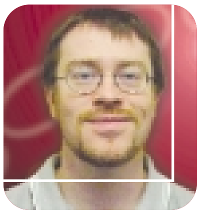
case STUDY

Rich-Client Deployment in a ZAC-less World

LEADING DENTAL ADMINISTRATOR MOVES BEYOND ZAC WITH SITRAKA DEPLOYDIRECTOR

Organizations deploying rich client/server WebLogic applications need to fill the void created by the deprecation of BEA Zero Administration Client (ZAC). The affiliated Delta Dental Plans of Michigan, Ohio and Indiana, a leading dental plan administrator, chose Sitraka DeployDirector to deploy the client-side portion of their claims-processing application to thier users.



BY
DAVE TRUMAN

AUTHOR BIOS...

Dave Truman is a technology writer at Sitraka. He has more than 10 years of experience working with and writing about application development tools, and has worked with Java since JDK 1.0.

CONTACT...

dave.truman@sitraka.com

Some days it seems like you can't win. You've developed a WebLogic application that has a Swing-based "rich client" front end and were planning to deploy it using BEA's ZAC (Zero Administration Client) deployment utility. Then you learn that ZAC has been deprecated. What do you do? One company solved this problem with Sitraka DeployDirector, a third-party Java provisioning and management solution.

The affiliated Delta Dental Plans of Michigan, Ohio and Indiana (Delta Dental) run the largest dental benefits system in the Midwestern United States, providing dental health benefits to large and small employers. It is part of the Delta Dental Plans Association, a nationwide group of 37 independent affiliated dental service plan corporations. The tri-state operation provides coverage to more than 6 million people enrolled in over 4,200 groups.

Processing dental claims for an organization this large and geographically dispersed is a big job. The varying levels of network bandwidth and infrastructure between affiliated offices and Delta Dental's many claims processing agents was one challenge. The most unexpected obstacle, however, turned out to be around the *deployment* of the rich-client component of the application.

A WebLogic Application with a Twist – a Rich-Client Front End

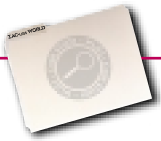
Delta Dental designed a custom claims processing system to work well in this environment. A cluster of BEA WebLogic application servers running on Unix formed the core of the system, handling transaction processing and business logic. Not exactly an atypical J2EE application architecture – except for the front end.

While most WebLogic application developers initially opt for a browser-based presentation layer (or front end), Delta Dental determined that they needed a rich-client front end. Claims agents enter a lot of data and needed the speed and productivity a rich client would provide. A browser-based thin-client presentation layer can have usability and performance limitations compared to a Swing-based user interface. In this case, the team determined that it would require too many round-trips to the server to repaint screens as users changed fields in the complex user interface.

A hybrid Java Swing GUI application on the client that connects to WebLogic to process business logic and transactions is a bit of a twist on the standard J2EE application stack, but one that worked well to address the application's performance and usability needs.

The Deployment Challenge

Of course, one big reason most WebLogic applications employ browser-based user interfaces is the hassle of deploying Java-based rich clients. Rolling out a Java-based application



across a large number of end-user machines in an organization is particularly challenging because a Java Runtime Environment (JRE) must also be present on each machine, and must be correctly configured. Using Java applets does not solve this problem entirely, and applets have major drawbacks of their own.

WebLogic applications that employ a browser-based presentation layer rely on the server to render pages to the user. Deployment is easy in this case, but applications with complex user interfaces require a high number of round, trips to the server, which impacts performance and usability.

Delta Dental had planned to use a feature in WebLogic called ZAC. But while ZAC was provided with earlier releases of WebLogic, BEA deprecated the feature in 6.0, and does not provide documentation for it in 7.0.

Therefore, deploying and managing the rich-client portion of the application to the many remote users became more of a challenge than originally anticipated. While the team could have chosen to use ZAC anyway, they felt it was not a good long-term decision, especially after learning that ZAC requires extra development work – work that would not migrate to a new deployment system down the road. Now was definitely the time to find a new solution.

Choosing a New Deployment Solution

Delta Dental turned to Amit Singh, a systems engineer with BEA's Professional Services Organization, for help. Mr. Singh had firsthand knowledge of the real challenges of deploying large Java client applications to a wide user base. Having used ZAC for past projects, Mr. Singh knew the effort required for that solution.

BEA's documentation for WebLogic 7.0 recommends migrating from ZAC to Sun's Java Web Start deployment system. However, Mr. Singh knew this solution wouldn't work for Delta Dental. While Java Web Start provides basic remote deployment and dynamic updating capabilities, it is not suitable for enterprise-wide application deployment and management. Specifically, Delta Dental required:

- A deployment solution designed for clustered application servers communicating with many clients
- Automated rollout and rollback of application updates
- Remote monitoring and administration capabilities to help IT staff troubleshoot problems with users

Delta Dental needed a deployment solution that would provide everything ZAC did and could meet the requirements. The team did some research and discovered Sitraka DeployDirector.

Sitraka DeployDirector is a Java application provisioning and management solution that helps IT organizations deploy, manage, and update rich clients. DeployDirector makes it easy to deploy, manage, and update Java rich clients across multiple client platforms from the WebLogic application server.

It provides WebLogic developers with control over application provisioning, updates, rollbacks, and version usage. In addition, DeployDirector provides powerful logging and reporting features, access control, JRE management, and immediate wireless and e-mail error alerts.

DeployDirector is essentially an automated deployment platform. It's server-side component stores the rich client application bundles and JREs, and manages the versions of the rich-client application. It also has a client-side component that wraps around the rich-client application, checking whether it needs to be updated, that it has the correct JRE, and so on. The team chose to evaluate DeployDirector to verify that it would meet their requirements and would integrate seamlessly into their WebLogic environment.

DeployDirector and WebLogic – Making It Work

The back end of Delta Dental's claims-processing system runs on a WebLogic cluster comprising two WebLogic servers and a load balancer. The key task was to integrate DeployDirector into the WebLogic servers because once integrated, a developer can access DeployDirector's tools to set up deployment of their own rich-client application.

The DeployDirector installer by default sets up and configures itself to use its bundled copy of the Apache Tomcat server. However, the distribution also provides a WAR file to integrate its server-side components with an enterprise-class application server such as WebLogic.

Mr. Singh and the Delta Dental team added the WAR file to each WebLogic server and configured the servers to run the DeployDirector server-side components. The team could also have chosen to run deployment centrally. Once configured, the team tested that DeployDirector was running in WebLogic. They then deployed the administration tools to a development workstation.

Setting up deployment of the client-side portion of the application was a point-and-click affair. The DeployDirector administration tool is a GUI-based application that creates and manages application deployment bundles. The tool enabled Delta Dental to create a deployment package for the client-side component of their application and specify which JRE it should run with, when to check for updates, and so on.

Delta Dental found that DeployDirector was not only able to replace ZAC, it was a better and more robust deployment solution than ZAC. "Everyone was very happy with how easily you can deploy and update the application on the fly with DeployDirector," says Mr. Singh. "Real-world use of ZAC requires a lot of custom coding, but DeployDirector didn't – Delta Dental was up and deploying in less than a week."

No ZAC? No Worries

After performing test deployments of the client-side application to pilot groups of users, Delta Dental was satisfied. Says Mr. Singh, "Delta Dental was very happy that the architecture they selected could be enabled with DeployDirector." Delta Dental subsequently realized that the DeployDirector management platform provides them with far greater control and flexibility than would have been possible with ZAC. Some key benefits include:

- Designed to deploy large applications over clustered environments to many users in a variety of locations.
- Applications can be set up for deployment easily within a GUI environment, without requiring application code changes or extensive scripting.
- Rapid updating and rollback capabilities, using class-level differencing to minimize bandwidth requirements.
- Easily monitors and reports on who is using what version of applications, including wireless and e-mail alerts when a rich client has runtime problems.

For More Information

For more information on the affiliated Delta Dental Plans of Michigan, Ohio and Indiana, please see www.deltadentalmi.com. For more information on DeployDirector see www.sitraka.com/deploydirector.