


Another Technology Report from Ken North Computing



Middleware in Action

Industrial Strength Data Access

May 2007

Executive Summary

Understanding Middleware for Data Access

The move to client-server architecture for databases brought a division of labor when processing SQL queries, the details of which are often misunderstood. The complexities of distributed processing and the importance of best-of-breed middleware can elude even senior technical people, such as system architects, IT managers and consultants.

Parallel processing, commodity servers and high-bandwidth networks advanced the distributed computing model and the distribution of data. Tuning database applications involved fewer variables when the databases were on centrally managed mainframes being accessed by local users. Distributed processing introduced the network round trip for a query, concurrency issues and the need for sophisticated client programming. It also created a class of middleware specifically for accessing server databases. Today's model for query processing introduced the performance effect of middleware, network latency and distributed data.

In a distributed processing environment, premium quality data access middleware can provide important functionality:

- Implementation of database-specific communication protocols
- Efficiency and security of network communications
- Type and state mapping
- Error checking
- Caching data and connections
- Authorization and authentication
- Scalability and load balancing.

This is the executive summary of an independent technology assessment. It presents a top level view of the role of data access middleware and its importance. The complete report provides more detailed information than this summary.

Section 3 of this report discusses the defining characteristics of premium data access middleware, including features that affect performance and scalability.

Parts, Components and the Emergence of Middleware

In the modern era of computing, monolithic applications have given way to servers, clients, middleware and components. Our understanding of information technology (IT), software, applications and web sites is analogous to our relationship with the automobile. Depth and breadth of knowledge comes not from use, but from design, building and repairing. The person who designs, builds or repairs cars must understand quality requirements for parts and assemblies. The same is true of computing and information technology. Solutions for web computing, mobile computing and enterprise computing rely on a collection of hardware and software components. Knowledge of parts and assemblies is essential to being able to create and maintain systems in this era of distributed computing.

The model of monolithic applications on a central computer gave way to resource sharing and partitioning logic across clients and servers. That meant an application's parts and data were distributed and not centrally managed. Clients and servers communicated via remote procedure calls (RPCs) and exchanging messages, implemented with various types of middleware. The reliability, performance and security of middleware in processing messages or accessing databases became an important consideration for mission-critical software. As the computing paradigm changed, consistency also became an issue. Consistent behavior and compliance with standards became a critical feature for data access middleware.

As a development paradigm, fabrication from parts solved problems of building monolithic applications. But in distributing processing across servers, clients, databases and middleware, we are reminded

A chain is only as strong as its weakest link.

Networks, RPCs and distributed computing

To build applications and web sites characterized by stability, security and excellent service, we must maintain quality when buying or building the constituent parts. Every part of a system must be robust, supportive of data integrity and transactional integrity, and consistent in its behavior. This is true of tool kits, servers, database software, network software and middleware.

Attaining that assurance for middleware is best addressed by adopting a best-of-breed mentality when it comes to buying or building data access middleware. The defining characteristics of best-of-breed data access middleware include:

- Broad connectivity and platform support
- Scalability
- Performance
- Robustness, reliability, stability, high availability
- Standards compliance, consistency
- Security
- Tools, language support.

Best-of-breed
middleware

Frameworks, Databases, Technology Behind the Browser

The creation of new applications has been influenced by various software development paradigms, from structured programming and modular development to prototyping, object-oriented, agent-oriented and aspect-oriented programming. Development methods today include component-based development, model-driven development, re-factoring design and architecture patterns, extreme programming and agile development. Regardless of the preferred methodology, many developers embrace integrated development environments and frameworks. These include Eclipse, Java Platform, Enterprise Edition (Java EE), Hibernate, Spring, .NET and various AJAX frameworks. Frameworks provide efficiency with tested, reusable components, including components that use data access middleware. Using frameworks and components raises the level of abstraction for developers, but they do not obviate the need for data access middleware.

Despite the appeal of data-aware components, Web 2.0, and AJAX (Asynchronous JavaScript and XML), not all information processing lends itself to mashup, scripting or point-and-click solutions. Classic applications, such as order processing, business intelligence, design engineering and supply chain automation, require sophisticated behind-the-browser technology. They can also be quite complex, such as Enterprise Resource Planning (ERP) suites with a database schema having thousands of tables. We've traded monolithic applications and centrally managed computers for distributed systems composed of many parts (servers, network gear, software libraries, databases and more). The tradeoff for distributing data closer to the user is an increased need for:

- Data integration software
- Versioning and configuration control tools
- High-bandwidth networking
- Distributed query capabilities
- Sophisticated authorization and authentication solutions
- Locales and globalization
- Load balancing and failover capabilities.

The growth of distributed computing and databases fueled the creation of middleware for SQL access to data; middleware that supports the capabilities described here.

The Emergence of Data Access Standards and Middleware

There was much debate about whether data access middleware should support a proprietary application programming interface (API) or a standard, multi-database API. The issue was settled by widespread adoption of multi-database APIs. Today tools, frameworks and developer languages make extensive use of SQL API standards, including JDBC™ and Open Database Connectivity (ODBC). Following the emergence of middleware based on those standards, the software industry raced to develop libraries, database engines and data-aware components that operate with standards-based middleware. It was here that many performance bottlenecks were found that were erroneously attributed to standard APIs and middleware. When the performance myths were de-bunked, the major software vendors started using standards-based middleware for multi-database access by Oracle Open Gateway, IBM DataJoiner and other products. Eventually data access middleware and drivers were bundled with application servers, integration servers, business intelligence servers and many platform products.

Data access middleware enables connected and disconnected clients, including middle-tier servers, to communicate with remote or local SQL servers. But distributed application design requires attention to security, scalability and performance.

Performance bottlenecks and myths

Mature Technology

The trade press and industry pundits have put a spotlight on services-oriented architecture (SOA), Web 2.0 and search engines. But enterprise architects and system architects understand the contribution of industrial-strength enabling technologies to the success of distributed applications and web sites.

There's a steady stream of innovation in areas such as data visualization, user presentation, 3-D graphics, social networking, streaming video and business intelligence. But innovative applications often rely on mature core technologies, such as TCP/IP and SQL. Database software vendors, such as IBM, Microsoft and Oracle, continue to evolve their SQL platforms, spurred on in part by competition from open source products such as MySQL and PostgreSQL. Nonetheless SQL is mature technology for persistence and applications such as online transaction processing (OLTP).

SQL and core technologies

The software architecture for multi-database access and the APIs represent a mature, enabling technology. Perhaps that's why some system builders incorrectly assume database middleware is commodity software, with all drivers and providers being equal in quality, capabilities and compliance with standards. Database gurus and knowledgeable system architects don't buy into the "all drivers are equal" argument. Nor do technologists at leading software vendors who select drivers and data providers for their products. More than 300 software companies license and distribute DataDirect middleware including:

- *The top four vendors in the SQL database market*
- *The six leading vendors of application servers for J2EE and Java EE platforms*
- *16 of the top 20 business intelligence (BI) vendors*
- *12 of the largest vendors of content management, knowledge management and portal software*
- *11 of the industry leaders with data integration and information integration products*
- *More than 80% of the leading application software, systems infrastructure and packaged software vendors.*

SQL and dropping hardware prices helped launch the distributed computing wave when DBMS vendors moved to client-server architecture. The decreasing cost of computers was a driver for application partitioning, client-server computing and distributed processing. A distributed architecture offers benefits such as eliminating single points of failure and throwing more hardware at performance problems. The adoption of distributed computing has gone hand in hand with organic growth of high-performance and high-availability computing. Distributed processing and distributed data were a causative factor in the surge of interest in integration with enterprise applications, mainframe computers and legacy databases.

More computing capacity enables serving more users and improving application and database performance. It improves the response time for users and reduces the time lag for making information available. For example, processing during the mainframe era was batch-oriented because of limited processing capacity. During that period, business users would accept a window of five days for month-end processing on the computer. Today IT departments deal with totally different expectations, such as users wanting real-time business intelligence. Today's software solutions must not only be robust, they must also be fast and scalable.

BEA, IBM, Microsoft, Oracle and Sun are among the leading software vendors who bundle data access middleware from DataDirect Technologies with their products.

The adoption of client-server SQL technology brought major changes. Data access techniques evolved, dynamic SQL gained prominence and a new type of middleware emerged. In the early days of SQL application programming, developers used embedded SQL or a proprietary API. Embedding static SQL or dynamic SQL in source code requires re-compiling programs for adapting to heterogeneous databases having different types and features.

Embedded SQL, compile-time optimization and static SQL are a classic, performance performance-oriented solution for database developers. But programmers who required greater flexibility opted for dynamic SQL and run-time invocation of data access libraries. Developers who were writing multi-DBMS client software weren't working with uniform data distribution, known queries and schemas. They wanted a solution that, unlike embedded SQL, didn't require recompiling the client code for each targeted DBMS.

The desire for database platform independence and a vendor-neutral API led to the emergence of Open Database Connectivity (ODBC), and subsequently JDBC. Both APIs enable developers to write dynamic SQL, however DataDirect middleware uses bind packages when possible to provide better performance than dynamic SQL. The .NET Framework introduced another platform-neutral data access solution. ADO.NET is an API, optimized for the .NET platform, which complements the Common Language Runtime, ASP.NET and other .NET technologies.

DataDirect software is in widespread use for enterprise computing using ODBC, JDBC and ADO.NET.

Eight of the top 10 U.S. banks license DataDirect data access middleware.

Scalability, Interoperability, High Availability

Applications with Windows clients accessing SQL databases could serve thousands of users, but Internet connectivity brought dramatic growth in the user population. Internet computing is distributed computing in the large and it places a premium on scalable architectures. Scalability is a key requirement for client-server databases, networking software and data access middleware. The Internet model supports large user populations and distributed processing based on platform-neutral technology.

The Internet and Extensible Markup Language (XML) emphasize interoperability, which has influenced middleware and services. Middleware today can provide connections between homogeneous or heterogeneous systems, clients and servers, using diverse operating systems and database platforms. It can operate in any tier of a multi-tier architecture (figure 1).

Internet, XML and heterogeneous systems

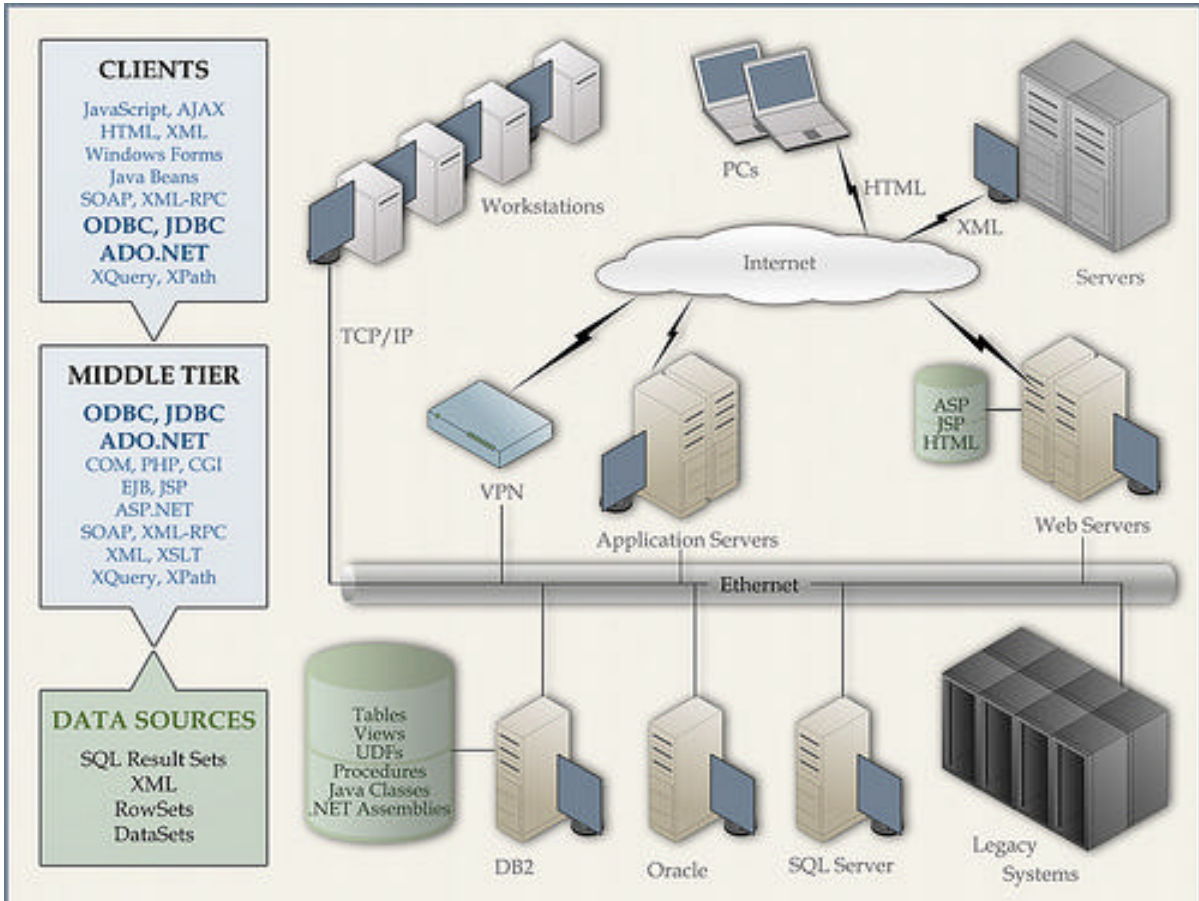


Figure 1
Multi-tier distributed application architectures provide performance and scalability. Data access middleware can operate from any tier of a distributed application.

Components, Services, Services-Oriented Architecture (SOA)

After object-oriented programming (OOP) and components became a favored approach to developing software, it was inevitable there'd be a call for data-aware components. These components simplify development by putting a layer of abstraction over SQL application programming interfaces (APIs). Not to be overlooked is the fact that under the covers the components use data access middleware. Components, object libraries and database engines sometimes introduce a performance penalty that's erroneously attributed to APIs and middleware.

The emergence of XML prompted a move by IBM and Microsoft to resolve interoperability problems with components. They unveiled a computing paradigm that used XML-based Web services to permit the creation of collaborative applications. The new model for collaborative computing using Web services and a services-oriented architecture (SOA) garnered widespread support among leading software and hardware companies and the open source community.

Recognizing the potential of XML, the major database vendors extended their SQL platforms to provide document processing, document queries, XML messaging and integration with Web services.

As SQL platforms added features to support document processing and data processing, data access middleware evolved to support SOA computing and a new XML data type.

Middleware under the Microscope

One approach to selecting software is to create a list of features required to sustain an organization's applications, services and web sites. For data access middleware, certain attributes appear consistently on a required features list.

The key attributes include multiple capabilities for boosting query performance, including caches and connection pooling. Middleware should support tunable data access performance, such as adjusting network packet size. For scalability and high availability, it should be multi-threaded and thread safe, with capabilities for client load balancing and failover to alternate servers (figure 2).

Performance and scalability

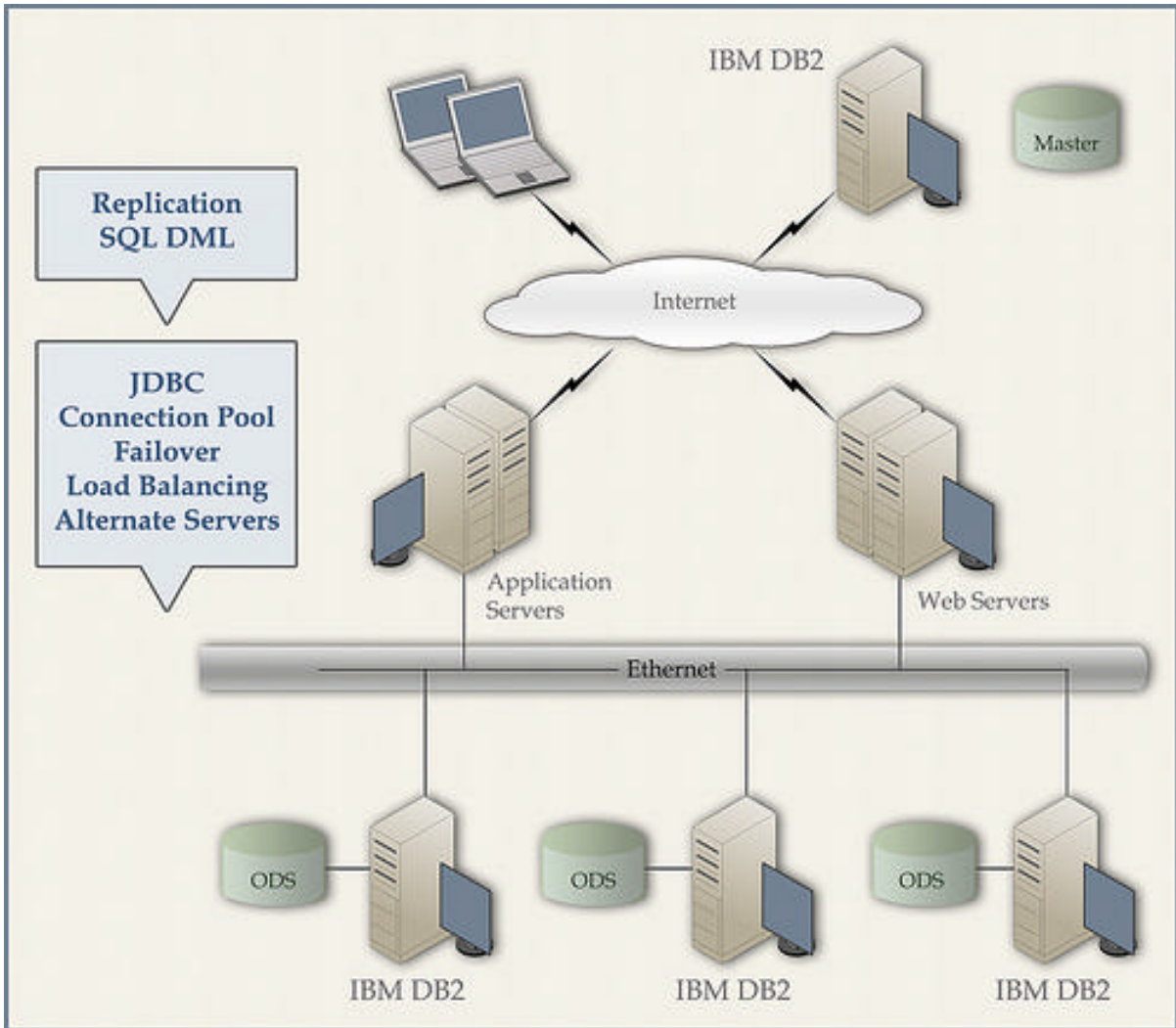


Figure 2
Premium-quality data access middleware plugs into scalable application architectures. These architectures use load balancing, alternate servers and failover capabilities, including client failover.

Licensing middleware for a specific application or database version is shortsighted, because organizations typically manifest a need for accessing disparate data sources for a variety of purposes. Middleware should therefore support disparate databases, types and features and interoperable SQL. That flexibility should extend to supporting diverse applications, such as business intelligence, data warehousing, transaction processing and legacy data integration. It should extend further to operation with multiple computing platforms, with different chipsets and operating systems. Another flexibility requirement is the capability of operation in systems and applications with diverse architectures - scaling successfully from workgroup to department, division, the enterprise and the Internet.

Multi-platform flexibility

In an era when criminal enterprises employ skilled hackers, organizations require middleware with a strong emphasis on security. It should fit into a defense in depth strategy for network security and database security, with secure communications and secure code. It should also integrate with multiple solutions for authentication and authorization, including three-factor authentication.

Security

To facilitate ease of development and debugging, data access middleware should include trace and spy tools. It should be standards-compliant and reliable, with each new release subjected to rigorous pre-release testing. One measure of quality and reliability is whether it enjoys industry support.

Reliability

Data access middleware from DataDirect Technologies meets these requirements. By all of these measures, it's a middleware of choice. This is reflected by the number of key companies in the computing industry, such as Adobe, BEA, IBM, Microsoft, Oracle and Sun, that license and distribute DataDirect middleware.

The checklist that follows provides a summary of features with an indication whether a DataDirect driver or data provider provides each feature. A more detailed discussion is found in section 3 of "Middleware in Action: Industrial Strength Data Access".

Data Access Middleware Checklist

This checklist presents critical data access middleware features that should be part of any evaluation of data access middleware. The second column is for indicating whether a feature is a requirement. There are three columns that indicate whether a feature is available or applies to one of the DataDirect Connect products. The last column is for entering data about other drivers or providers.

“To leverage the best of Java and its distributed processing capabilities, it’s important to have optimal database connectivity, data caching, and horizontal scaling solutions.

Premium-grade data access middleware allows system architects and developers to create first-class solutions with the Java platform, and is essential for performance and scalability.”

Rick Cattell

Distinguished
Engineer and Chief
Architect, Sun
Microsystems
Database Technology
Group

Platforms, Computing Environment, Connectivity

	Required? (Y/N)	Connect for ODBC	Connect for JDBC	Connect for ADO.NET	Other driver, provider
Windows operating systems		●	●	●	
Linux ¹		●	●	◐	
Unix, Solaris, AIX, HP-UX ¹		●	●	◐	
32-bit Platforms		●	●	●	
64-bit Platforms		●	●	●	
Requires VM, runtime for Java			●		
Requires CLR for .NET				●	
Operates without client libraries		●	●	●	
Flow over TCP/IP		●	●	●	
Flow over Named Pipes		●			
Clientless installation			●	●	
Kerberos, OS authentication		●	●	●	
JNDI support			●		
Active Directory support				●	
LDAP support			●		
Support JCA as resource adapter			●		
TNS Names support		●	●	●	
Globalization, NLS, user locales		●	●	●	
Unicode		●	●	●	
Distributed transactions ² , XA transactions		●	●	●	
Trace capability ³		◐	●	●	
Integrated spy capability			●	●	
Performance optimization wizard		●	●	●	

¹ Novell Mono offers an open source .NET framework for BSD Unix, Linux and Solaris. DataDirect has not certified its ADO.NET data providers for Mono.

² Windows platforms support distributed transactions using the Microsoft Distributed Transaction Coordinator / COM+. ODBC drivers and ADO.NET can use this functionality for distributed transaction support. JDBC drivers can work with XA-compliant resource managers.

³ On Windows platforms, the Microsoft ODBC Administrator provides an ODBC trace facility.

Performance, Scalability

	Required ? (Y/N)	Connect for ODBC	Connect for JDBC	Connect for ADO.NET	Other driver, provider
Asynchronous query execution		●	●	●	
Connection pooling ¹		●	●	●	
Multi-threaded, thread per connection		●	●	●	
Emit TDS packets (Sybase, Microsoft)		●	●	●	
Wire protocol, TNS support for Oracle		●	●	●	
Wire protocol, DRDA support for DB2		●	●	●	
Adjustable network packet size		●	●	●	
Statement caching		●	●		
Performance tuning w/ connect strings		●	●	●	
Prepared queries, parameter queries		●	●	●	
Use stored procedures		●	●	●	
Pre-fetch rows		●	●	●	
Use static SQL		●	●	●	
Use stored procs for prepared queries		●	●	●	
Create, modify, use DB2 bind packages		●	●	●	
Batch inserts and updates		●	●	●	
High-availability clusters, load balance		●	●	●	
Redundancy, connection failover		●	●	●	
Access REF CURSOR data		●	●	●	
Nested Savepoints				●	
Snapshot isolation level		●	●	●	
Efficient memory allocation/object use		●	●	●	

¹ Supported on Windows platforms. For UNIX and Linux, SQL Relay or other third-party software is required.

Features

	Required? (Y/N)	Connect for ODBC	Connect for JDBC	Connect for ADO.NET	Other driver, provider
Thread safe		●	●	●	
Escape clauses for interoperable SQL		●	●	●	
SQL leveling		●	●	●	
Metadata for adaptive programming		●	●	●	
Comprehensive scalar function support		●	●	●	
ANSI transactions, isolation levels		●	●	●	
SQL:2003 row value constructors			●		
Multi-database support		●	●	●	
Multi-version DBMS support		●	●	●	
Return query results as XML		●	●	●	
Serialization interfaces, Serializable			●	●	
Connected/disconnected queries			●	●	
JSR-114 RowSet			●		
.NET DataSet				●	
Secure login with encrypted password		●	●	●	
Query data encryption			◐	◐	
Secure code			●	●	
Managed .NET code assemblies				●	
Managed stored procedures				●	
OS authentication (Kerberos)		●	●	●	
FISMA-compliant e-authentication		●	●	●	
HIPAA-compliant authentication		●	●	●	
FFIEC-compliant authentication		●	●	●	
Complies with PCI DSS authentication		●	●	●	
SOX (PCAOB-compliant) authentication		●	●	●	

Choosing Best-of Breed Middleware

Decision makers will often choose the best quality equipment to provide responsive, reliable transaction processing, business intelligence and other capabilities for their organization. For the sake of consistency, they should apply the same yardstick of quality when it comes to middleware.

Organizations that invest in top-of-the-line equipment, servers, routers, tiered storage, high-speed interconnects and clusters should apply the same quality standard to databases and software. Best-of-breed middleware delivers scalability and performance that's a match for best-of-breed server software.

About the Author

Ken North is a consultant, author, speaker, industry analyst, software developer and company founder. He teaches Expert Series seminars and is the publisher of SQLSummit.com, WebServicesSummit.com and GridSummit.com. Ken was Contributing Editor for *Internet Computing*, *Web Techniques* and *Dr. Dobb's Journal*. He wrote the "Database Developer" column for *Web Techniques* and *Dr. Dobb's Sourcebook* and was XML and Web Services Editor for *Dr. Dobbs Journal*.

Ken has consulted and spoken at conferences and seminars in North America, South America, Asia and Europe. He was conference chair for the NEXTWARE conference and for the XML DevCon conference series in Europe and North America. He has organized technical content for several conference producers, including Penton Media, SIGS and Camelot Communications.

Ken wrote *Database Magic with Ken North* (Prentice Hall) and *Windows Multi-DBMS Programming* (John Wiley & Sons). He developed APIBench, the SQL API benchmarking suite and contributed to *Dr. Dobb's Database Development: Tools and Techniques* (R&D Books). He was a technical reviewer for *JDBC Database Access with Java* (Addison-Wesley) and *JDBC API Tutorial and Reference: Second Edition* (Addison-Wesley). Ken's articles have appeared in dozens of publications including *Intelligent Enterprise*, *SQL Server*, *DB2*, *Business Integration Journal*, *XML*, *XML-Journal*, *Web Techniques*, *Dr. Dobb's Journal*, *The Data Administration Newsletter*, *SearchDatabase*, *Java Pro*, *Software Development*, *DBMS*, *Byte*, *PC Week*, *Windows NT*, *Network Computing*, *Windows NT Systems*, *Windows Tech Journal* Prior to founding Resource Group, Inc. in 1981, Mr. North held management and software engineering positions with TRW and Computer Sciences Corporation.

"Ken is *the* world's expert on interfaces to SQL DBMSs, a topic of critical importance in the integration space. "

David McGoveran

President, Alternative Technologies

Senior Technical Editor, *Business Integration Journal*

Vendor Profile

DataDirect Technologies is an independent operating company of Progress Software Corporation, a publicly traded company (NASDAQ). DataDirect is a successor to companies that helped establish the data access middleware market. It retains many of the key people from those pioneering ventures.

In 2005, Progress Software acquired NEON Systems, a leading vendor of mainframe connectivity products, and integrated the company into DataDirect Technologies. It also acquired OpenAccess Software and added its software developer kits for ODBC, JDBC and ADO.NET to the DataDirect product line. DataDirect customers include leading companies from the software industry, including BEA, Business Objects, Cognos, IBM, Microsoft, NCR, Oracle, SAS and Sybase. DataDirect licenses data access middleware to more than 300 OEM customers, who integrate it into application server products, integration servers, BI suites and other products.

DataDirect Technologies has a track record of success in the data access middleware space, retaining key personnel from predecessor companies, and exhibiting consistent growth after acquisition by Progress. The company's middleware is in use by vendors of leading products for database management, business intelligence, integration services, application services and SOA.

DataDirect today continues a long-established tradition of being a leading software vendor with important middleware products for accessing databases. The company's business model targets corporate and OEM sales to independent software vendors, SQL DBMS vendors, mainstream computer companies and customers with mainframes and distributed data. Its expertise is recognized by the computing industry and the company regularly participates in activities such as W3C working groups and Java Community Process expert groups. Because the company is involved in developing standard specifications, its software products consistently comply with industry standards.

DataDirect personnel have played a role in the development of key specifications for querying and data access, including ODBC, JDBC and XQuery.

Copyright

Ken North Computing published this report as part of an ongoing assessment of software and information technology. For more information about our consulting services, send e-mail to knc@sqlsummit.com.

© Copyright 2007 Ken North Computing, LLC. Unauthorized reproduction is forbidden. All rights reserved.