# BPM and SOA

**Mike Rosen**
Chief Scientist
Wilson Consulting Group

Mike.Rosen@WilsonConsultingGroup.com

## 5 Key Requirements for SOA

I've been seeing a lot of press lately about the relationship of BPM and SOA. Some common themes have been the convergence of BPM and SOA, or the gap between BPM and SOA, or the virtues of a product release that addresses the convergence and spans the gap. So what is this gap, if it even exists?

The gap is really a difference in viewpoint. Viewpoint is a well known architectural concept to describe the audience and concern, of a particular description. If you're familiar with Enterprise Architecture at all, you've probably heard of the common EA sub-architectures of Business Architecture, Information Architecture, Application Architecture and Technology Architecture. Another way of describing these is as four different viewpoints on enterprise architecture, each focused on a particular part of the enterprise problem space. Each view point has its own set of abstractions, concepts, vocabulary and artifacts.

We can look at BPM and SOA as two different viewpoints. One might be called the 'Process Viewpoint' and the other the 'Service Viewpoint'. The process viewpoint is concerned with the overall, end-to-end lifetime of a business process. Each process is constructed from a set of individual activities, and the process is concerned with combining them together and then executing, managing and monitoring the current state, progress and performance of the entire process. The service viewpoint is concerned with the collection of services that are available to the processes, and how they are constructed, organized, provisioned, managed and maintained, but not so much with any specific process that they are used in. Obviously, these two viewpoints are related and synergistic.

We could also look at them as the 'Consumer viewpoint' (process) and the 'Provider viewpoint' (service). In other words, processes are the consumers of services. They see services as something that provide business capabilities and they have certain expectations about their reliability, Quality of Service, etc. Like any other provider, the service viewpoint should be concerned with meeting their customer's needs in the most efficient, and economical way.

So far, so good? But the rub comes when viewpoints use the same terminology, but mean different things. Is what service or SOA means to the process viewpoint the same as what it means to the service viewpoint? What do they mean by shared information or semantics? How about integration and governance? So the gap is created by using the same terminology while applying a viewpoint centric meaning to it, rather than a shared meaning. I've seen animated, almost violent discussions ensue until the parties realize that they we're actually talking about different things.

Back to the news, products are now claiming to support the convergence of BPM and SOA technologies. There have been major announcements this year of BPM products being acquired by SOA vendors, such as the purchase of Collaxa by Oracle and the recent purchase of Fuego by BEA. There have also been announcements of new product sets following previous acquisitions. IBM released a mind-boggling array of 11 new products and 20 enhanced products around it WebSphere SOA initiative last month. WebSphere Business Modeler supports the design of business processes. Those process models can then be imported into WebSphere Integration Developer to design the services that will implement the process activities. IBM is addressing the process view point with Business Modeler, and the service viewpoint with Integration Developer, specifically acknowledging the different viewpoints and target audiences with targeted products.

In a different announcement, TIBCO released version 10.3 of their Software Process Suite, based on the integration of Staffware into their product line. In their white paper on "Unified BPM and SOA Process Architecture", they boldly claim that "…the latest advancements from TIBCO deliver on every key point required to fulfill the requirements of unified process architecture (i.e. composite BPM/SOA)." Their strategy (which seems to be more common in the industry than IBM's) is to provide a single product that provides a unified viewpoint, rather than promoting two different viewpoints with complementary product. Unfortunately, they don't bother to describe what the key points for a unified process architecture are.

The problem, which I see over and over again, is that the typical 'SOA' is only about building services. "Yeah so?" you say (or "Whatever" if you're a teenager). But wait…there's more! What about the 'Architecture' part of SOA? Let's stop to examine why we are creating services in the first place. Certainly, services are a good way to provide common functionality. But generally, we are expecting more, especially at an enterprise level. We want services to be combined together, and to support business processes. And we are probably expecting that services will be implemented by different groups, and that service consumers and providers may be in different organizations. While the implementation of a service is important, SOA should be at least as concerned with the support of BPM and the totality of services as it is about any individual service.

The 'Architecture' part of SOA must provide a solution to how services are designed, used and combined to meet the overall goals and requirements of BPM and to facilitate traversing the organizational boundaries of the consumers and providers. In doing so, SOA will address services, but from the context of achieving commonality among services so that they can be implemented by different groups and combined into business processes. The SOA must provide a detailed definition of what a service is and how it uses the SOA infrastructure. Beyond that, the definition should also include: what interfaces a service is required to provide (if any), what a service contract looks like, how services are registered and discovered, what an SLA is and how it is managed, monitored and reported, and all aspects of a service lifecycle, etc.

Even then, this is not enough to ensure that independent services can be combined together in business processes. In order for services to work together, they have to have a common meaning for shared information. For example, if you want to combine services together to perform a function for a customer, all of those services need a similar definition of a customer. (Note, the information that is common about a customer must be the same, but each service can have its own internal definition of a customer -- the beauty of separating implementation and interface). So, SOA must address how shared semantics is defined, where it is stored, how it is used in the implementation of services, and how it supports the semantics of the business processes. Just as the gap between viewpoints can result in acrimony when different meaning are assumed for shared terms, individual services can be caught in a semantic gap. Unfortunately, when services get stuck in violent disagreement in the midst of an executing process, the results are probably more serious than a bruised ego.

So to help readers evaluate the current and future products that claim convergence between BPM and SOA, here is my take on the key aspects of SOA needed to support BPM.

## Shared Information and Semantics

There are three important facts about services that set the basis for their information processing requirements. First, services, especially at the business level, pass information in the form of documents. Secondly, the information in those documents needs to conform to the enterprise information model and semantics. Third, there is often a transformation between the enterprise semantics and the internal information model of the service. So, the SOA platform must provide:

- Document processing capabilities.
- Integration with existing, enterprise information models.
- Definition of documents based on the information model.
- Information transformation capabilities.

## Application Integration

Many enterprise capabilities are tied up in existing legacy, COTS and other applications. These capabilities and data need to be 'service enabled' so that they can be included into new business processes. However, we must also avoid the previous mistakes of EAI that led to fragile, expensive and out of control integration. For example, the functional and information model of the existing applications is different than that of the enterprise. Services need to map from the enterprise model to the application model as part of the integration, not impose the legacy model on the business processes. The SOA platform must provide intelligent service enablement that:

- Exposes function and data as services which exhibit all important service characteristics (such as contracts, discovery, etc.) .
- Transforms between the enterprise model and the legacy model .
- Does not expose existing functions and data directly as service interfaces.
- Never allows a business process to connect directly to a legacy system or database without using an integration service.
- Treats integration services differently than business services.

## Service Reuse and Governance

One of the goals of SOA is to provide a catalog of reusable services, and to have the business processes use those services. This doesn't just happen by itself. The process designers need to be able to find the set of available services, examine them to determine if they perform the required functions, to request changes and enhancements to existing services, and to include services in their business processes. The producers of services need to determine that a similar service doesn't already exist, and to understand the boundaries of the role and responsibility of new services to avoid creating redundancy and overlap. The SOA platform must provide:

- A design time service repository that supports the discovery and examination of services during the design of business processes.
- A distinct difference between the design time repository and the run time registry.
- An approach and method for organizing the overall set of services including the definition of responsibility boundaries.
- Service use and development governance policies, preferably automatically applied as part of using the repository.

## Versioning and Lifecycle

Once a service is used by more than one process, it will inevitable have multiple users, with multiple different needs and project timetables. This leads to the need to enhance and extend services in a way that manages the different requirements. Evolution of services ultimately leads to multiple versions of the same service. Note that this is just the facts of life of SOA, not something to be feared or avoided. The SOA platform must support this by providing:

- The ability to assign version numbers to service interfaces.
- The ability for a single service implementation to support multiple versions (typically multiple minor versions).
- The ability to have different versions of a service running simultaneously on the network.
- The ability to bind a client to the appropriate service based on version compatibility.
- A versioning policy that includes major and minor updates.
- The automatic implementation of the versioning policy as changes are made to the service interface.

## Management

As we use and rely on services, we depend on them to perform as expected. As we provide service level agreements at the process level, we need to aggregate the service levels of the individual activities (performed by services) together. Of course, like reuse, this doesn't just happen by itself either. We need to be able to define and measure the performance of services, and to manage those services according to their agreements. In SOA terms, we need the platform to provide:

- Service contracts that go beyond the simple definition of the service interface to include SLAs (at a minimum).
- The ability to monitor and report on service performance according to SLA.
- The ability to notify service providers and take corrective action when performance is not being met or certain thresholds are reached.
- Reporting of service performance across a variety of parameters and a variety of service collections.

Of course, these are not the only requirements. I didn't even bother to mention the basic integration of services into the BPM infrastructure. Instead, these go beyond the obvious into what really makes a different at an enterprise level. Don't expect vendors to provide them all, especially in their first product releases. But, use them as a guide to help cut through the hype and evaluate how effectively a product set can deliver on the promise of convergence and actually provide an SOA platform that enables a new level of flexibility and agility to BPM.