# WebSphere Application Server V5

*Delivering New Business Value to the Enterprise on a Java™2 Platform, Enterprise Edition (J2EE) and Web Services Base*

A White Paper by IBM Software Group

April 2002

# Executive Overview

It's all about business value. It's all about *your* business.

With *WebSphere Application Server V5*, IBM is making significant investments in a next generation application server environment that delivers compelling innovations built on an industry-standard foundation. This premier application server environment is further enhanced by associated value-added tooling based on the Eclipse open source project[1]. Together, this combination of runtime and tools lets you address three common, but critical, business imperatives:

1. **Reducing overall costs**
2. **Improving customer loyalty**
3. **Adapting quickly to new business opportunities**

Of course, achieving these objectives contributes towards complementary business goals such as increased profitability and market share, improved competitive position, and faster time to market.

The purpose of this white paper is to briefly introduce *WebSphere Application Server V5* from a high level perspective[2]. You are encouraged to read this paper if you are looking for an application server solution that provides positive answers to the following kinds of questions:

- *How can I introduce competitive changes into my business model in "real time"?*
- *How can I improve customer service at reduced cost?*
- *How can I minimize the time required to develop and maintain applications?*
- *How can my business grow more quickly and efficiently?*
- *How can I leverage existing investments when developing new applications?*

The paper begins with a brief overview of the traditional role played by application servers in an e-business solution. This is followed by a discussion about how application servers might be optimized and extended to deliver new business value – as in the case of *WebSphere Application Server V5*. WebSphere optimizations offer new levels of *performance*, *availability*, and *scale*. WebSphere extensions promote unparalleled *productivity*, *flexibility*, and *integration*. These concepts come to life through several business scenarios that are described in the context of a realistic case study. Key WebSphere differentiators are summarized at the conclusion of the paper.

Please read on to learn more about how WebSphere software can help you achieve critical business goals and gain competitive advantage in record time. We are eager to share the story!

---

[1] See www.eclipse.org for details on this industry-leading open source tooling initiative.

[2] Follow-on work is planned for second quarter, 2002 that will provide a more detailed and technical discussion about *WebSphere Application Server V5*.

# A Next Generation Application Server

As previously mentioned, IBM is investing in next generation application server technology. Before we consider what this means in some detail, we should first consider the traditional role that an application server plays in a typical enterprise topology.

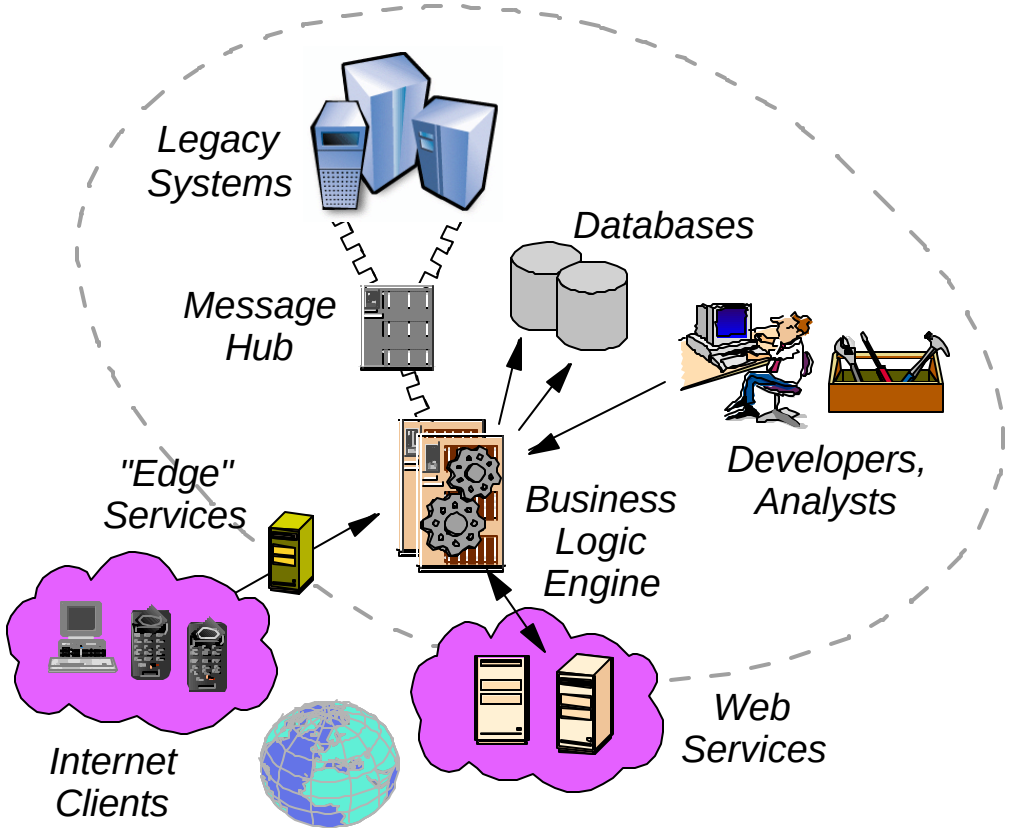


**Figure 1. Application Servers as "Business Logic Engines"**

The previous figure depicts application servers as *business logic engines.* Typically, application servers provide a place to execute policies; to enforce terms and conditions; and to apply business rules. Global clients commonly access these engines using browsers and pervasive devices that operate on both static and dynamic content. This content can be cached at the edge of the network to drastically reduce both network traffic and system load[3]. Increasingly, these engines are also accessed by partners in inter- and intra-enterprise Web services interactions. Of course, the application servers don't stand alone, as they normally need to access persistent information stored in one or more databases. In addition, existing systems and applications often need to be leveraged as part of an e-business solution, and messaging technology and message brokering are

[3] Based on our analysis of major retail customers, we've found that for some types of applications only three percent of the traffic really needs to touch the backend server environment. The rest of the traffic can be served from the edge of the network using cached information. Edge serving was also critical to meeting IBM's performance and scaling challenges at major sporting events such as the Sydney Olympic games, and IBM continues to make significant investments in "Edge of Network" technology.

frequently used to drive such legacy system interactions. We complete the picture by showing that developers need an easy way to create and update the applications that are hosted by the application server. And increasingly, business analysts need a way to model; monitor; and maintain the various business processes and policies that run there.

When we think about ways to improve application servers (that is, business logic engines), we should consider some general ways in which *any* kind of engine could be improved. There are basically two approaches that one could envision here. In one case, we might choose to *optimize* an engine with a "super oil" or "special oil" that is designed to improve efficiency and performance. We could also *extend* an engine by enhancing its capabilities, thereby giving the engine fundamentally new utility. These two concepts are depicted by the following figure.

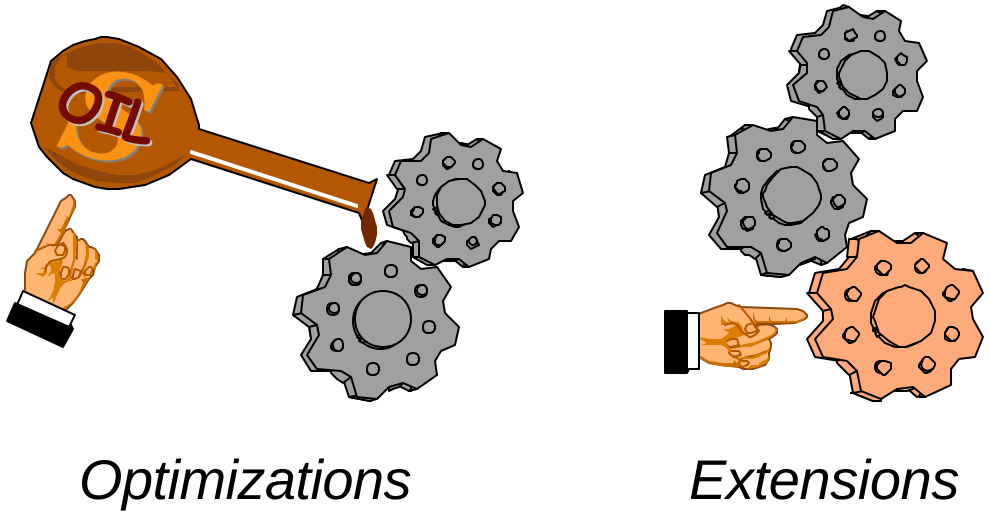


**Figure 2. Improving Engines with Optimizations or Extensions**

From an application server perspective, you should think of optimizations as a *deployment* concern. Optimizations are *administration-based* and they *do NOT involve source code changes.* An example of this would be the transformation of a standards-based J2EE application into a "turbocharged" standards-based J2EE application in which *NO* programming changes are required.

Extensions, on the other hand, are a development concern. Extensions *DO affect the programming model*, and they are frequently associated with a set of *supporting development tools.* An example of this would be the transformation of a standards-based J2EE application into an application that offers entirely new functionality (such as one that supports process automation services and message-to-component mapping services). This kind of extended functionality would generally be considered for incorporation into the evolving J2EE specification over time.

The optimizations and extensions delivered by *WebSphere Application Server V5* define a next generation application server environment. This environment offers outstanding

benefits in terms of meeting the critical business imperatives that were identified at the outset of this paper:

1. **Reducing overall costs**
2. **Improving customer loyalty**
3. **Adapting quickly to new business opportunities**

The next two sections provide a closer look at how these imperatives are addressed, beginning with a brief discussion on *Optimizations*, and then later on, *Extensions*.

# Optimizations Addressing Critical Business Imperatives

Our three critical business imperatives actually represent "value opportunities" for an application server. For example, consider the following questions from an application server perspective.

1. Since cost reduction is an imperative in today's economy…
   - How could we reduce I/T costs (people, HW, SW)?
   - How could we manage more with less?
   - How could we process work more efficiently?

2. Since building customer loyalty is a matter of survival...
   - How could we provide better customer responses?
   - How could we improve customer response time?
   - How could we improve system availability?

3. Since business models must evolve and adapt quickly...
   - How could we rapidly link to suppliers and partners?
   - How could we simplify access to their systems?
   - How could we provide secure, reliable, manageable, efficient and scalable access to our own systems?

## Performance

The first optimization that we should consider when addressing the previous questions is performance. What if you could take the same number of machines, and process more work through those machines than ever before? Taking that thought even further, what if you could take *fewer* machines, and yet process significantly *more* work? This would have a direct bearing on cutting costs. Faster and more consistent response times would improve customer service and customer satisfaction as well. So you can see a direct mapping between performance optimizations and business benefits.

Just imagine the efficiencies that would be gained if application servers could be taught to allocate proportionately more of the shared work to the most capable machines? What if application servers could be made smart enough to detect variable runtime conditions,

and then redirect work dynamically to the machines that are the least busy? What if application servers could access databases faster than ever before, and could support concurrent database usage more flexibly than ever before? What if application servers could leverage runtime information to intelligently and automatically tune themselves? All of these performance optimizations would be extremely valuable to a business.

## Availability

Now let's consider availability. Highly available systems need to have at least two of everything. This allows work to continue in the event of component failure. The more failure bypass that a system offers, including the failure of internal components, the less that an end user will be disrupted when something goes wrong. By reducing unplanned outages, you clearly reduce costs. You also make your business more accessible to other businesses as part of the evolved business models that link your company with others.

Beyond *running* applications, you should also note that fault tolerance is important in *managing* application servers. This includes the *deployment* of applications, making it possible to update the application environment whether the application servers happen to be on-line or off-line. And speaking of application servers that are off-line, another aspect of intelligently managing application servers is allowing for work in process to be completed properly before an application server is taken down.

Of course, you know that failures are going to occur at some point, for at least some part of the system. So what is needed is a way to capture the root cause of the failure and then quickly diagnose the necessary corrections. If remote support is needed to fix a problem, then everything that the remote support team needs for diagnosis should be *automatically* captured for further investigation. Still another aspect of effective problem determination is configuration verification to provide *early detection* of topology problems. This provides a highly effective problem avoidance strategy.

## Scale

The final optimization to consider is scale. Scaling means that you can easily handle more work, and this becomes more important as your business grows. Effective scaling ensures that all customers receive the service that they deserve. Effective scaling provides growth without breaking the bottom line. And effective scaling lets your business expand beyond its traditional boundaries to embrace new partners and suppliers.

Scaling means that you can deploy new applications into large environments easily and automatically. This includes applications that extend out to the edge of the network. Scaling also means that you can effectively manage those environments, and – through pluggability – you can leverage the investments that you've made in existing management and security software as well as operational skills.

A final point about scaling takes us beyond the immediate boundaries of the enterprise, and this ties in very much with the Web services story. Making Web services more

accessible and efficient means that your business can more easily scale its business model. This includes the publication of tested and approved business services through a private UDDI registry. It also includes monitoring and management of Web services invocations, and the ability to easily map Web service definitions to existing service implementations, wherever they happen to reside.

The following figure summarizes the three kinds of optimizations that we considered in support of highly dynamic, distributed deployments that meet enterprise-class needs. For each of these three categories – performance, availability, and scale – there are in turn three "details" that capture the essence of what we've discussed.

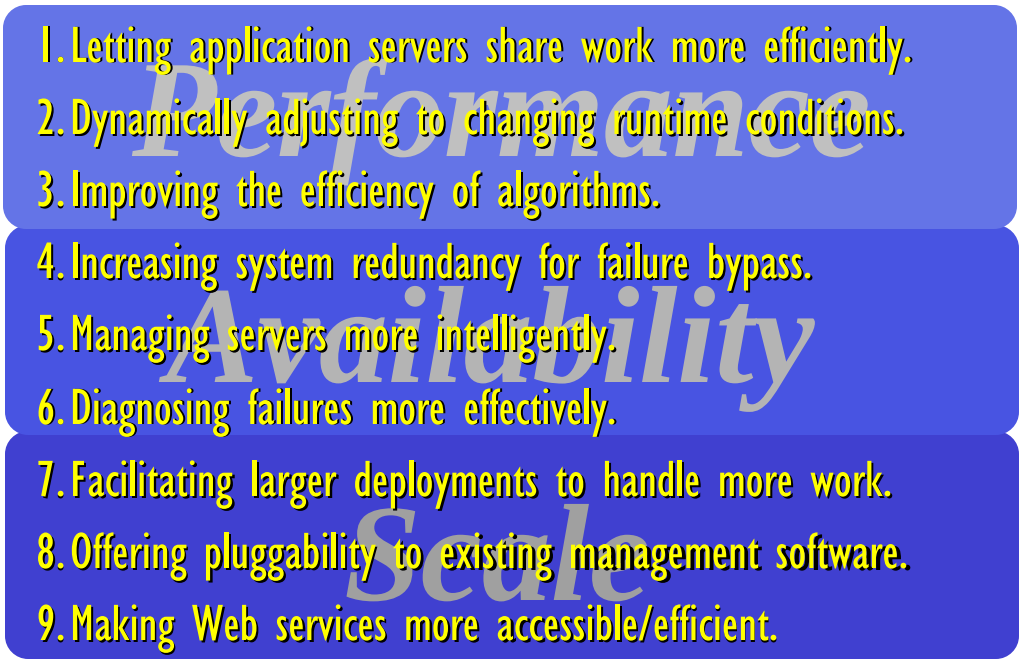


**Figure 3. Valuable Optimizations – Performance, Availability, and Scale**

The interesting thing to note about this figure is that *each* optimization maps to *each* of the three business imperatives. As one example, you can easily map the second point (dynamically adjusting to changing runtime conditions) to the second imperative (building customer loyalty). It should be clear that you could improve satisfaction and loyalty by giving customers more consistent responses, and faster responses, as they use your business applications. As another example, you can easily map the third point (improving the efficiency of algorithms) to the first imperative (cutting costs). Clearly, if you could make existing systems work faster and more efficiently, then you could also save money on upgrading those systems over time. As mentioned earlier, all nine of the optimization details can be easily and directly related to the three business concerns.

Having briefly looked at *optimizations*, let's now consider some valuable *extensions* that would help us cut costs; build loyalty; and adapt quickly to change.

# Extensions Addressing Critical Business Imperatives

As discussed in the previous section, the three critical business imperatives present opportunities for an application server to deliver value. Let's consider a new set of questions that relate to each of the imperatives.

1. Since cost reduction is an imperative in today's economy…
    - How could we improve developer productivity?
    - How could we leverage new/existing investments?
    - How could we facilitate componentization and reuse?

2. Since building customer loyalty is a matter of survival...
    - How could we provide top service to top customers?
    - How could we satisfy requests more quickly?
    - How could we ensure the highest data integrity?

3. Since business models must evolve and adapt quickly...
    - How could we change business policies in real time?
    - How could we rapidly support diverse global markets?
    - How could we improve development and operational efficiencies?

## Productivity

The first extension that we should consider when addressing the previous questions is productivity. Market analysis suggests that the time required to roll out new applications is a key concern across industries, and improved developer productivity is clearly a way to address this.

One of the ways that you could vastly improve productivity would be to reduce the need for handcrafted programming. This could be accomplished through powerful frameworks that absorb much of the work involved in development. This could also be accomplished through tools that generate code used by the runtime. Imagine a combination of frameworks and tools that work together to provide best practice implementations, a realization of the industry's best architectures. In this kind of world, developing and maintaining applications could be largely facilitated through something called *visual programming*. For example, you could create basic service definitions through "adapter tooling" that visually connects your Java applications to Enterprise Information Systems. Through *service choreography*, you could then combine these basic services into "composed services" that perform higher-level business activities. Wiring these interactions together in a visual fashion makes it easier for developers to create applications, and to preserve the flow structure of the application when underlying service implementations change over time. Still other productivity gains would come from the close integration of components and messaging systems. This includes the automated transformation and mappings required between message flows and components in order to satisfy diverse application needs.

As mentioned previously, the productivity benefits just described are realized in large part through world-class tooling. This is shown by the following figure in which visually constructed flows clearly define the sequence of logic within a business process step.

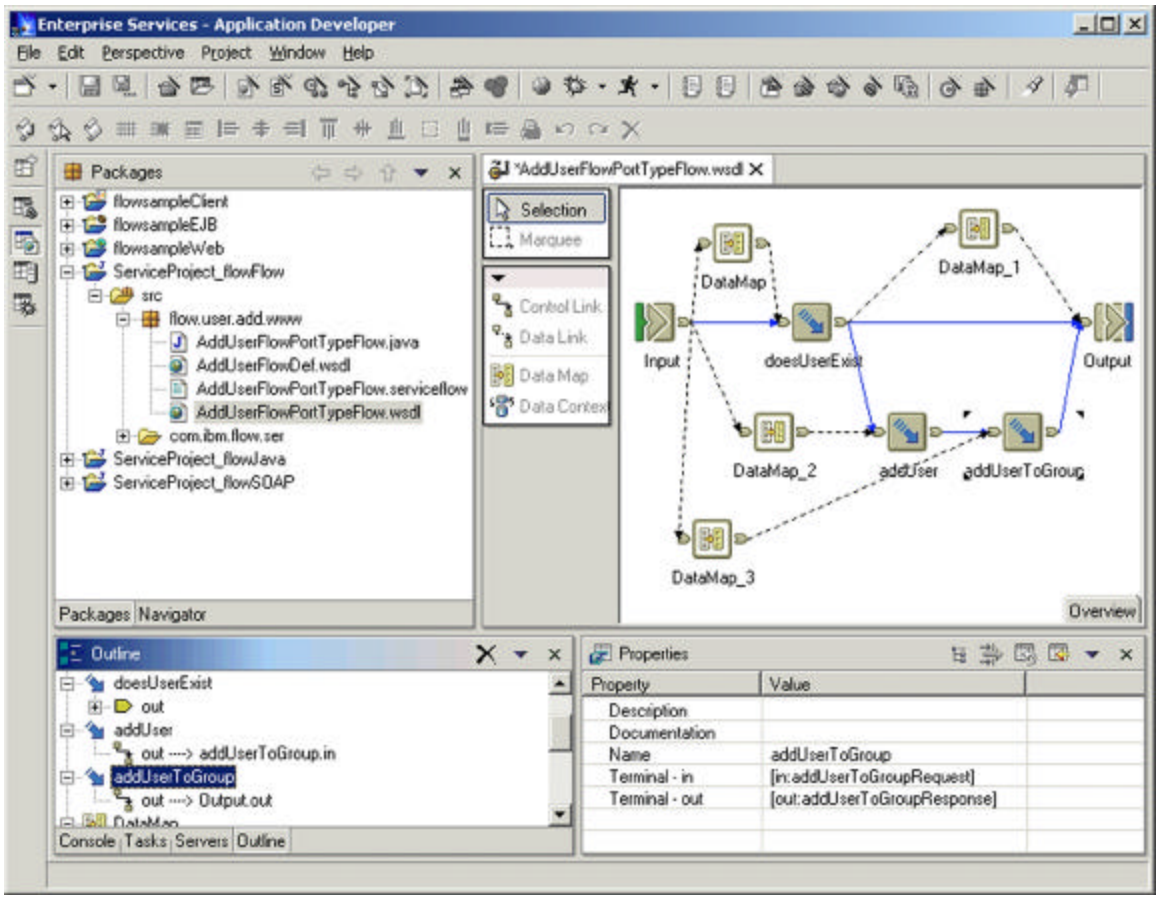


**Figure 4. Visual Programming and Service Choreography**

## Flexibility

Beyond productivity, another extension to consider is flexibility. There are many aspects to flexibility. One aspect of flexibility is the ability to externally administer business policy. In industries like insurance, there are many policies and many variable business practices that need to be maintained (like defining the risk category for each policy holder). Many industries also need to respond quickly to regulatory and legislative changes, and doing this through programmer intervention can be prohibitively costly. What's desired is a framework for defining business policies that can be invoked from business logic. Imagine a world in which these same policies could easily be maintained and scheduled by business analysts. That level of flexibility would promote *truly* dynamic e-business.

And there's more to flexibility. Cross-industry analysis suggests that many businesses find it difficult to expand into international markets. Imagine the business value that would result from an application server runtime with the flexibility to adapt to global constituencies. Customers around the world expect to be served in their own language; with appropriate formatting rules for currency, decimal points, and commas; accounting for differences that exist in various time zones; and so on. Applications that can dynamically adapt to such differences will enjoy much more flexibility in serving new geographical markets.

Yet another aspect of flexibility is the capability to provide on-demand access to information that is needed by the business model. The ability to express queries for information in a way that efficiently maps to the business problem being solved is extremely valuable. This represents another example of delivering on the promise of dynamic e-business.

## Integration

The final extension to consider is integration[4]. A key requirement that customers are placing on application server vendors is support for process automation facilities. The ability to host and schedule a living business model in the application server environment brings new opportunities for seamless process integration. This is accomplished by means of service choreography and it includes scripted interactions with Enterprise Information Systems and other services as well as longer running stateful workflows that tie together activities into more course-grained business process steps. Also related to this is support for business entities that adapt themselves to diverse business processes. Customers are asking vendors to provide a cost-effective application-serving environment in which these process integration capabilities can be easily leveraged, with a single administrative and operational view of the runtime environment. This includes the ability to not only run business processes, but to monitor them as well.

There are other aspects of integration beyond what we previously mentioned. Customers are looking for ways to manage sophisticated transactions. They want a way to update *all* critical resources inside of a transaction with high integrity. They want support for *long-running* transactions. They are looking for *automated* compensation in the event of a system failure. These are examples of deep transactional integration that provide both business and technical value.

Yet another example of deep integration is support for advanced component models. This includes the ability to leverage useful composition patterns, making it possible to form brand new components by aggregating or discriminating between existing ones. Another example of an advanced component model is one in which procedural systems like CICS and SAP can be integrated into the world of components as easily as relational databases. This facilitates higher degrees of reuse, which provides customer value.

---

[4] Integration is a very broad subject and IBM is well positioned to address the many styles of integration required to meet customer needs. Please refer to Appendix A of this paper for more discussion around this important topic.

Following the pattern used in the previous section of this paper, the following figure summarizes the three kinds of extensions that were just discussed. For each of the three categories – productivity, flexibility, and integration – there are three "details" that capture the essence of what's been presented.


**Figure 5. Valuable Extensions – Productivity, Flexibility, and Integration**

As in the previous section of the paper, it's interesting to note that *each* extension maps to *each* of the three business imperatives. For example, you can easily map the first point (reducing the need for hand-crafted programming) to the first imperative (cutting costs). The association is clear in terms of improved developer productivity and reduced errors. Similarly, it's easy to map the fourth point (externalizing business policy management) to the third imperative (increased business agility). By giving business analysts direct control over the setting and scheduling of business policy, it becomes possible to respond dynamically (virtually at the speed of thought) to change.

Perhaps you are wondering how these extensions – as well as the optimizations that were previously discussed – relate to *WebSphere Application Server V5*. Basically, the next generation of WebSphere Application Server significantly *extends* and *optimizes* a "core" J2EE and Web services-based application server, as illustrated by the following figure.

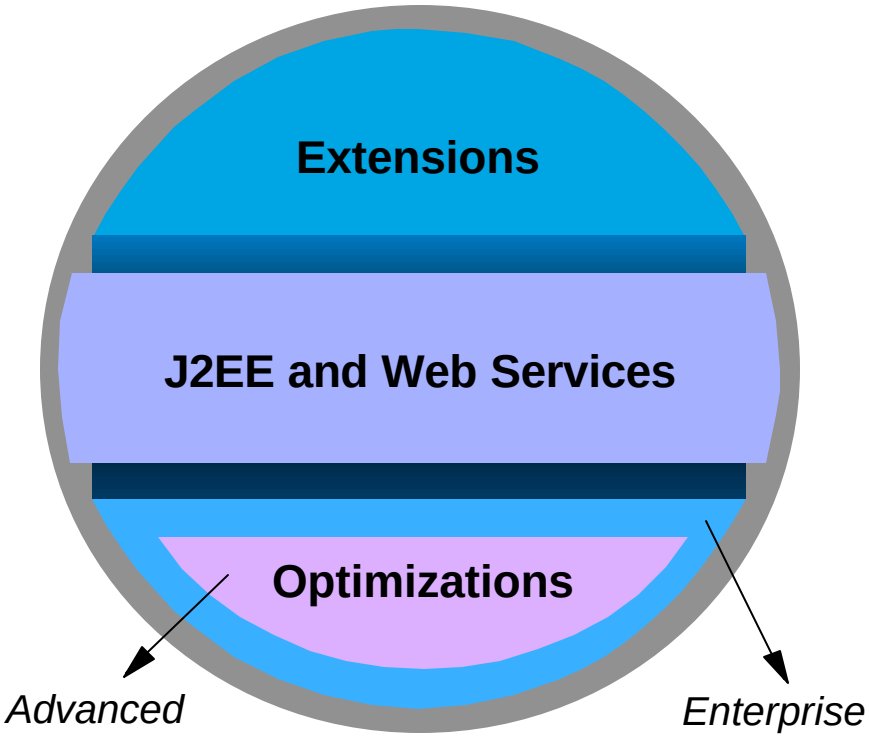


**Figure 6. An "Inside Look" at WebSphere Application Server V5**

This figure lets us conceptually "peek underneath the hood" of the application server. At the core we find the industry's best support for J2EE and Web services. By adding *advanced optimizations* to the core we find support for things like failure bypass and load balancing. As a proper superset of this, the *enterprise optimizations* raise the bar even higher by delivering things like cross-domain failure bypass and dynamic load balancing in support of enterprise-class deployments. And finally, the *extensions* complete the picture by providing a greatly enhanced programming model. Here we find support for things such as process automation services and dynamic business policy management[5].

The next section of this white paper demonstrates the full power of WebSphere Application Server and the associated WebSphere Studio tools using a realistic case study. The case study puts our previous discussion into a practical business context.

## A Case Study Scenario

Let's consider some of the challenges faced by a fictitious yet representative enterprise called *Web Banking Online* (or WBOnline, for short). This is a global finance company with a commitment to serve international constituencies. WBOnline has grown very quickly through mergers and acquisitions. The company now needs to *carefully control its costs* in an increasingly competitive market. In addition, *improving customer loyalty*

[5] Extensions typically introduce business value "ahead" of the associated industry-standard specifications. IBM actively submits Java Specification Requests (or JSRs) in order to drive valuable extensions into the formally adopted Java specifications.

and *rapidly adapting to new opportunities* are viewed as critical to continued business success.

WBOnline is looking for a high performance, standards-based e-business infrastructure that can provide the flexibility to support everything from centralized operations to departmental projects to remote branch offices. The environment needs to be highly secure, reliable, and manageable. It needs to handle unpredictable spikes in demand with continuous availability and consistent user response times.

WBOnline has already made a strategic decision to build new enterprise applications in Java, and these new applications need to integrate with a significant portfolio of existing assets. As one example of this, WBOnline continues to drive significant amounts of work through CICS applications that have been developed over the course of several decades. Rather than being "ripped and replaced", these CICS programs need to be operationally reused in the context of new component-based applications that are written in Java. Moreover, in supporting a broad set of business objectives, the Java applications must accommodate a wide variety of computing styles, including:

- *Composed interactions with legacy systems and packaged applications* to provide higher-level, reusable business services
- *Longer running, stateful workflows* that link both people and applications together in order to implement business process models
- *State machine processing* that allows business entities to respond appropriately to various events that occur over the course of an extended business process
- *Asynchronous batch jobs* that can be scheduled to process background work at predefined times and at various priority levels without any human interaction
- *Web services interactions* with various partners in support of business-to-business scenarios

Complementary to these diverse computing styles, WBOnline has identified the following (equally diverse) architectural objectives required for the rollout of new Java applications:

- Changes to the business model should be possible in "real time"
- New international constituencies should be easily supported
- Updates across disparate applications must be achieved with high integrity
- Existing legacy assets should be reused as much as possible
- New applications should consist of reusable services that can be choreographed to meet evolving business requirements

This last point is significant, as WBOnline (like many other companies) is looking very closely at *service-oriented architecture* as a way of achieving increased levels of pluggability and reusability in the implementation of critical business processes. A service-oriented architecture allows virtually any software resource to be seen through the lens of a business service interface. This generalized notion of *services* builds upon the Web services value proposition to provide a consistent and inclusive programming

model that easily incorporates existing assets into new solutions. The result is faster assembly of new applications, improved consistency, and greater leverage of existing investments.

With that brief case study introduction behind us, let's now consider some of the ways in which *WebSphere Application Server V5* and *WebSphere Studio V5* can uniquely address the range of computing styles; architectural objectives; and operational requirements that WBOnline has identified in order to meet its business goals. We will do this in the context of a new WBOnline "Credit Card Application" that highlights the unique benefits that WebSphere offers.

As we work through this case study, it's convenient for us to reverse the order of our previous discussion about optimizations and extensions. Thus, we'll first examine some key WebSphere *extensions* that promote new levels of productivity, flexibility, and integration. Later on we'll expand the case study to account for key WebSphere *optimizations* that deliver industry-leading performance, availability, and scale.

## A Credit Card Application – Focusing First on Extensions

As previously mentioned, WBOnline wants to leverage its existing CICS investment as much as possible. Many of the CICS applications have been developed as "psuedo-conversational" 3270-based applications, and WBOnline has critical business information and business services that are hosted by this environment (including customer account information). WBOnline wants to leverage these assets in a standards-based way as part of developing its new Credit Card Application.

WebSphere Application Server and WebSphere Studio provide an open and pluggable solution for leveraging Enterprise Information Systems. Resources accessible through J2EE Connector Architecture (JCA) can be extended to provide developers with interactive access to the hosting environment, such that existing applications can be traversed dynamically in order to quickly compose a flow of interactions that produce a meaningful business result. The composed set of interactions can then be exposed as a generic and reusable service in support of a service-oriented architecture.

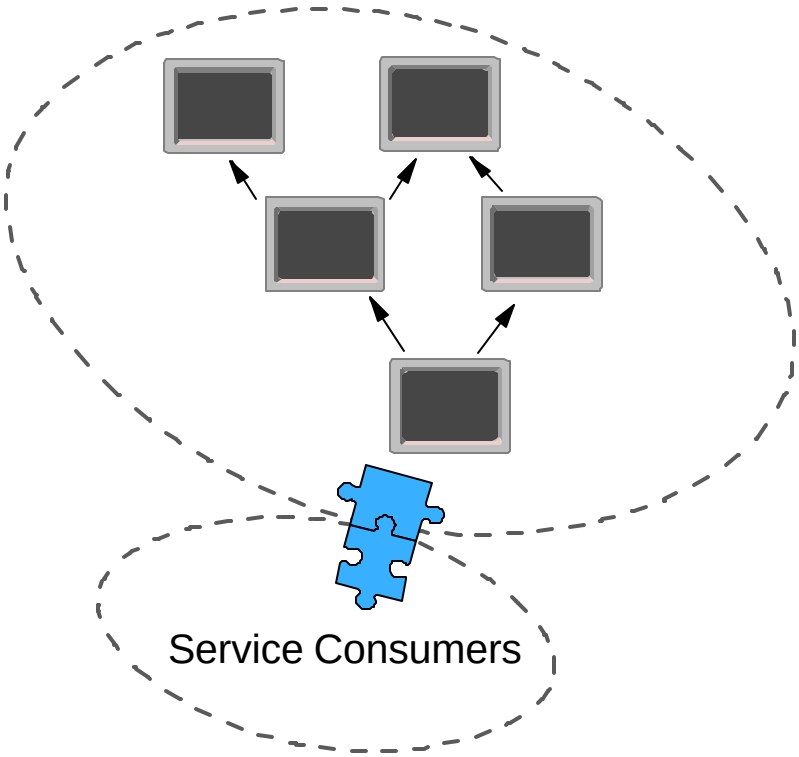


**Figure 7. *Host On-Demand* Interactions Delivered as an Aggregate Business Service**

Once defined, such services can be prepared for reuse in a variety of valuable contexts by providing easy access to Web-based clients; pervasive devices; traditional desktop applications; Web services partners; and messaging systems. Such services can also be adapted to accommodate different underlying protocols and to rationalize disparate low-level application interfaces. Finally, the services can be aggregated (or composed) into more course-grained services that will assume larger responsibilities in the implementation of the business model. Note that this style of aggregation nicely complements other WebSphere approaches to composition, such as tool-assisted support for "directly" merging components via patterns of aggregation and/or discrimination.

Returning to our Credit Card Application, we've still got some work to do. After WBOnline leverages WebSphere software to quickly create a customized and reusable service, it needs to integrate that service with other services and components, people interactions, and new business logic. This activity is essentially about implementing the workflow aspects of the larger application that's been previously defined (visually) by business analysts. WebSphere Studio plug-ins supplied by companies such as Holosofx provide the modeling linkage to such "top down" business process definitions. In any case, a simplified view of the workflow steps within our evolving Credit Card Application would look something like this:

1. **Obtain customer account information.**
2. **Determine if customer is "gold or above". If so, issue the card.**
3. **Determine the customer's credit rating. If "positive", issue the card.**

The first step has already been implemented through the "services encapsulation" of CICS, as previously described. The second step is dependent on a specific business policy. It's interesting to note that while automatic issuance of a credit card is *currently* based on the classification level of "gold or above", this policy could easily change in the future (to "platinum only", for example). WebSphere provides the technology for managing such business policies externally. Once the policies have been defined, business analysts can maintain and even schedule when they should take affect – without any further programmer involvement. This provides unprecedented flexibility and it promotes truly *dynamic* e-business as any arbitrary policy or rule can be managed in this fashion.

The concept of dynamic business policy management is depicted by the following figure. WebSphere supplies the framework for defining, executing, managing, and scheduling the rules that encapsulate variable business policy. Any discrete unit of business logic can be expressed as an externally managed rule. Developers initially create or select a rule that will be triggered from an application. Business analysts maintain the rule from there without further programmer involvement.

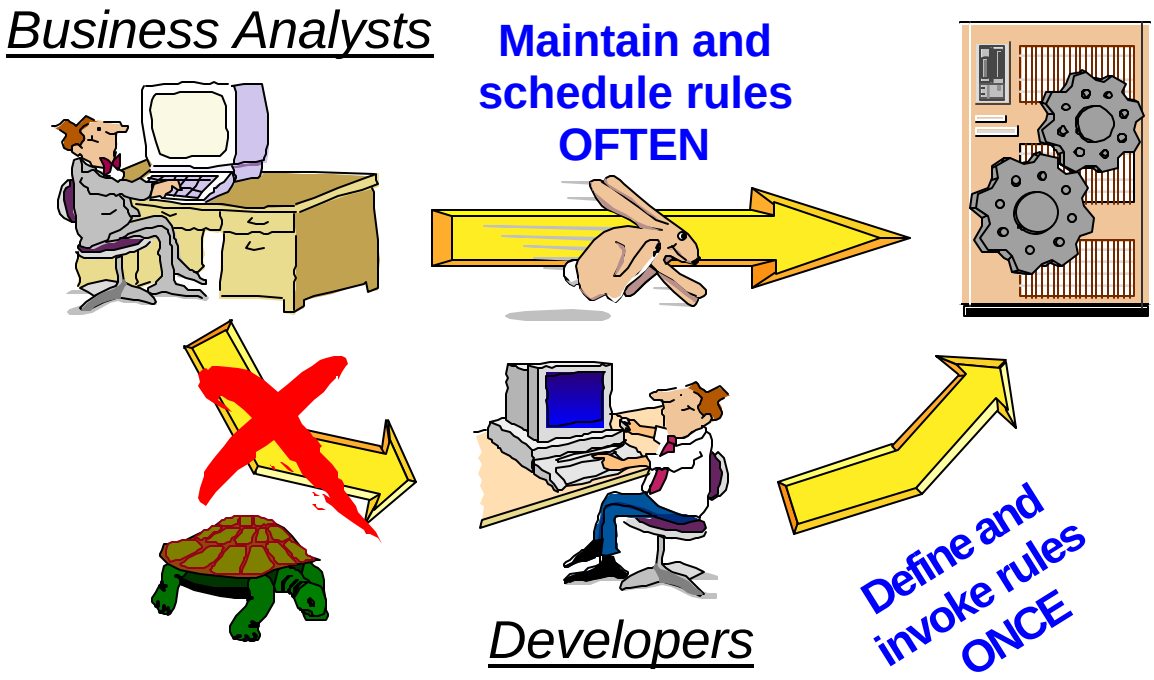


**Figure 8. Dynamic Management of Business Rules and Policies**

Can you see the benefits from externalizing rules and policies in this fashion? We suggest that some of these benefits include:

- Explicit documentation of business practice decisions.
- Clearer understanding of application behavior.
- Reuse of policies across business processes.

- Increased consistency of business practices.
- Decreased maintenance and testing costs.
- Improved manageability of business practice decisions.
- Increased confidence in predicting the business impact of proposed changes.
- Easy ability to identify and correct conflicting rules across the business.

Now we come to the third and final step in the Credit Card Application workflow. This merits significant discussion.

3. **Determine the customer's credit rating. If "positive", issue the card.**

This step is implemented using a Web services invocation to a Credit Bureau agency with which WBOnline is a partner. A SOAP message is sent over HTTPS in order to obtain the customer's credit rating. Note that WebSphere goes well beyond basic Web services support by providing a private UDDI registry that facilitates the sharing of approved business services (such as those provided by the Credit Bureau). The private UDDI registry supports internal Web services development efforts as well. WebSphere also includes a Web services-based transformation service that can intermediate between different XML formats. And WebSphere provides a standards-based, fully extensible Web services gateway. The gateway facilitates configurable mappings between internally and externally defined services so that WBOnline can leverage the most appropriate implementation protocols based on the location from which the service is invoked. Moreover, the gateway serves as a bi-directional control point for critical tasks such as security, validation, logging, transformation, metering, and so on.

Specific to our case study, WebSphere also deals nicely with a particular business issue related to the Web services invocation. As you would expect, the Credit Bureau charges WBOnline for performing credit checks, and therefore we need to ensure that restarting the Credit Card Application won't cause any "redundant" business-to-business interactions. WebSphere provides first class support for long-running, "interruptible" workflows to handle this case. Note that the same pluggable tooling and runtime elements that let you choreograph JCA-enabled resources and other services (as described previously – where we drove interactions with CICS) apply to this case as well. Indeed, ALL forms of service choreography are supported by WebSphere, whether the resulting flows are:

- *Microflows* (short running flows that execute as a single unit of work), or
- *Macroflows* (longer running, stateful flows that can be interrupted prior to completion and then automatically restarted from the last successful step).

Our case study reveals other important aspects of these long running flows. Specifically, if the Credit Bureau response indicates that the applicant is "unknown" then it becomes necessary for one or more people at WBOnline to manually investigate the applicant's creditworthiness. WebSphere Application Server addresses this requirement through integrated support for people and organizations playing various roles. In addition, WebSphere allows you to dynamically monitor the progress of the work by means of a

*dashboard*. The dashboard lets you identify precisely where you are in a given process at any moment in time. This ability to measure your work also gives you the ability to manage your work.

Now let's move ahead through our Credit Card process with the assumption that an applicant is to be awarded a credit card. In this case, the Credit Card Application needs to perform two separate but highly related tasks:

1. A message needs to be placed onto a queue informing a "detached" system to physically issue and mail the credit card at some convenient point in time.
2. An update reflecting the "approved applicant" status needs to be made to the heritage CICS system.

WebSphere simplifies the processing in task one through automated support for outbound (as well as inbound) messaging. This means that developers can focus on business logic instead of complex messaging API's. Handcrafted *Java Message Service* (or JMS) code is no longer required, as message content (and for that matter, transactional policies) are declaratively mapped onto the relevant business components. By blending component-oriented and message-oriented computing models in this fashion, WebSphere makes it easier for developers to quickly create high quality applications that integrate with other systems through messaging infrastructure such as WebSphere MQ.

The CICS update described in task two is also interesting. As with the first part of our case study, we drive one or more interactions with CICS in a microflow but this time we do so using the *External Call Interface* (or ECI). This style of communication supports the invocation of programs running in CICS under a *one-phase commit* protocol. In layman's terms, this means that if updates are made to another resource besides CICS, they will not be coordinated with the CICS updates as a single unit of work. This, however, is *exactly* what is required in the case of tasks one and two above – we want to ensure that an update to CICS will only occur if the update to the message queue is also successful (and vice-versa). From a business perspective these two updates logically go together hand in hand.

The J2EE specification offers no solution to this problem (that is, driving a single global transaction when one of the affected resources only supports a one-phase commit protocol). Indeed, J2EE demands that all resources used in a single global transaction support a *two-phase commit* protocol. While most of the world's databases and many messaging systems are indeed enabled for two-phase commit, this is effectively NOT the case for most of the packaged applications that are on the market. And in the context of our Credit Card Application, this is not true in the case of CICS ECI either.

So how could we address this common problem with WebSphere Application Server? It turns out that WebSphere *extends* the transactional capability defined by J2EE. Once WebSphere's transactional coordinator has ensured that all two-phase commit resources can be committed, it drives an interaction with the one-phase commit resource. Based on the outcome of that, the coordinator then directs the two-phase commit resources to either

commit or rollback, as appropriate. The net effect is straightforward – either ALL or NONE of the updates will occur, which is the required business result for our application. Just in case you are wondering, WebSphere also addresses the situation in which *many* one-phase commit resources are being updated as a group. In this case, each of these resources will be committed at virtually the same time based on an explicit application checkpoint. Any failures will then be reported back to the hosting application for appropriate resolution. The next figure summarizes some of the key points that we have made regarding the final step of the Credit Card Application workflow.

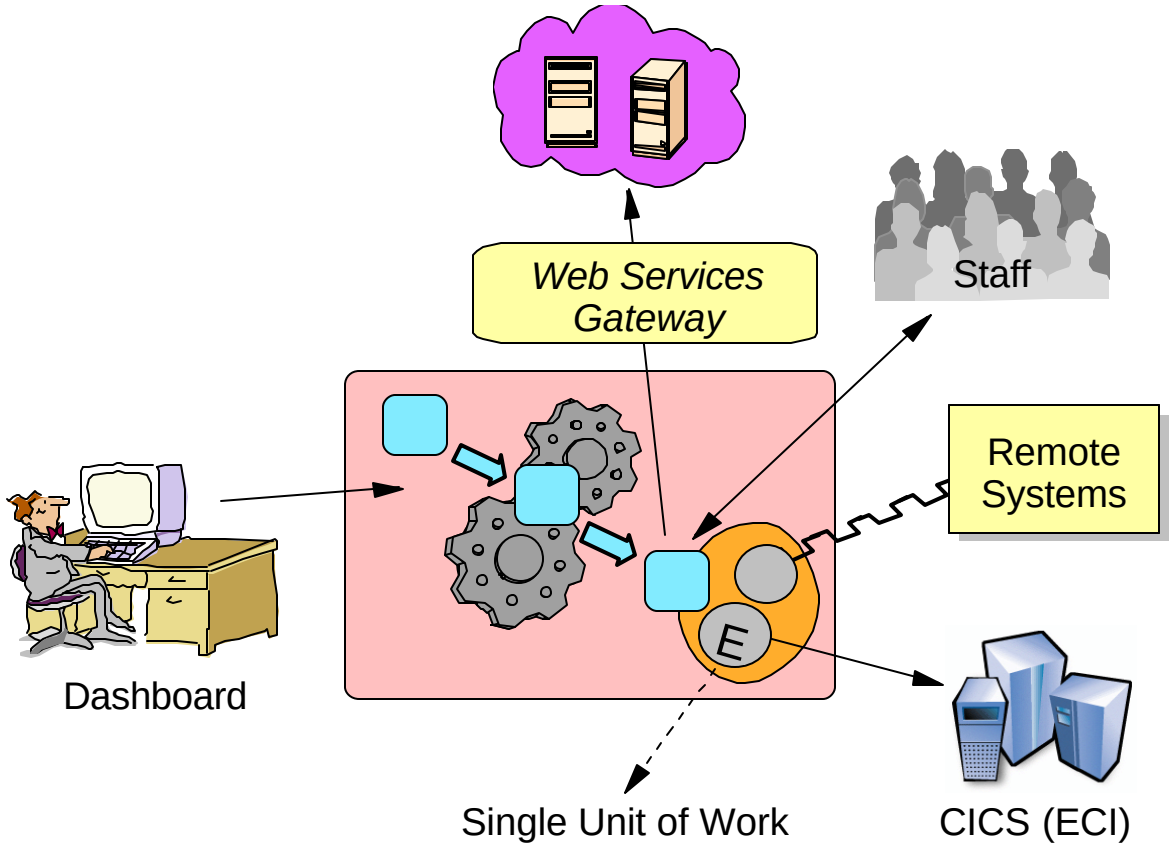


**Figure 9. Long-Running Workflow with Advanced Integration Features**

Perhaps you are wondering about the "E" that appears on the component related to the CICS environment. This designation stands for "entity", and it identifies another compelling feature about WebSphere Application Server. WebSphere makes it possible to represent systems and applications, not just databases, as persistent business entities. This is an advanced object modeling capability that makes it possible to render procedural environments in an object-oriented fashion, thereby mapping more naturally to the corporate business model and facilitating higher degrees of reuse.

There's another advanced object modeling capability worth mentioning here as well. This concerns WebSphere's advanced component query support with capabilities that extend well beyond what is defined by J2EE. Many real world component-based applications

need support for queries that return ordered results, maximum and minimum results, results based on date or time expressions, and results that depend on inherited attributes. Many applications also need support for truly *dynamic* queries. WebSphere allows dynamically composed queries to be executed based on real-time business needs. These queries can drive requests against both component operations and complex attributes. They can also provide efficiency gains by an *order of magnitude* for many query scenarios.

## Other WBOnline Applications – Still Focusing on Extensions

At this point we've investigated some of the ways in which the WebSphere runtime and tools deliver unprecedented productivity, flexibility, and integration benefits to WBOnline. Some of the features that we've considered include:

- Integrated runtime and tools support for service-oriented architecture including the ability to choreograph services that encapsulate standards-based access to Enterprise Information Systems; Web services; and more.
- Support for all forms of "flows", including microflows and long running interruptible macroflows that can also integrate people and roles into business processes. This includes dashboard support for monitoring and tracking as well.
- External business policy management that enables policies and rules to be maintained and scheduled by business analysts. Key benefits include adapting quickly to change, improved documentation and understanding, reuse of policies across processes, increased consistency, reduced maintenance and testing costs, and increased confidence in predicting the impact of change.
- A private UDDI registry that enables the publication of tested and approved business services. It also facilitates application development projects that need to leverage Web services technology.
- Web services-based XML transformation services, as well as an extensible Web services gateway that acts as a bridge between external and internal services; improves the ability to reuse application artifacts within and across the enterprise; and serves as a business and technology control point for Web services execution and transformation.
- A blending of disparate computing paradigms – for example, the "marriage" of component-oriented and message-oriented processing. This includes automated transformations and mappings between message flows and components in order to address diverse application scenarios.
- Advanced transactional integration that allows a one-phase commit resource to participate in global transactions, and that permits *multiple* one-phase commit resources to be committed at virtually the same time under application control.
- Advanced component styles that make procedural systems look like business objects, and query services that bring new capabilities and efficiencies beyond those offered in the current J2EE standard.

Now it's time to examine *additional* capabilities of WebSphere that bring still more value to WBOnline. Our ongoing storyline continues with two new applications, the first of

which updates customer accounts with year-end tax information. This work does not involve any user-facing elements, and it is the kind of (lengthy) processing that can be run in the "background" (or even in "parallel"). WebSphere Application Server handles this requirement through support for long running transactions with automated mechanisms for "retry" and "restart". Just as important, this work can be scheduled to run asynchronously through deferred execution that accounts for the notions of priority and context propagation (including the propagation of security credentials).

The other new application that we'll consider involves one of WBOnline's business partners in the travel industry. This application presents travel packages to WBOnline customers that are secured through bank loans. The travel partner generates significant business through this channel, and WBOnline drives new sources of lending income. As with the Credit Card Application, service choreography involving Web services and straight-through business processing is complemented by external business policy management. But this new application surfaces additional requirements that reveal other critical capabilities of WebSphere as well. For example, consider the steps that are involved in booking a trip:

1. Book a hotel.
2. Book a car.
3. Book a flight.

Suppose that the hotel and car reservations are successfully booked but the preferred flight is unavailable. In this case, the two previous (independent yet logically related) transactions need to be "undone". More specifically, *compensating transactions* are required to reverse the effects of the original work. Fortunately, WebSphere provides automated compensation facilities (along with associated visual development tools) that squarely address this kind of requirement. It's interesting to note that the precise nature of the compensation is basically arbitrary. Some kinds of work cannot be reversed as cleanly as one might like. This is why "compensate" is really a more appropriate term than "undo".

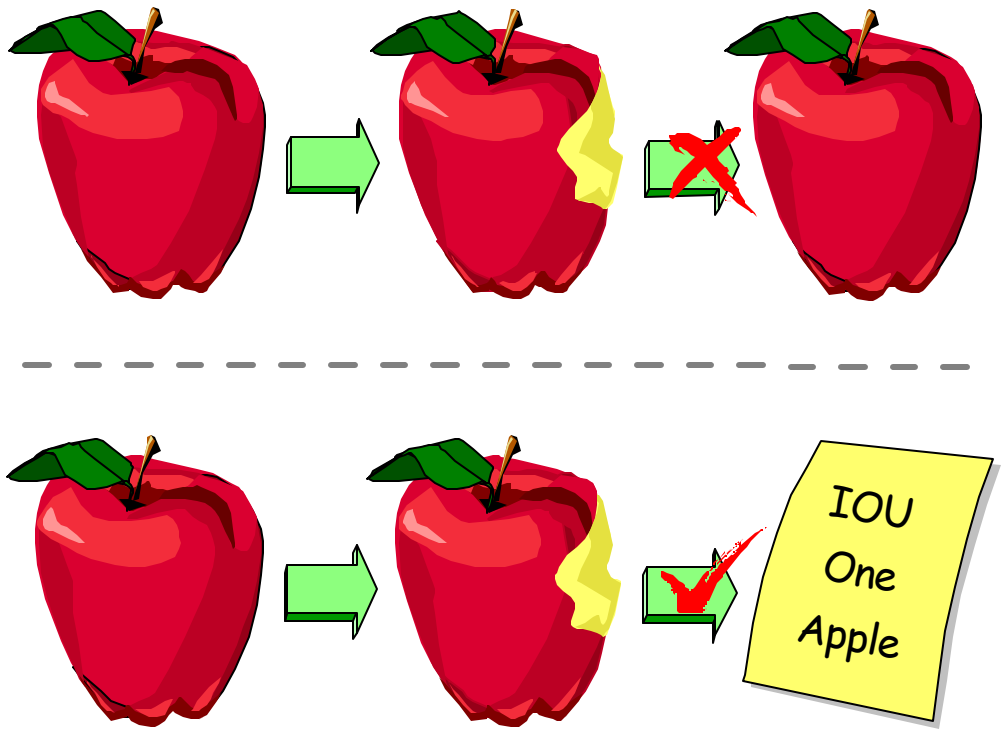


**Figure 10. Compensating Previously Committed Actions**

Continuing with our trip discussion, let's assume that the end user is able to successfully book the desired itinerary. From a business perspective, this will result in the creation of a "trip entity" that has its own lifecycle and that will need to be appropriately managed. At various points over the lifecycle, business events must be handled properly based on the terms and conditions defined for the trip. The next figure shows a *state transition diagram* that illustrates some key concepts associated with *state machines*. As business events occur, a business entity assumes different states against which business policies can be applied. The primary focus of this figure is on the effects of canceling a trip. There are different points at which a trip may be canceled and the business actions resulting from this depend on the current state of the trip as well as the business conditions that are in effect. Different actions may be associated with various transitions (such as penalty charges). Just as in the case of automated compensating transactions, WebSphere lets you visually define the critical states of your business model and the appropriate business actions that are associated with state transitions. WebSphere allows you to deploy these same definitions into a robust runtime environment without the need for extensive handcrafted programming.

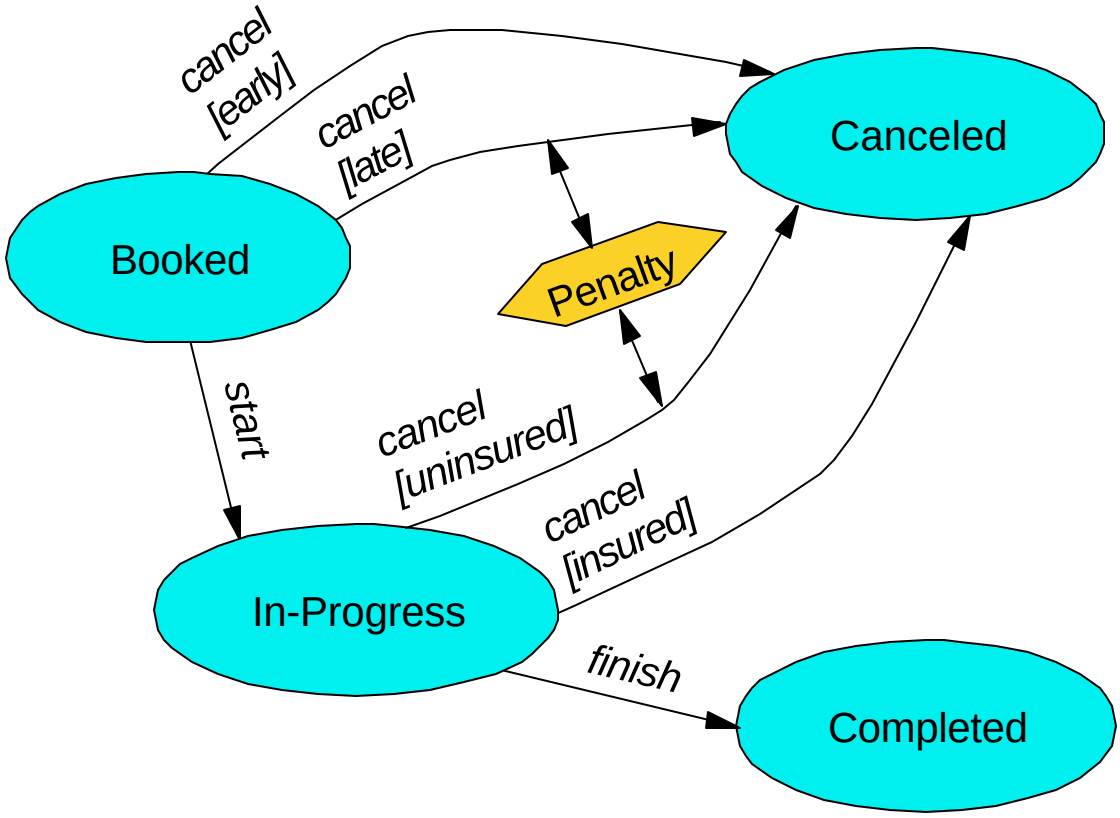


**Figure 11. Modeling the Effects of Canceling a Trip**

Before we conclude this section of the case study, let's recall the fact that WBOnline needs to serve international markets. A smart server needs to account for different cultural conventions, languages, and geographical considerations. It should also be "time zone aware" with respect to clients. WebSphere Application Server meets this requirement through an *internationalization service* that lets you to properly handle date and time information, formatting of currency and decimal points, language selection, sorting rules, and more. Please note that these capabilities are not just presentation-related. They are very relevant to business logic as well. WebSphere applications can easily leverage internationalization context across a distributed application *end-to-end*. This lets your business applications adapt quickly to new geographical constituencies.

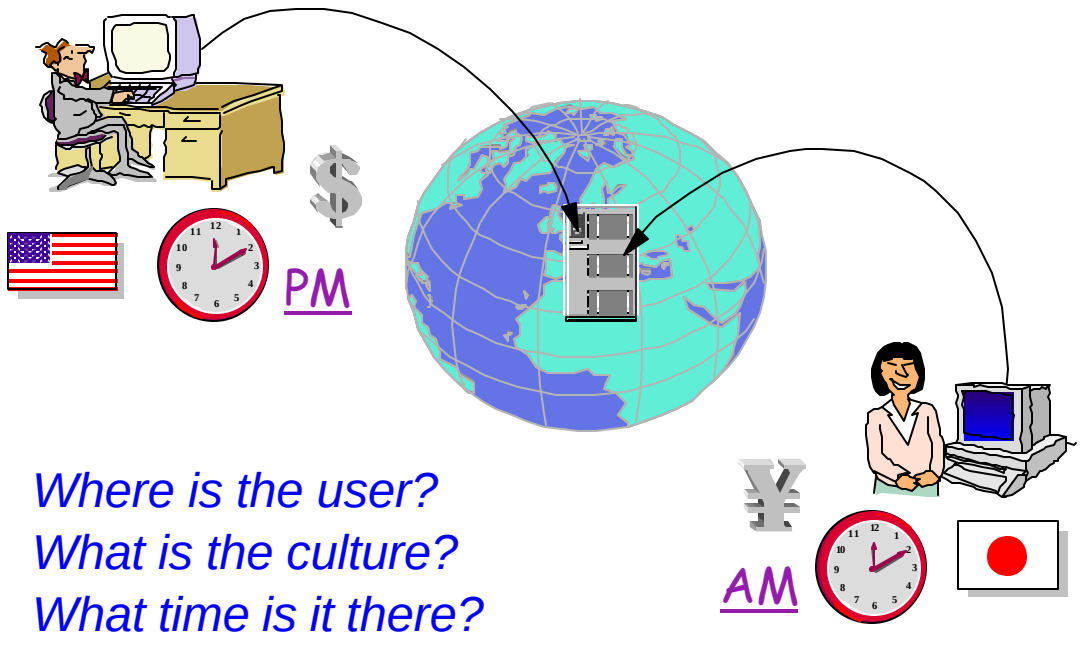


**Figure 12. Internationalization Considerations Apply End-to-End**

The internationalization service helps your developers to be more productive since the details about internationalization flow across the application *transparently*. This last point is conceptually related to another WebSphere extension in which *any arbitrary business information* can be shared non-intrusively across the distributed environment. This means that customer profile information, for example, can be efficiently leveraged wherever it's needed without the expense of reading and writing from databases or the development costs associated with opening up application interfaces.

Finally, consider the fact that WBOnline – like many companies – has made significant investments in building ActiveX, CORBA, and C++ applications. WebSphere helps customers preserve their investments in these technologies by providing:

- An ActiveX bridge that leverages high availability and load balancing features. This makes it possible for existing Microsoft clients and servers to reliably access J2EE components running in WebSphere.

- Bi-directional CORBA support for interoperating with various third-party ORBs. This makes it possible to leverage existing CORBA investments from WebSphere applications, and to invoke WebSphere components from CORBA environments[6].

- A C++ CORBA Software Development Kit for integrating various C++ assets. This lets C++ clients invoke J2EE components using CORBA technology. It also lets WebSphere applications incorporate C++ assets behind CORBA wrappers.

[6] Support varies depending on the ORB, platform, language, direction of interaction, and type of data. Contact your IBM representative for more specific details.

## Revisiting the Applications – Focusing on Optimizations

The features that we've covered up to this point in our case study represent valuable *extensions* that are delivered by WebSphere Application Server and WebSphere Studio tooling. These extensions clearly offer productivity, flexibility, and integration benefits to WBOnline. But *even more is required*. In serving its global customers WBOnline must run these applications in highly distributed configurations with unparalleled performance, availability, and scale. Let's continue our case study by considering some of the ways in which WebSphere *optimizations* can fortify WBOnline's application serving environment:

- **Profile-based access to data sources.** Distributed components dynamically receive instructions at runtime concerning access to data. The same component can behave differently depending on the application that calls it. This means that strategies about concurrency, update intent, use of collections, and pre-populating component state and data buffers can be carefully optimized to each application's needs without impacting application source code, or even requiring redeployment.
- **Predefined database access.** While profiles provide optimized access to data at the component level, it's also important to facilitate optimized access to data at the database manager level. WebSphere allows database managers to evaluate, define, and store a given access strategy *before* the access is actually required through support for SQLJ technology.
- **Dynamic workload management.** This capability goes well beyond the weighted selection of application servers. Indeed, it allows server selection to be based on unpredictable and highly variable runtime conditions such that work can be redirected to those machines that are the least busy.
- **Cross-domain failure bypass.** Even in the face of catastrophic loss such as the elimination of an entire network domain, work will be redirected from clients over to a mirror system so that critical business processing will continue. This provides first-class support for enterprise-class availability requirements.
- **Pluggable infrastructure.** WebSphere implements standards-based "plug points" that let you introduce new or existing security solutions, workload management solutions, or systems management solutions into your runtime environment. For example, *Java Management Extensions* (or JMX) provide hooks for integrating with systems management tools and also enable WebSphere's support for SMNP.
- **Flexible, self-enhancing infrastructure.** WebSphere utilizes actual runtime statistics to drive continual application performance improvements. In addition, proximity-based naming services lead to efficiencies in server selection and add flexibility when modifying deployed configurations. Other aspects of flexibility include roles-based administration and asynchronous application updates.

Each of the previous points brings competitive advantages to WBOnline, and each feature is really a topic unto itself. While a full treatment is beyond the scope of this paper, let's consider the first point in some more detail by mapping it back to WBOnline's application environment.

Having made a strategic commitment to J2EE, WBOnline has designed its applications in a way that maps the business model to a set of collaborating software components. In J2EE parlance, WBOnline is making good use of *Enterprise JavaBeans*™ (or EJBs). This approach offers strong traceability between the business problem and the software solution. It facilitates parallel, roles-based development and testing with optimized tools for different tasks. And it promotes reuse for improved consistency and faster assembly of new applications.

One of the ways in which WBOnline has applied component technology is in the creation of a reusable *Account* EJB. This component provides services to the Credit Card Application, the Year-End Tax Application, and many others. The *way* in which these applications need to use this common component varies greatly however. For the Credit Card Application, account information for a particular customer is retrieved but not updated. Indeed, the update occurs elsewhere in the application – to a *CreditCard* component, as you would expect. Since this application is "user facing", fast response time is required. The Year-End Tax Application, on the other hand, runs as a background task. As such, it can execute with a lower priority. Moreover, the nature of the application is more about *update* than retrieval, and the *scope* of this application extends to *all* customers, not just one customer.

To accommodate these diverse computing requirements, WebSphere Application Server provides profile-based access to data sources (also known as *application profiles*). The concept of application profiles is not really new. In CICS environments, IBM has long supported the notion of attaching different operational characteristics to different transaction names, where these names might actually be mapped to the same underlying source code. The ability to cleanly separate operational issues from business logic is the ultimate goal here. In the context of J2EE, what we'd like is a mechanism for instructing the same component (or more accurately, the so-called *container* or environment surrounding that component) to interact with a database; a workload manager; or any other part of the runtime infrastructure in a fashion that best serves the needs of the application which is currently running. The source code of the component should not be affected and the need to redeploy the component should be minimized. The net effect is that the very same component can provide optimized services to different types of applications at exactly the same time as a result of administrative decisions. Some of these concepts are related back to our WBOnline case study in the following figure.

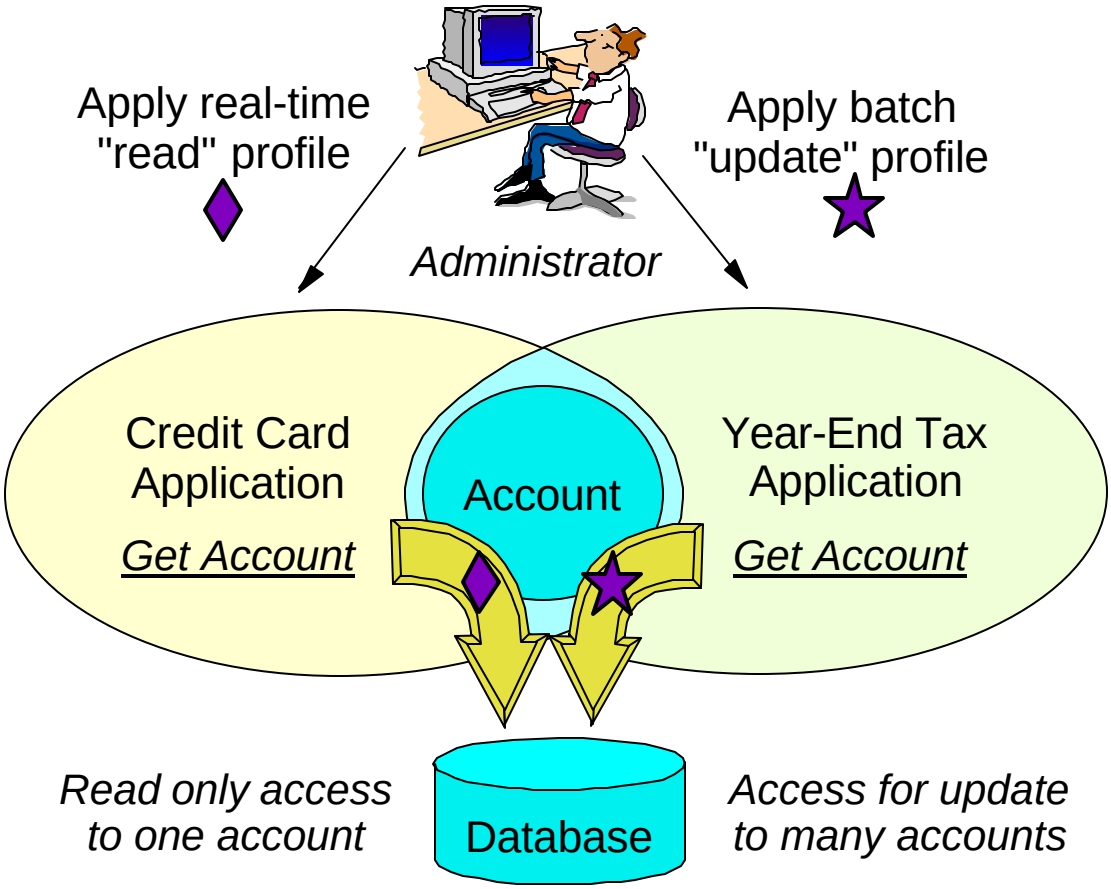


**Figure 13. Application Profiles Offer New Levels of Runtime Control**

## Going to the Edge – Still Focusing on Optimizations

WebSphere optimizations extend to the very edge of the network. This is critical to a global company like WBOnline, which offers core business services to constituencies worldwide. In many of WBOnline's application scenarios, the vast majority of the time required to service requests is consumed in network overhead. WBOnline realizes that optimizations need to be pushed out beyond the immediate boundaries of the enterprise in order to provide services closer to the clients that require them. In this regard, WebSphere Application Server offers the following kinds of sophisticated optimizations to support enterprise-class needs in highly dynamic distributed configurations.

- **Intelligent load balancing** that promotes very high availability, scalability, and performance by transparently clustering all distributed servers (including those at the network's edge). The load balancing includes support for site selection, session affinity, transparent failure bypass, custom advisors for determining server health, content-based routing, and both Cisco and Nortel consultants for load optimization and fault tolerance.
- **Enhanced caching services** that include support for dynamically generated pages as well as dynamic content invalidation to control the timing of refreshed pages.

Note that caching services are also integrated with *content distribution services* (which are described next) for higher scalability.

- **Content distribution** deploys published Web content to caches and re-hosting servers throughout the distributed network in a secure fashion, pre-positioning the content closer to the user for an enhanced user experience.
- **Policy-based quality of service** allocates computing and network resources based on a transaction's real-time business value (while at the same time interoperating with existing networking hardware). This prioritizes networking throughput for the most highly valued customers.
- **Security services** that include support for the *Secure Sockets Layer* (or SSL) protocol, together with SSL endpoint affinity and optional hardware-assisted cryptography for enhanced, secure connections to Web resources. In addition, with the integration of Tivoli Policy Director, customers can achieve a centralized security solution that includes user authentication at the edge, cross-domain single sign-on, and web-based access control for blocking unauthorized access of cached and non-cached Web resources.

WebSphere's capabilities at the edge of the network are bringing tremendous value to IBM customers today. In the words of Jeff Lucas, IT Director, The All England Lawn Tennis Club, *"WebSphere…enabled us to support 3.2 million unique users and 208 million page views at Wimbledon. The openness and speed of WebSphere was a tremendous asset for us throughout the two weeks."*

## WebSphere Application Server – the Right Choice

As demonstrated by the case study, *WebSphere Application Server V5* delivers new capabilities that are designed to help you gain sustained competitive advantage. In this final section of the paper we suggest that there are many additional reasons to choose WebSphere – and we hope that you'll agree.

**Support for complete solutions.** The value proposition for WebSphere Application Server starts with support for complete solutions. WebSphere Application Server provides the foundation for a complete e-business infrastructure that spans everything from "reach and end-user experience" to deep "business integration". When WebSphere won *Network Computing's* Editor's Choice award, the magazine stated that *"WebSphere Application Server is part of IBM's larger suite of complementary products that sits on top of this world-class solution."* Network Computing's most compelling remark about the application server was actually based on the things that run on top of it – the full set of solutions that it supports. Of course, WebSphere also benefits from IBM's expertise and ability to supply end-to-end services and enablement; the complementary experience and capability delivered by the Lotus, Tivoli, and DB2 brands; clear hardware leadership; strengths in Global Financing; and a set of very strong industry partnerships. And this discussion about complete solutions would not be complete if we didn't also mention integrated tooling – one of IBM's key differentiators.

**Unparalleled sustained investments.** WebSphere Application Server is backed by IBM's year-over-year multi-billion dollar investments and many thousands of IBM employees, with industry-leading developer programs that promote widespread WebSphere adoption. WebSphere has been one of IBM's "must-win" initiatives with top-level executive commitment and extensive focus across the company. WebSphere has also benefited from IBM's enormous investments in research and development (some $5.6 billion dollars in 2001 alone). There were 524 patents attributed to IBM software this past year – more than any other software company in the world. Of these 524 patents, 222 were specifically related to WebSphere[7].

**Core strengths and competencies.** WebSphere Application Server incorporates IBM's core capabilities in building system software. These capabilities include transactional and security leadership; an ongoing focus on interoperability; IBM's heritage in delivering messaging and persistence capability as well as component technology; strengths related to Web services and XML; industry-best support for manageability (including synergies with Tivoli); and significant experience in the area of application integration and connectivity.

**Clear industry leadership.** WebSphere Application Server has very strong industry momentum that is second to none (as measured by both analyst and press opinions, market share numbers and trends, and growing developer momentum). WebSphere has won key competitive battles in accounts like eBay and Abbey National, and more than 50,000 customers are now using WebSphere worldwide. Business partners are also increasingly selecting WebSphere as their application server of choice. WebSphere delivers first-rate support 24 hours a day, seven days a week. WebSphere Application Server and Java solutions from IBM have together won more than 30 industry awards. Beyond garnering an impressive number of industry awards, WebSphere has allowed customers such as Bekins Van Lines to receive first-place industry awards of their own.

**Excellent return on investment.** WebSphere Application Server helps businesses improve customer loyalty, respond more quickly to change, and reduce overall costs. This last point hints at WebSphere's advantages related to overall cost of ownership measured in terms of truly excellent price/performance, as well as the provision for customers to leverage their existing investments. WebSphere customers enjoy an integrated solution stack where each component has been engineered and tested to work within a larger solution space. Again, this solution stack includes integrated tools – tools that make it possible to build and deploy high quality e-business applications faster than ever before.

---

[7] Patentable code specific to the WebSphere Application Server includes performance optimizations such as application profiling; advanced caching and in-memory data replication; flexibility extensions provided by shared work areas and the internationalization service; and business integration extensions that include support for process automation services and advanced transactional models. Patentable innovations also include things like Java class loading and application isolation, HTTP session affinity and profiling, inventions related to threading and scenario-based testing, numerous security inventions, and – specific to IBM's tooling – advanced techniques related to code generation and object modeling.

**Support for industry standards.** WebSphere Application Server is J2EE-certified on more platforms than any other vendor. WebSphere is the first and only major application server brand to become J2EE 1.3-certified last year (through the Technology for Developers release). WebSphere engineers have contributed to more than eighty percent of the J2EE specification, and these engineers continue to define the next wave of standards through participation in the Java Community Process. WebSphere was first in line to deliver an integrated Web services solution with a complete set of associated tooling. WebSphere engineers have defined, co-authored, or significantly contributed towards all of the relevant standards in the Web services and XML space.

**Scale and performance across platforms.** WebSphere Application Server scored first place in PC Magazine's public benchmark. WebSphere also delivered superior price-performance in realistic configurations as part of the public ECPerf benchmark. Beyond that, WebSphere demonstrated the scale and performance results necessary to secure a victory in the highly visible eBay bakeoff. According to Evan Quinn, chief analyst with the Hurwitz group, the eBay contract suggests that WebSphere is *"reliable enough to handle the huge transaction volume served up by even the busiest Internet sites"*. WebSphere customers are able to achieve many tens of millions of page views per day in production on the Web. Kana recently set an eCRM Solution industry benchmark by running on WebSphere. And IBM has demonstrated more than 12,500 EJB-based transactions per second – serving 800,000 users – with approximately one quarter of a second average response time. WebSphere delivers the industry's best vertical scaling, and only WebSphere leverages the native scaling facilities that are offered on z/OS. WebSphere enhances base application server performance through advanced caching technologies and edge of network performance enablers. WebSphere is supported by a High Volume Web Site team, and by a High Volume Web Site Performance Simulator used in capacity planning. WebSphere includes performance-monitoring APIs and tooling that lets customers measure application load characteristics with both IBM and third-party software. WebSphere offers performance wizards and is backed by substantial performance-related collateral (such as redbooks and white papers).

Taken together, we believe that all of these things spell ***SUCCESS*** for your business.

**S**upport for complete solutions
**U**nparalleled sustained investments
**C**ore strengths and competencies
**C**lear industry leadership
**E**xcellent return on investment
**S**upport for industry standards
**S**cale and performance across platforms

# Conclusion

*WebSphere Application Server V5* and *WebSphere Studio V5* help you to cut costs, build customer loyalty, and adapt quickly to new business opportunity. With this release, WebSphere continues to raise the bar in the application server space.

- IBM is delivering "tomorrow's standards today" through industry-leading *extensions* that improve productivity, flexibility, and integration. In support of these extensions, WebSphere engineers are continually driving Java Specification Requests into the Java Community Process and they actively contribute to the work of open standards bodies such as W3C.
- WebSphere Application Server is leading the way in delivering sophisticated enterprise-class optimizations that enable the best in performance, availability, and scale. These optimizations address the very high-end requirements for deploying e-business applications into dynamic and distributed configurations.

Perhaps IBM's own customers say it best. To quote Thomas Leemann of UBS, a leading global financial services firm that serves more than four million personal and corporate clients, *"In today's market, robust e-business infrastructure is critical. We look to WebSphere's new software to support our efforts to maintain a highly satisfied, loyal customer base, and to do it more efficiently and cost effectively than our competitors. With the integration features, WebSphere helps us shorten time to market through faster application development, while reducing our ongoing maintenance costs. And, it integrates easily with our existing e-business applications, enabling us to leverage our prior IT investments. That boosts the return on our investment while enabling us to support thousands of users securely and reliably, day in and day out."*

So what do you do next? You may be wondering how you will gain the knowledge and skill to make the best use of this technology. We emphasize that WebSphere is about delivery of *expertise* and not just software. WebSphere solutions include the services of a talented and experienced team of engineers and consultants who stand ready to work with you on your most pressing business needs. WBOnline used these services to establish an enabling Center of Competency in record time – and so can you. We invite you to contact your local IBM representative *today* for more information about how you can achieve business results well ahead of your competition. IBM's WebSphere team is at your service.

Note that online information about WebSphere Application Server software and solutions can be found at www.ibm.com/software/webservers/appserv.

# Appendix A – Integration

This paper made the point that WebSphere Application Server and WebSphere Studio deliver significant integration benefits. Integration is a very broad term and there are many styles of integration that address different kinds of business problems. This idea is depicted by the following figure, which introduces a variety of integration themes that are supported by the overall WebSphere software platform.

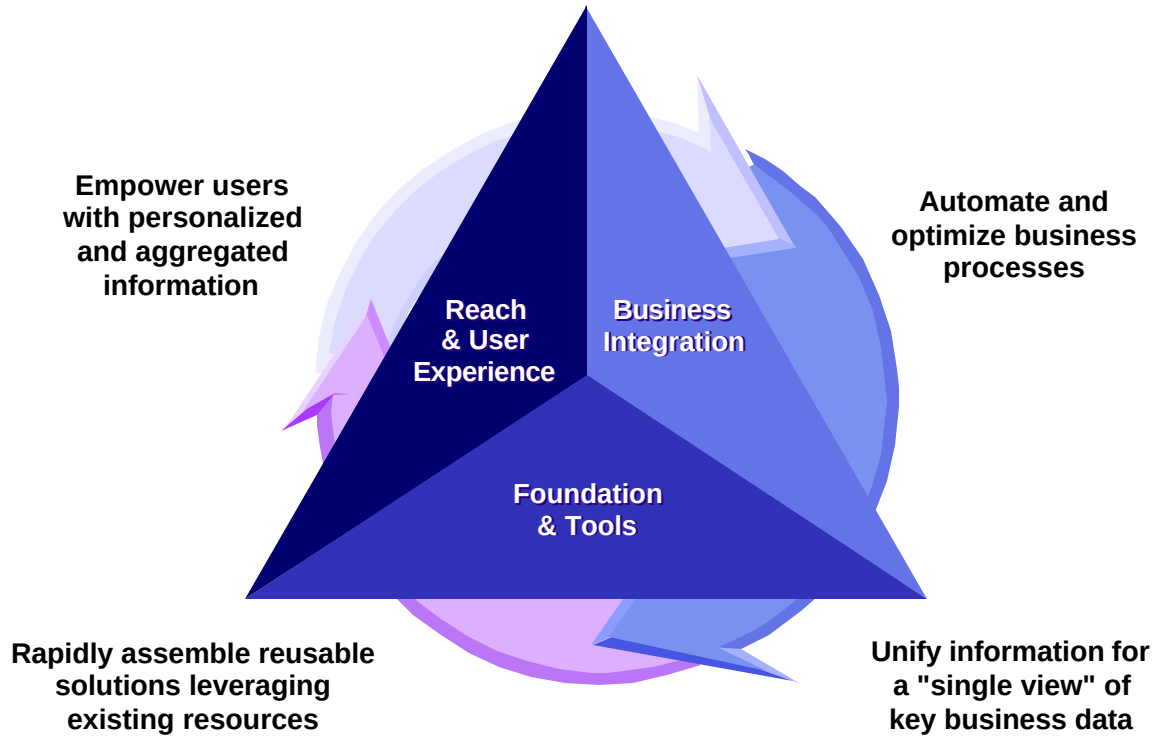


**Figure 14. The WebSphere Software Platform Supports Multiple Styles of Integration**

The triangle in this graphic illustrates that WebSphere – as a complete software platform that goes beyond the foundation layer of application serving and tools – is essentially about three things:

1. **Reach & User Experience:** Providing personalized and streamlined access to content and collaborative services on any variety of devices (including pervasive devices). Note that this also includes the ability to conduct electronic commerce.
2. **Business Integration:** Providing integration services both within and between enterprises in order to promote business agility and to strongly support business-to-business initiatives.
3. **Foundation & Tools:** Providing an infrastructural underpinning for a whole range of e-business solutions. Application serving and integrated development environments are some of the key elements here.

Around the outside of the triangle you'll notice that there are four brief descriptions. Starting at the upper left corner and moving clockwise, these relate to four distinct styles of integration (as follows):

## Empower users with personalized and aggregated information

Some customers need an integrated user interface for accessing diverse applications and content. They are looking for a rich user experience characterized by personalized and streamlined access to both content and collaborative services. *Key objectives* in this case include support for pervasive devices, single user signon, pluggability and extensibility in order to efficiently support new portal sources, powerful and flexible real-time search capabilities, and analytics support for measuring portal effectiveness.

The IBM solution for this integration requirement is *WebSphere Portal Server*.

## Automate and optimize business processes

Some customers are looking for the automation of business processes that span complex, heterogeneous, legacy environments – these environments frequently consist of packaged applications such as SAP and Siebel. They may also require business process integration of people and organizations. *Key objectives* in this case include the modeling, managing, and monitoring of both intra- and inter-enterprise business processes in order to promote business agility and to achieve high return on investment; support for robust connectivity; support for business-oriented cost analysis; and rapid deployment achieved through process templates.

The IBM solution for this integration requirement includes *IBM CrossWorlds* offerings, *MQSeries Workflow*, and complementary offerings from *Holosofx*.

## Unify information for a "single view" of key business data

Some customers need to unify, connect, and integrate diverse information for a single view of key business data. This includes the loose coupling of heterogeneous application information and data via transforms, routing, and messaging. It also includes services that provide message enrichment and data capture associated with message flows. *Key objectives* in this case include dynamic distribution and streamlined routing of information between different application sources, rules-based information connection, and content-based publication and subscription services.

The IBM solution for this integration requirement includes both *WebSphere MQ* and *WebSphere MQ Integrator.*

## Rapidly assemble reusable solutions leveraging existing resources

Some customers are strongly committed to J2EE and to the development of new Java-based applications that will be deployed into an application server environment. These new applications frequently need to leverage existing legacy assets in combination with the development of new business logic written in Java. *Key objectives* in this case include high operational productivity through a single administrative environment, high development productivity through an integrated development environment, standards-based connectivity, strong support for component models, and flexible "service choreography" facilities for defining both fine and course-grained flows (including people interactions).

The IBM solution for this integration requirement includes *WebSphere Application Server* and *WebSphere Studio.*

There are a couple of critical things that you should note about all four solutions:

- These solutions are not mutually exclusive. For example, customers using CrossWorlds and who need new J2EE applications with fine-grained integration to existing assets will also want to use WebSphere Application Server. Customers using WebSphere Application Server and who need support for information transformation and routing will also want to use WebSphere MQ Integrator. Customers using WebSphere Portal Server for integration “on the glass” and who also need to achieve integration between diverse backend systems will also want to use one or more of the other IBM integration solutions.
- Taken together, these solutions represent the industry's most complete integration story. IBM is exceptionally well positioned to address the broad and diverse requirements that are found in the "integration space".

# Appendix B –Technology Mappings

The following table maps a subset of the case study business discussion to the underlying enabling WebSphere technologies. The table captures a few of the more interesting associations, but is not intended to be exhaustive. The technologies listed here are candidates for a follow-on, deeper technical discussion to be provided by subsequent documentation.

| **BUSINESS FUNCTION** | **ASSOCIATED TECHNOLOGY** |
|---|---|
| Direct access to Enterprise Information Systems | *JCA Wizards* (with WSDL-based metadata import) |
| Choreographed interactions to produce an aggregated result | *Process Engine Support* (for *microflows* of J2EE and Web services assets) |
| Service choreography for workflow automation | *Process Engine Support* (for *macroflows* to automate business process steps including human interactions) |
| Automated business policy management | *Business Rule Beans* |
| Registration of tested and approved business services | *Private UDDI Registry* |
| Monitoring and management of business service invocations | *Web Services Gateway* |
| Deep application integration with messaging systems | *Container-Managed Messaging* |
| Extended unit of work support for a "one-phase" resource | *Last Agent Optimization* |
| Extended unit of work support for *many* "one-phase" resources | *Activity Session Service* |
| Representing applications and systems as persistent entities | *CMP Over Procedural* |
| Advanced query services | *EJB Query Extensions* |
| Background and parallel processing | *Deferred Execution*<br>*Scheduler*<br>*Asynchronous Beans* |
| Automated compensation and event processing | *Business Process Beans* (for *compensating transactions* and *state machine* support) |
| Adapting to global constituencies | *Internationalization Service* |
| Efficiently sharing information using "virtual scratch pads" | *Shared Work Areas* |
| ActiveX integration | *ActiveX Bridge* |
| CORBA integration | *CORBA Interoperability* |
| C++ integration | *CORBA C++ Software Development Kit* |
| Profile-based access to data sources for maximum efficiency | *Application Profiles* |
| High performance/availability in widely distributed networks | *Edge of Network Technologies* |