Advancing SOA With Real-Time Events

By Chris Garner

As industry pundits debate the proper terminology—some calling it 2.0, others next-generation or advanced—there's no denying that the latest evolution of Service-Oriented Architecture (SOA) is about the impact of real-time events on SOA and the implications this progression has for the real-time enterprise.

Understanding the advancement of SOA requires that we establish a baseline for how the term has come to be understood. SOA is an approach for building distributed computing systems based on encapsulating business functions as services that can be easily accessed in >

a loosely coupled fashion, enabling robust, flexible interaction between components. These services can be accessed via a network while the underlying platform implementation remains transparent to the consumer (whether a person or program), and the location of the source program or data—whether on the same machine or another—is irrelevant to that consumer.

While many elements of SOA were in place years ago, what really pushed those elements into widespread adoption as an architectural approach was the firming up of industry standards from the comparative anarchy that prevailed in the early days of object-oriented programming. With viable standards in place, enterprise IT administrators have quickly seized on the benefits afforded by SOA of rapid application development and agility in addressing changing needs.

The latest evolution of SOA entails the merging of SOA with other architectural types—especially Event-Driven Architecture (EDA)—to create symbiotic systems that can exceed the sum of their parts. EDA pertains to creating an architecture that fully leverages the ability to capture real-time state changes (events) as they occur, then dynamically respond or react in appropriate ways as defined by, say, business rules. Unlike SOA, where communications are solicited, EDA operates without the event-generating process having any awareness of the "consuming" processes—which may be innumerable and can receive and respond to the events in parallel. Where SOA processes are loosely coupled, EDA processes are so autonomous as to often be termed "un-coupled."

What advantages does EDA bring to SOA? When you combine the real-time intelligent aspects of event integration with interoperability and zero latency, this evolved form of SOA can enable businesses to realize what Gartner calls the "Real-Time Enterprise," defined as "an enterprise that competes by using up-to-date information to progressively remove delays to the management and execution of its critical business processes."

Consider a financial deposit at a bank. Traditionally, a batch system registers all such events and processes them together overnight. That's not real-time and has a high degree of latency. A complete facility for EDA, by contrast, provides the ability to capture each event as it happens and apply rules to it to perform certain actions, transform the composite data into some interoperable format, and provide a vehicle for publishing it. Capture-enrichment-transformation-publishing is the equation that defines the new modern tools for supporting an EDA.

It's important to distinguish between events that have an impact on an organization's business and those that don't. Event detection has commonly been employed in systems management software to use network events in normalizing system operations and maintaining system health. An example might be server load balancing, where real-time capture and response of systemic events prevents processing overload from occurring on any one server. The technical events we speak of in connection with EDA are more business-specific. Where the merging of advanced SOA and EDA have great promise is in using these business-meaningful events to realize aspects of the real-time enterprise.

So, what types of organizations stand most ready to take advantage of SOA complemented by event-driven integration? Applications regarded as SOA 2.0-ready include those in financial services and insurance enterprises. These organizations are already making heavy use of Web services and have been early adopters of SOA. A major reason is the vast amount of business-meaningful events (such as rate changes, investment grade changes, stock price changes, etc.) that their IT infrastructures must handle and react to daily, sometimes on a minute-to-minute basis. These industries are driving the need for real-time event capture and publish in an SOA.

Another major reason behind rapid adoption of SOA in the finance and insurance sectors is that these enterprises rely heavily on the mainframe and legacy applications that typically run on them. Legacy systems were initially designed without modern distributed computing in mind. Until recently, most mainframe integration methods also forced tight coupling.

Years of adapting to innovation and growth have increased the complexity of mainframe integration methods; this has led to latency in service performance and brittle implementations, often based on gateways. The mainframe world has been deliberate, if not slow, to support new applications. Legacy systems often involve sophisticated programming, requiring a technical expediency that's costly and increasingly scarce.

Even so, mainframes can't be ignored. In fact, the majority of *Fortune* 500 companies rely on mainframes, and more commercial transactions are processed on mainframes than on any other platform. If an enterprise is considering where best to incorporate events into an advanced SOA, it will likely find the preponderance of these occurring in the highly transactional mainframe environment. A properly implemented event-driven SOA system can, without touching application code, capture events in real-time from leading mainframe database systems and push them asynchronously via multiple messaging and communication protocols.

An advanced SOA should be enterprisewide. An SOA only achieves true ROI and universal integration throughout a large-scale enterprise by incorporating legacy applications. Enterprise

business integration journal takeaways

BUSINESS

- Enterprise IT administrators have quickly seized on the benefits afforded by SOA of rapid application development and agility in addressing changing needs.
- If an enterprise is considering where best to incorporate business-meaningful events into an advanced SOA, it will find most of them occurring on the mainframe.
- The evolved form of SOA can enable businesses to realize what Gartner calls the "Real-Time Enterprise."

TECHNOLOGY

- EDA operates without the event-generating process being aware of consuming processes, which may be innumerable and can receive and respond to events in parallel.
- A complete facility for EDA provides the ability to capture each event as it happens and apply rules to it for performing certain actions.
- An event-driven SOA system can, without touching application code, capture events in real-time from various databases and push them asynchronously via communication protocols.

developers would be best served leveraging a single, unified mainframe integration tool to isolate specific transactions within multiple legacy systems and expose them as reusable components and Web services.

To become a real-time enterprise, follow these SOA 2.0 best practices:

- **Implement enterprisewide:** Any advanced SOA should extend throughout the enterprise without being limited to any particular platform or technology. This must include legacy mainframes.
- **Implement a single, unified, bus-like platform:** An enterprisewide SOA implementation entails a distinct departure from point integration systems, involving a comprehensive Web services-centric platform that acts as a foundation for a distributed SOA approach to messaging and integration in the form of an Enterprise Service Bus (ESB). The heterogeneous mainframe-reliant enterprise also must incorporate into this model all mainframe connectivity—a mainframe services bus approach—where data, applications, or business logic are seamlessly available via industry standard technologies.
- **Seek ease of development:** An opportunity to harness the power of publishing a given event as a Web service is self-defeating if the custom development required is too time-consuming. So, the real-time enterprise should have a comprehensive IDE with an intuitive graphical interface. It should be able to leverage modern application development frameworks for .NET and Java as well as provide seamless integration to expose mainframe applications and data as Web services and consume Web services from distributed platforms, real-time business events, or as direct SQL calls. The ideal is to have a single development environment to handle any integration requirement without requiring detailed knowledge of the mainframe or distributed technologies.
- **Balance security and optimum performance:** The reach and flexibility SOA provides can potentially compromise security; this is particularly true on the mainframe platform. Mainframes just weren't designed with the idea that a consumer might sign on via a button on a Website which would activate a Web service that might hit that mainframe a thousand times a day. A means of managing and reducing the processing overhead for authenticating loosely coupled connections should be implemented. For any process requiring authentication (e.g., a Web service or SQL call), this mechanism should work in conjunction with the established client and host security protocols to optimize user authentication for sign-on processing.
- **Maintain transactional integrity:** Real-time events can provide a critical role in maintaining the consistency of data and process orchestration. Given the huge volumes of transactions in the mainframe portion of an advanced SOA, it's essential that proper support is provided for Two-Phase Commit (2PC) and emerging requirements for Web Services Business Process Execution Language (WSBPEL) and WS-Atomic Transaction.
- **Address maintenance, performance, and scalability:** Simply having integration without the ability to control it would result in a system lacking enterprise quality of service. Implement capabilities such as real-time diagnostics to ensure resource availability. Performance and scalability, while intangibles, must be addressed.

For an example of advanced SOA and the applicability of real-time enterprise approaches to business processes, consider the trading sector of the financial industry, where so many variables can affect markets and where EDA has begun to be adopted to automate immediate reactions to changes in those variables. Where the evolution of SOA can best accelerate productivity and effectiveness, though, is in scenarios where real-time processes have been lacking.

For example, consider the insurance and finance division of a large company in any unrelated industry sector. Say this division is running a homegrown legacy benefits administration system on a mainframe that handles health insurance and claims as well as 401(k) plans for tens of thousands of employees. The company's Website has a password-authorized, log-in area that lets employees register, complete, and submit forms, etc. Behind the scenes, however, division staff members are re-entering all this information into "green screens" accessing the legacy system; they're also identifying materials requiring further human interaction—such as underwriting verification, claims adjustment, and approvals—and routing them to the appropriate parties. As the available "Web services" increase, and as employees make greater use of them, more staff time is required for sorting and data entry.

This organization is a prime candidate for implementing a real-time solution using an advanced SOA to service-enable their mainframe assets in a real-time fashion. The mainframe interoperability would be best served by middleware in the form of a single platform for handling data access (SQL), screen transformation into Web services, and mainframe business event management. The solution would provide a real-time bridge between the mainframe system of record and the distributed servers, running on Windows or Linux. Data existing on forms that require no further interaction could be written directly to the legacy application database via the middleware, while forms requiring review could be pushed asynchronously to the appropriate parties using a readily available shared portal software product. Inquiries made via the Web portal can seamlessly interact with mainframe green screen transactions that have been transformed into Web services.

All application development needed to support this advanced SOA can be done via industry standard Microsoft .NET or Java tools, requiring no specialized mainframe programming skills. Without having to tamper with its proven, strategic mainframe software infrastructure, the organization will now be able to rapidly transform its legacy system into a more flexible, Web services-based application that better addresses the current and future needs of its employees and that slashes the staff time spent processing employee data.

This example shows the potential benefits of an advanced SOA when implemented with real-time enterprise best practices. The applications are virtually endless and the room for creative thinking is vast. With the addition of real-time events, SOA carries great promise for companies eager to seize the opportunities and competitive edge it affords them and points the way to the future of the real-time enterprise. **bij**

About the Author

Chris Garner is vice president of research and development for DataDirect Technologies' Shadow mainframe integration products. He has been on the front line of mainframe and data integration technologies for more than 30 years.
Email: chris.garner@datadirect.com
Website: www.datadirect.com