IDC
Analyze the Future

I D C A N A L Y S T C O N N E C T I O N

Carl W. Olofson
Research Director, Application Development and Deployment

Embedded Databases: The Invisible Engine That Could

August 2005

The embedded database management system (DBMS) market grew 15% to $1.86 billion in 2004, driven by continued demand for applications that can stand alone and work "out of the box" without user setup and without a database administrator (DBA). Types of applications driving this demand range across the board, from large server-based applications to small mobile applications and desktop tools. IDC expects the market for embedded database management systems to continue growing over the next five years, fueled by demand for self-contained applications that can manage data without a DBA. In particular, this trend will be supported by an increasing emphasis on fully self-contained packaged applications; falling prices for IT, resulting in increasing adoption by price-sensitive smaller businesses that require operationally affordable software because they have no DBA staff; and the DBA shortage, which continues to get worse, increasing demand for DBA-less databases.

The following questions were recently posed by db4objects on behalf of its independent software vendor (ISV) customers to Carl Olofson, Research Director of IDC's Application Development and Deployment group.

Q. What is an embedded database, and how does it differ from enterprise databases?

A. When IDC talks about an "embedded" database, we're referring to a database that's inside another software product. This kind of database is quite different from the type most people are familiar with, what might be called an "exposed" database.

An embedded database is one you don't see – you only see the product it's running inside of; the product for which it's managing data. The operating environment for an embedded database can range from big Unix servers to personal digital assistants (PDAs).

The key difference between an embedded database and an exposed database is that embedded databases aren't necessarily required to provide reporting. They're typically customized to the performance requirements of the application, which often doesn't include functions like reporting.

Perhaps the biggest difference, however, is that embedded databases must be so robust that they never fail. The reason for this requirement is that, in the environments where embedded databases typically operate, there is no database administrator. The database is invisible, so there's no one to come in an fix it if it breaks.

Another difference is that the performance characteristics of the embedded database need to match exactly the performance characteristics of the application it resides in — there can be no degradation. Similarly, the footprint of the embedded database needs to match the environment in which it's operating.

Embedded databases also usually have features that can be manipulated by the software in which it is embedded.

Q. What impact does the open-source phenomenon have on embedded databases?

A. Open source provides some interesting ways for vendors to offer creative licensing, which can bring the total licensing cost down. Usually an open-source vendor has a business model different from a closed-source or proprietary vendor. For example, typically a commercial database management system (DBMS) vendor charges per CPU or a per number of users price. Open source DBMS vendors sometimes don't even charge for the DBMS itself, but just charge for the support relationship. Then it may be that the DBMS vendor will have an escalator based on how much business its ISV partners are doing using the DBMS.

Licensing, however, is only part of the impact open source is having on the embedded DBMS market. Because an open-source embedded DBMS is distributed with its source code, this gives ISVs a tremendous advantage in packaging and debugging their own, final product. The ISV is able to examine all the best ways to use that embedded DBMS just by walking through its source code.

It's important to remember that ISVs have critical technical requirements that must be met. They're trying to make products that will be installed in widely different and unpredictable environments. Consequently, the ISV needs its product functionality to be tight and bulletproof. Being able to see the source code makes a big difference to these companies, because it greatly increases the range of testing that they can do.

Also, open source DBMS vendors typically offer ISVs the ability to offer only selective portions of the DBMS, which enables the ISV to keep the footprint down and streamline its own product.

Q. Object-oriented DBMSs are often seen as an unsuccessful technology. Do you think embedded object-oriented DBMSs will create a resurgence in this technology?

A. The object-oriented DBMS market went through a period of great growth, followed by a period of great contraction. The market needed to discover the value of object-oriented DBMSs, which is that object-oriented DBMSs function best as components of larger systems.

An object-oriented DBMS is a shared resource environment for an object-oriented application, which allows that app, for example, to be distributed across multiple systems in a network. The object-oriented DBMS offers another benefit when that shared data is used in managing business transactions. This is because the data they manage is persistent, so it creates a durable computing environment — if the system goes down, you just bring it back up and everything is restored.

Embedded object-oriented DBMSs have done a really good job of allowing application developers to build fluid, cross-system products. This software also has the advantage of supporting robust business processes that span systems and applications.

Already a number of the commercial object-oriented DBMS vendors have become aware of the opportunity to provide these capabilities to various components in application server environments. They have, in fact, become quite strong in that area.

©2005 IDC

An even more interesting opportunity for these vendors is in environments where they can provide a persistent shared-object capability. This environment can be spread across many servers and maintain knowledge of where the application is working and what it's working on over time. At the end of a transaction, for example, the application could store this data in a relational database. But while it's working, the app is keeping track of itself and managing shared resources.

The thing to keep in mind is that an object-oriented database doesn't necessarily manage the same data as a relational database.

The embedded version of object-oriented DBMSs is helping to move object technology into wider deployment — especially now that companies are adopting grid environments, for instance, where applications can be functionally distributed across many systems. Object-oriented DBMSs enable elements of a single application to run on different servers, which supports what people are calling a grid, clustered, or utility computing environment.

Embedded databases also enable software on occasionally-connected devices, including laptops, PDAs, Blackberries, etc., to deliver and handle enterprise information without requiring a constant signal. The DBMS in such cases should feature a very small footprint, but still offer normal data services to the application, operating as a local data source in cases when the device is disconnected from the network. This capability is vital for applications, such as field force automation (FFA), which require support for highly mobile information processing, and which need to provide capabilities for users who may find themselves in locations where they cannot connect to a WiFi or cell phone service, or where such connection may be prohibited, such as on commercial aircraft.

ABOUT THIS ANALYST

Carl Olofson performs research and analysis for IDC's Information Management and Data Integration Software service within the Application Development and Deployment research group. Mr. Olofson's research involves following sales and technical developments in the information and data management (IDM) markets, database management systems (DBMS) markets, data movement and replication software, data management software, metadata management software, and the vendors of related tools and software systems.

ABOUT THIS PUBLICATION

This publication was produced by IDC Go-to-Market Services. The opinion, analysis, and research results presented herein are drawn from more detailed research and analysis independently conducted and published by IDC, unless specific vendor sponsorship is noted. IDC Go-to-Market Services makes IDC content available in a wide range of formats for distribution by various companies. A license to distribute IDC content does not imply endorsement of or opinion about the licensee.

COPYRIGHT AND RESTRICTIONS

Any IDC information or reference to IDC that is to be used in advertising, press releases, or promotional materials requires prior written approval from IDC. For permission requests contact the GMS information line at 508-988-7610 or gms@idc.com. Translation and/or localization of this document requires an additional license from IDC.
For more information on IDC visit www.idc.com. For more information on IDC GMS visit www.idc.com/gms.

Global Headquarters: 5 Speen Street Framingham, MA 01701 USA P.508.872.8200 F.508.935.4015 www.idc.com