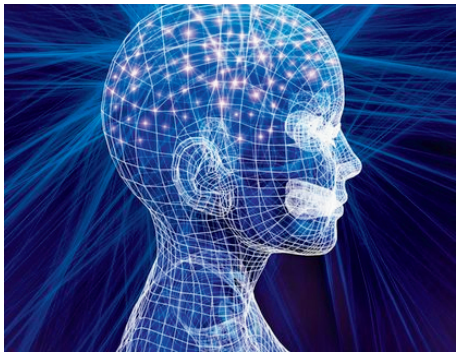
AS SEEN IN: SOA Web Services Journal

Fiorano SOA 2006 Platform

A Honey of A Product

REVIEWED BY **WARREN HAMPTON**



SOA, EDA, BCM, ESB and BPEL...More than IT Catch Phrases?

I recently had the chance to evaluate the next-generation Fiorano SOA Platform 2006 suite from Fiorano Software, Inc. As an architect and developer who's worked with previous versions of the kit over the last three years in addition to several competitor offerings, I looked forward to sitting down with Fiorano's latest release.

For those unfamiliar with the product, it's a feature-rich SOA/BPM development, deployment, and administration suite built on the company's J2EE ESB technology. It relies on standards-based services, shared component modeling, peer-to-peer communications, and high-performance event-driven messaging (pub/sub and queuing) across a distributed network. This latest release brings a standalone BPEL Editor, Composite Components, pre-built JCA-compliant adapters, a Business Development Component Kit and Shared Resource Pools among other enhancements (see Figure 1).

With intuitive interfaces, pre-built components, and data adapters, business analysts can build business process workflow applications with little or no programming. These components aren't graphical representations for diagram purposes; they are fully functioning feature-rich services that can be added to the application with drag-and-drop ease and are configured using simple parameters.

This is not to say that developers can't build complex coarse-grained custom services to leverage the full power of the suite. The extensive toolset has allowed our developers to concentrate on extending and refining richer applications to meet business requirements in a fraction of the time of competitors' offerings while encouraging code re-use, SOA design, and solidifying patterns and practices. Another key point is the platform's low overhead, full scalability, unassisted fail-over, and flexibility to run on a single server or across a highly distributed WAN environment or load-balancing cluster with ease.

Getting Started

First note the decision to rename the suite from Fiorano ESB to Fiorano SOA. This makes perfect sense. I've always maintained that previous versions were more than a toolset for building ESB-based solutions. Although Fiorano originated in high-performance message queuing, the platform has matured into a powerful, scalable service-oriented development platform on top of the proven messaging abilities. Earlier versions allowed for rapidly building robust workflows and quick integrations across disparate systems and relied on intuitive administration interfaces for views into daily development and production processes. I was eager to see what the latest iteration would add.

The installation is simple, a few point-and-clicks is all you need to get the full Enterprise version installed and ready to roll. Although the suite is intuitive I'd recommend browsing the Startup and the Platform Concepts guides before beginning the install to familiarize yourself with the primary components and the depth of the product. You can find extensive documentation and code samples at http://devzone.fiorano.com/devzone/dev_zone.jsp.

An improvement over previous versions is the built-in ability to launch both the servers and individual processes as Windows Services. This allows for auto-restarting services and applications as well as monitoring the services using commonly available network monitoring tools. I would, however, have liked to see this as part of the initial installation. The rules-based security is easily understood, set up, and maintained but also robust and flexible enough to pass stringent security standards. Integration with LDAP services is also an option.

Once installed you'll find the following:

- **Fiorano ESB Server 2006:** A web Services-capable middleware platform
- **FioranoMQ Server 2006:** Peer-to-peer JMS messaging platform
- **Fiorano BPEL Server 2006:** A distributed BPEL processing orchestration engine.
- **Fiorano Business Components and Adapters 2006:** Ready-to-use JCA-compliant components
- **Fiorano Process Orchestration Tools 2006:** Integrated development and admin toolset
- **Fiorano BPEL Editor 2006:** A standalone BPEL design, development, test, and deployment tool set

SOA, EDA, and ESB in the Real World

Although real-world sample applications are included I wanted to see how I could improve existing workflows created in previous versions as well as solutions built with other tools to see if there were some viable gains in this "major" release to further simplify the development process and make it easier to build, track, debug, extend, and monitor these solutions. I choose more complex workflows that get XML messages from external sources via HTTPS posts or

Web Services, rely on content-based XPATH queries for routing and XSLT transforms from external to internal formats, create physical archive files, update an SQL DB, and again transform messages to one of many positional or delimited file formats for delivery to legacy systems. Also included along the way are extensive error-handling alerts and simple business rules.

What I found was very encouraging. As previously mentioned the suite includes an Event Process Orchestrator to build flows. I was glad to see that this had retained its look-and-feel while the pre-built services have improved in many key areas in regards to continuity and configurable parameters. Also apparent was an improvement in speed when opening the configuration dialog screens. I was quickly able to build new loosely coupled versions of the existing applications (including non-Fiorano solutions) leveraging many of the enhancements such as improved services, data adapters, composite components, and shared resource pools to create highly efficient applications.

Several new or enhanced pre-built services and data adapters are offered, some of which include BeanShell script components, improved DB, HTTP, Web Service, and transform components as well as support for adding iWay JCA-compliant services to the flows (www.iwaysoftware.com). These, added on top of the existing extensive palate of services, protocols, and adapters, (60+) went a long way to letting me build richer enterprise-level solutions with little or no custom programming. Not that you're limited in this area; improvements have been made making it easier than ever to build custom components that can be added to the existing palette using Java, C, C++, and C# as required.

One of the main improvements is the enhanced ability to build composite components that are repeated many times in day-to-day workflows and integrations. Easily and effectively I re-created large pieces of a complex workflow that can be called on from multiple applications. This is a great time-saver and helps you adhere to code re-use principles and enforce standard handling of repeated processes. Another welcomed enhancement is the improvements to what was arguably the best mapping tool available. FioranoSOA 2006 now retains existing mappings when adding to the underlying schema. This is a real time-saver when you consider the implications of re-mapping an extensive transformation from scratch to add an additional tag to a complex schema.

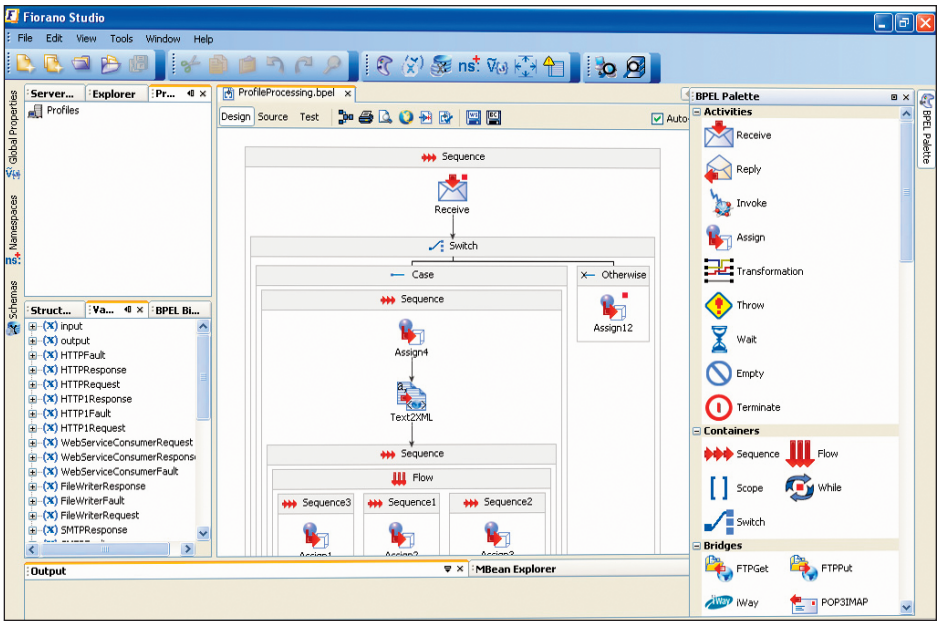


Figure 1

Business Process Meets Application Workflow...

One of the most striking things I always found with the platform was its ability to bridge the divide between business process and application workflow modeling. This is even truer today using a combination of the BPEL Studio, the Event Process Orchestrator, and an extensive palette of standards-based, pre-built functional services; everyone from business analysts to developers and administrators can conceptualize workflows, build applications, test, and monitor the full development process from a common interface without extensive knowledge of the underlying services.

The ease with which you can create and deploy working solutions must be seen to be appreciated. Many times I have built a working solution before someone's eyes and deployed it only to be asked, "Okay, so now what are the steps required to build, code, and deploy the application?". By immediately running the application and showing the messages flowing is it clear a fully functioning flow can be created almost as quickly as you visualize it. Compared to competitor offerings FioranoSOA excels in this area.

Production applications can even be extended, debugged, or corrected without downtime to the application, something unique to the platform and eye-opening to those accustomed to late-night builds, pushes, and testing. Application profiles help control versioning between development phases and push-to-production environments. The platform challenges the traditional development process methodology, saving significant time and money when a production issue has to be corrected. But it still fits perfectly into the standard process and eases many of these tasks compared to alternative methods.

The speed with which you can do change requests to a business process is where the application and toolsets really shine. I have faced major production issues after a deployment in which the ESB platform had no direct affect, acting largely as a transport/transformation layer. However, due to the power and flexibility of the platform I was able to correct the issue by applying business rules that altered the messages in the production application in less than one hour, whereas the legacy system sending the data in question would have taken four to six hours of development, testing, and a late-night build to resolve the issue by the next business day. Starting and stopping services at runtime, components like the Feeder Service that can send messages to the orchestration or the display service that shows you the output from any service port help to speed debugging and re-queuing from any point in the process. The ability to capture messages via event interceptors on the routes between services empowers you to queue messages, edit services, and analyze log files without disrupting other services in the flow.

The cost savings in these scenarios is hard to quantify since it's very far-reaching in a high-volume transaction-based business that directly affects revenues based on these messages; needlesstosay it's significant. The savings starts at the development and testing stage where the cost savings created by the speed with which adjustments can be made carries all the way through ongoing maintenance of the platform and solutions in production. At each stage significant gains are realized when compared to competitive offerings or traditional coding methods.

This flexibility also allows for simple scaling of the platform. You can easily add more peer servers, fail-over solutions, load-balancing flows, and error handling without significant impact on your applications. You can go from test to beta to QA to production in a very short period of time and deploy these updates with confidence and minimal risk. The support included for resource sharing across multiple instances of components in composite applications reduces the memory load of intensive services such as DB connections across multiple components further extending scalability.

An often-overlooked strength is the power of centralized modeling, development, deployment, and administration. Through rules-based security, roles can be restricted as required allowing all disciplines to view, alter, or administer the applications, servers, real-time messages, services, and related components from a common set of tools as required. Being able to have the BA, developer, QA analyst, and administrator all look at the same visual interface with no risk to the application and work through testing, deployment, monitoring, and debugging is invaluable. The ROI continues well past the development stage of your applications. The flexibility of the platform also excels at extending the lifecycle of legacy applications by adapting to the older platforms, then allowing you to extend the legacy applications' abilities by integrating newer protocols, standards, and abilities such as HTTPS, XML Messaging, and Web Services.

Conclusion

With this release Fiorano has firmly established a mature development platform further increasing the ability of a business analyst to create application flows and transformations and freeing highly skilled developers to build richer solutions, custom services, and loosely coupled composite services. With minimal training the ROI can begin immediately and principles like RAD (Rapid Application Development), SOA, and code re-use become reality. The cost savings begins with the first stage of Business Process Modeling and continues all the way through day-to-day administration, extending the lifecycle of existing platforms where applicable.

Also noted are the improvements in performance and throughput in several key areas, enhanced simple and distributed transaction handling, improved ANT script support, and AXIS and Eclipse integration. On the horizon a native C# version of platform and enhanced Web Service support will further extend the scope of the platform. Consuming Web Services is a breeze but there's still work to do regarding incoming Web Services. Overall I was very impressed with the new release and have already realized immediate gains in time-to-production, quicker ROI, and richer applications.

About the Reviewer

Warren Hampton is a senior e-commerce architect with Quicken Loans/Title Source Inc.

warren.hampton@authors.sys-con.com

Fiorano Software, Inc.
718 University Avenue, Suite 212
Los Gatos, CA 95032
Web: www.fiorano.com
Phone: 800 663-3621