A Detailed Analysis of BPM Suites

1 Product Overview

The Product Overview sets the scene and communicates the core product features that differentiate the product from its competitors.

2 BPM Engine

We continue our analysis by describing how the tool is organized to manage business processes. Most BPM Suites are organized around client-server or multi-tier architectures. The repository of process models and the core BPM engine usually live on a server that provides infrastructure support to developers, employees, and managers who access the server via Web browsers. Traditionally, vendors provided a desktop client environment that was loaded on the PCs of employees and managers, although most now use the Web browser for this purpose. The vast majority of vendors still provide a desktop PC-based process modeling client for use by developers and business analysts.

The Server Environment

Different vendors describe their BPM engines in different ways. We also have to reflect the multiple, overlapping middleware technology products and acronyms. The term *Server* is a generic term that is used to describe either a software component that performs some specific purpose or a piece of computer hardware that provides shared resources to a range of *Client* machines. Within this study we generally mean a software component of some kind.

Most environments make use of a modern Application Server from one of the major IT vendors, such as Microsoft, BEA, or IBM, although a few have developed proprietary approaches that do not use this infrastructure. An Application Server is a software product/component that dynamically generates Web pages. Over time, the Application Server products have taken on more and more responsibility for managing other software requirements such as load-balancing and clustering support for applications.

The BPM Engine is another type of Server. The BPM Engine or BPM Server is responsible for executing, controlling, and monitoring all business processes. It orchestrates events within and across multiple processes. The BPM Engine handles employee interaction, routing work to employees, and ensuring that the work is accomplished (managing the state of the case of work). In the past, this functionality was often described as workflow management.

The BPM Engine is usually also responsible for coordination of third party applications into the process and manipulation of process-related data (which is normally stored in an enterprise-level database system such as SQL Server or Oracle).

When integrating third party applications we find a number of approaches. The BPM Engine may employ a distinct EAI software component or Integration Engine; it may reuse functionality provided by the Application Server;; or it may directly invoke the external application, using that product's Application Programming Interface (API). In some products, the core process engine is separated from another internal component that handles the integration of third party applications. A wider discussion of Integration follows later.

The BPM Engine is generally responsible for allocating resources, ensuring thread-safety, and de-allocating resources and database connections at the conclusion of a transaction (some of these tasks may be supported by Application Server). Technically, two approaches predominate to provide transactional integrity, security, scalability, redundancy, and dynamic load-balancing. Either the product relies on a scalable set of components, such as Microsoft .NET or J2EE (focused around the Application Server), or the core engine is subdivided into a wider group of interoperating proprietary

components that achieve a similar aim.

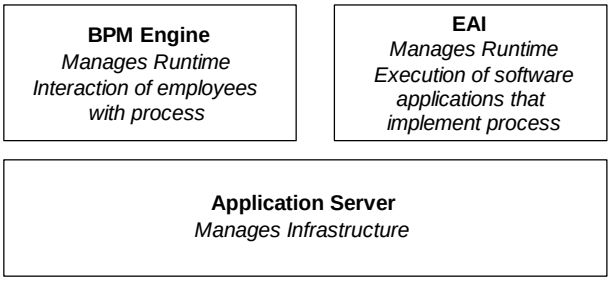

Figure 1. The Application Server may also provide the EAI support needed, or there may be a separate EAI engine

Most products rely on a direct connection between modeling tool clients and the BPM Engine that stores developed processes models (in its own proprietary repository). When BPM applications are executed, they generate large amounts of information on specific process instances or cases of work. All of this information, both modeling constructs and runtime data, is then stored using established database products such as Oracle or SQL Server.

Client Elements

Users normally access their work items through an Inbox application of some kind – either directly via a standard Web Browser (via a custom plug-in), and/or via email notifications which then take the user directly to the relevant work item via an embedded URL.

Apart from the browser-based Inbox application, most vendors provide separate user interface capabilities for systems administration and process monitoring. These are often integrated within the same environment in which users access their work, although they can exist as stand-alone tools.

Developing process models and enhancing them for the runtime environment can also be thought of as an additional user interface. We explore this aspect more thoroughly later. These process-modeling tools are sometimes accessible in a browser environment but usually require a proprietary desktop PC client.

Web Services

We are now seeing widespread adoption of Web Services. Web Service functionality usually provides support to the developer in one or more of the following ways:

- To support the integration of external applications: As long as the external application is exposed in terms of Web Services, it is fairly straightforward for process modelers to integrate the desired functionality into a process application (see Integration
- To re-use existing process models in a service oriented fashion (as an invoked subprocess) and also to similarly make the current process reusable. (See Subprocesses)
- To facilitate the development of a Shared Data Space: By introspecting and importing the WSDL of a Web Service, the process developer can rapidly develop a framework for the relevant contextual data required by the process. (See Shared Data Space)
- To allow an external application to query and report on the status of process instances.
- As an adjunct to Business Activity Monitoring functionality, where the system invokes a Web Service to alert (or invoke a process) should a metric or SLA go past/over its alert status (whether that be over time, too many work items in a queue, etc.).
- To expose the administrative capabilities of the product, allowing work to be reassigned, or

starting and stopping processes/servers.

- To allow the functionality of the BPM Engine and associated repository of process models to be accessed by (embedded within) external applications. The range of support in this area can vary widely.

Some vendors are starting to incorporate support for the Business Process Execution Language for Web Services (BPEL4WS – usually referred to as BPEL). BPEL was initially developed by BEA, IBM, and Microsoft and is currently going through a final ratification by OASIS. BPEL is an open, interpreted XML-based language that can be used to drive process execution in a BPM Engine. However, for all practical purposes, BPEL is limited to managing software applications. Without further extension, it cannot support employee interactions – although a future version of the standard probably will. [1]

At the moment, BPM Engines that rely on BPEL as the native execution language are rare. Indeed, within the products comprising our study, only IBM talks about native BPEL language support, and that discussion very quickly drops into proprietary extensions (somewhat negating the rationale for a standard execution language). What is far more common is the capability for a BPM Engine to import a BPEL definition and then translate that into its internal process definition format before executing it (using the proprietary features of that BPM Engine). Although this sounds like a subtle difference, it is important. Buyers should understand and clarify the vendor's support for BPEL and its implications for the way in which applications can then be constructed.

2.1 Platforms

Different products or elements require different software and hardware environments. Companies that rely on Java and use WebSphere or WebLogic will probably prefer a BPM Suite that is tailored for these environments, while companies using Microsoft Server will probably prefer a BPM Suite tailored to rely on Microsoft Server. A few engines run natively on both J2EE and Microsoft environments. Companies that rely on specific databases for their enterprise work will want to know which databases different BPM Suites require or support.

2.2 User Interface

The industry standard these days is a web-based browser such as Internet Explorer or Netscape. Sometimes this is supported by a plug-in for the browser which allows the user to load different views of their Inbox. This is often combined with email to deliver work notifications to the user's normal email Inbox. Each email embeds a URL that launches the browser to take the user directly to the appropriate item.

Most vendors also support their own proprietary, thick user interface clients. These tend to be used for so-called *heads-down* or power-users. Some products support the ability to embed the process Inbox within Microsoft Outlook.

Increasingly, we are finding that vendors are directly supporting Portal technology, providing standard *Portlets* that can then be embedded in any portal environment. The prevailing standard in this arena is JSR 168. This specification defines a set of APIs for Portlets and addresses standardization for preferences, user information, Portlet requests and responses, deployment packaging, and security.

Other technologies employed in this area include DHTML and XHTML support and use of the Jetspeed portal framework (which should include support for JSR 168 later this year).

[1] BPEL is somewhat constrained in other areas as well. For instance, it has no support for Transactions, Business Rules, Task Management, Mobility or Human Interactions. Despite these weaknesses, the market has *chosen* BPEL as the execution language going forward (primarily because of the support it has received from the major vendors).

Either way, out-of-the-box user interface functionality is still composed of relatively standard tabular views of work to which the individual has access (based on Role assignment or shared access to Queues). Other customizable views may include information on completed work items, or the ability to search for cases of work and drill down on their status.

2.3 Scalability

Most products rely on the *container* provided by J2EE or .NET for scalability, enabling a single logical BPM Server to be split across several machines in a cluster. At their hearts, the engines tend to be multi-threaded, with some using an advanced *stateless* process execution model ensuring that the server components do not needlessly consume machine resources. In virtually all cases, information required to track the state of a business process is persisted in the database.

Some vendors have off-loaded part of the server functionality (normally work queue management or monitoring) into discrete components that can be replicated and/or distributed. This has the effect of moving the processing requirement closer to where it is needed, leaving the central logical process engine with less work to do (and hence greater scalability). Where work queues have been distributed, the product tends to depend on a reliable messaging infrastructure (such as JMS or WebSphere MQ). In a couple of products, a separate integration engine is used to marshal resources from third party applications, removing the strain from the core logical server.

Those that have chosen to work outside of the J2EE or .NET application server infrastructure have greater control over how components are distributed across a cluster (although they then need to manage the whole environment within which the server operates). In these situations, it is often possible to have multiple BPM Engines that are configured to interoperate (running either centrally or in distributed locations). Master-slave relationships are relatively rare, although they are more common in those products that have a long pedigree. A few products have the capability to survive node failures in a cluster without impacting the capabilities of the engine.

3 Processing Modeling

This is the area where the effective capabilities of the product are most apparent – it is where the *semantics* of the vendor's interpretation of process are made visible. A key issue here is the degree of accessibility of the environment and the amount of IT specialist support required. Looking at the user interface of the process modeling application often gives a good idea of what is delivered out-of-the-box (rather than requiring the development of suitable programs).

Virtually all products claim that end-users can develop their own processes, but this is usually just a first step. What is normally meant is that business analysts or expert end-users can develop procedural definitions – the *Activities* of the process, the *Order* in which they are carried out, and the *Roles* to which work is assigned. Typically, process models are then tossed over the fence to IT developers who extend and further develop the process description, making it ready for deployment by suitably authorized personnel.

In some products, the business analyst and expert end-user are provided with a distinct process modeling environment, allowing the vendor to simplify the user interface for this group of users, yet still deliver rich functionality to meet the needs of IT. In such scenarios, developers then build on these relatively simplistic process models, converting them into deployable solutions integrated with LOB databases, electronic forms, business rules, other processes, third party applications, etc. This sort of functionality is normally beyond the capabilities (or interests) of the end-user population.

However, this points to a critical issue for BPM environments generally – balancing the sophistication and capabilities of the process support environment with user accessibility and control over processes. There is a trade-off here. Some vendors have chosen to leverage widely used modeling tools such as Microsoft Visio to alleviate the problem.

Either way, the greater the involvement of specialist developers (IT specialists), the more time is required before any desired changes are introduced. As a general rule, the more development end-users do for themselves, the lower the total cost of ownership. On the other hand, greater care and control can be exercised over changes to the system by specialist IT developers. Certainly it is a good idea to ensure IT involvement in the design of the overall process architecture (how all the processes fit together).

One of the key challenges is the integration of third party applications. While this is explored more extensively in a later section of this report, integration capabilities are worth noting in the context of a discussion on end-user accessibility to process modeling. To get around this problem, the expert developer is often given the capability to develop custom *tasks* (often called *activities* or nodes) and place these on the modeling palette for use by the business analyst. These tasks are already *pre-integrated* with the surrounding technology infrastructure (facilitating reuse of existing IT assets).

And then there is the question of access. Who should see and edit individual process descriptions? There is normally a fair sized team of people working on a project. Again we see a range of capabilities delivered in the modeling and system building environments:

- In the majority of products, users deal with each model individually. There is no prospect of two or more users interacting with the model at the same time.
- Another group of vendors have implemented a proprietary, multi-user repository that facilitates the access of several users to the model at the same time. There is a downside to this level of sophistication.; It usually means that it is impossible to work on the model unless one is connected to the central repository. Some get around this through a check-in/check-out approach to the management of process models in the repository. Security is provided either on a named user basis within the repository, or via a role-based mechanism implemented via a Directory Server. (See Organizational Structure)
- Other vendors support the situation where the repository only covers one project (where a project encompasses a series of models associated with a given solution). This approach supports the situation where an individual may want to take responsibility for the project in the early stages, yet still enable multi-user access at a later date.
- Another approach to this problem is to implement a bridge to enterprise oriented modeling repository tools. This allows multiple users access to process models in the third party environment (where the modeling repository manages all access to individual objects). Initial implementations tend to be uni-directional, where the BPM Engine imports process information from the third party repository. Problems can then arise in making those models ready for deployment. A lot of rework can be required when small changes in the process must be embellished with proprietary extensions required by the BPM Engine for effective operation. A two-way bridge normally removes the need for this sort of rework, with tighter integration between the modeling repository and the BPM engine.

Either way, once a process model is deemed complete, it is published to the BPM Server. Of course, some vendors support more than one of the above approaches.

And then there is the problem of complexity and completeness. In the past, products provided little or no support for the firm in trying to spot and resolve errors that may creep into process models. These can include business rule conflicts, collisions, gaps, overlaps, inconsistencies, or redundant processes. However, it is now quite common to see validation and testing functionality built into the modeling environment (often referred to as a *pre-production simulation* capability).

In the past, virtually every product had a proprietary modeling notation. The semantics of their environment was thus inextricably linked to their modeling tool. With the advent of an industry

standard process-modeling notation, the Business Process Modeling Notation (BPMN) from BPMI (www.bpmi.org), we now see vendors starting to adopt this standard graphical notation. Indeed, a few of the vendors we have looked at within this report have fully BPMN compliant modeling user interfaces. Most vendors who have adopted BPMN have yet to release versions that support *Compensating Transactions* and role assignment through the use of *Pools* and *Lanes*. Over time, we expect to see more vendors adopt this approach.

3.1 Subprocesses

Most products incorporate the ability to handle subprocedures as part of the overall process model. Indeed, it is around subprocesses that vendors tend to incorporate opportunities for flexibility in process architectures. Precisely how this is done and the resulting process architectures that are possible vary a great deal.[2] Again, we find a range of approaches:

- The simplest level is really little more than a graphical construct to simplify process models (often called *Submaps*). Effectively, the subprocess is embedded in the parent process. When a case of work following the parent model arrives at the Submap, control is handed to the subprocess to complete before control is handed back again. Sometimes, the product allows a Submap to be called from a variety of places within the parent process (further simplifying the construction of the parent).
- Another approach is to develop a library of subprocesses, which are then copied into the parent process at design time. Effectively, the end result is the same as the situation above, although reuse is facilitated through the library function.
- A more flexible approach is to *call* the subprocess at runtime, with only the reference to the specific subprocess linked in at design time.
- The most flexible approach is to allow the call to the subprocess to be decided at runtime – either programmatically by the engine, based on some business rule, and/or by humans involved in the process exercising their judgment on what is required.[3]

Calling subprocesses as stand-alone *Process Objects* has both advantages and disadvantages. On the one hand, it allows loose coupling of processes. Individual process fragments are developed independently of each other, allowing much more flexible process architectures. Individual processes are insulated from the functionality of other application areas. Think of such Process Objects as Services in a Service Oriented Architecture. (See Figure 2.) On the other hand, it means that more work is normally required up front to ensure that the Process Objects product is robust and capable of operating in a variety of different situations (the developer does not necessarily know to what future uses the process might be put). To get around this issue, some products support both embedded subprocesses (where the subprocess is copied into the parent at design time) and the calling of stand-alone subprocesses (subprocesses as Process Objects).

Further sophistication is possible when one considers how the subprocess is called. In some situations, the usage scenario requires that the parent process wait for completion of the fragment (we sometimes refer to this synchronous behavior as *triggering* the process). Alternatively, the parent is not concerned with what happens to the child (we often use the term *spawn* to describe this asynchronous behavior). It kicks off the child process and carries on.

---

[2] We often use the term *process fragment* to describe a piece of process model that can be called as a subprocess. Think of a process fragment as a sort of process *object* that has a life of its own (i.e., it must be maintained and controlled almost in the same way as an entire process). The key difference is that a process fragment is instantiated by another process.

[3] In many sophisticated environments, this sort of decision is farmed out to a separate business rules engine that effectively decides which subprocess to bind into the parent at runtime.

Furthermore, it may be necessary to instantiate the subprocess a given number of times. For example, if the parent process were collecting budget information from a variety of corporate subsidiaries, the parent process would probably spawn a separate subprocess for each. The parent might then wait while each subsidiary formulated and posted their responses (which might, in turn, instantiate further processes in each subsidiary). At a later point, the parent would want to re-synchronize with each of its spawned process fragments, incorporating the returned figures into a corporate budget. Using the trigger functionality in this situation would mean that if one subsidiary did not post its response, then the whole process would grind to a halt (which may not have been a desirable outcome).

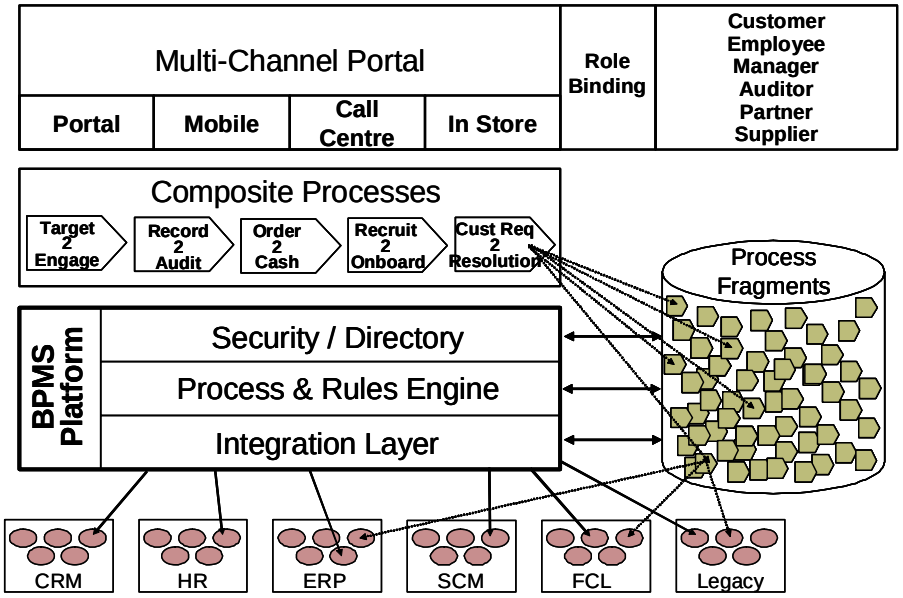

Figure 2. Composite processes are best defined in terms of constituent process objects/fragments that in turn may integrate 3rd party applications.

Further sophistication is possible through the use of *arrays* as process variables in the Shared Data Space (see next subsection). In this situation, an array is populated with information relating, for instance, to each subsidiary. A different subprocess could then fire for each subsidiary. This method would be effective in handling the needs of,, say a marketing campaign. Overall process support for the campaign is required, integrating cleanly with different processes for handling each customer who responds (each with potentially different requirements).

Subprocess constructs can also be used to implement goal-seeking behavior in the process. When the parent process arrives at the point where the subprocess is defined, it evaluates the needs of the process against the available Process Objects, instantiating those that meet the criteria of the parent. This sort of functionality is usually found in separate Business Rules Engines, but is sometimes found in the BPM Engine. (See the section on Business Rules for more information.)

When we look at how a Process Object is called, at the technical level, this has traditionally been done in a proprietary fashion (i.e., each vendor has invented its own approach, based on a database reference or directly invoking byte code). With the advent of Web Services, this mechanism is now becoming standardized (indeed, we would be surprised if all vendors do not quickly support this method). Assuming that the vendor has gone down the Web Service route, a process can be published as Web Service and can also consume Web Services. This allows any process to call a Web Service enabled subprocess, either by programmatic conditional action or through human decision at any step in any process. Moreover, a process running on one BPM Engine could make use of process functionality running on another.

Thinking about subprocesses, one also needs to keep in mind how Line of Business (LOB) data is managed between the parent and the child. A few products provide specific facilities to ensure that these are consistent; some provide validation mechanisms, others support a mapping facility where the elements of the parent are matched to the capabilities of the child. Another approach is to base all subprocesses on specific templates that already have a Shared Data Space defined. Most leave it to the developer to spot potential conflicts.

3.2 Shared Data Space

When looking at the product's capabilities with regard to managing information, one needs to remember that most process models integrate third party applications and capture information associated with the case.

All products provide facilities to create variables of some kind and associate them with the process model. These variables are really placeholders that are populated with information as the case moves through the process model (either programmatically from third party applications or via direct user entry). This information is then available to support automatic decision-making, or is presented to users in electronic forms, or saved back to a LOB application. Alongside these discrete elements of data, there may also be documents of virtually any sort.

Collectively, we refer to this capability as the *Shared Data Space* (SDS) of the process model (our term). This is all about how the BPM engine manages *context*. Data gathered at one step can be reused at another. As far as the case of work is concerned, related information is available (stored in the supporting RDBMS, rather than extracted every time it is needed), allowing data to be pushed or pulled to and from third party applications. It is worth noting that the information itself probably belongs to (is persisted in) some third party application and must be flushed back to the correct transaction based system.

While virtually all modern BPM products provide a Shared Data Space of some kind, often this is limited to the discrete variables associated with the case. However, some offer more extensive capabilities associated with how the product handles other types of content. Here we are referring to the letters, attachments, and other documents. Indeed, the ability to manage processes alongside the content to which it refers is at the heart of some of these products.

As we noted earlier, a key capability to watch out for in this area is the ability to support an array concept. Arrays provide an important construct necessary when designing more flexible process architectures. Arrays can be supported in various ways, either as a direct variable type, a pointer to a more complex object, or even through a spreadsheet.

Another feature that has started to emerge (with the more modern BPM Engines) is the capability to quickly construct the required SDS structure by reusing the data structure of either an associated Web Service or an existing Form of some kind (HTML, InfoPath, PDF, ASP, JSP, etc.). Indeed, if the application is to integrate a third party application, then it can often make sense to first wrap that functionality within a Web Service and import the WSDL structure for the process. Some products also provide the capability to selectively map the WSDL of a Web Service or Form structure to the existing SDS (using a graphical tool).

3.3 Forms

In most products, electronic forms are used to present and gather information (handled by the Shared Data Space and any underpinning LOB applications). Some vendors provide proprietary thick client forms environments, although most now employ browser-based approaches. Often we find that a default form generated by the system can then be adapted and modified in the form-building environment of choice.

Depending on the overall architectural orientation of the vendor, you may find support for a range of

form types including JSP, ASP.NET, HTML, PDF, InfoPath, ActiveX, etc. Some of these products also allow for drag and drop creation of forms and the graphical linking to process variables. Accessible functionality in the modeling environment is a key enabler toward lowering the cost of ownership of developed applications.

3.4 Time

BPM Suites vary quite a bit in their handling of time. Usually it is perceived in terms of how much time a Step or Activity has to reach completion before some escalation or error condition is activated. Escalation is normally handled via email notification (to a specified role) or by placing the work item in some special queue visible to the supervisor. Some vendors include several levels of escalation and/or the ability to manage specific server side error conditions.

However, one could think about other requirements for time handling. When a step is instantiated, is this time value set relative to the time that the case of work was instantiated, or when the last task was completed, or in relation to some fixed (target) end date or other date related variable? From a business rules perspective, one might also have time related conditions based on a fixed point (e.g., collect all timesheets from employees on Friday afternoons, start the production run only if goods can be delivered by Saturday between 10 and 11 am).

A few products support highly distributed business operations with all time references calculated back to Greenwich Mean Time (GMT) – also known as Universal Time Clock (UTC). This enables engines and workers in different time zones to see time references in their own local time.

3.5 Process Optimization & Simulation

Some products include a bundled simulation engine that allows process owners and developers to assess the impact of process changes. These facilities aim to identify potential bottlenecks, project time, and the costs of specific processes or activities, and to optimize the overall flow of the processes.

While this is an admirable aim that appeals to most managers, it is worth mentioning that simulation models can be misused. Too often we see people building deterministic models that are designed to try to determine how much money will be made as a result of a process change. Such models usually bury their underlying assumptions. It's far better to surface and examine simulation assumptions.

Another factor to keep in mind is that most products that have bundled simulation capabilities normally limit the scope to a single process model. Of course, this does not reflect the real world in which such processes would operate. In a sense, business processes compete with each other for the resources of the business. People are usually involved in a great many different processes, and it is difficult to accurately represent their switching behavior or the impact of one process peak over the performance of another. Of course managers often make similar mistakes when they seek to redesign processes in isolation.

Given these qualifications, simulation can be very useful. Furthermore, it is becoming much easier to employ simulation in support of overall process optimization efforts. In the past, constructing effective simulation models required significant skill and resources. One needed to define exactly the resources that were consumed by each activity, the hours that these (human) resources were available, and the probabilities that a case would travel down one path or another (using one of many available statistical methods). One also needed to ensure that the behavior of the simulation model reflected the real world. Most companies that used simulation found that they needed a simulation specialist or consultant to obtain useful results in a reasonable amount of time. Moreover, it took a lot of effort to gather the supporting data to validate the model.

All of that has become a lot easier now that the simulation capability is directly reusing the process models of the BPM Engine. The Engine also gathers all the necessary data to support model validation. Rather than having to guess what sort of statistical method was needed to approximate the real world,

the simulation tool can now directly reuse existing history data from cases in the system. One can now explore the impact of re-assigning resources from one part of the process to another.

Of course, this information is only relevant where the process itself has not changed fundamentally. Simulation, by itself, usually does not provide insight into ways one might improve a process. Most of the innovations that result in the optimization of processes result from people sitting down and discussing with each other what will work best, what is unnecessary, etc. No tool will automate the generation of ideas.

Note that some vendors use the word *simulation* to describe a pre-production testing and validation capability (designed for process developers). In these situations, the model is *exercised* by the BPM Engine to support the developer in spotting errors and problems with the underlying process and its integration of external applications. Some of these products also feature *what if* scenarios and other analysis tools to identify improvement opportunities, prior to deployment.

4 Business Rules

Business Rules can mean different things to different people.[4] Certainly, different vendors interpret the need for business rules quite differently and have differing objectives. Business rules within a BPM system can be used to support the automation of business decisions, to structure loose informal business practices and policies, or to apply rigorous sets of conditions that can help business analysts and domain experts express the essential requirements of the system behavior. For others, business rules are limited to a set of conditions that control specific business events. In a process, business rules are the components that implement the decision-making logic.

All BPM products allow conditional routing – where a transition between Activities/Steps carries with it a condition. If the condition evaluates to true, the process follows that path (often creating a fork in the process). Depending on the approach taken by the vendor, the condition is not placed on the transition between Activities, but is instead placed on the Activity itself (think of it as a precondition that must be met for the Activity to become instantiated).

Indeed, constructing process definitions based on such preconditions is a very powerful way of reflecting the true needs of the process, without unnecessarily inhibiting the overall pattern of work. In some situations, the process need not be as neatly ordered, as one would initially expect. Certain parts of the process could be carried out in any order, while other Activities can only be done once certain conditions are met.

In many businesses, the rules are far more complex than can be easily represented with simple conditional statements. Although it might be possible to represent all possible paths through a moderately complex process with conditional statements, the resulting process map would become almost unintelligible. As the complexity increases, it becomes more difficult to represent the necessary logic with conditional statements. As one vendor put it, "The problem with detailed process maps is that their graphical nature, while initially of high value to promote understanding, loses their value as complexity increases. Even when following best practices for mapping, the effort to follow graphical flows (and understand the details behind the pictures) is too great. Instead of promoting clarity and understanding, they increase complexity and confusion."

To get around this problem, some vendors have built sophisticated business rules features into their core product. The overall aim of business rules functionality is usually to enable business users to interact with the system through English-like rule-based statements (rather than getting too involved in all the complexity of procedural modeling tools).

[4] For the purposes of this discussion we will limit ourselves to discussing business rules within the context of BPM initiatives.

For example, a rule might be "If the customer has a good credit rating, then apply the fast track loan approval process." This might combine with "If the applicant is an existing customer and has never defaulted on a repayment, then assign a good credit rating."

One can store rules in a database and manipulate them with a database management system, but complex rule systems are usually managed by a Rule (or Inference) Engine that evaluates the rules at runtime. Briefly then, we can describe the ability of a product to manage rules in terms of three possibilities, each able to handle more complex logic:

- Constraint-based systems.
- Rule systems that manipulate a rule set with a database management system.
- Inference-based systems that can evaluate rules at runtime.

Independent of the technology used to evaluate the logic of a set of constraints or rules is the interface that a developer uses to create and evaluate rule sets. Most developers find it easier to enter rules via a spreadsheet-like interface. More complex systems can require each rule to be entered separately via a rule editor.

From the user interface point of view, BPM Suite vendors that have developed their own business rules capabilities have normally employed a spreadsheet metaphor. Users describe the condition using Boolean logic, exceptions, events, and the values of variables attached to the process instance (when modeling, this is based on the structure of the SDS). If the condition evaluates to *true*, then the engine moves the work item to the process state described in the spreadsheet, bypassing the procedural core of the process. The same sort of functionality can also be used to bind in selected subprocesses.

Initially, most BPM vendors integrated stand-alone third party Business Rules Engines (BREs), allowing the vendor to respond to the requests for business rules functionality of potential customers. Over time, we have observed that some of these BPM vendors have since extended their own product offerings to support more fully the business rules functionality.

Given a rapidly evolving business climate and a related set of business rules, there are profound implications for audit and compliance. Indeed, the whole notion of lifecycle support for business rules development, deployment, and eventual replacement is of concern. Who should have access to designing and changing business rules?[5] While all systems require certain levels of authorization to change and adapt rules, firms employing this functionality still have to be rigorous about how the capabilities are deployed.

Business rules facilities also aim to simplify process maps, pushing the complexity into the specialist rule handling functionality. Others see the addition of business rules functionality as a way of enabling on – the-fly evolution of the process model. Instead of developing a robust process model that reflects every possible path, business rules are used to provide an incremental process development capability. Given the volatile nature of business processes, it is often difficult (if not impossible) to capture all of the necessary details prior to implementation.

Still another use of rules is in providing superior management over how work is distributed to employees. Given the raft of scandals in the financial services industry relating to mis-selling of products, regulatory bodies are insisting that only suitably qualified (trained) personnel should undertake certain tasks. In these situations, the rules must factor in the current organizational structure and the skills (and quality indices) of the employees within it. (See Figure 3.)

[5] Quite how this sort of functionality fits into a control-centric view of Sarbanes Oxley legislation is open to interpretation. Some would argue that it directly opposes the aims of such regulations, while others would maintain that, with suitable controls and safeguards, the benefits outweigh potential disadvantages.

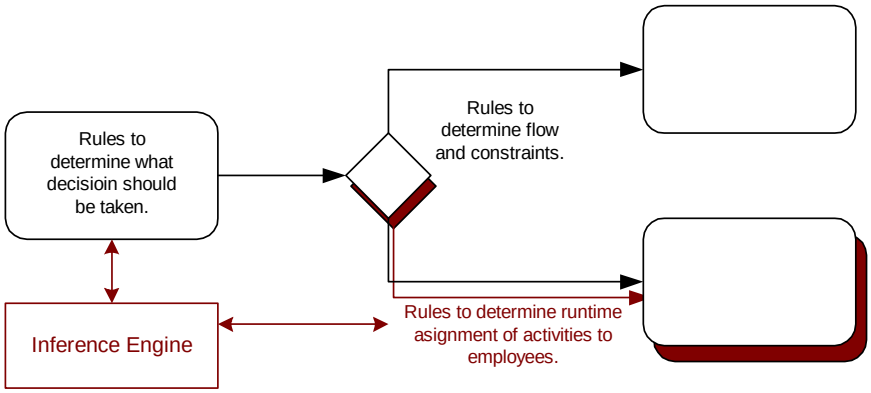

Figure 3. Three ways of using rules

When considering business rules capabilities, firms should keep in mind that as the complexity of the rule set increases, system performance (scalability) of the resulting application often deteriorates. Similarly, scalability is impacted by the number of variables considered by the rule set. On the other hand, with certain sorts of applications and the falling cost of computing power, this may not be a problem.

Another issue to keep in mind is that as the number and complexity of rules increases, it becomes increasingly difficult to determine the validity and completeness of the overall system. Further, it becomes increasingly difficult to see how a given business objective is reflected in a set of rules.

Those companies that conceptualize their processes in terms of rules will want a BPM Suite that supports the use of rules during process execution. Companies that already have independent rule systems may prefer to keep existing rule systems and simply link the BPM Suites, but many will decide that they want rule capabilities integrated within their BPM Suite.

5 Integration

In the past, Integration was the *Achilles Heel* of process-based systems. Integration at the back end could consume an inordinate amount of time and resources (representing as much as 50% of the cost of ownership for a process enabled application throughout its effective lifespan).[6] However, advances in technology over recent years have transformed the integration landscape. EAI and Messaging Oriented Middleware (MOM) products pushed the boundary considerably. In the past few years, XML and Web Services have enabled a quantum jump in the productivity that is achievable. They have simplified the task of integrating even the most complex application, lowering the bar of technical expertise required. Now, a business analyst with a basic understanding of XML can integrate most applications. The *introspection* of Web Services leading to rapid development of a Shared Data Space for the case, along with graphical tools for mapping variables onto that structure, have allowed non-programmers to undertake much more of this sort of work.

Those products that have a legacy from the EAI and Workflow domains still tend to require specialist IT support. Products that have emerged more recently tend to concentrate on the XML and Web Services route (rather than bespoke scripting).

Some products delegate the integration challenge to a separate Integration Engine. And some BPM Suites leverage existing integration facilities of their host environment. For instance, in the Microsoft arena, virtually all products make use of Microsoft's BizTalk Server 2004 (a .NET environment). Some

[6] This figure is based on research undertaken during 1996 around a 33,000-seat deployment of workflow technology in a large retail bank and related anecdotal evidence gathered at the time.

vendors deliver their own bundled Integration Servers which work closely with the core process engine of the BPM Suite. Examples are IBM's WebSphere BPI Server (a Java environment), TIBCO with BusinessWorks, and CommerceQuest's TRAXION.

When linking in third party applications we can see three distinct levels of sophistication:

- *Brittle Spaghetti* – Some leave it to the process (software) developer to write suitable programs. Most application vendors facilitate this with sophisticated API sets that allow developers to pass contextual information to the third party application. But this can result in what we have come to call *brittle spaghetti* where disjointed external program calls are peppered throughout process definitions. Over time, the management of this integration code becomes a nightmare as back-end systems change and the process models themselves evolve.

- *Adapters* – A more sensible approach is to develop standard interfaces to these applications through which the process communicates with the backend application. Sometimes called *Connectors*, a whole industry has sprung up with third party software vendors providing Adapters to major enterprise applications such as PeopleSoft, Siebel, and SAP. While much better than the brittle spaghetti described above, they do have limitations. Purchase an upgrade on your SAP system and you will likely need a new set of Adapters to go with it.

- *Introspection* – The best approach we found was a self-generating *adapter* facility that is driven by automatic service discovery. The interfaces to third party applications are *wrapped* to create a set of reusable *components* that are then made available for use in process models. Known as *introspection*, the developer explores the methods and properties of the target application via its API set or a Web Service interface. Selected methods and their properties are then wrapped with an executable program, which connects to the target API and, through it, to the back end application (creating reusable components).

Some vendors make these components available on a modeling palette as pre-integrated, custom *Tasks* that are then dragged and dropped onto the modeling canvas, as needed. Others catalogue them and make them available as a set of discrete integration objects that are sewn into the back end of system-oriented task types, as needed. This latter approach is preferable as it normally includes mechanisms to manage the library of components, making sure they are available, as needed, on target systems.

Either way, this introspection approach allows specialist IT developers to ensure that components work effectively, yet delivers an environment that sophisticated end-users can access without having to resort to coding. Moreover, it provides a better way of managing the interface to third party applications, insulating them from each other and the process environment. Change the back end system and all the developers need to do is re-introspect the application – the process model and end-user application need not change at all.

Reusing BPM Engine Functionality

Another aspect of integration is the need to allow a third party application to reuse the process functionality of the BPM environment itself (i.e., the exact opposite of the above need). These options are most relevant to ISVs looking to process enable their existing applications, or major end-users looking to develop custom user interfaces for specific categories of employee.

This sort of requirement is normally implemented via a published API set (of the BPM Engine). Products vary greatly in their overall approach, although we are now seeing more use of Web Services in this area.

Some products expose key elements of functionality as a set of standard objects at the client level. An alternative is to provide access to functionality via a server-side model – COM, C++, Native Java, RMI, and EJB flavors are common.

6 Organizational Structure

An essential feature of any BPM solution is the ability to route work items to the correct person. In order to do that, the system must know about the organizational structure and the way in which processes interact with it. At the simplest level, process definitions reference *roles* or *queues*. In some way or another, the BPM environment associates individuals in the company with these roles/queues, allowing the engine to resolve routing decisions and push work through the system to the appropriate people.

But allowing the process to reflect the companies, divisions, departments, sections, job functions, groups, matrix reporting structures, and communities of interest within an enterprise is a little more challenging. And the complications only increase when you consider that the notion of process is no longer limited to the boundaries of the enterprise; the scope now extends up and down the value chain, incorporating customers, suppliers, and partners. How do I route the work item to the supervisor of this person? Who should have sight of the current state of the process instance? How do we route work to our partner (yet allow them to own their own parts of the process)? How much work can this individual user (or team) handle, and what skills do he/she/they need to carry out the work? All of these types of questions require some notion of organizational scope and structure.

Proprietary Approaches

In the past, vendors had to provide their own capabilities in this area, building proprietary notions of organizational structure into the product. Some products had a very simplistic approach (with just shared queues to which users are assigned). Others used roles (assigned to the queues) with an independent structuring of roles to reflect either the reporting structure of the business and/or abstract roles representing a type of work carried out.

Directory Servers – LDAP & Active Directory

While the BPM environment may have its own proprietary data stores to reflect the organizational structure, integration with an industry standard Directory Server is now becoming the norm (at some level). Directory Servers can be used to maintain information about the structure of the organization and, sometimes, its customers, partners, and suppliers, making this information available to a variety of applications. The Directory Server stores the assignments between employees and roles (either abstract or organizational reporting based).

Directory Servers come in two fundamental flavors: The *open systems* version is LDAP (Local Directory Access Protocol), whereas Microsoft has its own Active Directory (AD) Server. Most products now incorporate a capability to reuse this information.

It is worth noting that there is a marked difference between end-user firms when it comes to the understanding and use of AD and LDAP technologies. Larger firms have embraced the power of these environments to underpin a variety of systems initiatives, whereas the vast majority of smaller firms have done little more than distinguish between systems administrators and end-users. For many firms, it is challenging to build and maintain a comprehensive model of the organization along with all of its complex reporting structures and teams. This usually requires considerable attention in many BPM initiatives.

As we will see in one or two of the product reports, getting control of this area is critical to achieving ongoing productivity improvements. Without an accurate organizational structure, it is impossible to deliver effective management information or operational performance data.

When it comes to making use of a Directory Server, we find a number of distinct levels of use:

- *Import Organization Data*: The majority of vendors simply import this information (into their own internal structures) and then enhance it with proprietary information to reflect a richer

diversity of organizational forms than those one tends to find in a typical Directory Server deployment.

- *Reusing Organization Data*: Rather than import organizational information, these products directly reuse the data recorded in the Directory Server for routing decisions. They tend to provide tools to modify the structure of the information held in the Directory Server, adding further categorization mechanisms.
- *User Authentication*: Some products not only use the data for routing decisions, they leverage the Directory Servers inherent capabilities for user authentication. This can be taken a stage further and applied to authentication for access to associated content repositories.
- *Deployment*: Although not covered in the products reviewed here, we have also seen vendors that use the Directory Server to support distribution of applications to the appropriate part of the organization, ensuring that the necessary components, processes, subprocesses, and third party application connectors are deployed to the correct local servers.

7 Process Adaptability

In this section, we consider the underlying capabilities of the BPM Suite to support adaptation of processes at runtime. This sort of functionality is inextricably linked with that covered by the next section, Process Lifecycle Management.

Over the last 70 years an enormous amount has been written on how organizations "get things done." Two broad approaches are apparent. At one extreme, individuals are controlled (to the n^{th} degree). Workers only carry out tasks as instructed. At the other extreme, workers are *empowered*. In this case the workers share an understanding of the vision, goals, and business rules of the organization. Empowered workers tend to work independently of management with little or no control imposed from above. In reality, most organizations adopt a position somewhere in between these two extremes. Different groups (within an enterprise) are usually in different positions along this spectrum – usually based on an individual manager's style and circumstances, rather than on any conscious decision on the part of the firm's management.

These issues point towards one of the critical issues for BPM deployments. How does one achieve that fine balance of control and empowerment, varying the stance to reflect the needs of different parts of the organization and distinct phases of activity within the process? Empowered cultures have disadvantages with respect to control and communications, although the benefits, in terms of overall adaptability and flexibility, can be enormous. However, when one looks at the products and BPM implementations, most reflect management's preoccupation with control, often resulting in highly mechanistic process architectures that reduce organizational flexibility.

A critical point, often overlooked, is that, prior to the selection of supporting BPM technology, managers should consider the organizational and cultural context of the firm. The capability of those environments to support end-user adaptability of the process is critical: No matter how much analysis is carried out up front, the process models used to drive the business will almost certainly be required to change very quickly. Moreover, individual cases may present unforeseen exceptions.

Products vary considerably in support for the challenges of change. We have always maintained that BPM environments reflect the views on organizational behavior held by the original software developers. The products varied considerably, based on whether the vendor thought the problems were all about control of employees or, at the other end of the scale, whether they sought to enable knowledge workers to achieve their goals.

Moreover, one of the key challenges in a process driven production environment is that all the manpower is spent handling exceptions (rather than the instances that follow the standard, automated

pattern of work). If you do not have an effective way of handling each exception (for that particular instance), then it rather defeats the purpose.

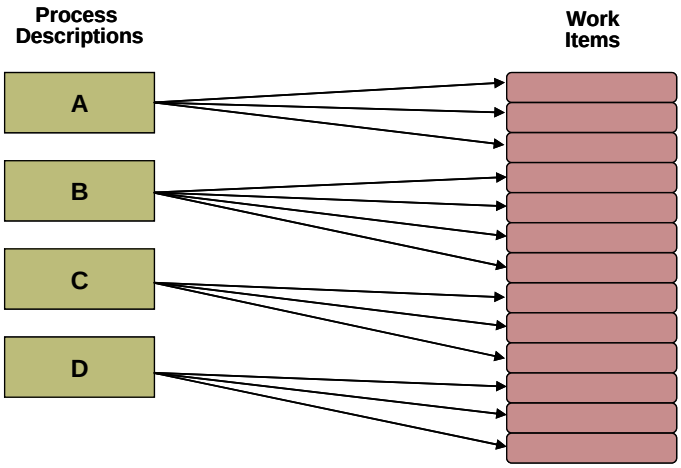

Figure 4. The majority of approaches rely on multiple work items sharing a common process model; the process is primacy and the work item is secondary.

Most products start from the position that all changes must be reflected in the process model a priori (beforehand), and that, at the simplest level, all work items of a given class share a single process description (rather than copying the model to support each work item). In such situations, the engine will normally allow the end-user little or no choice, and no option to step around the pre-defined procedures. The end-users are restricted to entering information; any exceptions are gathered up manually by a supervisor who may have the capability to bypass a step in the process or cancel the work item. In such situations, the supervisor will normally need separate access to any third party applications. Very often, the supervisor becomes the bottleneck as cases mount up that do not seem to follow the rigid pattern laid down beforehand.

Change is, of course, possible through re-development of the process model. New cases of work will then follow the modified process description (most products incorporate some form of version control to support this transition). Change to an individual case requires that all threads of work be deleted and the case recreated at the appropriate point (compromising any future audit).

The core problem with this underlying approach is that all possible permutations of the process must be predicted in advance – something that is virtually impossible to achieve. To get around this situation, some products incorporate features that provide the ability to bypass, redo, and rollback steps.

A more flexible approach is to develop what has become known as a Case Handling environment.[7] Although it is rare to see out-of-the-box support for this, it is often possible to enable much greater adaptability through the use of advanced process architectures.[8] Effectively, a parent process is used to support the overall case. End users (and/or the system) select the appropriate process object from a library, which is bound into the parent at runtime to reflect the needs of the particular case in hand.

Traditionally, this selection was achieved with a library of subprocesses (submaps) embedded within the parent process. While somewhat effective, this design was not as adaptable as one based on distinct

[7] See the Business Case For Case Handling at http://www.enix.co.uk/caseman.htm

[8] This sort of design approach is possible with a few of the major BPM environments, leveraging the flexibility offered around the way in which subprocesses are called and instantiated. It will usually require great care in the design of the process architecture itself, and may involve the development of an external application.

process objects maintained independently of the parent process. The latter approach enables each process object to be developed and maintained independently of the others.

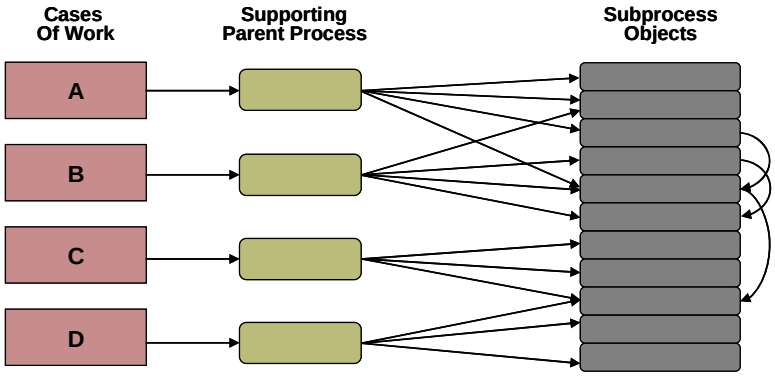

Figure 5. *Case Handling* systems allow multiple process objects to be associated with a given case of work; primacy is with the case, rather than the process.

The advantage of this approach is that process developers can then concentrate on the essential path of the process, leaving it to end-users to exercise their own judgment. Products in this category normally incorporate a pre-condition function that allows developers to build in checks that ensure corporate policies are enforced. For example, in an insurance claims situation, this sort of flexibility would enable Customer Service Representatives (CSRs) to follow any number of paths through the process, yet would still allow the firm to ensure that the claim went through appropriate approval before any payments were made. Such systems offer users great flexibility while also allowing the firm the greatest level of control. Each process object is developed in advance and is tested (often by the specialist process developers) to ensure that it functions appropriately. It is also worth noting that with the advent of Web Services, we are likely to see more and more Case Handling type process architectures.

The highest level of adaptability in Case Handling is enabled when end-users have the ability to create their own personal process fragments and apply them selectively within an individual case. New process objects are defined by the user, as needed. Alternatively, they are copied and refined from existing templates and bound into the parent. The user is now able to interpret the task in hand, invoking a private/public process object to undertake the work. Suitably authorized users then have complete control over the process definition. Process models evolve as users adapt existing definitions or as new fragments are added when required.[9]

While this last level of adaptability might appear to many readers to be outside the scope of a BPM Suite, it is essential for a whole class of applications. It would be most relevant in project-oriented environments and where knowledge workers must continually exercise their judgment – examples would include software development, architecture, offshore oil and gas exploration, and advertising. Each scenario is unique, yet the benefits of process reuse are massive.

The precise mechanisms used to enable adaptability rely on the underlying architecture of the BPMS and whether it can handle more than one process associated with a work item. With a couple of notable exceptions, products use a single process model to support all cases of a given class. Some have sought a halfway house where the a priori model drives work, but, when unforeseen exceptions occur, the process instance is *referred* to the process owner (or suitably qualified expert) who adjusts the business rules to reflect the needs of the case in hand. The process model is effectively modified through the business rules, allowing the case in hand to continue.

[9] This degree of flexibility is even more rare than Case Handling. None of the products reviewed in this study provided this type of functionality.

Those vendors that have elected to copy the process model for each case of work generally have a much more flexible and adaptable stance. They can truly respond to the needs of individual work items without having to revert to changing the original process model. Indeed, most products that do not support this approach assume that exceptional cases can be neatly erased from the system and then restarted with the new version of the process, which, in turn, affects audit ability and traceability.

8 Process Lifecycle Management

One could say that all BPM suites provide some degree of support for the lifecycle management of business processes. Indeed, supporting that lifecycle from the initial vision, through modeling, deployment, execution, performance monitoring, and modification, is at the very core of most BPM initiatives. In the past, we have seen mechanisms that solicit input from key people and record reasons for changes made to models. [10]

However, products vary significantly in their support for lifecycle management. What we are particularly interested in here is the ability to track the lifecycle of a process model, supporting the organization as it optimizes the process over time. Given the need for BPM products to support an adapting, evolving business climate, lifecycle management of a firm's process assets is a critical area that is generally poorly supported by most vendors.[11]

Virtually all products provide an administrative console to facilitate and control the deployment of process models. While it is normal for the process developer to have the right to publish his or her own models, some organizations look for more precise control over this aspect (allowing them to apply quality assurance and compliance tests to the production environment).

As we mentioned earlier, most BPM environments support some sort of version control of the business process models. Some take this a bit further, with an ability to transparently support the deployment of different versions and variants of a process, depending on some specific factor. For example, imagine a bank and its credit approval process. It would need to support subtly different processes in different geographies, organizational units, or through different channels. The deployment mechanism of the product would then need to ensure that different versions and variants of the process are deployed to the right places.

In some products, the administrator has an option of overwriting an existing published process (effectively replacing it on currently running cases of work).[12] However, most insist that the new version of the model is used to drive new cases of work, while the older model is maintained until existing cases have completed.

Generally, products leave it up to the systems administrator to worry about how these models evolve and how older versions of the process description are managed. Apart from that, support for the company managing its *process assets* is delegated to third party repository-oriented modeling tools. A couple of our vendors in this study have used associated functionality, such as business rules or the content repository of their engine, to manage the way in which processes are deployed and managed.

---

[10] Although almost totally unrelated to BPM technology, a useful methodology that can help alleviate this is Issues Based Information Systems or IBIS. See the open-source initiative at www.compendiuminstitute.org

[11] This requirement is distinct from the ability to simulate process models and carry out what-if analyses.

[12] Clearly, there are consistency issues associated with replacing the process model used by cases that are in existence already. The current process state of a case may become invalid. Products that support this sort of replacement option should also include a search option to ensure that all existing cases of work are in a valid state.

9 Monitoring, Measurement, and Management Information

Many senior managers have embraced the idea of BPM deployment in hopes of obtaining better information about their business processes. They want the ability to look into the end-to-end profitability of a product or service, drill down to assess, compare and contrast the impact of one channel over another, and to see how individual departments are performing against Service Level Agreements (SLAs). They want the ability to track and monitor their business, identifying teams that are failing or to spot quickly any bottlenecks that are impacting performance.

Virtually every BPM solution we looked at promoted the idea that, by using that product, customers will get better management information, but, once again, different vendors mean rather different things.

Consider Figure 6. As events occur, data about them are stored in a database (a History File). A monitoring utility can use that information to provide feedback to employees (1). It could, for example, tell them how many work items they had to deal with in the past hour. Some BPM Suites are set up to generate reports, usually relying on some 4GL system. Thus, a system might generate a report for the supervisor or the supervisor's immediate manager that would tell how many units were produced per hour, per employee (2). This same or related data could be used by the BPM Suite or by IT managers to monitor and adjust the performance of the BPM software, redirecting work, as needed, to alternative resources.

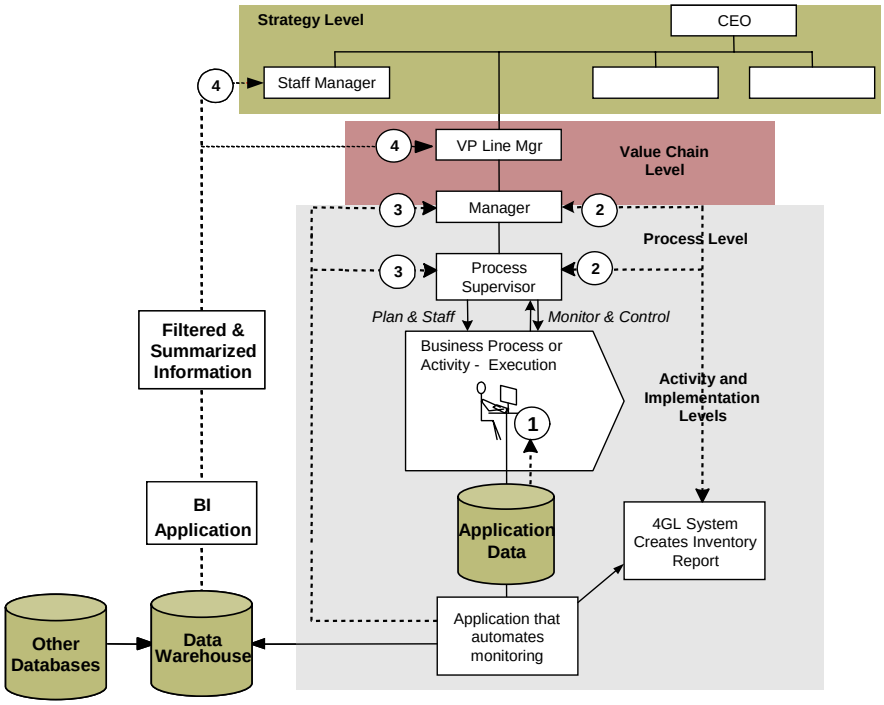

Figure 6. Different ways of gathering and presenting process data

We hear vendors say, "The history file provides all the audit information you will need – a complete history of the work item, including who handled it when, etc." Usually, this audit information is limited to a description of work items, changes in activity state, and information on who handled it. A couple of vendors have taken this sort of information several steps further. For instance, M1 Global maintains not only a history of the changes at the activity level, but also of each interaction channel across a case – whether that is via call-centre, the web, or even instant messaging. On the other hand, eg work manager uses quality and compliance sampling metrics and skills to support tight production management

disciplines that drive increased productivity.

Generally speaking, with the right information gathering and reporting regime, the business can more easily sense important changes in the market and customer behavior – changing the process, if necessary. Based on a better understanding of operational needs and capacity, managers can then make more informed decisions, adjusting resource levels in different departments – effectively load-balancing the business in line with peaks and troughs of demand. Users and managers can then view average process cycle times, which types of cases and activities take the longest, or even how long cases take to get to a particular stage. This information can then be used to find process bottlenecks and identify areas for improvement.

Other monitoring systems can make the data available on line for the supervisor or the manager (3). Some, in effect, generate reports that the manager can access on line, while others can provide the manager with real time information about the events occurring in a given process.

The key thing to note is that (1), (2), and (3) all provide managers with more or less *raw data*. If the manager does not understand the specific activity or process, he or she will probably not be able to understand the data being generated. Moreover, the data needs to be limited or the volume of the data will overwhelm the manager.

More sophisticated monitoring systems go well beyond this. They combine the data resulting from a given process with other data from other sources. This is usually done by combining multiple data sources in a *Data Warehouse*. For example, one might combine data from the specific process with data from the upstream and downstream processes, or even from sales and customer satisfaction data. This data would be truly overwhelming if it were sent to a senior manager without being filtered. Most companies that are developing more sophisticated monitoring and management information systems, use Business Intelligence (BI) systems to analyze the data and to look for patterns. In addition, they use filtering systems and tailor the manager interface to present the information in a highly summarized format. This information is suitable for senior executives and strategy teams seeking to exercise high-level oversight or to determine the overall value of a business process (4). This is the approach that IBM uses in its WebSphere Business Integration Modeler.

At a minimum, users evaluating BPM Suites ought to ask if the suite is reporting data about the specific process, or if it is combining process data with other data, analyzing it, and filtering it to provide information for senior managers.

In the past, managers were left to reflect on their own special needs for Management Information. Developers had to write suitable programs to extract this information and present it to the appropriate member of management. But now we are seeing a far more systemic approach. Thus, vendors have begun to include *Analytics* functionality within their products.

Indeed, we even have one specialist vendor (eg Solutions Ltd) that focuses exclusively on delivering effective Management Information and production planning capabilities (on top of another BPM product). One should remember that there is a distinction between having process metrics available and using them effectively.

10 Templates and Frameworks

In addition to selling a BPM Suite, many vendors provide templates or frameworks that give the business a start point in developing specific types of applications. For example, a vendor could provide proprietary business rules for popular industry processes, or even offer software components to facilitate the development of a specific type of business process. Some go so far as providing support for entire applications focused on a specific vertical or horizontal application.

In a similar way, some vendors provide industry frameworks that define vocabularies and metrics for specific types of processes. ACCORD provides a framework for the insurance industry and the Supply

Chain Council offers the SCOR framework for companies working on supply chain systems. The use of a framework or packaged components and rules can substantially reduce your development time. If you use a business process framework at your organization, its worth inquiring if the BPM Suites you are considering will support that framework as well.

11 Vendor

In each review, we briefly describe the vendor, noting its history, and its current situation. Everyone should be aware that the interest in BPM Suites is still developing. At the same time, the technology is evolving and new standards are emerging – while the majority of potential users are still evaluating BPM and have yet to decide exactly what features they will want in a BPM Suite. Of course, in such a dynamic situation, where innovation is still the watchword, most of the BPM Suite vendors are relatively small. This creates a volatile situation. Predictably, as the market grows rapidly in 2006, and then in 2007 or 2008, and as standards and technologies mature and companies settle on the features they want in BPM Suites, there will be a shakeout in the BPM Suites market. Thus, companies that are considering BPM adoption should consider both the technological sophistication of the vendors and their financial strength. These considerations may not be critical in 2006, if your company is simply buying one or a few copies to experiment with, but they will become critical when companies began to standardize and make more extensive commitments to specific vendors.

12 Cost

Finally, we provide basic information about how each vendor prices its product.