

OpenSpan White Paper Series: **Moving Applications Off the Desktop**



Moving Applications Off the Desktop

INTRODUCTION	p. 3
IDENTIFYING SERVICES IN LEGACY APPLICATIONS	p. 4
MAKING USE OF THOSE SERVICES	p. 5
BENEFITS OF THE OPENSAN-BASED SOA STRATEGY	p. 6
CONCLUSIONS	p. 7

Moving Applications Off the Desktop

INTRODUCTION

The principles behind Service-Oriented Architecture (SOA) enable organizations to treat applications as a set of discrete services that can be configured to quickly produce new applications in response to changing business requirements. Sometimes the SOA process involves creating entirely new services, based on an analysis of business requirements.

More often, however, services are derived from existing applications. The business logic required to power the operations of an enterprise in most cases already exists, but is locked away in legacy applications, often running on mainframe or client server systems. These legacy applications serve a single business purpose, and often lack documentation and even source code. There is little opportunity to extract logic programmatically using traditional means.

This means that the business logic exists, yet cannot be accessed to be used in other applications. At the same time, attempting to replicate that business logic through new code will almost certainly be costly, and may in some cases be impossible due to a lack of human expertise in the problem domain.

Enterprises are essentially prevented from using existing IT capabilities for new business models or changes in the business climate. The business logic is essential to the enterprise, yet is locked inside a legacy application.

This is especially true of applications that, for one reason or another, remain essential to the organization, yet are difficult to use from the modern desktop computer. And the existing applications tend to have text-based user interfaces, making effective use more difficult. In most cases, they were used through transaction terminals, or on PCs through telnet or other text-based interface.

In some cases, logic may be contained in more modern client server applications, with a rich client interface to a database on the back end. These applications came to define the architecture trends of twenty years ago, yet lack the scalability and reach needed in most enterprises today.

Is there a way to “unlock” this business logic, so that it can be run on the server yet accessed in some manner by new enterprise applications? What if, from the standpoint of an enterprise SOA, it was possible to service-enable legacy application functionality, and apply those services together into applications that are delivered to the enterprise via standard Web services or messaging interfaces and to the user through a portal, browser, mobile device, RSS feed or other modern interfaces?

These services, once free of their legacy applications, can be wired together into more complex workflows that span these applications, and provide the business with more complete and comprehensive applications that can be developed and deployed rapidly. Further, they can be brokered to run entirely on the enterprise servers, and delivered via a thin client architecture to the desktop if desired.

The advantages of doing this are readily apparent. Legacy applications, whether mainframe or client server based, can be made accessible away from the mainframe terminal or from specialized software on the PC. If the user interface is delivered through a browser, this allows enterprises to move closer to a true thin-client architecture and control the logic and presentation of any such application entire through its servers.

This approach enables the IT department to fully manage applications that are delivered to the desktop. The applications can appear on the desktop in the browser, rather than using a transaction monitor or rich client interface. Maintenance and updates are done strictly on the servers, while users have no disruption of application service. The result is a set of new application components that can be easily assembled and accessed by users on the Web using a custom front end, yet maintained and enhanced strictly from the server. Many of the benefits of moving applications off the desktop in terms of operational flexibility and application delivery are well understood which is why technologies such as Citrix, presentation services and application virtualization have flourished. However, this approach adds a different dimension in that the legacy applications themselves are no longer required to be used by the user at all.

IDENTIFYING SERVICES IN LEGACY APPLICATIONS

Identifying business logic in legacy applications can be one of the most difficult parts of this process. Searching out legacy applications that may have value in discrete parts comes up against some realities of both the organization and the applications. Often source code is not available, and even if it is, it may not always be possible to understand how a specific function operates simply by examining that segment of code.

If source code is not available, or if it is difficult or time-consuming to study source code and identify business functions that can be turned into services, in many cases the organization either attempts to rewrite those functions as services, or gives up altogether on service-enabling those applications.

With the OpenSpan Platform, it is possible to interrogate a wide variety of applications, including various legacy applications, to determine interfaces that can be called during runtime to provide select relevant business functions. By interrogating these applications, it is possible to identify business logic that can be called as a service, even from within the application itself.

Further, this interrogation identifies the gateway to that logic, the programming interfaces that enable other applications, and services to call into the logic. Those programming interfaces may or may not be public, and may or may not be documented, but they provide OpenSpan with the ability to link into those services while running, pass data in, and receive processed data back.

Once OpenSpan interrogates these applications and identifies candidate services and programming interfaces, IT developers can assemble a composite application by calling the interfaces and passing the data needed to execute those services. Once the services and interfaces are known, developers can treat the accompanying business logic just as they would any other type of service.

From within OpenSpan Studio, IT developers can connect these services, and pass data between them, by simply connecting boxes representing them with a line in the visual programming environment. With this environment, developers can easily call services, pass and receive data, and use that data in subsequent service calls.

Entire business processes, consisting of a combination of services from multiple legacy applications, other types of services, and manual operations can be rapidly designed, codified and deployed. These business processes can be implemented by ordering selected services in OpenSpan Studio and passing data from one service to the other with any additional required business logic being added to the workflow. If human intervention is required, for example, developers can build a rich client user interface from within OpenSpan Studio or design the most relevant user interface using Web or native design tools and have OpenSpan interface with them.

For organizations utilizing an Enterprise Service Bus (ESB) or messaging infrastructure such as JMS or MQ Series, OpenSpan also supports the triggering of the legacy functionality and consumption of the new exposed services via messages, topics and queues as an alternative to, or in addition to, invocation by Web service calls.

MAKING USE OF THOSE SERVICES

The OpenSpan Platform enables organizations to extend SOA implementations to virtually any Windows application, host system, Java application, Web application, PowerBuilder, and even DOS applications. It also provides access to third-party applications, Software-as-a-Service (SaaS) applications as well as proprietary applications for which the organization does not have access to APIs, application connectors or the underlying source code.

The OpenSpan Platform provides access to business logic inside these applications while running, and enables those logic components to be called as designed from within OpenSpan Studio. Deploying this application is a matter of employing an optional OpenSpan component known as Virtual Broker. The Virtual Broker is an application fabric, or middleware, for managing multiple OpenSpan sessions. This environment can include virtualized sessions if used by the organization and so desired. The Virtual Broker supports both the .NET platform as well as leading Java EE-based application server platforms.

Virtual Broker enables the coordination of these service components across multiple servers and hosts. It ensures that the services from multiple servers are delivered to the .NET or Java EE application server and exposed via a single proxy service, enabling those platforms to deliver new and comprehensive applications via a Web browser directly to the desktop or via any other application or device that invoke Web services or send and receive messages via an ESB.

In that sense, you can think of Virtual Broker as an orchestrator of those OpenSpan-enabled services, enabling them to be available to multiple users in the order they are called by the application server. The application server accepts requests from users, calls the Virtual Broker to execute the service or services required to fulfill that request, then creates the response and delivers those results to the users.

The combination of the OpenSpan Platform with Virtual Broker enables organizations to turn existing legacy application functionality and workflows into services, and deploy those services in a secure, managed middleware environment, no matter what the host platform. Using traditional SOA techniques, this can take years of architecture and development, while at worst it may be technically impossible.

Once the Virtual Broker environment is in place and able to orchestrate and deliver these services, enterprises can then build enterprise mashups, complete with (formerly) desktop legacy functionality. Organizations are limited in the way they combine data and legacy application functionality only by the imagination of their architects. They can immediately respond to new business requirements, and create new business opportunities, by combining formerly siloed applications and components into a coherent and valuable set of new services-based application functionality.

BENEFITS OF THE OPENSAN-BASED SOA STRATEGY

The use of the OpenSpan Platform with Virtual Broker provides the ability to deliver existing applications and application components as services in an SOA. This approach to SOA can provide a number of advantages, including:

- ❑ **Acceleration of the enterprise SOA strategy.** In all but the most trivial circumstances, implementation of a comprehensive Service Oriented Architecture takes years of design and development. The use of the OpenSpan Platform with Virtual Broker enables organizations to achieve perhaps the most difficult part of this strategy, that of service-enabling legacy applications, in months or weeks, rather than years.
- ❑ **Risk mitigation of new IT applications and development projects.** By leveraging business logic and data access from existing applications, organizations significantly reduce the business risk inherent in major new development projects. Business logic is reused, rather than re-implemented, which dramatically reduces the risk of error in interpreting and implementing that logic. Any required development is of a significantly smaller scale than a full-blown new “rip and replace” application development effort, with a simpler design and project structure and correspondingly fewer areas of risk. When circumstances and budgets permit, these legacy capabilities can be systematically deprecated over time. As new services are developed that replace the legacy functionality, these can be deployed with minimal disruption to the end user.
- ❑ **Cost containment by minimizing new development and leveraging existing applications.** By reusing existing code, organizations don't have to invest significant amounts of resources in new software. Instead, using the OpenSpan Platform with Virtual Broker enables them to focus on adding new capabilities, rather than replicating existing ones. And because OpenSpan applications run on existing servers, the only new hardware required may be for the application server and Virtual Broker. These characteristics significantly reduce the cost of ownership of SOA in any organization. Additionally, in some cases dramatic reductions in maintenance cost may be realized if hundreds or thousands of copies of desktop software are no longer required.
- ❑ **Operational flexibility in the form of a streamlined desktop environment.** Desktop preparation and support become significantly simplified if mission-critical applications are only delivered to the users via browser. This enables a more rapid transition to a true thin-client environment.
- ❑ **All applications are now under the direct control of IT.** Application delivery is significantly more straightforward, because no logic resides on the desktop system. Instead, browser delivery makes it possible to run applications on any system equipped with a browser. This potential includes non-desktop devices such as PDAs, cellular phones, and other handheld and mobile devices. Last, IT control means that applications can be maintained and enhanced without interaction with hundreds or thousands of desktop systems. Changes can be made and immediately deployed to all users simply by installing those changes on the servers.

CONCLUSIONS

Being able to design and build SOA projects which can take advantage of business logic contained within a combination of legacy applications can be a huge advantage to an organization in terms of risk, cost and time-to-market but implementing it is a challenge. If one of the goals is to streamline the user environment and move applications off of the desktop and fully under the control of IT, the challenge becomes greater. Delivering applications that contain components of existing applications exposed via standard services or messaging interfaces means that organizations can rapidly implement the SOA design, reduce the emphasis on the desktop, and focus on server-based application components.

Additional benefits for IT operations include managing that composite application entirely within IT, on the servers, rather than spend resources to maintain desktop rich clients. In addition, IT maintains and enhances each individual application running on server systems, and ensures that the applications operate properly and can handle the user load.

OpenSpan's approach to accelerating SOA design and deployment enables organizations to service-enable existing functionality rapidly, rather than take the years required to plan and execute SOA through new development only. By service-enabling and then removing the applications from the desktop with the Virtual Broker, the OpenSpan Platform enables an organization to substantially reduce the time until legacy desktop applications are no longer required by the user population without losing all the benefits of the man-years of effort and testing that went into making the original applications work. This complements a traditional SOA strategy, and delivers new, enterprise mash-up applications and significantly reduces the time needed to see real benefits from a SOA initiative.