

db4objects, Inc.

The Business Case for the Open Source Object Database

Object-oriented database technology was first introduced in the early 1990s with great fanfare. However, its promise to displace relational database technology remained unfulfilled, partly because of technical deficiencies (most products were “non-native” and hence couldn’t realize the true benefits of object-oriented software development), but more so even due to the lack of an appropriate market focus and business model.

Today’s consequence is that nearly all market entrants have consolidated and focused on “vertical” target niches such as defense and healthcare. These proprietary object-oriented databases (ODBMS) tend to be expensive and usually require heavy vendor support for their complicated, proprietary interfaces. On average, 70% of these companies’ revenues stem from professional services rather than licensing. Among these are Versant (VSNT, consolidated with POET); ObjectStore, now the Real-Time division of Progress (PRGS); and Objectivity. As a result, most IT decision makers have ceased to consider the object database a mainstream technology.

Yet developers continue to love object database technology because it is the ideal match with object oriented (OO) environments like Java and .NET, and overcomes one of developers’ biggest hurdles: the inherent incompatibility between OO languages and current mainstream relational databases (RDBMS) – the so-called “object-relational (OR) mismatch.”

The OR mismatch has become enormously significant over the past three years, as most programming languages are now fully based on OO methodology – especially since Microsoft shipped C# in 2002 and thus validated Java’s OO concept – while datacenter legacy and a lack of affordable ODBMS solutions largely forces developers to continue using RDBMS (e.g., from Oracle, IBM, or MySQL) for data persistence.

An undeniable indicator of this growing pain is the emergence of so-called object-relational mappers (ORM), such as Toplink and Hibernate. This solution “cures” the symptoms of the OR mismatch by adding a layer into the software stack that automates the tedious task of linking objects to tables. However, this approach creates a huge drain on system performance, drives up software complexity, and increases the burden on software maintenance, thus resulting in higher cost of ownership. While the mapper solution may be feasible in large, administered datacenter environments, it is prohibitive in distributed and zero-administration architectures such as those required for embedded databases in client software, mobile devices, middleware or real-time systems.

Contents

1. The Market Opportunity
2. db4objects’ Vision
3. Open Source Business Model
4. Driving a Platform
5. Native OO Advantages
6. Future Prospects

db4objects, Inc. was founded in 2004 to fill this OO database gap and give developers a genuine choice when it comes to persisting data in OO environments. While OR mappers can be helpful in server-side applications, db4o's native object database provides for zero-administration and lean implementation of embedded object-oriented persistence. This harnesses the power of OOP to offer added flexibility and the ability to create more feature-rich, competitive products while slashing development time and cost.

db4o is designed to be a universal, affordable product platform, that is easy to learn, easy to implement and free or available for commercial use at very low cost. db4object's open source dual-license business model combines the power of the open source development community with servicing commercial customers' needs for product roadmap predictability, indemnification, single point of contact, and full tech support with fast response times.

As a result, db4o is now one of the most popular object databases in the world. If anyone has ever told you that object databases were a dead end, it is now time to think again. This paper will provide an eye-opening look at db4o's market focus and open source business model, and will illustrate why db4o's native object database technology is here to stay.

The Market Opportunity

“Building Java applications that use relational databases is perhaps the single most underestimated challenge in enterprise development today. More projects are delayed, under-featured and difficult to maintain because of this underestimation. The problem lies with the use of fundamentally different technologies... The object world and the relational world do not match up.”

– Oracle Corporation¹

“Using tables to store objects is like driving your car home and then disassembling it to put it in the garage. It can be assembled again in the morning, but one eventually asks whether this is the most efficient way to park a car.”

– Esther Dyson²

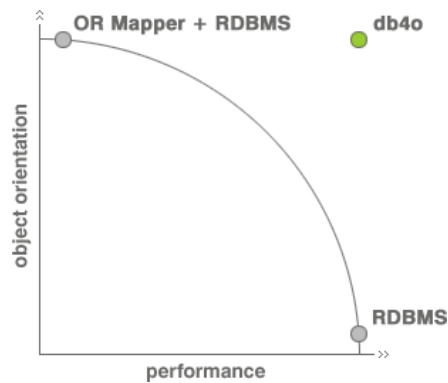
The emergence of distributed data architectures – in networks, on clients and embedded in “smart” products such as cars or medical devices – is forcing companies in an array of industries to look beyond traditional RDBMS technology for an improved way to deal with object persistence. They are searching for a solution that can handle an enormous number of often complex objects, offers powerful replication and querying capabilities, and slashes development time. This is especially true for updating software versions and object models that must be distributed over a large installed base without the need for an administrator.

¹ Donald Smith, Building Superior Java Applications with Oracle9iAS Toplink, Oracle Whitepaper, September 2002

² Esther Dyson, Debunking Object-Database Myths, Byte.com, October 1997

The major reason for using relational databases today is legacy software, i.e. retaining old enterprise data and the set of existing applications relying on it. But beyond the server-centric persistence, there are a multitude of enterprise applications, consumer products and other examples of client-side persistence for which conventional database technology falls short when OO languages like Java and .NET are used. Here db4o's technology offers new levels of performance and functionality.

Currently the most salient solution for the object-relational mismatch is the use of an object-relational mapper (ORM), such as Hibernate, a popular product that "wraps" relational databases to store objects directly. But benchmark results demonstrate that this type of solution is only viable if resources are abundant and performance is not critical (see www.db4o.com/about/productinformation/benchmarks/).



As a new-generation object database, native to both Java and .NET, db4o eliminates the traditional trade-off between performance and object-orientation. Recent PolePosition benchmark results show that db4o outperforms object-relational mappers by orders of magnitude, up to 44x in use cases with complex object models.

db4o uniquely offers object persistence with zero-administration, cross-platform applicability to Java and .NET, object-oriented querying, replication and browsing capabilities, and a small footprint of only some ~350KB. Its single library (JAR/.DLL) is easily deployed and runs in the same memory process as the application, making it a fully integrated and tunable portion of the product.

Customers, analysts, and experts agree that the db4o object database is one of the world's best and most popular choices, because it stores object natively, with just one line of code and not only eliminates the overhead and resource consumption of an ORM, but is also much leaner, faster and easier to integrate into an OOP development than any RDBMS.

db4o Case Studies

www.db4o.com/about/customers



AVE High Speed Trains



BOSCH's Packaging Robots



Massie Labs' RetCam II

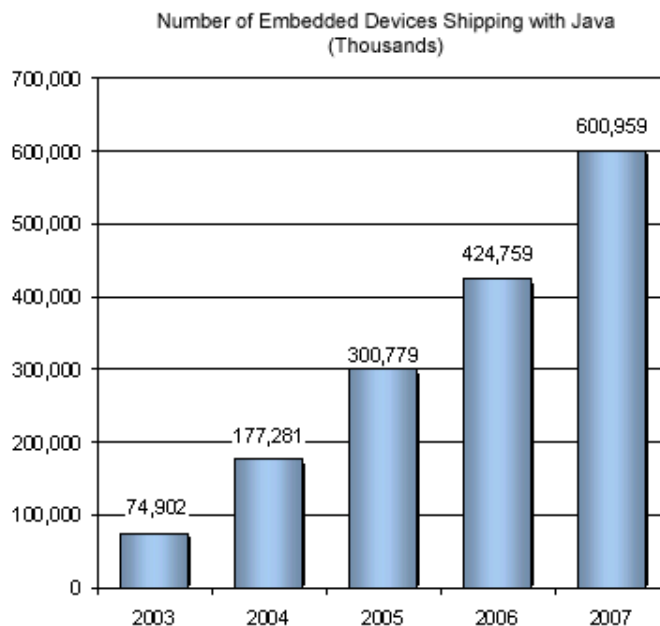


Eastern Data's Mobile Apps

www.db4o.com/about/customers

The target environments for db4o are therefore persistence architectures where there is no legacy, i.e. primarily on clients and in the middleware. It is also applicable in the enterprise data center, but used less often here due to the prevalence of legacy data and systems. Typical applications are embedded use in products such as mobile (both consumer and enterprise), medical and gaming devices (or software running on these devices), automotive, real-time control systems, and packaged software products. Existing customers range from multinationals like BMW, Hertz, and Bosch to a range of small, "boot-strapped" start-up companies.

To accommodate this wide field of interest, db4o adopted a dual license model, which supplies a full version of db4o either under the General Public License (GPL) for the open source community, or under a highly affordable conventional proprietary license for commercial product developers who wish to embed db4o into their non-GPL commercial products.



"Java's success in embedded systems is now all but assured, especially in consumer and handheld devices," according to Venture Development Corporation (VDC) market research³. This growth is complemented by Microsoft's .NET platform.

The embedded DBMS market grew 15% to \$1.86 billion in 2004, according to IDC⁴, and is expected to grow to \$3.18 billion in 2009. db4objects is confident that its business model, focus, and unique product benefits will make it the leading technology in this segment, and while decreasing unit prices, will also dramatically increase the technology's reach and number of installed units within the next few years.

³ www.vdc-corp.com/embedded/press/03/pr03-62.html

⁴ Worldwide Embedded Database Management Systems 2005-2009 Forecast, IDC #33449, May 2005

db4objects' Vision

As devices become “smarter” and enterprises increasingly mobile, relational databases slowly sink to their knees, and the need for a lightweight, apt object-oriented persistence solution becomes increasingly urgent.

db4o provides OO developers with a choice beyond relational databases when it comes to persistence. Our vision as a company is to become the mainstream persistence architecture on all mobile and embedded devices running on Java or .NET, achieving consolidation in a market now overrun with hundreds of small vertical niche vendors offering predominantly outdated or unsuited pre-relational or relational technology at exorbitant prices.

The template for this consolidation process can be found in the rise of embedded Linux as a real-time operating system (RTOS). Linux consolidated a wide array of proprietary, closed-source RTOS offerings to one common, low-cost and universally accepted platform offering huge benefits to users in terms of product commoditization, interoperability and liberty from high, proprietary licensing fees.

We believe that the availability of a popular, affordable and embeddable open source object database will have a similar, profound effect on the IT industry and the innumerable enterprises that rely on it to move forward. db4o not only alleviates the pain OO developers suffer in current projects, but also opens a world of new applications, products, and smart devices.

As a client-side, embeddable database, db4o is particularly suited to be deployed in mobile devices or products with embedded software. Germany's Mobilanten, for example, was an early adopter of db4o's mobile offering. This company gained a competitive edge by providing a PDA-based solution for fieldforce workers of mid-sized utilities, whereas competitors using relational databases (RDBMS) required bulky laptops to process assets, orders, materials and customer information. Replicating some 300,000 objects was just not feasible using RDBMS on a PDA, while synchronizing objects via db4o proved to be extremely efficient.

Additionally, electronics giants like BOSCH Sigpack confirm that they have been able to cut time-to-market for their products by about 10% when using db4o, because db4o can facilitate the complex object models required by their high-performance packaging robots with ease.

INDRA Sistemas has built a new real-time control system for Spain's high speed bullet train, called AVE, using db4o. No other system was able to handle the incredible load of processing over 200,000 heterogeneous objects per second.

Many saw OO databases as propagated by closed-source vendors in the 1990s as a dead end. But db4objects has inverted this view by using the open source model as a strategic tool to enable focus on the right target market – embeddable databases – and by benefiting from the mainstream adoption of OO programming environments, marked by Microsoft's .NET platform and C# programming language releases in 2002, validating Java's OO approach.

Following a four-year development period with some 100 pilot customers helping to create a technologically stable platform, db4o was released under a dual open source/commercial license in November 2004.

The power of the open source model has given db4objects a jumpstart to achieve its vision at warp speed: In just six months, db4o has realized more than 200,000 downloads and is now supported by a registered user community nearing 5,000 users. It is already one of the world's most popular object databases, in essentially no time.

"Dead" technologies look quite different. db4objects has effectively put object database technology back on the map.

Open Source Business Model

While offering software for free might seem like nonsense at first sight, it turns out to be a smart triple-win strategy.

db4objects uses the now-established, open source dual license business model as pioneered by MySQL, one of the world's most popular relational databases. In this model, products are made available in their entirety under the GPL and also under a commercial license. Any developer wishing to use the software in an open source product that falls under the GPL can use the free open source version. Those developers wishing to embed db4o into a for-profit product can choose the affordable commercial runtime license. Other uses and licenses including those for evaluation, development, and academic application remain free under the GPL, creating a large and lively community around the product at very low cost to the vendor.

Note that the GPL license is the "cleanest" open source license, because it clearly states a "quid-pro-quo": if the providing vendor is open source, then users need to be open source as well. The dual license model extends this by offering an affordable commercial license for those who wish to use the technology commercially. This alternative removes the limitations of the GPL license obligations for commercial developers, therefore making the product available to a much larger audience. It also allows db4objects to meet commercial users' needs with indemnification, a single point of contact for fast-response support, and directed product development. That is, as opposed to "pure" open source software that floats freely among the community and is altered at random, db4o's core product development is guided by the company, with an eye toward filtering community input and implementing the best recommendations. Also, the company sets the agenda and establishes a roadmap, so that developers can predict where the technology is going for better planning and decision-making.

The vast community and extended use of an open source product platform like db4o is the key to its success. All users, commercial or not, benefit from its wide adoption: better-tested software, better suggestions (with users actually looking into the code), better peer exchange of experiences – positive and negative – eliminates the gatekeeping function and also the enormous overhead associated with Outbound Sales and Marketing departments. These functions are actually the biggest cost factors in most closed-source software companies today. Customers end up sharing the burden of paying for sales efforts aimed at explaining to them why they should purchase a product that they weren't looking for in the first place. But

in the open source world, this sort of outbound proselytizing adds no value. Open source companies like db4o (can afford to) wait for customers to come to them, in search of an appropriate product and hence realize enormous savings in sales and marketing efforts.

db4objects also has a huge cost advantage when it comes to production. Using open source development processes and tools, db4o can productively employ a distributed workforce, blending internal and external personnel to hire the best person for the right job regardless of location. Utilizing open and free collaboration platforms such as newsgroups, CVS, Bugzilla, and VoIP, to name just a few, db4objects can foster a productive dialogue between users and core developers. This not only advances its transparent development model, but also eliminates the bulk of high software production costs.

With these various savings, db4objects' model allows it to incur much lower cost of doing business, and can therefore sustainably offer its customers more affordable prices and higher quality than closed-source competitors can. While our customers will not be "pampered" by sales personnel, they will enjoy the huge benefits of an affordable, innovative, stable, and well-supported product platform that makes their life easier and eliminates a number of major headaches in their software development processes.

Our interactive and completely transparent development model makes product evaluation much more efficient, as deficiencies are evident and openly discussed. This drives db4objects to invest in product improvements rather than in sales collateral. Effectively, most commercial customers start off as members of the open source user community, as they usually begin by evaluating the product for free before they are ready to purchase. The resulting co-existence of community, commercial customers, and vendor translates into a win-win-win situation: It is not until customers are ready to ship complete products that db4objects asks for a fair and proportional reimbursement for the materialized benefits of the enabling technology.

Paradoxically, it is the paying, commercial customers who benefit most. The open source community offers them efficient, free evaluation in the decision-making stage, and its low-cost production and distribution mechanisms provide them excellent quality software at much lower prices in the long run.

Driving a Platform

db4objects aims to take the open source business model a level beyond what classical open source players like Linux and MySQL have done by effectively introducing an innovative product platform by means of the open source community's pervasiveness.

That is, we are working to advance object database technology not to commoditize a common, mainstream software category, but rather to drive the creation of a new, viable platform that will foster multiple product families and expansion into new markets.

A platform can be defined as "the smallest set of technologies common for a range of products that determine the ultimate competitive performance of those products, i.e. an explicit entity which remains unchanged in all the products incorporating the platform."

Platforms allow expansion into new markets defined by different performance/segment combinations via modularity and subsystem components that are compatible with the system interfaces of the platform technology.

The process of driving a platform involves a number of key steps, including creating a statement of strategic intent, a situation analysis, a strategic plan, defining the basis of competition, developing product family maps, and finally, business valuation.

For platform products like databases, the key to business success is a minimum installed base that makes a product acceptable to commercial developers. Too often, closed-source companies find it difficult to drive user adoption and earn money at the same time. In contrast, db4objects has successfully embraced a model that drives fast user adoption at very low cost, while still reliably collecting revenues from the subset of (commercial) users deriving the most commercial benefit from db4o.

No other object database experienced similar growth to that of db4o following its open source release in 2004, when more than 5,000 users registered in only 6 months and 200,000 downloads and many more deployments materialized within a very short timeframe. At the same time, db4objects has been periodically breaking financially even since inception, and has successfully closed commercial deals with the likes of BMW, Hertz, and BOSCH.

As noted, db4o is now one of the world's most popular object databases – setting the stage to quickly become a winning platform across markets.

Native OO Advantages

"Ask any object oriented developer about their list of frustrations and you're likely to hear them mention the difficulty transitioning from object-oriented thinking to relational persistence. db4objects aims to eliminate this divide entirely, without the overhead of an object/relational mapping layer, by providing native object oriented persistence, thereby allowing developers to store data in the same manner as the application generating it: as an object."

– Stephen O'Grady, Senior Analyst, RedMonk

Object orientation is not just a technology, but a methodology for structuring and representing data in a more "naturalistic" way than the methods used in early architectures, i.e. hierarchical and relational models. The industry has recognized the value of this methodology, and moved to make OO the mainstream programming paradigm, with Java (pushed by IBM via Eclipse) and .NET (Microsoft) technologies competing head-to-head for the lead. db4o not only runs under both platforms, but has also proven strong and efficient in cross-platform deployment. As a consequence, companies can choose heterogeneous platforms to increase their competitiveness and mitigate vendor lock-in. A typical example of this advantage is a heterogeneous architecture with clients like PDAs running on Windows Mobile (.NET), while the server-side runs on Java and open-source components. Another example is an ISV's product to run on both Java and .NET-enabled devices like smart phones and game consoles or on a desktop computer.

There are many advantages of using “native” object technology over RDBMS or RDBMS paired with an OR mapper, and these technological advantages significantly impact an organization’s competitiveness and bottom line.

First, we have mentioned that object databases not only simplify development by eliminating the time and resource-consumptive OR-mismatch entirely, but they also foster more sophisticated development through gains in flexibility and productivity brought on by “true” object-orientation.

db4o’s ground-breaking object-oriented replication technology solves problems arising from distributed data architectures. Partially connected devices need to efficiently replicate data with peers or servers. The challenge lies in the creation of “smart” conflict resolution algorithms, when redundant data sets are simultaneously modified and need to be merged. With db4o’s OO approach, developers can build smarter and easier synchronization conflict resolution and embed the necessary business logic into the data layer, rather than into the middle-tier or application layer. This creates “smart” objects that can be stored in a distributed fashion, but easily consolidated, as the smart object itself knows how to resolve synchronization conflicts.

As a result, developers can now more consistently persist data on distributed, partially connected clients than ever before, while decreasing bandwidth requirements and increasing the responsiveness and reach of their mobile solutions or smart devices to make products more competitive in the marketplace.

Second, with an object database, the object scheme is one and the same as the data model. Developers can more easily update their models to meet changing requirements, or for debugging and refactoring. db4o lets developers work with object structures almost as if they were “in-memory” structures. Little additional coding is required to manage object persistence. As a result, companies can add new features to their products faster to stay ahead of the competition.

Third, developers can now use “true” object-oriented approaches when it comes to querying, since db4o provides several object-oriented query APIs that take advantage of the particularities of OO methodology. This capability makes querying more search-oriented, aligning db4o’s applicability nicely with the industry’s trend to rely on search rather than hierarchical data management.

Fourth, db4o also allows for more complex object models than its relational or non-native counterparts; as the persistence requirements become more complex, db4o’s unique design easily handles (or absorbs) the added complexity, so developers can continue to work as though new complexity were never introduced. Complexity means not only taller object trees and extensive use of inheritance, but also dynamically evolving object models, most extremely if development is taking place under runtime conditions (which makes db4o a leading choice for biotech simulation software, for instance). db4o could be referred to as “agnostic to complexity,” because no type or amount of complexity will change its behavior or restrain its capabilities, as is the case with RDBMS or non-native technology. The real world is more complex than IT architects wish. With db4o breaking through this complexity, developers are able to write more naturalistic – i.e., user friendly and business-case appropriate – software components without incurring such high costs that such an undertaking becomes unfeasible.

In sum, db4o's native, cross-platform OO architecture enables its users to build more competitive products with faster update cycles, more naturalistic object models to match more realistic use cases, and more distributed data architectures to increase the reach of products. Native OO database technology is clearly more flexible and powerful than software that stems from the world of functional programming languages.

Future Prospects

db4o is going places. As a company, db4objects is a well-funded, highly respected startup that has become one of the hottest technology innovators in Silicon Valley. As a platform, db4o is supported by an avid, rapidly growing user community that is spreading the word of its value at lightening speed.

db4objects is a privately-held company based in San Mateo, California. The company was incorporated in 2004 under CEO Christof Wittig, with the financial backing of top-tier Silicon Valley investors including Mark Leslie, founding CEO of Veritas; Diane Greene, president of VMware; and Audrey MacLean, named by BusinessWeek as one of the 50 most influential business women in America.

db4o was created in 2000 by Chief Architect Carl Rosenberger, and first shipped in 2001. Some 100 commercial pilot customers and a loyal user community have endorsed db4o from its earliest days, and proved it ready for mission-critical applications prior to its commercial launch in 2004. Since then, dozens of prestigious new customers helped realize upwards of 200,000 downloads and many more deployments, and built a fast-growing, devoted community of 5,000 registered users.

With the open source code effectively being escrowed to the public, db4o has the world's best insurance that it will continue to thrive as a powerful, living platform in 10, 20, or more years from now. Thus, OOP developers can rest assured that they can safely choose db4o for projects with a long-term outlook.

All OO developers stand to benefit, along with the developer community at large, not to mention the many industries to which object database technology brings revolutionary capabilities – in mobile solutions, in bio-medical, in automotive, in real-time control and industrial automation, and in many other systems.

db4objects has become a high-profile forerunner in the open source wave that is changing the software industry unlike few paradigm changes before. Embraced by players like IBM, HP, and SAP, open source has become a major “game changer,” creating a new ecosystem in which object database technology has finally found its rightful place as a viable database choice that makes the life of Java and .NET developers so much easier – and thus saves huge amounts of development time and cost.