Selecting the Application Interface

A Raima Inc. Technical Brief

Published: August, 2008
Author: Paul Johnson
Director of Marketing

Abstract

Control of the Raima database embedded is made available through three different API types (not counting JDBC and ODBC): Native, SQL, and JAVA. The API selected is a matter of the application design goals including run-time and maintenance.

This article is relative to the following versions of RDM:

- ✓ RDM Embedded: All
- ✓ RDM Server: All

Contents

Overview

Raima's embedded databases are unique in that they give total control of the data management system to the application. In other words, the database only does what the program instructs it to do. This means NO surprises to the application (and the developer).

Multiple APIs

Control of the Raima databases is made available through 3 different API types: Native, SQL, and JAVA. The API selected is a matter of the application design goals including run-time and maintenance. We will consider each of these API types.

Before selecting the interface, the design goals need to be fully considered. The API selection can be a function of footprint, performance, portability, or language. By understanding the balance between these design goals, the optimum API can be selected.

If a small footprint or code size is a design requirement, using the Native (or Java for Java based applications) is the interface of choice. RDM Embedded's native API offers over 150 functions providing a comprehensive set of functions to control the run-time and administration of the database.

The run-time API is a C/C++ compatible API set that offers functionality including Set and Record Manipulation, Set and Record Navigation, User Settings, XML Import/Export, Database Mirroring, and Database Administration. You will note that all of these APIs are run-time controlled which gives comprehensive database control to the application.

In cases where the application uses SQL either due to portability, familiarity, or desired abstraction, RDM Embedded's SQL API would be used. Using RDM Embedded's SQL interface layer allows you to use a standard database control language to manage the database. RDM Embedded's SQL layer mirrors a subset of the ODBC 3.51 standard and provides a standard for the database management. Our selection of ODBC API functions and SQL language is based on the available native API.

Finally, as of RDM Embedded 7.0 we now offer a Java interface that will allow you to develop your application in one of today's most exciting languages. Java developers have always been interested in the performance benefits of the Raima RDM Databases for their Java programs. The Java API on the RDM Embedded DBMS is based on Java Native Interface (JNI) technology and supports the Sun Microsystems JVM version 1.2 and above (Java 2 SDK). By extending the C API to the Java programmer via the JNI, RDM Embedded allows you to organize and access information efficiently, regardless of the complexity of your data.

The benefits of a JNI interface are obvious: Performance and Size. This combined technology provides tremendous speed advantages and minimizes data redundancy. With RDM Embedded's Java interface, Java developers can now take advantage of the inherent high performance of the RDM Embedded database for their critical applications while running in a Java environment.

Conclusion

So whatever your design criteria, with RDM Embedded, we have an interface that will meet the needs of your development environment.

Contact Information

Website: http://www.raima.com

WORLDWIDE

Raima Inc.
720 Third Avenue, Suite 1100
Seattle, WA 98104
Telephone: +1 206 748 5300
Fax: +1 206 748 5200
E-mail: sales@raima.com

EUROPE

Raima Inc.
Stubbings House, Henley Road
Maidenhead SL6 6QL United Kingdom
Telephone: +44 1628 826 800
Fax: +44 1628 825 343
E-mail: sales@raima.com