The Enterprise Service Bus: integrating core back-end applications with web services

An Arcati Research Report
October 2005

Contents

Point-to-Point Integration	3
EAI and Messaging Middleware	3
Application Integration in today's Web-Centric Environment	4
The IBM Enterprise Service Bus: One Size does not fit All	5
The Adoption Model and Self-Assessment	8
Bottom Line	10

Arcati Research Report

The Enterprise Service Bus: integrating core back-end applications with web services

by Mark Lillycrop, Arcati Research

Today's heterogeneous IT users face a big challenge as they attempt to integrate lightweight web sevices with mature, high-performance enterprise applications. The Enterprise Service Bus concept, embodied in products such as WebSphere ESB and WebSphere Message Broker, is evolving to tackle this problem head-on.

There is nothing like the issue of application integration to remind us just how complex the IT business really is, despite its immaturity. Commercial business systems have existed for little more than 40 years, but in that short time we have generated wave upon wave of – often incompatible – technology. Large enterprises are typically built around islands of business function and information (silos, as they have become known), which are not only technically dissimilar but cannot even share the data and business logic they contain without considerable 'integration' effort.

It's not difficult to see how this complexity has occurred. Constant pressure to deliver improved business benefit has encouraged companies to explore new development tools and techniques; new hardware platforms have brought with them an ever broader selection of operating systems, databases, and software packages; and the continuous process of company mergers and acquisitions has forced IT departments to cope with an increasingly heterogeneous infrastructure.

What has become clear in recent years is that integrating dissimilar technologies is invariably more cost-efficient than the 'rip and replace' approach. Large companies have invested hundreds or thousands of man-years in developing core business applications and tuning them to achieve optimal performance in a mission-critical environment. Mainframe applications in particular (which in many cases were developed in-house and still offer unparalleled performance and data management characteristics) need to interact effectively with newer web-facing applications without losing any of their inherent value to the business.

In today's rapidly changing extended enterprise, it is no longer sufficient to provide fixed points of integration with high-overhead static translation methods. There is immense pressure on commercial organizations to be more responsive to customer needs, and to combine disparate sources of information to create new business opportunities, to be responsive to external drivers or to identify trends and changes. This means that today's integration tools must be flexible enough to handle data and process integration 'on the fly', around the clock, using lightweight or

heavyweight solutions depending on the characteristics of the application or task in hand. And with security and regulatory compliance high on the corporate agenda, all data integration must be conducted in a totally secure and manageable way.

Before considering the products and concepts that are evolving to cope with the new requirement for 'secure flexibility', let us first review how application integration has evolved.

Point-to-Point Integration

Application integration really became an issue in the early 1990s, when large companies first recognized the need to share resources between core centralized systems and the newer breed of client/server technologies. Early point-to-point integration solutions were often designed in-house, and were intended to cope with a relatively limited number of data format translations and network protocol conversions.

What characterized these early integration projects was that applications were tightly coupled; once joined together, they were not easily put asunder. Communication between the two applications involved was generally synchronous in nature, and the interfaces at either end needed specific knowledge about the data routing and translation processes involved. Any changes in this process required considerable programming effort and technical expertise.

Fixed point-to-point integration was quite sufficient for simple interoperability: internal applications exchanging data at predictable levels of performance, and needing to generate minimal reporting information. Indeed, this approach to integration was often extremely reliable and secure, but it was notoriously difficult to change. As corporate architectures and data exchange requirements became more complex, users developed more sophisticated requirements and began to look for a more generic, platform-independent solution to their integration needs. Support also became an issue: what started as a way of integrating legacy data with new logic became a legacy issue in its own right, and the overhead involved in maintaining fixed integration technologies discouraged all but the most essential changes.

EAI and Messaging Middleware

The solution to the problem came in two forms. With the arrival of MQSeries (now WebSphere MQ) and message-oriented middleware in 1993, users had at their disposal a backbone for application integration – a simple bus structure that offered asynchronous communication. This meant that applications could be loosely coupled: each app needed little or no knowledge about the data formats or protocols supported by the other. It also meant that the delivery of dispatched messages would be guaranteed by the backbone – once and once only. Now larger numbers of applications could interact and exchange data – a process that often involved complex sequences of actions – and the messaging infrastructure would manage

the data flow and ensure the integrity of the corporate information crossing the network.

A parallel development throughout the mid- to late-1990s was the emergence of new Enterprise Application Integration (EAI) technologies that reduced the overall integration effort. Companies such as BEA, IBM, Microsoft, and Constellar developed a whole range of message brokers, hubs, adaptors and connectors which converted data, interpreted calls, and performed routing and other functions. Application logic and data were 'wrappered' to a greater or lesser degree to separate them from the integration process and maintain their integrity.

The important thing about these products was that they were non-intrusive. They did not affect the application code itself, and provided a more generic mechanism for exchanging data and events between networked systems.

In terms of flexibility and support, these EAI and early messaging products made life a lot easier for the corporate integration specialist. They externalized the integration logic from the application itself, and allowed changes to be made relatively easily. But although many EAI functions were absorbed into the application servers that came along afterwards, they didn't offer the real automated 'on the fly' capabilities demanded by today's on-demand service-oriented e-business environment. Moreover, in many cases they actually added to the complexity of the infrastructure: companies often found that different EAI tools were deployed in different parts of the organization and could not then be successfully integrated with one another.

Application Integration in today's Web-Centric Environment

With the development of Internet-facing applications, many of the rules and objectives of enterprise integration have changed. One of the great positive developments of recent years has been the evolution of service-oriented architectures (SOAs) and web services, built around technologies such as .NET, SOAP, XML and J2EE. Although still in their infancy, web services overcome many of the fundamental problems with application integration, in that they offer to complement existing heavyweight interfaces with simple services that exchange information in a more transient, unstructured manner than with traditional applications.

Not all Internet-led changes have been beneficial for the large enterprise. One of the main problems is that many of the tools and technologies underpinning the brave new world of web services are considerably less mature and functionally capable than the enterprise products that they are intended to replace or with which they interoperate.

Web applications often proliferate in the furthest corners of the extended enterprise, well away from the disciplined data center environment, and the management tools used to control and integrate them are far from ideal. Simple Network Management Protocol, for example – the TCP/IP-based toolset for managing web-

based traffic – has taken many years even to approach the level of sophistication of enterprise management tools such as NetView. Similarly FTP (File Transfer Protocol), which was designed for nothing more than simple file transfer in the non-commercial world, is increasingly being used as an integration tool between secure, business-critical applications. And XML, which has revolutionized the way that data is deployed within web services, is nevertheless very resource-intensive and can seriously affect enterprise network performance.

This is where Service Oriented Architectures become so important. A properly designed SOA allows businesses to define precisely how each IT service and application component must behave for any given business process: how data security and integrity are maintained; how performance is measured and managed; how networking limitations and protocol conversion issues are resolved; and so on.

Within the SOA environment, IT departments attempting to bridge the gap between mature, high-performance enterprise applications and the new generation of flexible but relatively unmanageable web services need a consistent approach to integration; one which will allow them to focus on the relative performance, security, and integrity characteristics of each business process and transaction under their control.

There are many reasons why data centers need to achieve this level of consistency between enterprise applications and web services. This is not simply a service management consideration. In some cases the immediate driver is regulatory compliance. With legislation such as Sarbanes-Oxley, Basel II, HIPAA, ISRS and the COBIT Act imposing a high level of data and application transparency on businesses across the world, technologies that contain little or no function for ensuring the integrity of data need to be tightly controlled (particularly when the data being managed moves outside the company boundary).

But the main reason for moving towards an SOA philosophy is to improve business agility. Once the controls are in place to manage system resources, application services and data on the fly, businesses are in a much stronger position to respond quickly and efficiently to changing market conditions, competitive pressures, and other commercial imperatives. For large, complex IT functions the SOA is likely to be a long-term goal as it can involve considerable organizational change; but it is a change that needs to be made. Companies that have already invested heavily in application integration are particulary keen to make the transition, as they appreciate the level of flexibility that a SOA can provide (and they know that their competitors are heading in the same direction).

The IBM Enterprise Service Bus: One Size does not fit All

If the SOA is the architectural solution to enterprise-wide integration, the technical framework on which the solution is built is the Enterprise Service Bus (ESB). This is a relatively new concept for many businesses (although the marketplace is already very competitive), and it takes the messaging backbone to a higher level,

one that is required to create an efffective SOA.

ESBs perform a number of core services: routing messages between services; converting transport protocols and message formats between requestor and service; and handling various types of business-generated events. Furthermore, they provide interpretation of XML-based content, which gives them an essential routing capability for web services.

ESBs need to handle a very diverse range of traffic types, as they play a fundamental part in achieving the integration consistency discussed earlier. Nimble, lightweight low-cost web services are increasingly combined with business-critical, highly available processes and transactions, built on other standards or even fully customized and proprietary. With multi-hop considerations and bandwidth

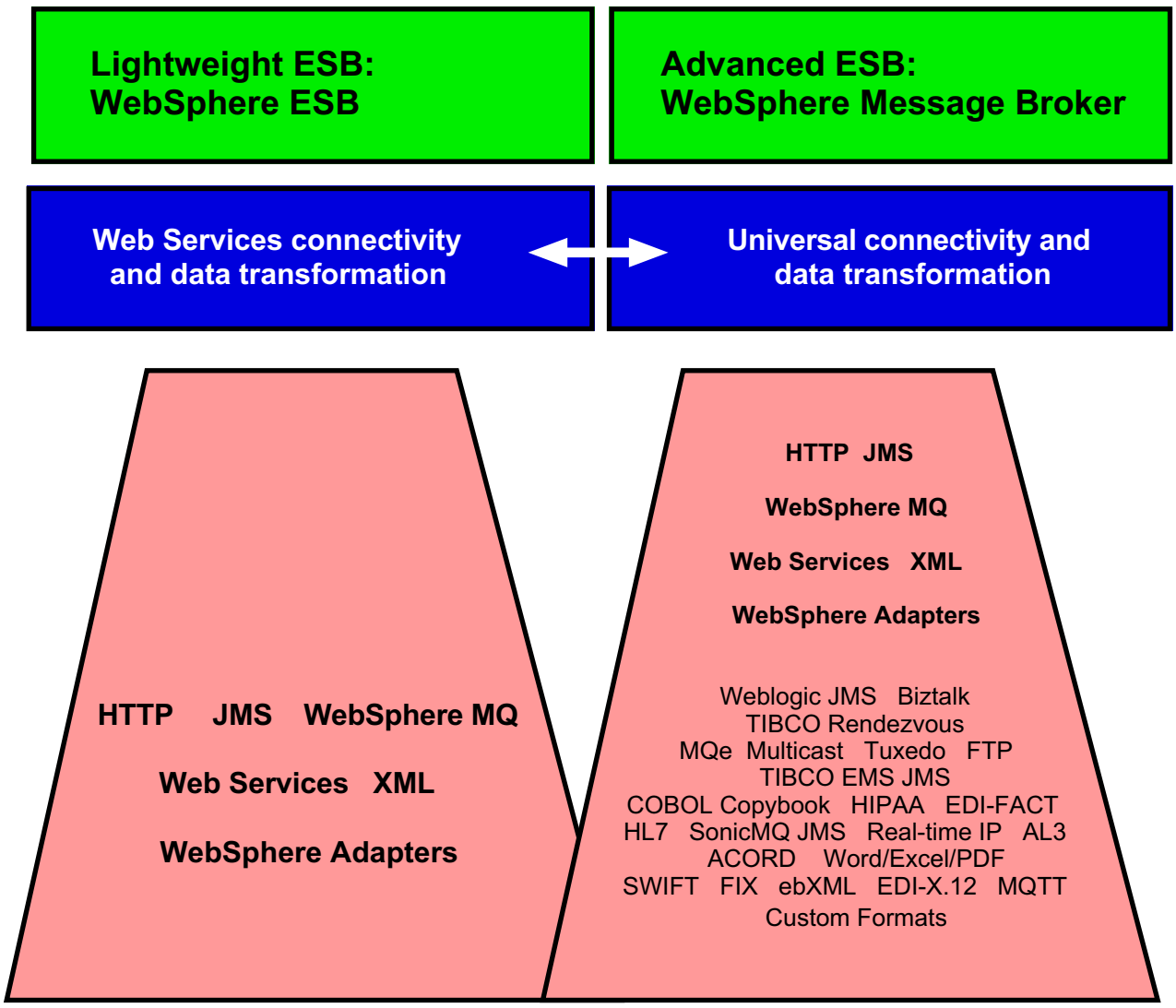

Figure 1: IBM's WebSphere ESB and WebSphere Message Broker

constraints across the Internet, it is essential to keep traffic overheads as low as possible for web services; conversely, with mainframe-oriented business-critical data on the internal network, bandwidth is far more manageable and the 'value' of the data in transit demands that a more heavyweight delivery mechanism be used. The need to support these two extremes and everything in between, while giving the business the flexibility needed to change service management priorities quickly and easily, is at the heart of the ESB philosophy.

All ESBs are unique and, in view of the required diversity, ESB implementations may not be built around a single product; they may need a combination of products and services that interoperate to provide optimal routing, conversion and interpretation services for each specific business task. IBM's recently announced ESB strategy distinguishes between 'advanced ESB' for combining mature enterprise-level application messaging with web services traffic; and what we would term 'lightweight ESB' for web services and peripheral network use. As shown in Figure 1, IBM now has separate offerings to target these two specific needs – the new WebSphere ESB and the more mature WebSphere Message Broker.

WebSphere ESB (WESB) provides connectivity and integration for web services, and is designed for situations where flexibility, low-overheads and ease of operation are high priorities. It offers a general-purpose web services backbone which is likely to appeal to a broad range of new and existing customers.

Message Broker (WMB) offers universal connectivity and any-to-any data transformation, and is ideally suited to more complex heterogeneous environments. It originally grew out of WebSphere MQ Integrator, and it builds on MQ's 'exactly once' asynchronous delivery mechanism to provide a range of availability options, using business rules to determine the required quality of service. Message Broker also benefits from WMQ's unprecedented level of support for mainframe and mid-range applications, including tools for integrating key CICS-based transactional applications with J2EE and .NET. WMB version 6 also brings with it direct integration with any JMS provider, improved performance and an even wider range of transformation options.

IBM does not foresee WESB and WMB being used in isolation. Customers are very likely to combine the products; the latter would sit at the centre of the enterprise, typically close to centralized mainframe-based data and resources, and the ESB would perform more web-service-based functions in distributed locations. The two products share a common heritage and are both closely integrated with the other components of the WebSphere family, so smooth interoperability is guaranteed. Morover, both products can be used in conjunction with the WebSphere Process Server, which works alongside the messaging tools, providing higher-level monitoring and analysis of the way that traffic and events relate to changing business processes.

There is, of course, a broader market for WebSphere ESB in smaller companies supporting a relatively small number of standard web-oriented data formats. For such businesses, the new product alone may be an attractive choice. However, we anticipate most interest among the larger companies that have an immediate

need for ESB support and which will benefit from the distinct characteristics of the two products.

The Adoption Model and Self-Assessment

Having the technology in place to support a broad range of connectivity standards, protocols and styles is only half the battle. For most companies that support heterogeneous applications, the idea of an Enterprise Service Bus is very attractive. Getting to the point where the ESB can be used to its full potential, however, is another issue entirely. For many, it might be unclear how technology of this kind can be used specifically to help address the organization's strategic business objectives, and to assist the enterprise in moving forward to a service-oriented architecture while continuing to maintain and enhance its existing base of legacy technologies.

For users SOA often requires a period of self-analysis, at the business level as well as the IT level, when the customer examines its current portfolio and the requirements of the business, and sets out a roadmap that will allow the IT function to evolve into a service-driven infrastructure, closely aligned with the business.

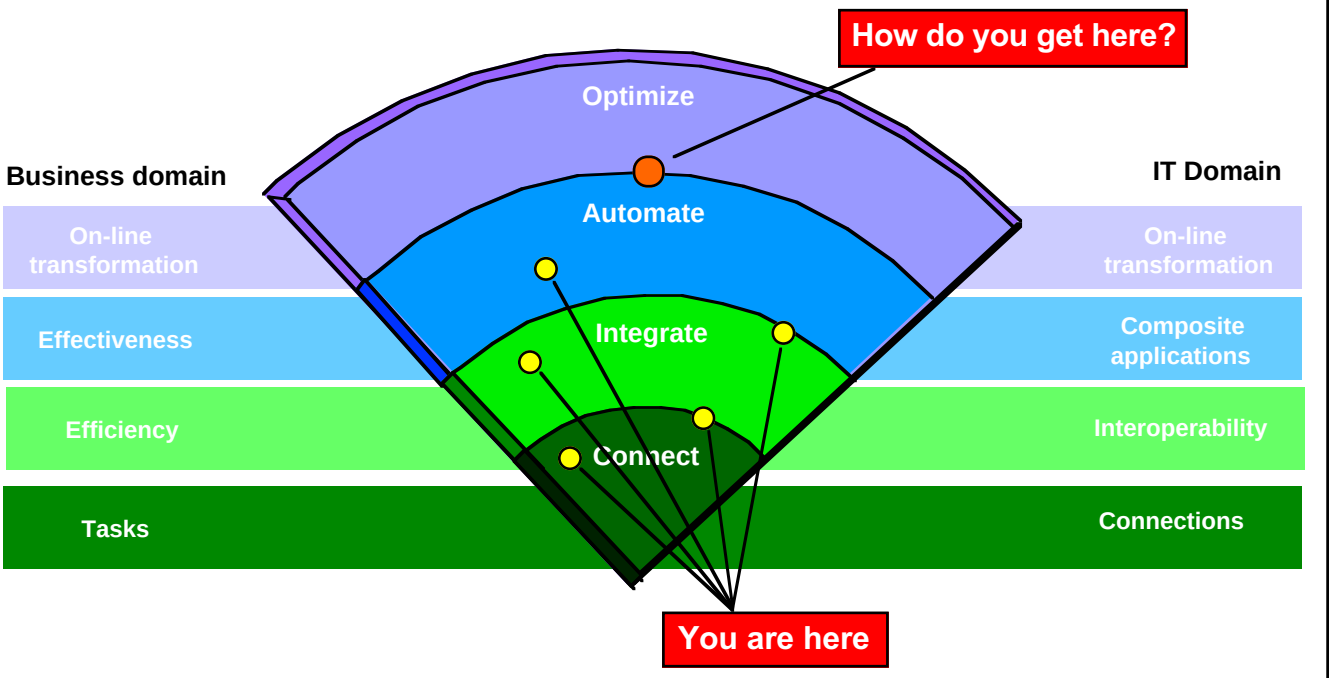

Figure 2: WebSphere Business Adoption Model

In response to this need, IBM has recently launched the WebSphere Business Adoption Model (http://www.ibm.com/websphere/soa_assessment), which incorporates an on-line SOA Self-Assessment. The model describes the stages through which companies typically move as they start to adopt a service-oriented approach to developing new business applications:

* *Connect*: Basic connectivity between applications that allows secure and reliable exchange of data.

* *Integrate:* A more flexible framework that allows interoperability between heterogeneous environments, and begins to overcome the barriers that inhibit integration of web services and traditional enterprise applications. This is described as the 'efficiency layer', as it is the stage of development where businesses are primarily optimizing and streamlining their application delivery services and making increasing use of their IT assets.

* *Automate:* At this level, businesses are putting automated tools and procedures in place to align business and IT processes. This is the 'effectiveness' level, where application integration should be sophisticated enough to contribute substantially to corporate revenue growth and cost reduction.

* *Optimize:* At the 'nirvana' level, described by IBM as 'on-demand transformation', business and IT processes and services are closely aligned and highly automated, to the extent that IT can enable and support substantial changes in business strategy with minimal operational disruption.

In terms of the approaches to application integration that we discussed earlier, one of the most significant features of this model is that the integration process becomes more flexible and less intrusive the further the user progresses, with the technical practicalities becoming increasingly transparent to the pursuit of true IT/business alignment.

The final transformation stage remains a pipe-dream for many heterogeneous businesses, and even the more pioneering integrators of web services and enterprise applications are likely to place themselves at the 'integrate' level or the more tentative stages of 'automate'. Indeed, some large enterprises would no doubt position themselves at more than one location on the integration road, with some internal operations more able than others to adapt to a service-based development approach.

IBM's SOA Self-Assessment allows customers to gain a clear picture of where they are today and where they need to go (in terms of the Adoption Model) to embrace the service-oriented flexibility offered by product suites such as WebSphere.

Once the self-analysis is complete, they receive a report which outlines their level of SOA sophistication and details the options available to them and the benefits that can accrue.

Bottom Line

Today's heterogeneous IT environment is a very different beast from the secure internal data center of a decade ago. Integration is as much about demonstrating data security and integrity (to satisfy regulators and stakeholders) and about responding rapidly to changing business requirements, as it is about handling multiple data structures and routing methods.

The emergence of the SOA and the Enterprise Service Bus are a direct response to the integration challenges facing the enterprise. While we believe that the ESB concept is still in the early stages of development, products such as WebSphere ESB and Message Broker offer a very rich implementation of the concept as it stands today. Enterprises that need to improve their level of IT responsiveness by embracing web-based technologies (built around XML, SOAP, J2EE and .NET) while maintaining and enhancing the value of their existing investments should consider the ESB route very seriously.

An SOA Self-Assessment might be a very good place to start, as it helps businesses to understand which ESB products are needed within the enterprise and how they can be expoited most effectively.

This paper was sponsored by IBM. The author, Mark Lillycrop, is Chief Analyst at Arcati Research.