CocoBase Enterprise

O/R V3.1 Service Release 9.0

by THOUGHT Inc.

REVIEWED BY JIM MILBERY

AUTHOR BIO

Jim Milbery is a software consultant with Kuromaku Partners LLC (www.kuromaku.com), and is based in Easton, Pennsylvania. He has over 17 years of experience in application development and relational databases. Jim is the applications editor for Wireless Business & Technology, the product reviews editor for Java Developer's Journal, and a coauthor of Making the Technical Sale (Muska & Lipman).

jmilbery@kuromaku.com

THOUGHT Inc.
657 Mission Street.
San Francisco, CA 94105
Web: www.thoughtinc.com
E-mail: info thoughtinc.com
Phone: 415 836-9199

Test Environment: sony vaio pentium 2 366mhz 256 mb ram

Last month in ***JDJ*** (Vol. 6, issue 4) I introduced the topic of object/relational mapping. Databases such as Oracle8*i* or DB2 store data in tables and columns. Thus, customer data is stored in a "customer" table and information relevant to the customer such as ID, name, and address are stored as columns. All the data for a single customer within the customer table is equivalent to a "record." From the EJB perspective customer data is represented by a customer "class" and the data elements are represented by "attributes." Conceptually, the mapping process is a simple one. Each database table is an EJB class (CMP or BMP), and each and every column in the table becomes an attribute. Individual customer records are instantiated as EJB objects as necessary.

This month I tested the latest release of THOUGHT Inc.'s CocoBase Enterprise O/R mapping tool on my Windows 2000 server. Java-centric software companies such as THOUGHT Inc. take full advantage of both Java and the Internet when it comes to software development. They continually enhance their products with new features and bug fixes.

Last month I previewed "Service Release 8," and this month I was able to upgrade to "Service Release 9" – which included more than a dozen feature enhancements. The installation of CocoBase is packaged as a Java class file; it's a simple process to extract the installation files and install the software. CocoBase doesn't create desktop icons or menu items – all the software must be accessed via a series of command scripts. Nevertheless, it's a simple process to create your own desktop icons that point to the most common functions. The starting point for working with CocoBase is the Enterprise Administration Interface. THOUGHT Inc.'s choice of terminology here is a little off the mark as the admin interface is really the heart of CocoBase. All the major functions of the software are accessible from within this one interface – which is itself a Swing-based GUI program (see Figure 1).

CocoBase can connect to any of your enterprise data through a powerful JDBC interface. The software also comes equipped with a built-in SimpleText database, and you can use this data source to get accustomed to the many tools within the CocoAdmin interface. I elected to work with the SimpleText database and to create a new database map (as shown in the upper window in Figure 1).

You're free to map multiple CocoBase objects to a single database table, and I quickly created a variation on the Employee table that would list only those employees that make over $20,000 annually. I called this new object EmployeeGT20000 and generated a new data map for this object from within CocoBase within a few minutes. I chose to build this object based on a CocoBase-supplied "where clause" against the salary field using a variable for the salary. Technically speaking, this object could be used to represent any group of employees by supplying different values for this parameter. This new object extends the Object class and implements a number of interfaces including Cloneable, CBProp (a CocoBase class), and java.io.Serializable.

Once I had defined the EmployeeGT20000 map, I used CocoAdmin to generate Java source code into a package that I defined as "jmpkg". I could easily have added custom attributes and derived fields as part of the generation process. In either case, the resulting code is straightforward Java code as shown in the right-hand panel in Figure 1. The generated code includes get/set methods for all the attributes, a "where-clause" facility, and a facility for managing query information. Feel free to extend and modify this code. CocoBase can be used to manage data using both the BMP and CMP entity bean types. THOUGHT Inc. provides interface code for a wealth of third-party application servers including Allaire, Borland, IBM WebSphere, HP/Bluestone Sapphire, iPlanet, BEA WebLogic, and Sybase. (The development environment integrates with a number of popular Java IDEs such as VisualAge for Java, Borland's JBuilder 4, and iPlanet's Forte for Java.) This review covers only a fraction of CocoBase's capabilities and I'd encourage you to review the PDF documentation for additional details.

Summary

CocoBase's O/R tool can greatly simplify the task of interfacing Java code with relational data. In contrast with complete "frameworks," CocoBase improves productivity without sacrificing low-level control. You're free to modify the Java source as necessary to address your specific application needs. I'd encourage you to put CocoBase on your short list of products to consider when building a data-centric J2EE application.. ●

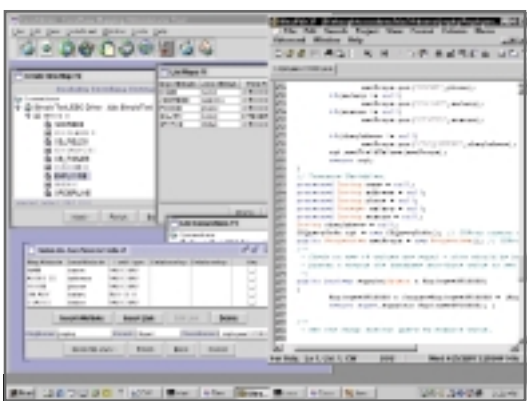

FIGURE 1 CocoAdmin with its many functions on display