Business Process Management In An Application Development Environment

Overview

Today, many core business processes are embedded within applications, such that it’s no longer possible to make changes to business processes purely through a manual change in procedures; instead, those applications must be able to change as frequently as business processes might change. Business processes change not just to improve efficiency, but to stay ahead of what the competition is doing: agility, or being able to change business processes quickly, is a key competitive differentiator in many industries.

Business process management systems (BPMS) can help accelerate the lifecycle of process application development, through model-driven development and many other capabilities such as analytics that are built in to the BPMS. This paper is about how BPMS work in the context of traditional application development environments.

BPM Defined

Business process management (BPM) has two components. First, it’s a management discipline for improving business performance, primarily through considering business processes as assets or capabilities. Second, it’s a set of
methods and tools used to optimize business processes. This includes process optimization methodologies such as Lean and Six Sigma, and technology tools ranging from process modeling to monitoring and analytics.

A BPMS provides all of the tools to create process-based applications: process discovery, usually done by the business; process and application design, a collaborative effort between business and IT; a build stage to customize and integrate processes to other applications, then an execution environment to run the process application, automating tasks and delivering work to process participants. Statistics are gathered from the executing process, analyzed, and fed into optimization and discovery of new or improved processes.

A full BPMS typically includes the following capabilities:

- **Model-driven process development**, where graphical models are transformed into executable applications. This always includes process models, but may also include data, organizational and other model types.
- **Execution engine** to run the process application.
- **Orchestration and integration**, leveraging service-oriented architecture (SOA) and other interfaces to piece together disparate systems into a single
orchestrated business process, passing data and control between them during execution of the process.

- **User interface (UI) and composite application development** for assembling applications and portals.
- **Social and collaboration** to allow users to collaborate on tasks within a process.
- **Work management** to control the work flowing to each user based on skills and roles, and allow business managers to modify work assignments.
- **Monitoring and analytics** for visibility into processes as they’re executing.
- **Mining, simulation and optimization** to find bottlenecks and other problems in the process, and suggest modifications to improve the process model.
- **Business rules** to make business logic explicit and external from the process applications for reusability.
- **Application templates** for vertical applications, such as a telecom or healthcare framework that includes pre-defined processes, rules, dashboards and user screens that need only to be configured for use.
- **Process asset repository** for versioning and reuse of processes, rule sets, UI, dashboards, reports and other assets. These repositories often enable collaboration and sharing in addition to source code control.

This is the range of functionality that comes out of the box with many BPMS, but would have to built in code within a more traditional application development environment.
Model-Driven Application Development

In a traditional application development environment, the business process is embedded in the code, and is not explicit. This creates two key issues:

- Changes to the process require code changes, or the use of parameter tables or configuration files. Parameter-driven agility can be effective, but can be very brittle since the code and parameter tables may be modified independently. It’s also more difficult to version parameter tables, hence may not be clear which parameters were in effect for any given transaction.
- The business stakeholders can’t see an understandable representation of their process, so they can’t get involved in the design and development cycle.

Development in a model-driven environment such as a BPMS can greatly reduce coding: graphical models are created and executed directly without writing code. BPMS also include features such as work management, monitoring dashboards and analytics, all available without writing code. It’s usually not “zero code”, but development is accelerated dramatically. This makes it a good environment for rapid prototyping, and minimizes rework when moving into development.

Furthermore, since all stakeholders can understand and participate in the design of the models, it is much easier for the developers to collaborate with business users, trading partners and customers. Involving the non-technical stakeholders directly in process discovery and modeling can
dramatically increase the quality of the process application relative to the business needs. And, because a BPMS-based application can be more easily modified than code, it can be quickly adjusted to suit changing needs.

Combining a BPMS with a traditional application development environment is a matter of picking the right tool for the job: use the BPMS for creating executable business processes, including orchestrating and integrating other systems, and for creating user interfaces. For services and other complex functionality, it can be more computationally efficient to develop code in a lower-level language.

For existing applications, it may be a matter of exposing that application’s functionality as services to be orchestrated by the BPMS within the processes.

**Leveraging BPMS: Scenarios**

Adding a BPMS to application development enables a service-oriented architecture. In the diagram below, the process in the BPMS (top) – which tends to be changeable – calls services (bottom) – which tend to have fixed functionality. The services may be from an ERP system, a custom application, or even external web services; these may be loosely coupled using REST or SOAP web service standards, or more tightly coupled using an application programming interface (API).
Since the process is model-driven, it can be easily modified to call different services, or in a different order. The services, which rarely change, may be called from any number of processes, or from other applications.

There are several architectural scenarios for combining a BPMS with a more traditional application development environment.

First is the full service-oriented architecture described above: for new process-based applications, this provides the best mix of agility and functionality. The BPMS is used to develop a process-centric user interface in the BPM portal, which communicates with the BPM server for executing tasks. Much of the work may be done by the underlying services, since any step in the process may call a service.
The next scenario is the opposite: there is an existing user interface portal and application, and some of the processes within the application are moved into the more agile environment of the BPMS. In that case, the BPMS becomes a set of services called from the application to perform process management functions, although they are not exposed to the end users. The BPMS can, in turn, call other services.

The third scenario is a coexistence architecture: the existing user interface is expanded to interact with the BPMS for process management, but the existing application remains untouched. This leaves existing processes in the application, and adds new processes alongside it.

**Process Monitoring For Greater Visibility**

Process monitoring and analytics add a process-centric spin to business intelligence by providing visualizations unique to the time and task-oriented nature of business processes. This can range from simple dashboards showing aggregate information, such as a chart of how many process instances are waiting at each step, or more complex interfaces that allow a user to drill down in a specific instance, or perform
historical analysis on how well a process is performing relative to its service level agreement.

Emerging capabilities include predictive process analytics that allow the system to project forward from a point in time to predict if a specific process instance will complete on time, and take corrective action. This type of process analysis can also feed into process optimization, since it will highlight inefficiencies in processes and may even suggest improvements to the process model.

Process monitoring and analytics have been developed for processes automated in a BPMS, but most business processes aren’t in a BPMS: they’re in other applications and systems, such as packaged ERP systems and custom applications. These other applications generally don’t have robust process monitoring capabilities, but they can generate events about what’s happening in the application. To take advantage of that, some BPMS include event listeners to capture events from other applications and include them in the process monitoring along with the processes that execute within the BPMS. This, in turn, enables process improvement efforts across all of the heterogeneous systems and processes, not just for processes within the BPMS.

Cross-application process monitoring provides a broader view of business processes, which improves process management and promotes process-centric thinking across the enterprise.
Using Rules For Automation and Agility

Business rules are the business policies that define an action that will be taken at a point in a business process: a decision to be made, or a constraint applied. Some rules are implicit, buried within enterprise applications or well-worn policies and procedures; fewer are explicit, having been identified and documented as separate corporate assets.

Decisions in business processes may be performed by custom code within an application, or by a front-line worker applying them on an ad hoc basis. There are many types of business decisions buried within business processes, ranging from simple work routing to more complex scoring or strategy evaluation. These decisions – based on market conditions, legislation, policies and other factors – typically change much more frequently than the processes and applications that use them, and the ability to change those decisions independent of the applications is key to business agility.

Business rules management (BRM), like BPM, is both a discipline for the discovery and management of business rules, and the methodologies and tools used to manage the rules. BRM systems (BRMS) – also known as decision management systems – provide rules modeling, execution, monitoring and administration capabilities, and enable rule visibility and reuse across the enterprise. Rules become explicit and business-readable, similar to the transformation of business processes when moving to a BPMS.
Although most BPMS include some rules capabilities, a full-featured business rules platform will include more sophisticated rule design functions, rule harvesting from enterprise applications, versioning and simulation. A separate BRMS can share rules between processes and applications: complex rules such as credit scoring or risk assessment can be defined in the BRMS, then invoked from any application.

Moving manual business rules into a BRMS reduces errors and costs by automating decisions that do not require human decision-making, and improves policy enforcement and compliance. It also can allow rules to be changed in real-time in response to changing conditions, unlike if the rules were encoded within a process or application, allowing business managers and analysts to update rule parameters to fine-tune decisions for current conditions. This is often used for work management, such as lowering the percentage of work sent to a QA function during peak volumes.

Process instances don't access a decision service until it's required, thereby applying the most up-to-date rules that support the current business policies, not what was in effect at the time that the work was created. This is particularly important for long-running processes that may be active for weeks or months, such as insurance claims.

As with other services, decision services may be called from a process in a BPMS or other application. Since the specific rule set used can significantly change the application behavior, rules can be used to create customized versions of
an application – for a different customer or product, for example – without changing the process or application code.

It’s critical to embody more intelligence in today’s business decision-making and have consistent, automated decisioning built into business processes in order to remain agile and competitive in today’s fast-moving market.

Summary

Adding BPMS to traditional application development environments can provide significant benefits over using the application development environment alone:

- Model-driven development, a core capability of a BPMS, enables faster development, greater stakeholder involvement and improved application agility.
- Tools and features of the BPMS can be leveraged from other applications, including process/event monitoring and analytics, and business rules.

Accelerating the application development lifecycle while providing better collaboration with all stakeholders provides strong wins for both business and IT.

About the Author

Sandy Kemsley is an independent analyst and process architect specializing in business process management. She performs engagements for both end-user organizations and BPM vendors across North America, writes the popular “Column 2” blog at www.column2.com and is a featured conference speaker on BPM.