，jBPM5对PVM的放弃也带来了几个不小的问题：第一是对开发人员来说只支持BPMN，不再支持jPDL（当然提供了迁移工具）；第二是流程执行的可扩展性回到了jBPM3的年代，仅仅支持自定义动作（相当于jBPM3里的Action）。此外，Web建模器由Signavio替换为了Oryx Designer。

总而言之，jBPM5通过引入流程仓库和BAM继续向BPMS迈进（目前的进展是与流程仓库的集成还未完成，BAM基于日志进行分析），同时，由于不再支持PVM和jPDL，带来了流程扩展性的降低和社区开发人员的未来流失。

**六、Activiti5的反击**

Activiti5是Tom Baeyens加入Alfresco后推出的新的基于jBPM4的开源工作流系统，1号刚刚发布第一个版本。Activiti的开发团队相比与jBPM强大了许多，有23位核心开发者。当然这也是由于activiti规划的功能所致：包括核心引擎、Web的流程建模器、协作工具Activiti Cycle、Activiti Probe、Activiti Explorer、与Spring的集成、与Mule的集成等。



**图 5：Activiti5的组件**

如上图所示，Activiti5由三种类型的组件组成，分别是：专用工具（Dedicated Tools）、内容存储工具（Stored Content）和协作工具（Collaboration Tool）。

专用工具包括以下：

* Alfresco—Alfresco公司的企业级内容管理产品

Alfresco 是一个开源的、企业级的内容管理系统，功能包括：文档管理、协作、记录管理、知识库管理、Web内容管理等功能。Alfresco与Activiti的深入集成实现了流程及相关文档的可视化。更重要的是Alfresco支持组织模型，能够提供在组织结构内进行不同层次之间的流程导航。

* Activiti Modeler—建模器

基于开源Signavio Web流程编辑器的一个定制版本，提供了对BPMN2.0图形化规范的支持，建模后的流程以文件格式进行存储。

* Activiti Designer—Eclipse插件形式的建模器
* Activiti probe—管理及监控组件

对流程引擎运行期实例提供管理及监控的Web控制台。包含部署的管理、流程定义的管理、数据库表的检视、日志查看、事务的平均执行时间、失败多次的工作等功能。

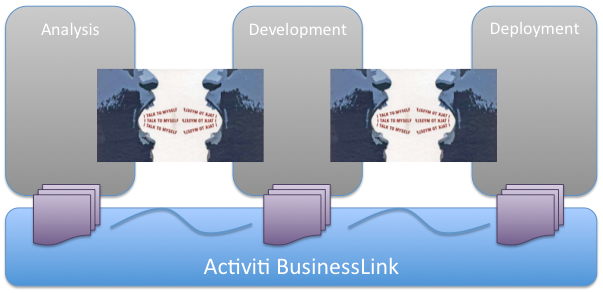
* Activiti Explorer—任务管理组件

提供任务管理功能和对案例、任务基于历史数据的统计分析（报表）功能。Web应用程序。

内容存储工具：包括了文档仓库、模型仓库、SVN仓库、MVN仓库和Activiti引擎。其中文档仓库、SVN仓库和MVN仓库三个组件为协作工具（Activiti Cycle）提供底层的支撑。Activiti引擎则是以前的PVM。

协作工具：与jBPM4相比，Activiti5最令人瞩目的特性就在于它的协作工具组件。

Activiti Cycle完全是一种新类型的BPM组件。它是一个用来促进业务人员、开发人员和IT运营人员协作的Web应用程序。 在现实的场景中，业务文档有业务人员所持有，而软件程序由开发团队所管理，被部署的软件应用则被IT管理人员所管理。三者之间不能很好的协作。我们可以想象这样一个场景，业务经理用文档来维护需求和visio格式的流程图，开发人员管理可执行的流程和大量的Java源文件而IT维护人员则管理部署在Tomcat中的.war文件和存储在Activiti数据库中的流程。



**图 6：Activiti cycle协作组件逻辑示意图**

Activiti Cycle通过BusinessLink将与流程相关的业务人员、开发团队与IT维护人员关联起来，实现他们之间的协作。

总而言之，与jBPM4相比，Activiti5目前最重要的增强就是实现了流程的可视化以及创新的Activiti Cycle协作组件，此外，通过与Mule的集成加强了其集成能力。其对PVM的保留使其继承了jBPM4强大的可扩展能力，对jBPM的老用户来说，这是向其迁移的重要理由。

**七、总结**

jBPM3是一个完整的工作流系统实现，面向开发人员，目的在于简化对组织核心流程进行支撑的软件创建，不支持标准。

jBPM4引入PVM，使其拥有更强大的扩展性，同时增加BPMS特性，这些特性包括了对BPMN的支持、面向业务人员的Web建模器和简单统计分析功能的加入。

jBPM5基于原先的Drools Flow，支持BPMN，通过与Drools的合并支持BAM，通过内容仓库增加对流程可视化的支持。由于放弃了jBPM4的PVM，引擎的可扩展性受到损害，并且不再支持jPDL。

Activiti5基于jBPM4，与Alfresco的集成增加了其流程可视化与管理能力，同时通过创新的Activiti Cycle协作组件支持流程相关人员之间的协调，最后，它加强了集成能力。

对于工作流应用或者jBPM3、jBPM4的老用户，建议转向Activiti5。

**关于作者**

荣浩，ThoughtWorks咨询师，关注敏捷和企业流程改进过程，目前正与辛鹏合著《Head First Process-深入浅出IT流程》一书。博客地址[http://ronghao.javaeye.com](http://ronghao.javaeye.com/)。