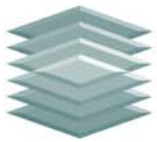
WebSphere Application Server Enterprise V5: Building, Deploying, and Managing On-Demand Applications

Web Application Infrastructure/
Product Lifecycle Management Infrastructure

D.H. Brown Associates, Inc.
222 Grace Church Street
Port Chester, NY 10573
(914) 937-4302
www.dhbrown.com

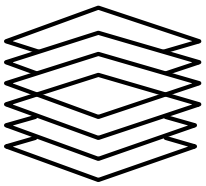


May 2003

About D.H. Brown Associates, Inc.

D.H. Brown Associates, Inc. (DHBA) is a leading research and consulting firm that concentrates on the strategic analysis, assessment, and evaluation of technologies, products, and market trends in the Information Industry. DHBA thoroughly understands and continuously updates its knowledge of the core technologies and trends underlying current and future products. DHBA's services enable its customers to consistently fashion well-informed decisions adapted to their needs.

Acceleration in technological innovation has driven price/performance by over 30% annually for open systems since 1983. This technological explosion allows unprecedented opportunities for world-class corporations, but demands far greater discipline and attention to detail. DHBA provides its clients with a high-level, balanced perspective coupled with the technical depth required for making tactical and strategic decisions that will capitalize on existing and emerging technology. For additional information, please contact marketing@dhbrown.com.



WebSphere Application Server Enterprise V5: Building, Deploying, and Managing On-Demand Applications

Pierre Fricke, EVP and Research Director

EXECUTIVE SUMMARY

IBM recently introduced the latest version of its flagship application server product – WebSphere Application Server Enterprise V5. With this release, IBM has significantly strengthened the product's capabilities in support of a Service-Oriented Architecture (SOA) that begins to deliver on the vision of on-demand applications.

Companies are beginning to use web services as a means of delivering application functionality to all the communities with which it works. As they do so, and as the integration of business processes within and outside of an organization increasingly becomes the means of delivering services and achieving interoperability, new tools and enhanced deployment environments will be required. These new capabilities will be targeted at supporting the efforts of developers who are working to help their companies stay competitive. Under current economic and business conditions, faster time-to-market of new applications (and changes to existing applications) and lower IT budgets pose two constraints to IT organizations as they attempt to deliver on their commitments to the business.

A service-based approach is evolutionary and builds on several generations of distributed computing solutions that began during the client/server era. Since then, two trends – a gradual but accelerating move toward standards and a return to a consolidation of distributed resources in order to gain competitive advantage – have driven the growth of new, distributed software infrastructures. Today, the application server represents the core of such an infrastructure. IBM, with its WebSphere Application Server family, is one of the top two vendors in this highly competitive but maturing market.

Successful vendors in the application server market will have to meet several key requirements:

- First, they must offer an integrated software stack that incorporates all the elements required to support the development, deployment, and management of service-based solutions. This stack must also support the incorporation of point products that may exist within an enterprise as a result of prior decisions.

- They must provide tools that are tightly integrated with the rest of the software stack.
- They must make the development of new business logic, and the integration of that business logic with new and existing applications, a straightforward exercise.
- They must provide tools that enable the construction and integration of business processes. The deployment environments they offer must support the "choreography" of business process execution through the use of workflows.
- The tools they offer must approach all development and deployment activities with an Integrated Development Environment (IDE) that provides all the required development capabilities – for coding as well as business process design. The IDE must also give Java developers the extensions they need to automate their building of services and increase the performance and flexibility of applications.
- It is absolutely critical that solutions allow applications to be changed in response to dynamic market and competitive conditions with little or no disruption of service.

In creating WebSphere Application Server V5, IBM focused on all of these factors and has brought to market what it positions as a next-generation application server for delivering on-demand applications. Marketing messages aside, IBM has created an arguably strong product offering that not only serves as an enabler for SOA adoption by enterprises, but offers Java developers new and valuable capabilities that make the J2EE specification the foundation of an emerging and more powerful web services platform. IBM has also introduced some interesting resource management capabilities that D.H. Brown Associates, Inc. (DHBA) believes should raise the competitive bar in this important product space. IBM does face some challenges, including a perhaps undeserved perception that WebSphere implementations require a high level of professional services involvement that may come at a high price. IBM's dedication to enhancing the integration among elements of the product family, improving development efficiency through better and more highly automated tools, and strengthening its position in relation to solutions that address specific vertical markets are among the ways IBM mitigates this challenge.

TABLE OF CONTENTS

IT PREPARES FOR WEB SERVICES

THE EVOLUTION OF DISTRIBUTED COMPUTING

As a means of developing, deploying, and maintaining applications, web services represents a relatively new concept. The subject of a great deal of attention over the past two years, the widespread implementation of web services solutions is only beginning to gain serious consideration.

Underneath the hype, the web services idea is actually quite evolutionary – building upon concepts and technologies that have been available for some time. Among these concepts are object-oriented programming (represented by C++, Java, C#, and other development environments), distributed object computing (as exemplified by CORBA and DCOM), and multi-tier, distributed architectures. These technologies have been used for years by IT organizations to support their user communities and, ideally optimize the organization's use of resources. The trend started over twenty years ago with client/server computing, and continues with great energy today. These concepts are now being tied together to form a body of technologies called "web services."

A major driver behind this acceleration, of course, has been the inception and increasing usage of the web to conduct business. The initial wave of web applications was developed in many cases by small, department-level IT functions employing immature tools with little or no help from corporate IT organizations. Today, the importance of a strong, efficient web presence is a *sine qua non* for success. The days in which organizations create a web presence without planning, context, or corporate-level support are over. Today's corporate IT organizations endeavor to gain greater control over the often wide array of infrastructure and applications that may exist within a given corporation. The goal is to gain competitive advantage through a more coordinated and integrated approach to delivering IT applications and services.

From the business perspective, this revived interest in a more centralized IT strategy is easy to understand. The technologies and tools that support the Web services concept are now widely viewed as applying IT resources across all the communities with which that organization must interact (customers, partners, suppliers, governments, and so on). And these services are provided in a targeted, personalized, secure manner. Improved business process efficiencies, greater customer satisfaction, and the ability to deliver goods and services at a much lower cost stand out as the potential benefits of a service-based approach.

From a technology viewpoint, the introduction of the application server ranks among the most important innovations of recent years. Application servers have been around for almost ten years now. They originally provided the runtime foundation for distributed, multi-tier architectures that surfaced during the 1990s as a means of offloading application overhead from desktop systems, while

allowing decentralized control over which applications were going to be supported. As the decade progressed, market maturity and the growing need to achieve a high level of application portability across existing operating system platforms stimulated two significant market dynamics: merger/acquisition activity that reduced the number of serious application server vendors; and the technology convergence of application server products toward J2EE. It is noteworthy, however, that the one strong exception to the latter dynamic is Microsoft, which offers all the elements of an application as part of its .NET environment.

Driven by interest in the new web services paradigm, and the aforementioned cost, flexibility, and time-to-market drivers, customers look increasingly to make decisions that go beyond the application server – focusing instead on total solutions that provide as much functionality as possible to meet their particular needs. As a result, offerings from a number of vendors form a more comprehensive software "stack," including the application server, an integration server, portal, identity and security management, and tools. The internal integration of this stack is rising as an important decision criterion for users, who consider it a metric of how quickly and cost-effectively solutions can be built and deployed. At the same time, since a given organization may have standardized on one or more point products from other vendors, the stack must be flexible enough to integrate these products easily. Vendors are also beginning to focus some energy on solutions that meet the needs of specific vertical industries, either by creating the needed technology internally, acquiring it, or forming close business and technology partnerships.

The greatest challenges IT decision makers face span integration, reduced costs, more intense time-to-market demands, and flexibility. Assuring these decision makers that solutions exist to meet these challenges will pose an adoption hurdle. Vendors must demonstrate leadership in standards and product innovation and this ability will separate the winner from the losers in the emerging web services arena.

The important role of the application server as the core element of the software stack will not change. Successful vendors understand this, and will enhance their products to better address the important challenges discussed above.

EVOLUTION BRINGS NEW CHALLENGES

Delivering on the benefits offered by web services poses an array of hurdles. Significant challenges face organizations as they prepare to move forward with solutions based on a web services approach. These challenges can be summarized as follows:

- *Coming to terms with complex infrastructures* – Given the evolutionary nature of web services, bringing together existing infrastructure elements, applications, and data sources poses a significant challenge for organizations. Many began dealing with this issue a number of years before web services became hot,

driven in great part by the need to address the web as a means of doing business. New and existing software vendors responded with products designed to address this issue. Web services has been perceived by many as easing this challenge, despite its reliance on standards that were not generally incorporated into these integration solutions. In fact, while web services holds the promise of facilitating the interoperability of IT resources *across* environments (which may employ a wide variety of platforms and technologies), integrating legacy assets still presents a significant hurdle.

- *Challenges in creating new applications and business logic* – Application developers must deal with a number of new technologies and standards generated by web services. Interoperability standards, based on XML, that support the required interoperability within a secure environment, must be learned and understood. Developers must also incorporate them into the development tools used to create web services solutions. Given the rather fluid nature of standards efforts to date, fixing a set poses a challenge. Likewise, more fundamental skills, such as those required to fully leverage the robust features of an EJB-based approach (if that is indeed the choice), remain difficult to locate. The technologies themselves present development challenges. To move forward, most organizations will need tools support that adds both automation and ease-of-use features to the development process. They also require the augmentation of the technologies that improve both performance and the ability of applications to handle transactions – two critical characteristics of web services implementation.
- *Making business systems more responsive to changing market conditions* – Through the web, organizations possess the potential to make changes to their business models quickly. Those that achieve this goal gain a significant competitive advantage. The ability to respond quickly to rapidly changing market conditions has therefore become a major competitive issue. The implications are that the design and redesign of applications must be made faster and easier – and entail a component-based approach to application development and deployment. The adoption of software component technology, often the subject of very aggressive growth predictions by analysts, will finally reach its potential over the next few years as web services begins to catch on in earnest. Responsiveness will also be facilitated through the use of tools that will allow business processes to be constructed and modified quickly – likely by bringing business analysts into the development cycle.
- *How to "do more . . . with less"* – Competitive pressures, coupled with a volatile economy that has affected budgets, have forced IT organizations to deal with these challenging issues with fewer resources, and in less time. The result has been a more cautious (although likely not longer) decision cycle for IT product purchases, with decisions requiring much stronger justification in terms of demonstrable and substantiated Return On Investment (ROI) and/or Total Cost of Ownership (TCO).

What does this mean for solution providers? They face a community of prospects and customers who will require them to demonstrate the cost-effectiveness of the solutions they provide. Solution providers must also show how their solutions can ease the integration challenge both inside and outside their company walls, must demonstrate their support of standards, and the degree to which their products can help developers master and leverage new and existing enabling technologies to build competitive solutions in record time.

WEBSPHERE APPLICATION SERVER ENTERPRISE V5: IBM'S NEXT STEP IN MAKING WEB SERVICES A REALITY

A NEW OFFERING IN THE WEBSPHERE APPLICATION SERVER FAMILY

IBM has recently announced the availability of WebSphere Application Server Enterprise V5. The Enterprise version of WebSphere is one of several application server offerings provided by IBM, each designed to address a different set of customer and application needs.

The WebSphere Application Server family is shown in Figure 1. The entry point for users is WebSphere Application Server – Express, a product that facilitates the construction and deployment of static and dynamic web pages and web applications, leveraging technologies such as Java Server Pages and Java Servlets. WebSphere Application Server, the second offering, forms the core offering of the family, and adds full J2EE 1.3 support including Enterprise Java Bean (EJB) capability. WebSphere Application Server can be deployed as a single server or through a Network Deployment configuration that supports manageable clustering in highly available distributed environments. WebSphere Application Server for z/OS allows owners of systems that use z/OS and OS/390 the full capabilities of the WebSphere Application Server. It also leverages features unique to these environments such as integrated service level agreement management, transparent recovery from major subsystem failures, transactional control over mainframe resources, and scaling across a Sysplex.

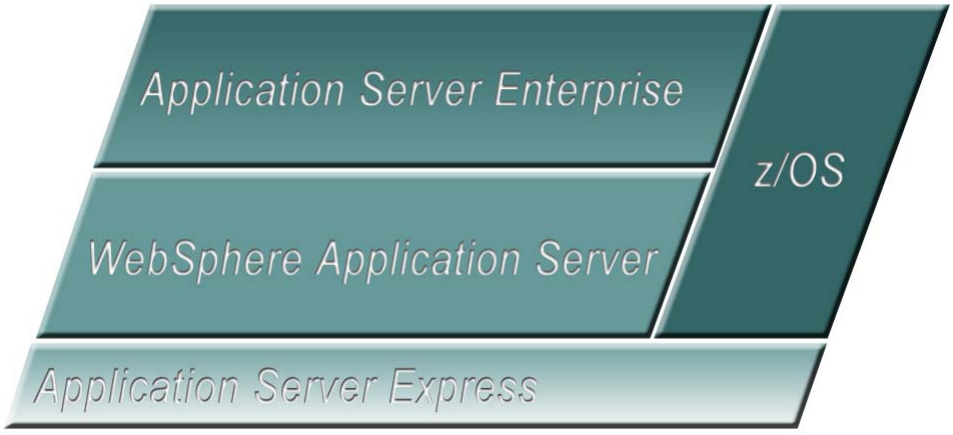


FIGURE 1
The WebSphere Application Server Product Family (IBM Corporation, 2003)

WebSphere Application Server Enterprise offers the most advanced programming model within the family of offerings and stands out as the product that facilitates the integration of J2EE components and legacy assets into business processes. This is achieved by leveraging web services and by enabling the easy construction of workflows among these assets that are both flexible and yield good performance. IBM touts WebSphere Application Server Enterprise V5, IBM's latest version of this product, as a next-generation application server.

ADDRESSING THE INTEGRATION CHALLENGE

ADOPTING A SERVICE-ORIENTED APPROACH

To understand the magnitude of the problem at hand, it is important to note the extent to which organizations have deployed a mix of back-end systems, middle-tier business logic, and databases. For an IBM customer, for instance, back-end systems based on CICS and IMS (and associated legacy applications and databases) operating on mainframes may need to interoperate with J2EE applications running on application servers. As needs change, additional business logic may be needed, implemented presumably in J2EE. In addition to integrating these assets, they must be turned into services that can be described, identified, and delivered. The need for the resulting services to interoperate with services and applications outside the enterprise adds further complexity. The legacy assets may be large, numerous, and implemented using a variety of technologies and languages.

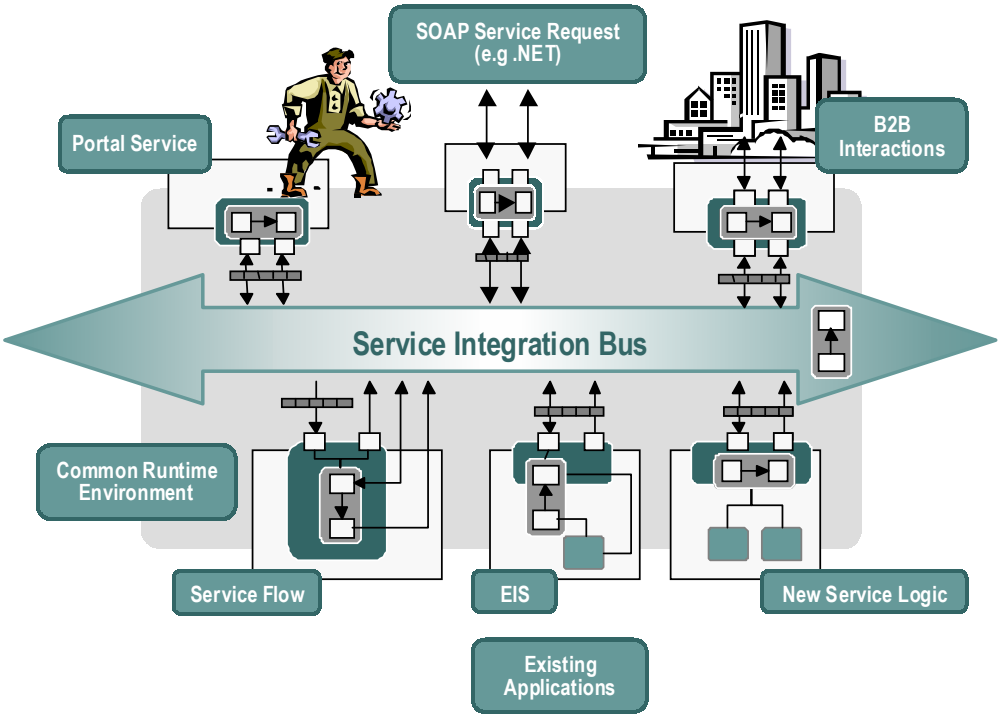


FIGURE 2
A Pictorial Representation of IBM's Service-Oriented Architecture, IBM Corporation, 2003

As organizations begin to address the integration challenge, a service-based solution can yield real benefits. For IBM, this takes the form of a *Service-Oriented Architecture (SOA)* as shown in Figure 2. Simply put, an SOA approach allows all software assets to be represented as services, regardless of their age, technology base, or geographic or organizational location. Every service represents a self-describing asset and therefore any change in the implementation of a service does not impact overall application functionality. The SOA works independently of the communications stack allowing services to be invoked through a variety of protocols, including Simple Object Access Protocol (SOAP)/HTTP, JMS, MQ, RMI, and others, by leveraging IBM's Web Services Invocation Framework. This

framework is part of WebSphere Application Server Enterprise V5 and was donated by IBM to Apache.

IBM plans to establish a leadership position vis-à-vis SOA by creating an environment that is based on industry standards, on a component-based approach, and on tools that facilitate and accelerate the development and deployment process. This will be achieved by demonstrating to its customers that applications that allow the service-oriented model to be implemented, and which means they allow them to achieve the advantages this model has to offer, can be built within the cost, time-to-market, and competitive constraints they must deal with. Similarly, IBM will work to assure its customers that the SOA it employs will allow a level of flexibility that makes them highly responsive to the markets they serve and the competitors they face.

The SOA concept lends itself well to solving the integration problems discussed earlier. To see why, we begin with a general view of a service, which consists of three major elements. The first is the *implementation artifact* –the actual asset (EJB, Java class, CICS application, and so on); the second is the *interface* – a description of the operations and messages that are available from the artifact; and the third, *binding* – the means by which an artifact is accessed. All services are described in a Web Services Description Language (WSDL) document. The initial task, then, is to create services from the existing assets using this model. The development environment used to accomplish this will be described later in this paper.

It is important to note that, while much of the discussion of SOA here relates to the highly visible concept of web services, any kind of service – including one created from an EJB, a Java object, and so on, can be a service within the SOA.

The services created are now available to be integrated into business processes. In WebSphere Application Server Enterprise V5, IBM has added the concept of *choreography* – the ability to define and manage a workflow that can be designed, deployed, and controlled within a single environment called the *workflow engine*. This engine, fully integrated with the application server, includes what IBM calls the *Process Choreographer* component. As the name implies, this component can be used to manage workflows within business processes in an SOA environment.

The concept of choreography is not new for IBM. In fact, choreography of what IBM calls "microflows" was supported in previous versions of the company's VisualAge for Java Enterprise Edition. A microflow is a short-running business process segment defined on the level of individual transactions. For example, the input to a flow that is part of a Customer Relationship Management (CRM) system might be a customer number. This flow could trigger process steps that (1) make a connection through a JCA connector to a CICS system containing customer information, (2) request and obtain details on the customer, (3) disconnect from the CICS system, and (4) obtain output. To the application developer, however, the workflow steps are transparent – he or she need only deal with the details of inputting the customer number and retrieving the results.

The Process Choreographer takes this a step further – by adding the capability to manage "macroflows." Here, the microflow described above could form the first step in a series of such flows (which, in this example, could include determining customer status, checking credit rating, issuing credit, and so on). Macroflows can be long running, involve multiple transactions, and can connect to other organizations using SOAP (SOAP over JMS support is planned by IBM in WebSphere Application Server Enterprise V5.02). Macroflows can also branch off to do EJB operations as part of implementing a step in a flow, and can send off JMS messages.

The language used to implement both macroflows and microflows is called Flow Definition Markup Language, a precursor to BPEL4WS (Business Process Execution Language for Web services). BPEL4WS (which IBM helped to define) is expected to become the future workflow standard, and IBM plans to implement it within WebSphere as part of an upcoming release.

EMPOWERING THE DEVELOPER

We noted earlier the need to support an architecture such as the SOA with tools and technologies that support increasingly severe time-to-market constraints, lower costs, and, by implication, an emphasis on components and reuse. At the same time, developers understand that to truly reap the benefits of a service-based approach, it must be possible to easily integrate, change, and manage business processes using advanced tools. Finally, features must be available to developers within those tools to augment technologies that support application development (i.e., Java) and web services to more robustly support the transactional, performance, and security demands of service-based applications.

To illustrate this point, one can note the contrast between the approach taken by developers using the choreography features described above and the direction taken by IBM's WebSphere Business Integration. The latter case takes a higher level business process-focused approach, due in part to the foundation technology used (based on the former CrossWorlds product). Here, a greater degree of "out of the box" functionality is employed, requiring few Java programming skills. This functionality will evolve toward packaged vertical solutions. In contrast, the choreography capabilities are targeted at Java developers – Java is in fact a prerequisite for using the product. The focus lies in building new applications, integrating them with existing applications, and, with new features to be described later in this paper, giving the Java programmer the ability to improve application performance and transactional reliability.

The product IBM provides for developers to accomplish all of these tasks is *WebSphere Studio Application Developer Integration Edition*. Based in great part on an existing product (WebSphere Studio Application Developer), the product allows customers to visually develop business processes that incorporate a variety of important features to support processes and flows. A visual editor, designed specifically for this purpose, allows developers to construct business processes.

The tool also incorporates the capability to specify *activities* that support the execution of individual process steps. For instance, the activity "invoke service" allows the process to call a service as defined in WSDL. The activity "invoke EJB" allows a remote method on an EJB to be called. The activity "invoke process" allows a business process to call another business process. This last functionality is particularly interesting because it illustrates not only the capability, but also the ability to "nest" business processes.

A number of other activities allow business processes to be executed and managed within an SOA approach. Business processes built using the tools in WebSphere Studio can also be designed to include *control links* that can be used to allow, restrict, or reroute business process execution flow; multiple business processes can be executed in parallel when it makes logical sense to do so, making it possible to improve overall performance.

One of the key requirements in addressing the integration challenges discussed earlier is the ability to build high-quality adapters that allow the requisite interoperability among existing applications and with new business logic. Within the context of a service-based approach, this first means being able to build and deploy rich adapters that are based on open standards and that include adapters supplied for popular applications such as SAP and CICS. WebSphere Studio allows developers to employ adapters based on the Java-based standard JCA 1.0, and then leverage a set of wizards to create a service offering easy access to a variety of back-end applications. One interesting example is the wizard to create a 3270 service that provides the capability to navigate 3270 screens. The concept of microflows described earlier is employed here as the means of permitting the navigation of screen flows – an example of the use of flows within adapters.

IBM's Java Connector Architecture (JCA) support includes enhancements to JCA 1.0 that build on that standard's support for the pluggability of enterprise applications into a J2EE-based application server runtime. This is achieved through the use of adapters supplied by vendors or other parties. Specifically, IBM's JCA Tool Plug-In, an IBM extension of the JCA standard, allows these applications to be pluggable into WebSphere Studio, making it easier to build services out of existing applications and integrate them into business processes. IBM is contributing this plug-in capability to the JCA 2.0 standards activity; when JCA 2.0 becomes part of the Java specification, the plug-in will be usable within other popular IDEs.

With WebSphere Studio Application Developer Integration Edition, IBM provides developers with a feature-rich environment for building and deploying business processes, creating new business logic, and initiating and integrating services from all application assets to support business process execution in a way that attempts to optimize performance, control, and flexibility. WebSphere Studio is a highly integrated environment that encompasses the ability for developers and business analysts to do their work together, and within a common framework. The foundation for this environment is Eclipse, a "pluggable" platform designed

by IBM and others to allow the building of IDEs that encompass a wide array of functions and capabilities. Eclipse.org, a consortium of tool vendors, manages the process of development environment creation and integration. IBM has certainly leveraged Eclipse in WebSphere Studio, which adds the capabilities described in this paper. The remainder of this paper offers developers and business analysts the full breadth of tools and technology support required to make IBM's "on-demand" vision a reality.

MAKING TRANSACTIONS MORE EFFECTIVE IN THE REAL WORLD

Despite the best intentions of all parties, business processes sometimes fail. A service-based approach to integrating business processes must take into account scenarios in which a change in conditions results in the need to decommit specific transactions. In other cases, the business process might be complicated by the need for one or more one-phase commit resources to be involved in a coordinated transaction. The web of interoperability made possible through the use of a service-based approach – often spanning a number of groups, organizations, and communities – can escalate the potential ramifications of an inability to decommit transactions or to coordinate multiple transactions. These ramifications can be costly in dollars, time, goodwill, and trust.

In an ideal scenario, a deployed business process executes based on the conditions built into its controls, including inputs from other applications, or via human interaction that represents events used to trigger specific flows. Transactions are committed when specific conditions are met, but what happens when one or more of those conditions is no longer valid and a committed transaction needs to be reversed? A simple example can be used to illustrate the issue. Suppose an employee makes a reservation for an airline ticket for a deal that will expire at a specific point in time. All transaction steps are committed, but failure to meet the deadline or actually take the step of purchasing a ticket could cancel the need to follow through on the steps of the transaction.

In this and other similar cases the transaction involves multiple operations, executed over a defined span of time, which will have to be undone. Here is the problem: This activity falls outside the traditional boundaries of a transaction. To deal with this, IBM offers what it calls *intelligent compensation support*. In the example above, a service would presumably exist called *book plane reservation*. Compensation in this context involves the definition of an "undo" service, which in our example could be called *cancel plane reservation* and would be logically associated with the original service in what IBM calls a *compensation pair*.

In practice, when a particular process starts, it launches a *compensation sphere* – a set of data that represents the activities executed by the business process. This sphere is registered with a *compensation coordinator*. As the process executes, all activities are tracked and a record is stored in a database. If for any reason a failure occurs, or if there is any other reason to cancel the transaction, the compensation coordinator

facilitates the execution of the business process in reverse order. The compensation mechanism is completely transparent to the execution of the business process itself, making it non-intrusive in terms of performance or flow. No programming is required on the part of the application developer to take advantage of this feature.

The newest version of WebSphere Application Server Enterprise supports connectivity with back-end systems in two more very important ways. In addition to the dynamic JCA-based adapter support discussed in the previous section, IBM has added a feature called *last participant support* to WebSphere. To understand the relevance of this feature, note the reality of circumstances regarding existing applications and systems. Many employ a one-phase commit capability when involved in the coordination of a transaction, a condition common to a large number of packaged applications. This is an issue encountered with CICS transactions as well. Dealing with this scenario is challenging from the integration standpoint and is a frequent problem.

Last participant support basically involves the coordination of the use of a single, one-phase commit resource with any number of two-phase commit resources within a single global transaction. J2EE developers who have had to solve this problem in the past performed a potentially high level of hand coding. IBM's approach reduces significantly the amount of work involved. The methodology used can be described as follows:

- When transaction commit takes place, the two-phase commit resources are prepared using the two-phase commit protocol.
- If this preparation is successful, the one-phase commit resource is called, and it commits the transaction.
- At this point, the two-phase commit resources are either committed, or rolled back, depending on the response received from the one-phase commit resource.
- A risk exists in the situation where the one-phase commit resource does not complete. This risk is of a mixed outcome in the global transaction, in which case the two-phase commit resources are rolled back – but the outcome of the one-phase commit resource may be unknown. To deal with this, developers need to configure their applications to operate through an extended deployment descriptor. IBM's TX service can also be configured to write a log entry, prior to the point where the one-phase commit resource is agreed to. In this manner, the proper heuristic reporting of the failure is cataloged, rendering it useful to the process of rollback.

While last participant support defines the mechanics of coordinating single one-phase and two-phase commit resources, there is an important, additional requirement – the ability to define a session, similar to a "unit of work," that manages multiple one-phase commit resources and directs them toward reaching a specific and common outcome. To address this, IBM has introduced the *activity session service*, which provides a scope of work – an *ActivitySession* – in which non-

recoverable one-phase commit resources are managed. The ActivitySession typically takes the form of a session bean. When this service starts, it calls methods on other beans that are part of the transaction flow. These beans then coordinate one-phase commit resources. The session ends, and if there is a failure, the failed resource identity is reported, and the appropriate action can be taken.

ActivitySessions can be used outside the context of services-based applications, and can also play a role in improving performance in ways that are outside the actual scope of the transaction. An example is the process that is executed when a user signs on to a website and initiates transactions. Typically, the user's web browser goes to a Java servlet and then establishes an HTTP session. If the session bean involved executes two transactions, the sequence of steps would be to first activate the EJB involved, execute the first transaction, and then passivate the bean. This sequence would also have to occur in a similar way for the second transaction, followed by the termination of the HTTP session. If an ActivitySession is used, two steps can be eliminated: the passivation of the EJB after the first transaction completes, and the activation step for the second transaction. Here, the EJB can remain in the application server container throughout the duration of the ActivitySession, a benefit that is made possible through the use of a user sign-on activation policy. The end of the ActivitySession is triggered by the completion of a user sign-on period governed by that policy.

Given the integration requirements of a service-based solution, compensation, last participant support, and the ActivitySession represent effective ways to resolve the daunting task of bringing together the transactional characteristics of existing applications with business logic built using J2EE. They are therefore significant enhancements to WebSphere Application Server Enterprise.

BUILDING AN INFRASTRUCTURE TO HELP MAKE BUSINESS MORE AGILE

Earlier sections of this report recognized key issues facing businesses today. Competitive pressures are forcing organizations to become more agile – often on a global level – and to leverage IT solutions more intensely. Building and deploying applications that allow for the integration of business processes – within the electronic walls of a company as well as outside those walls – is only part of the picture. The flexibility to change those applications and adapt business processes to respond to the changing market and competitive pressures that a company faces is also a key success factor. Customers, partners, and other communities that make up the value chains of an organization will demand solutions that can readily adapt to their needs.

Among the most difficult issues to deal with in exercising application agility is the requirement to take processes offline to make changes, test new application functionality, and redeploy. The challenge is to be able to readily adapt applications without disrupting the activities of the business and those with whom it must interact.

WebSphere Application Server Enterprise V5 now takes advantage of a business-rules-based approach that permits the creation of highly adaptive applications, allowing them to be easily adjusted to changing business conditions. Business rules are extremely practical in this type of situation because they can be used to separate business policies from application code. Any changes to business rules ideally would not require a change in the application itself, and can be performed without the involvement of an application developer. An example is useful here. A company may have different "tiers" into which it divides its customers, with each tier having parameters associated with it such as discounts, delivery terms, and so on. The criteria that determine the tier placement of a customer, and the parameters associated with each tier, can be defined using a set of business rules. The execution of application logic to fulfill these rules is independent of the rules themselves. If the company decides it wants to change the discount associated with a given tier, for example, it can task a business analyst to make that change using a simple, straightforward user interface. These rules then access the required Business Rule Beans (created previously by a Java developer) to make the required changes. The changes to business rules can also be scheduled to take effect at a future time – presumably during a period when no transactions or user interactions occur, but in a way that will not disrupt the execution of business logic.

One of the additional limitations of the J2EE 1.3 specification is the need to define – using the query language that is part of the standard – all queries statically during the development of an application. Queries defined at runtime (referred to often as "ad hoc") are not permitted. In addition, from the standpoint of data, the specification only allows for the extraction of keys and objects, and data sets

cannot be sorted or ordered. The "select" clause used as part of the EJB query language can only return one data value.

WebSphere Application Server Enterprise V5 provides the ability to dynamically build queries at runtime. These queries can include the ability to select, sort, and join data, and perform operations on this data at runtime. It makes it possible to mine data without breaking encapsulation, and to execute methods as part of a query. Mining data can be done without going through a potentially long series of sequential steps.

LEVERAGING INNOVATIONS TO INCREASE DEVELOPER PRODUCTIVITY

IBM has added additional capabilities to WebSphere Application Server Enterprise V5 focused on the product's ability to support the development and deployment of enterprise applications. These capabilities are numerous. All are important, but a few represent significant enhancements to both the programming model and execution capabilities of J2EE-based applications.

IMPLEMENTING AN ASYNCHRONOUS APPROACH TO RUNNING J2EE APPLICATIONS

The first of these is based on the concept of *startup beans*. In essence, a startup bean is a session bean that is loaded when a J2EE application is started up, and enables the application to execute business logic automatically. What this gives a Java developer is the means to initiate execution in response to an application lifecycle event, i.e., the normal start or stop of an application. This automation stands in contrast to the manual use of servlets to initiate the startup of a session bean. The developer takes session beans and tags them as startup beans, using a standard set of remote interfaces specified by IBM. Because a startup bean is essentially an EJB, there is no special tooling required to build and deploy it.

The second important feature is intended to address the issue of performance of operation execution. The J2EE execution model, as implemented to date in application servers, does not include the ability to use threads. In a typical J2EE application, servers can only execute code when an HTTP or IIOP (Internet Inter-ORB Protocol) request is received by the server from a user, or when a message is received by a message-driven bean. When that request completes, there is no longer any code running on the server. Additional execution threads are simply not part of the model.

Past workarounds involved the use of multiple JVMs that created a high level of complexity and a serious management and coordination problem. The other option was to package the required code in the client, making clients "fatter" and increasing the caliber of client machine required. Another complication for programmers is that if they build their own threads apart from a J2EE thread, those threads will not contain any of the contexts associated with the original thread. Important elements of this context – such as authentication – will not be available to the threads, resulting in a disruptive break in the programming model. Needless to say, vendors have generally not supported problems encountered during these workarounds.

To address this, IBM has created a new technology – another extension of J2EE – called *asynchronous beans* that will allow threads to run in a managed, J2EE-based application server environment. An asynchronous bean is able take a snapshot of the J2EE environment at the same point in time that the asynchronous bean was

created, and to propagate the context associated with that snapshot – including a security credential – to a different thread. This enables the simultaneous, parallel processing of multiple threads while ensuring that the context under which the original thread operated is maintained.

The programming model for asynchronous beans is essentially the same as that for EJBs. More specifically, the asynchronous bean can inherit most of the J2EE contexts, and can run using the security associated with its creator. From the standpoint of transactions, the asynchronous bean automatically gets its own local transaction, which is important because it is less "expensive" and requires less overhead than global transactions (although the latter is an option if distributed transactions are required).

Asynchronous beans offer the promise of allowing programmers to significantly improve performance by permitting multiple threads to execute simultaneously. *WorkManagers* are also provided, which act as thread pools. They are created by an administrator, and are tied to web applications and/or EJBs through the use of resource-references and/or EJB descriptors, respectively. The deployer of an application can then map a WorkManager to the resource-reference or descriptor when the application is deployed. Asynchronous beans can also be used in contexts where programmers need a way for J2EE components to signal each other about asynchronous events within an application or when programmers need high-speed transient alarms to enable activities such as logging to occur at sub-second intervals.

The *scheduler service* is an additional feature of interest new to V5. This service makes it possible to initiate work at a given point in time, or based on the passage of a specific interval of time. Pluggable calendars can be used for handling rules related to scheduling tasks. This service is complementary to the workflow support described earlier, since it allows business processes to be initiated at pre-defined times. This service also brings the traditional notion of "batch processing" into the world of J2EE applications. IBM has also added *object pools*, a feature that allows instances of objects to be pooled and reused instead of being destroyed. This can increase performance by keeping objects that may be reused immediately available at all times.

FURTHER ENHANCEMENTS TO SUPPORT DEVELOPERS

For those instances where information must be shared across elements of a distributed application, IBM has incorporated the *WorkArea* service. Accessed via J2EE-style APIs, the service permits developers to attach a "context" to an execution thread and have that context propagate through the distributed system so it is available through business logic for execution later. Without this service, the developer would have to create pervasive parameters, which can result in overly complex interfaces. Alternatively, an infrastructure could be created that would read the information and write it to a database for later access. This alternative would incur an overhead cost. The WorkArea service can therefore

make it easier for developers to implement information sharing in a distributed application, and can improve solution performance.

An interesting enhancement to the messaging capabilities of the product is *Extended Messaging Support*. Here, messaging becomes "container-managed" (an analog to the concept of container-managed persistence as it applies to data), allowing a separation of business logic and the logic used to implement messaging. Tools provided enable the developer to create *sender beans* and *receiver beans* that represent the mechanism by which JMS messages are processed.

An additional aid to increasing application performance is what IBM calls *application profiling*. This capability consists of two elements – access intents (policies that permit customization of services provided by an EJB container and persistence manager), and application profiles (which make customization of access intents for specific applications possible). The ultimate result is the ability of administrators to tune EJBs in ways relevant to specific applications. The unique feature is the notion of tuning at the application level, a feature not found in most J2EE application servers on the market today.

Ultimately, many organizations attempting to exploit the revenue, cost, and customer satisfaction benefits possible through web services implementations will be compelled to go "global." Doing so means tackling the difficult problem of internationalization ("i18n"). i18n capabilities now included with WebSphere Application Server Enterprise V5 allow the delivery of localized time and date information, currencies, languages, and decimal formats to any locale of interest to an enterprise. This is accomplished through the invocation by a local client. In this procedure an object on a server at the "host" locale can return proper information to the local client.

MANAGING RESOURCES IN AN SOA ENVIRONMENT

The web services concept brings with it both the promise of unprecedented flexibility and benefits based on business opportunity and cost containment. At the same time, it introduces some interesting possibilities and challenges in terms of the management of IT resources. The modular, componentized nature of the paradigm may permit the real-time deployment of a service on any system available, running one of many operating systems. The potential of optimizing IT assets, including systems, storage, networks, and other elements in real time and within a web services context is very real. In the limit, all IT resources become components that can be utilized as needed, with the entire mix optimized in terms of cost, performance, and responsiveness to user needs. The business benefits are both substantial and obvious. At the same time, managing such an environment requires capabilities not previously available.

To begin to address the need for the type of resource management required, IBM has announced *IBM Server Allocation for WebSphere Application Server*. This product will allow users to balance workloads and allocate resources as needed to support

the execution of WebSphere applications. For instance, organizations will be able to workload manage multiple applications across multiple server clusters. What this means is that during times when web applications are not experiencing peak volumes, computationally intense applications can be deployed in parallel and be managed to make use of available excess capacity. This management capability can be applied across an entire grid of WebSphere application servers as if they represented a single resource – as opposed to just a single cluster. IBM plans on making this new product available to a limited number of early adopter customers with a services engagement beginning in June, 2003.

IBM Server Allocation for WebSphere Application Server is, in the opinion of DHBA, a strong and innovative step in fulfilling the goal of enabling comprehensive management of IT resources in a services-based environment.

POSITIONING IBM'S WEBSPHERE AS ENTERPRISE STRATEGY

The recent introduction of WebSphere Application Server Enterprise V5 is a significant milestone in IBM's efforts to maintain its position as a leader in the area of web services and to reinforce the importance of its concepts of SOA and on-demand applications availability. IBM already commands a major share of the existing application server market and in recent years has added virtually all of the elements of the software stack required to build and implement a service-based infrastructure. V5 includes a variety of new capabilities that not only promise to give IBM the ability to maintain its strong position as a web services software vendor, but will strengthen IBM's position with the very developers who will build and deploy those services. In particular, IBM has recognized the need of developers to implement some key capabilities in their applications (such as real multithreading and higher performance), and has given Java developers many new features that extend the J2EE specification in a variety of important ways. Its intent to submit these enhancements to the Java Community Process should further enhance IBM's position among Java platform vendors, and as a standards leader.

IBM's strategy is not without its challenges. Despite its large installed base of WebSphere application server implementations, there is a perception that solutions built using WebSphere often require high levels of professional services involvement – often at a high price. This is not unique to IBM – many customers find that going from product purchase to actual implementation can be fraught with surprises along the way. These surprises often take the form of more manual programming than anticipated, or difficulty in getting various components of the stack to work in an integrated fashion. IBM tends to work with larger enterprises whose environments are highly complex. Coupled with the company's visibility and presence, this experience makes its perceived "skeletons" more visible and likely to cause concern. But with its V5 release, IBM addresses this perception through an engineering effort to better integrate its software stack among the products of which it is composed. Many competitors are striving to create an improved "out of the box" experience, because their customers are experiencing the same integration issue, and because these vendors have previously witnessed IBM's vulnerability.

Ultimately, product capabilities must translate into achievable business benefits to their users. Greater developer productivity can enhance competitive position through quicker time-to-market for new applications and by means of rapid changes to existing applications. Performance enhancement features can significantly reduce costs in terms of resources required. Robust transaction support and internationalization keep business processes moving in a reliable manner, allowing an organization to be more responsive to customers, business partners, and others. In these areas, WebSphere Application Server Enterprise V5 stands out. This achievment should put IBM in a strong position on the list of vendors to be considered by those seeking to enable web services through the use of leading tools and infrastructure.

CUSTOMER CASE STUDIES

USING WEBSPHERE APPLICATION SERVER V5 FOR WORKFLOW SOLUTIONS: STRATEGIC THOUGHT

System integrators play a key role in helping organizations implement solutions that require a high level of integration. This is hardly a new phenomenon, but it is evolving to include the construction and deployment of solutions based on service-based architectures. System integrators can also leverage the component-based nature of such an architecture to more effectively leverage reuse of the solutions they create.

Strategic Thought is a system integrator that knows the integration challenges faced by a long list of large companies in Europe. Based in the United Kingdom, Strategic Thought focuses its efforts in two major areas: Risk Management, where it offers a customizable product called *Active Risk Manager,* and Solution and Integration Services to provide business integration capabilities to its clients. The company employs 36 people, with revenues of approximately 5.6 million pounds, and has nurtured expertise in three major industries – retail, telecommunications, and financial services. Based on its view of IBM as a technology leader, it uses IBM as its integration partner. The company's integration business has evolved from traditional middleware to embrace IBM's MQSeries. More recently, it has been able to leverage the WebSphere Application Server, as well as other elements of the WebSphere product stack (including WebSphere MQ Workflow and the WebSphere Portal).

Despite the company's relatively small size, it delivers highly scaled solutions. For instance, based on a previous version of WebSphere Application Server Enterprise, one application could see up to one million simultaneous users during peak periods. These users would be executing real transactions such as purchasing tickets to major concert events. Another application – developed for a major bank – is required to handle up to ten million transactions per hour.

The decision to be part of the beta program for WebSphere Application Server Enterprise V5 was based on the company's desire to get an early start on new IBM product capabilities (seen as a competitive advantage), and to build on their extensive experience with IBM by getting an understanding of the value of the V5 workflow engine. The company's specific goal was to complete a proof-of-concept for a part of an existing workflow solution; the desired result was a sizing of the migration effort for that application which served as an example of how feasible such a migration would be for other solutions it has built.

Citing an insurance industry application, Strategic Thought explained that its customers in that industry would use a tool to generate their own workflow. Strategic Thought would then execute those workflows – which would generally involve passing documents from one person to another – using the workflow engine. This task proved difficult using previous versions of WebSphere

Application Server Enterprise. Strategic Thought noted that V5 significantly eased the task. The company indicated that the development tools – based on Eclipse 2.0 – were more user friendly and gave it impressive improvements in productivity. The company also cited the ability to take a Java class and, with a few mouse clicks, build beans and easily integrate them into business processes. Strategic Thought was also especially impressed with the tighter coupling between the workflow engine and the application server.

Strategic Thought was one of those organizations whose developers did the work of implementing by hand what asynchronous beans make possible in V5. The company not only noted a noticeable increase in developer productivity, but saw the opportunity to reap the performance benefits this new feature can offer. The enhancements of V5 span greater control over transaction scope and external processes – key benefits.

In Strategic Thought's view, the success of its beta exercise will make it possible to migrate complete solutions to a technology base, a procedure that should provide a greater possibility for reuse and faster time-to-market. From a business model perspective, reality dictates that the cost of migrating solutions will have to be measured against these benefits. To mitigate this, Strategic Thought believes that customer adoption of V5, driven by IBM's marketing efforts, will create client projects that will result in opportunities to cover the fixed costs of these migration efforts.

USING WEBSPHERE APPLICATION SERVER V5 AND SOA IN FINANCIAL SERVICES: DANSKE BANK

Denmark-based Danske Bank is the second largest institution of its kind in the Nordics, and controls a variety of distinct but related businesses under its corporate umbrella, including traditional banking services for individuals, corporations and other institutions, life insurance and pension products, asset management, mortgage finance, brokerage, real estate, and leasing services. With all these businesses servicing about three million retail customers alone (almost a third of whom use online services), Danske Bank has had to face some significant integration challenges in its quest to provide its customers with a full, robust set of financial services.

The bank's approach has been to see these challenges as opportunities – to carry forward a strategy it has had for some time. Danske Bank has adopted a "one group – one system" strategy, a departure from the approach taken by many companies (including some of the bank's competitors). While many institutions approach the IT integration issues resulting from a merger/acquisition event by tying together all existing systems at a high level, Danske Bank has energetically applied a services-based approach that first creates actual services out of existing capabilities (literally hundreds of them), and applies the functionality provided by those services uniformly across all products and channels. Instead of a large number of systems integrating disparate applications, all services are run across

one system. The bank's vision is to be able to expose functionality as Web services, and reuse that functionality everywhere in order to provide its customers with real-time access to all data and information as needed. One could envision a scenario in which a customer could prepare a budget, pay bills, apply for a mortgage, transfer money, and perform a variety of other tasks online via a portal that brings together all the functionality required as needed.

The alignment of this strategy with IBM's concept of SOA is obvious, and Danske Bank has been working with IBM toward this goal. The bank uses mainframes and software technologies and products such as DB2, and has been a WebSphere user for some time. What was missing until recently was the ability to create workflows directly from services – a capability that IBM is delivering in its *Process Choreographer* component. Danske Bank wanted to convert its current workflow solutions into a common, integrated workflow management system. It was also interested in the prospect of having some of its manual interactions accessed as services as part of that system. From the bank's perspective, whether the service of interest is a COBOL program, a .NET service, or a manual step performed by a single employee, it should be easily incorporated into the workflow solution. The key differentiator here is that every step in the workflow is represented as a service, permitting an unprecedented level of flexibility of integration and reuse of existing functionality.

IT professionals at the bank spent a portion of last year surveying the market to find a solution that would allow them to build the bank's "one group – one system" environment. Their conclusion was that the only solution they saw that provided the architecture and functionality they needed from the perspective of workflow were the enhancements incorporated by IBM into WebSphere Application Server Enterprise V5. At that point, Danske Bank became an enthusiastic participant in that product's beta program. To the bank, the workflow engine provided in the product represents a "new paradigm" for designing and managing workflows in a services-based environment.

The bank's use of the product thus far has been focused on building a pilot application that incorporates a number of relatively simple workflows focused on the work some of their employees perform to handle exceptions to the bank's many control procedures. Back-office users must handle such exceptions on a daily basis (the pilot application will be used by about 25 of these users), so workflows are being built and deployed to support their activities. Deployment of the pilot is scheduled for June 2003.

Danske Bank has targeted the fall of 2003 as the time when it will deploy a full production system for its mortgage business. The system will be used by up to 2000 users, 300 of which will be back office users who will use the system full time and concurrently. The remainder includes salespeople who will use the workflow and sales tools out in the field with mortgage customers on an as-needed basis. All workflows in this application will be implemented in Process Choreographer.

Danske Bank has also discovered another benefit provided by V5. The bank's experience with older versions, including V4, indicated some issues with going from application development to deployment. Danske Bank's IT people had found this process less than ideal in the past – both in terms of the manual effort and the time involved. V5 offers them a variety of APIs that allow them to make this transition a much more automated one, which for Danske Bank translates into cost savings and a greater level of responsiveness to its customers' needs.