

Network Computing

For IT By IT

FEBRUARY 21, 2006 | WWW.NWC.COM

Fiorano SOA Platform 2006 3.7

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

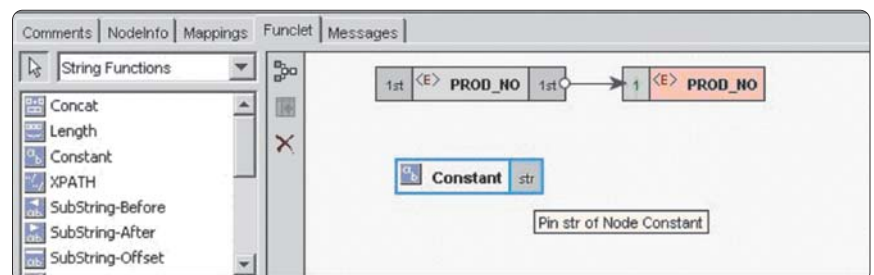
servers are configured to retrieve their configurations from the ESB server, and communication between the peer server and ESB server is accomplished over JMS. The differentiator Fiorano touts is that communications among peer servers uses a proprietary protocol over JMS, as opposed to the pure Web services 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.

