CMP
United Business Media
Network Computing
FEBRUARY 21, 2006 I WWW.NWC.COM
For IT By IT

Fiorano SOA Platform 2006

Fiorano SOA Platform 2006 confounded us a bit with its JMX terminology, and its pricing may leave some enterprises with sticker shock. But we liked its security focus, and the Fiorano Mapper is one of the best visual data mapping tools we've seen. BY LORI MACVITTIE

Fiorano SOA Platform 2006 is a J2EE ESB with several moving parts. All administration is accomplished over Java fat clients, and there are several to choose from depending on your role in the organization. Fiorano employs a peer-to-peer hybrid architecture, requiring the definition of peer servers as the nodes on which ESB processes run. This means all service orchestrations must be deployed on specific nodes, and you can specify backup and failover nodes during deployment.

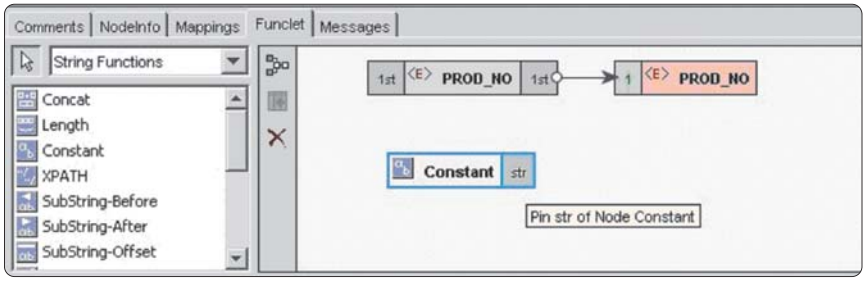
Fiorano manages its components using JMX (Java Management Extensions), and the facilities within Fiorano Studio make this painfully obvious. All BPEL processes are orchestrated in Fiorano Studio and in future releases the modeling tool--currently a Java fat client--will be moved to this tool. All JMX management is accomplished through this tool as well, though it's a bit confusing to move around in and lacks flexibility. In addition, the JMX terminology is overwhelming and makes administrating the environment tedious. Fiorano said it is in the process of "English-izing" its JMX interface and removing pieces that aren't necessary for 99.99 percent of all configurations. We were pleased to hear this, as well as the decision to move functionality from other Fiorano tools into Fiorano Studio.

Peer servers are defined using profiles, a concept that should be familiar to the JBoss/WebSphere administrator and is similar in nature to BEA's and TIBCO's "domain" management paradigm. Peer servers are configured to retrieve their configurations from the ESB server, and communication between the peer server and ESB server is accomplished over JMS. The differentiator Fiorano touts is that communications among peer servers uses a proprietary protocol over JMS, as opposed to the pure Web services model used by most products we tested.

Fiorano includes a heavily security- and rules-oriented model to define which components can be run on any given peer server. We found the Services and Security Manager easy to understand, and we defined rules limiting the components that could be run on one of the two peer servers we configured without any glitches. Fiorano's Event Process Orchestrator includes a resource-validation option, which flags as errors orchestrations whose component deployment configurations violate those rules. We configured a rule on a peer server to disallow execution of our database service, then tried to deploy that component to the same peer server. The result was an error during validation--including a lengthy Java stack trace--rather than a user-friendly message saying the deployment configuration violated the configured execution rules for the peer server. We'd prefer the latter, and Fiorano agrees; it says it will address this issue in a future release.

We modeled our service orchestration quickly in Fiorano's Event Process Orchestrator. EPO offers a wide variety of protocols and data formats out of the box, and aside from the expected requirement to configure OpenJMS connectivity and a JDBC connection to NWC Inc.'s Oracle9i database, we had no problem modeling our scenario. Fiorano supports BPEL, but its primary service orchestration uses a proprietary modeling notation. We orchestrated a service using BPEL, then exported it to the ESB server as a business service; this let us choose our service from the palette within EPO for inclusion in our service orchestration. The option to export the BPEL orchestration as an Axis Web service is also available.

We preferred Oracle's BPEL editor over



Fiorano's visual mapping tool makes transformations painless.

Fiorano's. Fiorano Studio was quite particular about connections, and we were unable to move connections once they were laid down. This meant we had to delete and re-create the connection, resulting in the loss of associated mappings between activities. We preferred Oracle's more flexible BPEL editor, which let us move and insert activities into an existing flow. Fiorano's integration of BPEL was along the lines of that offered by Software AG; smaller orchestrations can be modeled in both products using BPEL and then exposed as Web services for inclusion in the products' native orchestration environments.

Fiorano's mapping tool, Fiorano Mapper, was one of the best visual data mapping tools we used in this review. We visually mapped fields using a drag-and-drop paradigm that let us easily incorporate XSLT functions.

In addition, Fiorano lets you test orchestrations before deployment by incorporating Feeder and Display components into the orchestration. We dropped a Feeder component onto the palette and used it to inject a message with the appropriate protocol for the orchestration. We used a Display component to exhibit the resulting output document. This setup was not as elegant as those offered by BEA, Oracle and Cape Clear, but unlike BEA and Oracle, Fiorano--and Sonic--gave us the means to inject messages into a JMS queue, a common endpoint protocol on the ESB. We'd like to see both options in a single product, but we aren't holding our breath.

Fiorano's Web services support, like that of the other products in our review evolving from an EAI-focused world, is provided by Tomcat/Axis and is not yet well-integrated into the product. The model we created had to be built as a separate Web service process and then exported to Axis to support a SOAP-over-HTTP entry point into our orchestration. We were pleased with Fiorano's consumer-side support of Web services, though incorporating an external Web service into our orchestration revealed a few problems, such as the lack of user feedback when importing external WSDL files. Fiorano indicates only errors, not successes, and subscribes to the "no news is good news" camp when dealing with WSDL files. Despite the lack of feedback, we easily incorporated the external service required by our scenario.

Deployment of orchestrated services was just a button click away, and we liked Fiorano's ability to synchronize changes in an orchestrated service with processes already running on the server. Fiorano's capabilities in terms of managing services running on our server was more limited than those provided by Oracle, but we did view running processes as well as stop and start individual components at run time. We started the components required by our orchestration, then stopped one of them and launched the associated service. We could verify that the message had persisted in Fiorano's file-based repository because the next component in the orchestration wasn't available. We could configure the repository to use an external RDBMS instead of the default file-based system; if an external RDBMS is used, messages are persisted in the RDBMS instead of the file. After verifying that the message was being properly persisted we restarted the component in question, and the message was correctly routed.

Fiorano's prices its product on a per-CPU basis for the server ($40,000 per CPU) and a subscription-based developer model ($5,000 annually per developer). Needless to say, we are not pleased with the per-developer, per-year part of the equation. The company also charges on a per-CPU basis for adapters, so we added in $10,000 per CPU for our database adapter, and $4,000 per CPU for each SMTP and JMS. We also had to add in $995 per administrator for the use of Fiorano Tools (BPEL Editor, ESB and admin tools). Ouch.

■ FIORANO SOA 2006 PLATFORM, $170,995 as tested. Fiorano Software, (800) 663-3621, (408) 354-3210. fiorano.com/frontpage.htm