

Strategies for Implementing an SOA

Today's IT organizations must respond quickly to demanding market and environmental changes. This requires software and infrastructure that is flexible enough to quickly support unforeseen changes. One way to provide flexibility is through a Service Oriented Architecture (SOA). The primary goal of SOA is to achieve loose coupling among an environment's software applications. Loose coupling promotes reuse, enabling faster response times to the applications that control an organization's business assets. This paper addresses critical aspects of implementing an SOA and technologies that work hand-in-hand with an SOA infrastructure.

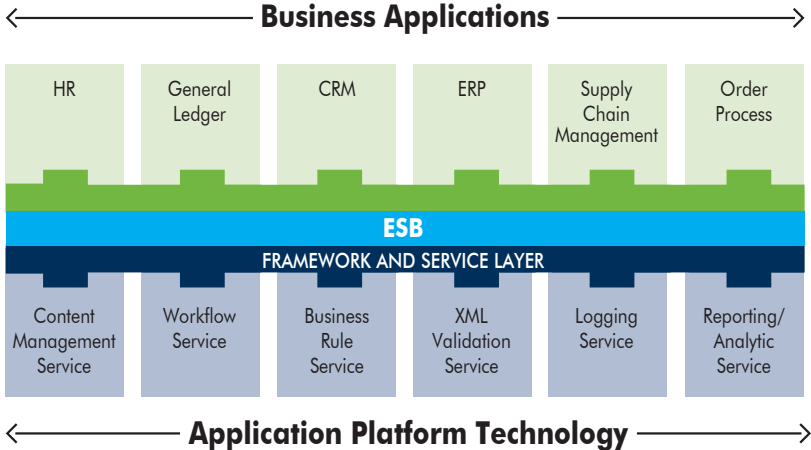
An SOA provides benefits both for an organization that creates the service and for organizations to which the services are exposed. The obvious advantage is the internal loose coupling of reusable service functionality. Additional indirect advantages are realized as selected services may be exposed to outside entities to allow data transmission and consumption. Outside organizations may post data via a service call or consume a service that provides information to an internal application. This distributed nature is where the real power of internal and web services are realized.

As businesses move forward to implement SOA, two important considerations should be noted. First, all new projects should be analyzed to determine what functionality may be reused by future applications. Those reusable pieces are ideal candidates to be implemented as services that will reside within the organization's core framework. Ideally, a small project with minimal scope will provide the best backdrop to introduce an SOA to an organization. Second, legacy applications that provide needed enterprise functionality can be "service-enabled" by developing a service interface that exposes functions within the legacy product. By doing so, legacy applications may live within an SOA, allowing them to be used with new application development and extending the lives of these applications.

Once an organization has worked toward service-enabling its new, current and legacy applications, the real benefits of an SOA begin to take shape. At this point many services exist within an organization's core framework, very few of which are dependent on each other. In a traditional implementation scenario, a custom built application will consume various services both inside and outside of the organization. Within an SOA, an application can be created by "gluing" services together. In this manner, the definition of application development changes. The majority of business logic is then executed within service calls rather than within the application itself.

Along with the recent development of SOA in the marketplace, numerous other technologies have emerged providing service orchestration, management and messaging. Business process management (BPM), business rules engines (BREs), and an enterprise service bus (ESB) all offer additional benefits to an SOA.

Business process management software, also known as workflow software, can tie the loosely coupled services of an SOA together in a way that creates a highly maintainable and flexible application. Further, additional functionality can be added to the workflow, such as decision making based on content or consumer decisions. BPM software supports human-to-human, system-to-system, and human-to-system interaction. Using aspects of BPM allows an organization to truly experience the benefits of an SOA.



As companies continue to work toward being able to respond quickly to policy changes, it is very important that business assets can be found within a common repository. Further, businesses often require changes that can be implemented by non-IT staff. The use of a Business Rules Engine (BRE) moves all business assets to a common repository. BREs have other built-in features such as versioning of rules, logging of rule changes, simulation, and the ability to service-enable rules and rule sets, to name a few. These business rules can be called from multiple applications, removing business logic from within each application. The advantages can be seen immediately in the form of maintainability, the ability of non-IT staff to implement rule changes at a policy level, and consistency of a company’s business rules no matter the application that is using them.

In the past, organizations have spent a significant amount of time integrating disparate systems. Additionally, when outside applications have to be communicated, application code is typically embedded in every application where the handshake must take place. This type of coding is often brittle and doesn’t respond well to changes within an organization. To alleviate this dependent code, various companies have introduced a product known as an enterprise service bus (ESB). An ESB’s role within an SOA is to allow communication between disparate applications both inside and outside an organization. With the use of an ESB, an application will call a service through the ESB, isolating the application from any changes within the service. Message transformations, security, reliable message delivery and other SOA necessities are handled by the ESB. By doing so, services may be changed with no effect on the numerous applications that may be using it.

The information outlined in this paper provides a good start to learning about the numerous advantages that an SOA can deliver. With planning and guidance, the benefits of an SOA can be fully realized and provide the flexibility that all organizations need.



Put the Agile Edge to work for you today.

Contact: Nicholas Coenen • 614.228-6522 • ncoenen@agiletech.com

www.agiletech.com