Foreword

When we first introduced the IBM® WebSphere® Application Server software (WebSphere), we had a vision that we were creating *the* foundation for information computing for the next several decades. We had as inspiration the IBM traditions of TPF, IMS, CICS®, DB2® and MQ—middleware platforms that have been hosting mission-critical business applications for the last 40 years. Only the future will tell whether our aspirations will be realized but in the short history that has amassed so far every indication is that we have established a solid foundation for achieving exactly that. WebSphere is used at some of the largest corporations in the world for both web-based information-computing as well as core on-line transactional computing. Companies such as eBay, Bank of Tokyo-Mitsubishi, Daimler-Chrysler Corporation, Dresdner Bank, Office Depot, SAS Airlines, Toshiba, Siemens Medical Solutions, as well as thousands of other customers use WebSphere to run their businesses.

At the end of 2003 we saw WebSphere take a commanding lead in the Java® 2 Enterprise Edition (J2EE) Application Server marketplace—somewhere in the vicinity of 29%[1] to 41%[2] market-share, depending on how you count it. WebSphere has gained market-share over BEA, Oracle, Sun or any other J2EE vendor. More importantly, the basic style of computing supported by WebSphere as defined by the J2EE specification is rapidly growing in importance for mission-critical business application deployments, representing a US$2.8B opportunity in 2003[3] just for the middleware to support these environments.

But great stories like these are not just created from marketing charts—they result from the culmination of an understanding of the marketplace, deep insight and experience in computing technologies, a sharp and detailed long-term vision for what information computing can offer to users and, at the end of the day, a great deal of hard work and perseverance—a dedication and passion for pursuing the right things.

As you may be aware, IBM is a large and globally distributed company with development organizations in dozens of cities throughout the world. This is one of the things that makes IBM a great company—it allows us to maintain a high degree of cultural diversity. This, in turn, ensures

[1]IDC, May 2004
[2]Gartner Dataquest, May 2004
[3]Gartner Dataquest, May 2004

that we retain a strong sensitivity to the needs of different users operating under different economic and environmental conditions, and to remain tuned to the patterns of local concerns throughout the world. However, that same diversity can also create huge challenges to development processes. Diversity of culture also means diversity of opinions and perspectives and those differences show up in the technical debates about what's important and why.

Diversity of locale means operating over different time zones, and without the inherent benefits of face-to-face communication. It means working through different holiday and vacation schedules. It means having to matrix across different organizational, legal, geo-political, and sometimes even language barriers. It means forging a common and mutual understanding where none might occur naturally.

When we started the WebSphere project we had a choice between developing the entire WebSphere offering at one location and developing it in a distributed fashion across multiple development sites. The decision was not hard to make. We knew that WebSphere would have to incorporate many different technologies—transaction management, state persistence and query, distributed communication, security, administration, web processing, caching, workload management, messaging, resource management, multi-tasking and contention management, session management, serviceability, high and continuous availability, performance and monitoring, and on and on. We chose to leverage the skills, knowledge and deep histories that already existed within IBM, at the locations that major in each of those technologies. We tapped messaging resources from the MQ team in Hursley, England; query resources from the DB2 team in Santa Teresa (now Silicon Valley Labs), CA; administration resources from the Tivoli team in Austin, TX; integration and componentization resources from the OS/400® team in Rochester, Minnesota; tooling resources from the VisualAge® and now Rational® team in Toronto, Canada; web processing and performance resources for the Servlet Express team in Raleigh, NC; scalable systems resources from the zOS® team in Poughkeepsie, NY; transaction management from the Encina® resources in Pittsburg, PA and CICS resources in Hursley; etc. Sure, that made managing the team and product more difficult (*a lot more difficult!*), but the alternative would have meant reinventing all of that knowledge and skill and that was something that we did not want to do.

As I said earlier, building a successful product like WebSphere takes a lot of work. Going essentially from scratch in 1999 to WebSphere V5 in 2004 has required painstaking refinement of technologies to get them to blend well together, to meet the latest and (at first) rapidly evolving industry standards, to balance and tune the disparity of workload types that different customers experience, and to achieve both something that is easy to install and use and something that will scale up to the largest of installations. We have done that in WebSphere by keeping to a few basic architectural principles, including: *maintaining a clean separation of concerns*—don't let one set of issues coerce a different set of issues; *minimizing the amount of functional duplication*—it is better to put more focus on getting one really good implementation than many poor implementations; on the other hand, *not assuming one-size-fits-all*—use the separation of concerns principle to differentiate quality-of-service and scale differences for different users of the same function.

All of this, then, is set to a more fundamental backdrop. First, WebSphere is a *machine*—it is an aggregation of functional parts that work together as a tool to help you get your job done more efficiently than if you had to do all the work yourself. Like all machines, the elegance and utility of WebSphere can be measured by how well we have blended the characteristics of *strength*, *precision* and *tolerance*—strength to hold up against the most demanding workloads; precision to execute accurately, securely and efficiently every time it is used; and tolerance to adapt to a wide variety of conditions and uses.

Second, the world is not static. WebSphere needs to support the extremes of usage scenarios, but just as important, customers must be able to transition between different usage scenarios with as little friction as possible—*minimizing the friction of movement*. You may start out with a small web site but, over time, as your business grows so will the demands of your applications—both in terms of the capacity of the underlying hosting environment as well as the sophistication of what you need to do. Other customers will start out with simple tightly integrated business functions, but later will grow to en-compass their entire business functionality across an enterprise service bus. WebSphere needs to be able to facilitate these transitions easily.

Third, the computing system will be touched by many different roles—end-users of the application, deployers, developers, monitors, administrators, service centers, etc. Each of these roles will develop their own, somewhat independent, perception of the utility of the middleware in achieving their goals and responsibilities. WebSphere must optimize the productivity experience of each of these roles.

Fourth, modern distributed computing systems are fundamentally and inherently complicated. Distributed systems combine a large number of moving parts with fragile and inefficient transitional boundaries. This is like putting horse-drawn buggies, freight trucks, and Formula-1 race cars all on the same highway and then managing the entire thing to ensure everything and everyone gets to where they need to go, does what needs to get done, and does so safely and reliably. WebSphere must be able to manage this complexity, and mask out as much of that complexity as possible.

The corollary to this is that the middleware is responsible for virtualizing the infrastructure through an abstraction model that lets businesses focus on their business domain expertise and adding value to their business. Without middleware, we often see application developers spending tremendous amounts of their time dealing with issues like connectivity to data systems, including recovering from connection failures; managing state caches, including handling invalidations; protecting access to information and processes, including administering access policies; and on and on—none of which has anything to do with taking orders, depositing checks, checking inventory, tracking shipments, managing risk, and so forth. The responsibility of WebSphere is to manage the distributed system on behalf of the application—freeing application developers to focus their attention on other, more valuable things.

Fifth, to expand the macro-economics of computing we must maintain a high degree of consistency in the fundamental aspects of information computing. As an analogy, the macro-economics of transportation depends on there being more similarity between trucks than differences. Trucks have the same basic limitations on height, length and width which enables them to fit on the same road structure; they use essentially the same driving configuration which enables a larger pool of skilled drivers; they use the same fuel formulae which enables a broader distribution channel for power; they use similar engine technologies which enables a large support structure; they conform to a uniform set of deck height assumptions which enables interchangeable loading and terminal implementations. That's not to say that there aren't differences between Mack trucks and Volvo trucks, but those differences are primarily around quality of service and scale differences, not basic functional differences. Said a different way, WebSphere must conform to a set of broad industry standards. Those standards enable the expansion of the marketplace for application serving middleware, and more importantly for enabling customers to derive value from their investment in applications designed to work on WebSphere.

The final backdrop to the principles of WebSphere architecture has to do with the extended nature of information computing. WebSphere is a hosting environment for J2EE-based applications. But WebSphere is also the foundation of a much broader platform for information computing—including portal-serving, business integration, service- and event-oriented architectures, service buses, client interaction and collaboration services. In this capacity, the application server defines the fundamental execution model for the entire platform—the integrity model, the protection model, the availability model, the serviceability model, etc. WebSphere is, in some sense, the network operating system for the information system.

This book is a microcosm of WebSphere itself. Ann, Mike Everett, Mike Casile, Keith, Alan, Kyle, Dave, Tom, Yu, Dipak, Michel, Mihaela, and Radhika all exemplify exactly the kind of dedication, expertise and diversity that you can find throughout the WebSphere development and execution team. They hail from many different locations within IBM, and, like the rest of the WebSphere team, have had to orchestrate and manage the development of this book across timezones and organizations.

The result, however, has been nothing short of outstanding. This book covers the full breadth of the operational experience with WebSphere—from a basic overview of the product through to configuration, deployment, administration, tuning and troubleshooting. Like WebSphere, this book represents a tremendous effort—especially so when you consider that all of the authors have real jobs to attend to through the day. I have absolutely no doubt that you will find this book to be hugely informative—a great aid to getting the very best from your WebSphere runtime.

I truly appreciate the effort these folks have made in producing this book. Just as importantly, I know I speak for the entire team when I convey how enormously grateful we are to the many, many customers, administrators and developers that have embraced WebSphere; that have discovered the value of WebSphere and have come to exploit WebSphere for the many benefits it has to offer. For that we thank you.

—Rob High, IBM Distinguished Engineer;
WebSphere foundation, Chief Architect; and
Member, IBM Academy of Technology