

DATABASE

TRENDS AND APPLICATIONS

Solutions for the Information Project Team • www.dbta.com

Volume 20, Number 2 • February 2006

APPLICATIONS INSIGHT



GUY HARRISON

Is Java at the Crossroads?

It's been a long time since I wrote programs for a living, however, I recently had the opportunity to review the database access APIs in Java and a variety of other languages (.NET, PHP, Perl and Python) while researching an upcoming O'Reilly book on MySQL stored procedure programming.

For my research, I essentially created a mini-application, which was driven by MySQL 5.0 stored procedures in each of the languages. The experience was extremely enlightening--especially in regards to Java. Although the Java JDBC interface has undergone virtually no changes in the past seven years, very few Java programmers use JDBC -- or even SQL statements -- directly. Database access is mediated through Object-Relational Mapping (ORM) schemes such as J2EE Enterprise Java Beans and Hibernate, or through alternative frameworks such as Spring. These frameworks allow for object-oriented access to relational data, eliminate the need to construct SQL statements and avoid much of the tedium that is required when coding directly in JDBC.

However, these frameworks are

imposing a significant barrier to the casual Java user. Although they may assist in the development of complex applications, they can make the development of simple applications more difficult. For instance, the construction of my first EJB almost reduced me to tears. Even though my experiences in the "alternative" Java frameworks of Hibernate and Spring were less traumatic, they still involved substantially more work than PHP or Python.

Java may be becoming less productive for simple applications and utilities. I'm unconvinced that abstracting relational database access and avoiding SQL is altogether a good thing. The popularity of the relational database is largely due to the ease with which business data populated by production systems could be retrieved for business intelligence purposes. I'm suspicious of any paradigm that makes the data more difficult to retrieve for the non-programmer.

While contemplating this, I picked up *Beyond Java* by Bruce Tate. A well known Java guru, Tate is co-author of *Better, Faster, Lighter Java* (O'Reilly, 2004). He

argues that in many ways Java has become a constraint. In addition to the problems introduced by abstracting relational database access, Tate argues that Java's strong typing, incomplete object orientation (primitives in particular), lack of strong GUI capabilities and over-reliance on XML configuration are causing fundamental productivity problems, especially for applications that do little more than provide a GUI management layer on top of a relational database. Perhaps the scene has been set for the emergence of an alternative development environment for "smaller" tasks. Tate nominated Ruby with the Rails development framework as the most promising, saying that Ruby probably provides a 400 percent productivity improvement over Java for certain tasks. It is definitely worth watching for in 2006.

Guy Harrison is chief architect for database tools at Quest Software. Harrison has specialized in application and database administration, development, performance tuning and project management. He is the author of "Oracle SQL High Performance Tuning" (Prentice Hall).