Technology Assessment from Ken North Computing, LLC

Middleware in Action

Industrial Strength Data Access

May 2007

Table of Contents

1.0 Introduction

In the modern era of computing, monolithic applications have given way to servers, clients, middleware and components. Our understanding of software and information technology (IT) is somewhat 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 a car must understand quality requirements for parts and assemblies. The same is true of people creating systems that rely on a collection of hardware and software components. Knowledge of parts and assemblies is essential for enterprise architects, network architects, system architects and software designers.

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 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*.

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.

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 that simplify development.

To manage and manipulate data, we've developed sophisticated database management system (DBMS) technology. Databases are omnipresent because they provide a robust solution for applications and services using persistent information. IDC reported the typical Fortune 1000 company has deployed an average of 14 databases and 48 applications. Gartner Research has reported 35-40% of all programming efforts are

devoted to “developing and maintaining programs to transfer information between databases”. To query databases, we often use SQL, which has been an international standard since 1986, to query databases.

Despite the appeal of Web 2.0 and AJAX (Asynchronous JavaScript and XML), not all information processing lends itself to a mashup or scripting solution. 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 above.

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.

Mature Technology

The trade press and industry pundits have put a spotlight on service-oriented architecture (SOA) and Web 2.0 (a new generation of Web technology). But enterprise architects and system architects understand the importance of industrial-strength enabling technologies for enterprise applications, web sites, mobile computing, and distributed processing. The technical community prizes innovation and we often want to

work with glamorous new technologies. Building a mission-critical application on a foundation that's completely leading edge technology is not a formula for success. To build robust software and web sites, we must not overlook tried-and-true technology.

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 many innovative applications 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, PostgreSQL and Open Ingres. Nonetheless SQL is mature technology for persistence and applications such as online transaction processing (OLTP).

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. Neither do platform providers such as BEA, IBM, Microsoft, Oracle and Sun. They've shown an overwhelming preference by shipping middleware from DataDirect Technologies with their products. More than 300 software companies that license DataDirect middleware include:

- 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.

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 was 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.

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 first phase 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.

Mature SQL platforms such as IBM DB2 and Oracle have long supported static SQL as a solution that offers performance and security. They provide for compiling and storing queries and invoking pre-compiled queries at execution time, thereby reducing query startup time. Oracle, for example, delivers static SQL performance by having PL/SQL cache cursors and reuse statements.

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, although DataDirect middleware uses bind packages when possible to provide better performance than dynamic SQL.

Scalability, Interoperability, High Availability

Applications with Windows clients accessing SQL servers 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) emphasized 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.

Components, XML and Services-Oriented Architecture

After object-oriented programming (OOP) and components became a favored solution for software development, it was inevitable there'd be a desire 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 they operate over data access middleware. Components and object libraries sometimes introduce a performance penalty that's erroneously attributed to APIs and middleware.

Widespread adoption of component and object programming fed demand for remote objects, which resulted in competing object technologies (CORBA and COM). A generation of developers found those technologies wanting and sought instead a lightweight, interoperable solution for collaborative computing.

Extensible markup language (XML) turned out to be a Swiss Army Knife solution. XML was not only a vital markup solution for web publishing and content management applications. It also provided interoperability using XML-encoded messages for purposes such as e-business applications and business-to-business (B2B) exchanges. In 2000, IBM and Microsoft unveiled a new computing paradigm that used XML-based Web services to permit the creation of collaborative applications with interoperable components. Loosely coupled components gained traction as replacements for heavyweight technologies such as COM and CORBA.

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. An SOA and Web services provide the basis for building applications as collections of distributed services. The collection of specifications that defines an infrastructure for Web services provides solutions for describing services, publishing and exposing services, orchestrating process flow, defining authentication and authorization policies, accessing resources, establishing identity and trust policies, and processing transactions.

Recognizing the potential of XML, the foremost database vendors extended their SQL platforms to provide document processing, document queries, XML messaging and Web services integration. Vendors such as IBM and Oracle provide tight integration between databases and message queues, supporting SQL databases for storing message streams and XQuery for querying them. As SQL database managers became SQL/XML platforms with document processing and data processing capabilities, data access middleware evolved to support SOA computing and a new XML data type.

The best-of-breed SQL servers became a Swiss Army Knife proposition, with rich types, extensible databases and a spectrum of capabilities for analytics, transaction processing, document processing, message queue handling and clustering. Data access middleware also evolved from a minimalist RPC solution to a vital technology that's a part of many applications, suites and business systems. And in a manner similar to the SQL DBMS marketplace, best-of-breed middleware products separated themselves from the pack.

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 interconnect and clusters shouldn't be shopping the bargain basement for data access middleware.

Not every organization does a detailed purchase evaluation of middleware. The use of inferior data access middleware is more often a sin of omission than a studied decision. Some people, even some with impressive technical credentials, don't understand the importance of data access middleware and the effect it can have on applications.

Simply stated, the core 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

A more detailed discussion of these characteristics follows in later sections of this report.

Pay Now or Pay Later

Recognizing the value of best-of-breed middleware is easier if one considers the downside of inferior middleware, which includes poor performance, inadequate scalability, inconsistent behavior, and flawed security. Best-of-breed middleware delivers scalability and performance that's a match for best-of-breed server software.

Complex, distributed applications are like an automobile. It isn't necessary to understand the workings of today's systems to use them, but it is to build them. And like an automobile, using inferior parts in distributed systems can affect reliability and performance.

Organizations with high standards for servers, routers, software, storage and other equipment should exhibit the same standard of quality for software. This is true when selecting data access middleware, the software that's a key player in the process of executing database queries.

2.0 Architectures for Distributed Computing

One characteristic of the mainframe-computing era was centralization and distributed data was rare. Data access performance was primarily a function of database design, query optimization and application programming. With today's client-server databases in multi-tier, distributed applications, the architecture has changed, and with it the demands on data access software.

When the Internet model for applications became popular, it was a period when distributed processing (in the form of client-server SQL applications) was already entrenched in many organizations. Software companies recognized a marriage of Internet computing with distributed databases was inevitable, with consequences for security, data confidentiality and scalability. Traditional enterprise software evolved and a new generation of data access middleware emerged. The new middleware scales with features such as connection pooling and load balancing. It encrypts passwords, supports strong authentication, and does not require prior installation of client libraries.

Behind-the-firewall enterprise applications didn't expose data to an audience of anonymous users. Internet connectivity means threats and vulnerabilities that call for solutions such as database security, encryption, firewalls, virus checkers, protocols for secure networking, and Kerberos for secure single sign-on. The overhead of added security measures and the need to serve Internet users forced system architects to focus on scalability and performance. Multi-tier architectures, load balancing, caches, tiered storage, replication and database clusters became tools of the trade.

Network latency, middleware, database design, server performance and the distribution of workload are factors affecting distributed computing performance. Computer hardware is inexpensive so IT decision makers tend to throw hardware at the problems. A more comprehensive approach will include putting system architecture, network topology, communication protocols, software, data models, design processes, application requirements, programming techniques and load testing under the microscope.

2.1 Leveraging Infrastructure

Programmers who developed software a few decades ago linked programs with libraries that provided basic capabilities, but developing systems was primarily an exercise in creating software from scratch. There were few published standards, little reuse, minimal interoperability and no software component industry. The architecture, design and development of systems today are quite different.

Off-the-shelf software, re-factoring, components, standards, and interoperability have become part of the lexicon of CIOs, CTOs, IT managers, enterprise architects, system architects and software developers. Instead of starting from scratch, it's possible to leverage an infrastructure that's a product of decades of experience of software and systems development. When creating systems and software today, it's prudent to examine how application requirements and capabilities align with industry standards and available technology.

Creating a successful blueprint for a new system involves knowledge of architecture (hardware, network and software), specifications and standards. Turning the blueprint

into reality also involves an understanding of frameworks, software products and components that can provide a foundation.

To provide context for a detailed look at data access middleware, this section explores architecture and other aspects of the environment in which it operates. It's easier to evaluate middleware features when you understand their importance and their place in the overall scheme of things.

2.2 Multi-Tier, N-Tier Architecture

The classic model of a multi-tier or n-tier architecture involves resource sharing and partitioning logic across several layers or tiers. It provides a separation of presentation, application logic, Internet connectivity and data services. The multi-tier architecture includes a presentation layer or client tier, a data layer or database tier and a layer containing business rules or other application objects.

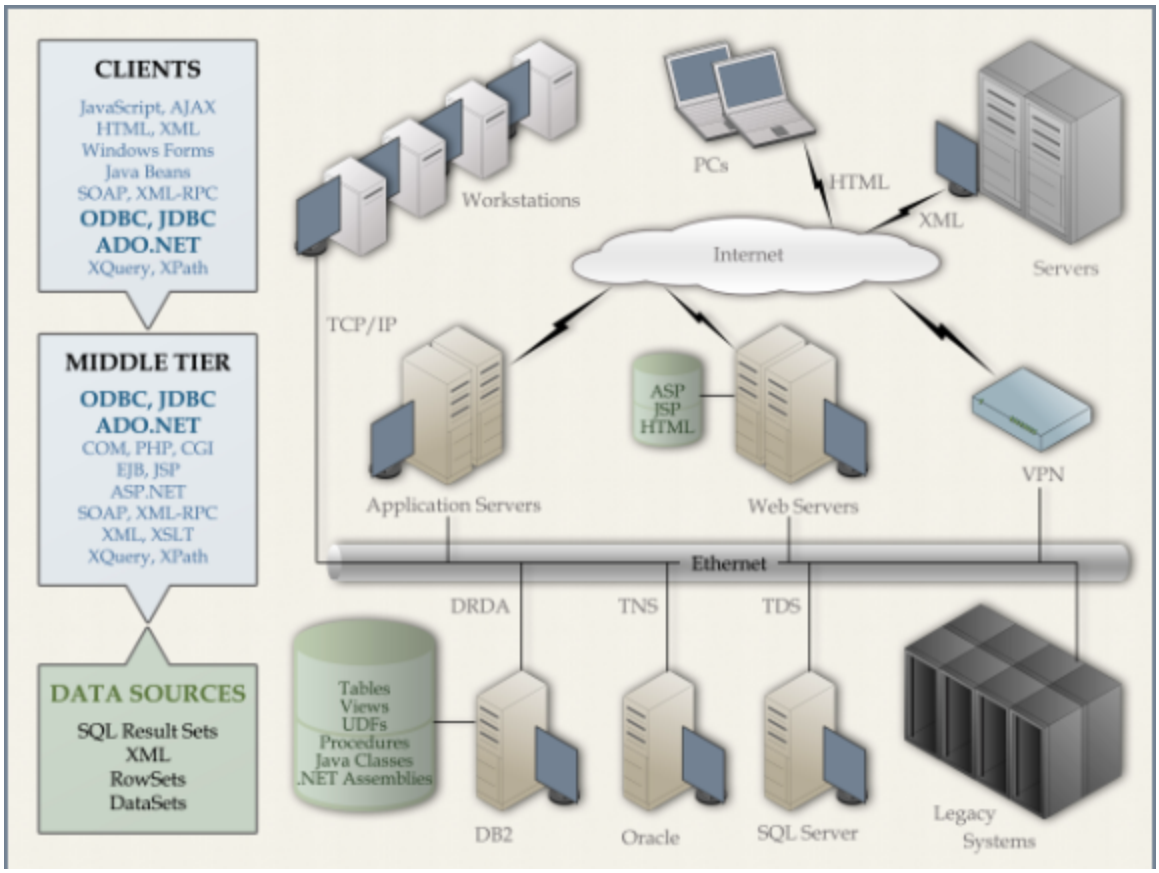

Figure 2.1 Multi-tier Distributed Application Architecture

N-tier architectures can exploit multiple tiers of specialized servers, including web servers, application servers, integration servers, print servers, FAX servers, transaction servers and database servers. An e-business transaction, for example, may originate at a consumer's PC or a partner's B2B server, cross the Internet, be handled by a Web server, integration server or application server before it produces a query request from an SQL/XML server.

That type of activity can involve multiple hops across the Internet. Data access middleware plays an important role in the connection between database servers and middle-tier servers that are database clients.

2.3 Persistence, Client-Server Databases, Distributed Data

Databases became prominent decades ago as a persistence solution supplanting ad hoc data stores. SQL databases have proliferated in organizations of all sizes and in mobile, desktop, workgroup, department, and enterprise settings.

A diversity of SQL platforms led to the development of the international SQL standard. The SQL standard also included a standard application programming interface (API) for embedded SQL. As computer hardware prices dropped, it became attractive to add dedicated computers to networks to act as database servers. Database development moved past the monolithic application model with SQL servers capable of servicing query requests from multiple clients.

The desire to simplify client programming for disparate SQL databases provided motivation for the Open Database Connectivity (ODBC) and JDBC ™ specifications, and for the Call Level Interface (CLI) defined by the SQL standard. When Microsoft undertook development of the .NET Framework, it included data access technology known as ADO.NET. The advantage to the software developer of using these APIs is abstraction by data access middleware. A driver or data provider (the tool for database connectivity) hides nitty-gritty details of processing SQL queries in a client-server environment.

Client-Server SQL Processing

The fundamental concept of the client-server SQL architecture is distributed processing, with the server receiving and processing query requests from clients. A database client sends a query request to a server and receives a response with data or status information.

The set of rows produced by a query, the result set, is also known by names such as record set, data set and row set (depending on the client language and API). With the SQL standard evolving into the SQL/XML standard, XML schemas are also a part of the picture. Java provides an example. The JDBC WebRowSet object includes query results in XML format, with a WebRowSet schema that conforms to XML schema annotations.

The notion of client-server SQL has evolved to encompass middle-tier servers acting as database clients. Likewise yesterday's SQL servers have evolved into today's SQL/XML database servers that support both SQL and XQuery processing.

The client and database server are part of an analog or digital network, such as a local or wide-area network. They exchange information, such as submitting and responding to query requests, by using remote procedure calls (RPCs). The information exchange in client-server SQL processing follows steps defined by standard and proprietary communication protocols. A communication protocol defines the processing to provide connectivity and the data flow across different layers of network architecture.

Basic communication protocols and standards define how to physically connect to networks, uniquely address computers, assemble and exchange message packets. They specify message header formats, checksums, blocking and de-blocking, acknowledgement, error checking and so on. There are standards for protocols that provide transport mechanics, such as the Internet's Transmission Control Protocol (TCP) and User Datagram Protocol (UDP).

Operating at a higher level are protocols for application services, such as the File Transfer Protocol (FTP), the Post Office Protocol (POP), Simple Mail Transport Protocol (SMTP), Hypertext Transfer Protocol (HTTP) and Simple Object Access Protocol (SOAP). SQL clients and servers can communicate using standard transports such as TCP/IP. For distributed SQL processing, they use application-level wire protocols. These platform-specific protocols can flow over lower level transports such as TCP/IP, NetBIOS and IPX/SPX.

For application-to-application service functions, Oracle created the Transparent Network Substrate (TNS). TNS provides a common interface for protocols such TCP/IP, Named Pipes, SDP and TCP/IP with SSL. It provides open, close, send and receive functions and it supports encrypted, sequenced message digests.

For client-server communication, Microsoft SQL Server uses the Tabular Data Stream (TDS) protocol that originated at Sybase. The Sybase Adaptive Server products also use TDS but the Microsoft TDS and Sybase TDS protocols are not interoperable. Microsoft SQL Server 2005 and Sybase Adaptive Server Enterprise 15 do not, for example, support the same data types. Microsoft and Sybase have each made changes to their version of the protocol since TDS 4.2, the last version supported by both platforms.

TDS is a token-based protocol that operates over a connection-transport and provides presentation, session and application services. Client requests can contain multiple SQL commands and server responses can contain multiple result sets. TDS supports authentication, login negotiation and redirection, encrypted passwords. It returns information to clients in a table-oriented format that's self-describing. TDS is a half-duplex protocol that supports variable length Protocol Data Units.

To promote database interoperability, IBM created a Distributed Relational Database Architecture (DRDA) that defined client-server protocols. DRDA originally operated over proprietary IBM SNA networks. IBM contributed the DRDA specification to The Open Group for standardization. The Open Group released a version 3 DRDA standard in 2004 and today DRDA works with TCP/IP networks.

The DRDA Distributed Data Management (DDM) architecture provides the definition of the commands and responses used by a client (Application Requester in DRDA) and a DRDA Server. DRDA defines data access in the form of a Remote Unit of Work, a Distributed Unit of Work or a distributed request. A unit of work is a single logical transaction or SQL statement. A remote unit of work affects one remote location, whereas a distributed unit of work can read or update data on multiple DBMS platforms with each statement directed at a specific platform. A distributed request can access multiple DBMSs for each SQL request. It supports joins and unions across system boundaries and insertion queries with data from other sites. DRDA version 3 uses the Open Group XA+ standard for distributed transactions.

Although these wire protocols can operate over standards-based transports, they differ in implementation details such as message and data value encoding. They do share common traits such as asynchronous query processing and providing SQLSTATE and SQLCODE information for SQL requests. An understanding of these wire protocols is essential to the development of best-of-breed data access middleware.

Client Libraries

DBMS vendors supply client libraries that implement proprietary data access APIs, such as Oracle Call Interface (OCI). They also supply a network library that provides session layer capabilities, such as net-lib for Sybase; and Net 8 and Oracle Net for Oracle databases. The network libraries provide the implementation of the application layer wire protocols discussed above.

DataDirect Connect middleware includes a class of drivers and data providers that are capable of communicating using the wire protocol. A DataDirect wire protocol driver for Sybase, for example, can emit TDS packets so it does not require client libraries.

Multi-Database Access

Data access middleware is also vital for applications and services that require multi-database access, such as connecting to Oracle 10*g* and Sybase Adaptive Server Enterprise (ASE). Both platforms offer client libraries that implement proprietary APIs. A developer writing a multi-database application would have to understand multiple proprietary APIs if he or she is not using a vendor-neutral API, such as ODBC, JDBC or ADO.NET.

Distributed Data, Distributed Transactions

Gartner Research's findings that organizations typically have 14 databases is one reason some applications, such as transaction processing, can involve distributed data and disparate databases. Organizations involved in mergers, acquisition or consolidation can inherit production applications that use heterogeneous SQL databases. It's not unusual for a multi-national organization to have different databases at the workgroup, department, division, region, and enterprise level.

Technologies such as federated databases, data replication, data integration, and distributed transactions play a role in addressing issues related to distributed data. Distributed transaction solutions, such as the Open Group XA standard, provide behavior we expect from local transactions, maintaining the ACID properties of atomicity, consistency, isolation and durability. Advanced database managers, data access middleware and application servers support distributed data and distributed and local transactions.

Federated Data

Federated data and the federated database provide a unification of data from multiple sources that enables us to operate on it as a single, logical database. We are able to define integrated views across distributed, diverse data sources and we can run queries without having to merge multiple databases. A federated data server provides location transparency and capabilities such as schema mapping, type mapping and global views.

Leading SQL vendors such as IBM, Microsoft and Oracle support federated data technology. SkyQuery is one of the largest applications of federated data using Microsoft SQL server. Oracle BI Suite provides enabling technology for queries over federated data. IBM DB2 federated technology and IBM WebSphere Federation Server provides access to Informix, Oracle, Microsoft SQL Server and Sybase data. To access federated data, standard JDBC programming techniques can be used in session EJBs.

A number of other companies offer software products for federated data access and analysis. These include Actimize, Actuate, Attunity, BEA, Sybase, Callixa, CopperEye, Decision Support, Extensio Software, MetaMatrix, SAS and XAware.

Virtual Organizations

Virtual organizations include partnerships, initiatives, and collaborations sharing resources in a dynamic manner. One example is the VOTES project in the UK, a clinical virtual organization that supports clinical trials and epidemiological studies. Virtual corporations include companies, such as Dell Computers, that depend on business partners to provide major parts of their supply chain. Other companies make most of the components of a Dell computer and Dell integrates the parts into its product.

Federated data and grid computing are enabling technologies for virtual organizations, which also have requirements for authorization, authentication and accounting. The trend towards virtual organizations will increase demand for federated data technology and middleware that provides data security and heterogeneous data access.

2.4 Data Integration

Because computing today is a global phenomenon and many organizations have distributed data, data integration consumes a fair share of IT resources. The IT budget of many organizations includes funds for purchasing or developing software to aggregate and integrate data from multiple data sources.

> "35-40% of all programming efforts today are devoted to developing and maintaining programs to transfer information between databases".
> Gartner Group

Data access middleware that provides multi-database connectivity helps simplify data integration problems. In addition, the W3C Extensible Markup Language (XML) is a widely adopted solution that permits the exchange of self-describing data. The capability to import XML is *de rigueur* for the newest generation of integration software and database servers. DataDirect middleware simplifies integration by augmenting traditional SQL processing. It includes the option of persisting SQL query results as XML files.

2.5 Application Servers, Platforms

Whereas monolithic applications were once dominant, today developers build applications, even mission-critical computing, with partitioned logic and distributed computing. Partitioning distributes presentation, data access and application logic across multiple tiers of computers. This created a new category of specialized server known as an application server.

Application servers are a container for application logic, excluding presentation logic. They act as a client in SQL client-server operations and they typically offer pluggable components, message queues, load balancing, caching, transaction coordination, authentication and authorization.

There are several classes of application servers. Mainframe and midrange systems, such as z/90 and AS/400 systems, are in use as application servers. Microsoft has embedded application server functionality in Windows Server 2003. Microsoft supports the .NET Framework on Windows Server 2003, and pre-installs it in Windows Vista.

Vendors of the top six commercial Java EE application servers bundle DataDirect drivers with their product. DataDirect drivers are also certified for the JBoss Java EE application server.

There are a number of open source projects and software vendors that offer Java Platform, Enterprise Edition (Java EE) application servers. The latter group includes IBM, BEA, Oracle, Adobe, Sun and Borland. Application servers typically have a requirement to connect to databases, sometimes to heterogeneous databases. For that reason, all of the leading commercial suppliers of Java EE application servers bundle DataDirect JDBC drivers with their product.

For several years, Linux has growing as a server operating system for many enterprise applications. The leading database server and application server products have been released for various flavors of Linux, as have ODBC and JDBC drivers.

2.6 Internet Architectures, Web Commerce

The Internet has become the backbone for a variety of computing activities, including web-facing applications such as portals and commerce web sites. Departmental applications may have dozens of databases users. Enterprise applications, such as ERP suites and business intelligence, may have hundreds or thousands of users. But Internet-based order processing, portals, online services and other web-facing applications may have tens of thousands of users.

- April 2006: Almost 90% of U.S. Internet users at work used a broadband connection
- November 2006: Netcraft reported the Internet had grown to 101,435,253 web sites
- November 2006: Nielsen NetRatings reported Internet use was 1,493 pages viewed per month per person.
- December 2006: Global Internet users exceeded 1.076 billion (source: *World Internet Stats*)
- December 2006: Nielsen NetRatings reported the audience for online shopping exceeded 30 million per day on four days of the November/December 2006 holiday period.

There is still great potential for growth of Internet applications and web commerce. Internet use among the population of Europe is below 40%. Asia has 378 million Internet users, but that figure is only about 10% of the population, so there's great potential for growth in Asia. Likewise greater broadband penetration in Europe is likely to spur

significant growth of the Internet audience.

The increase in Internet use and Web commerce is staggering, with many web sites accessing SQL/XML databases. Besides Internet businesses, many traditional brick-and-mortar companies have expanded with a substantial web presence. That expansion often involved integrating web commerce sites with back office systems and legacy databases.

Web 2.0 applications and mashups using Asynchronous JavaScript and XML (AJAX) have been in the spotlight. Mashups involve bringing together information from disparate web-accessible information sources. The technology also has potential for new classes of enterprise applications, such as using Google Maps to visualize business intelligence data. The emergence of Web 2.0 applications will undoubtedly contribute to the importance of SQL/XML databases and data access middleware.

Providing 24/7 availability and rapid response to thousands of concurrent users requires technology such as clusters, load balancing, data replication, caches, failover and high-performance databases and middleware. Database design and query optimization often receive attention when fine-tuning mission-critical applications, but data access middleware can have a significant effect on query performance and the online customer experience.

The explosion of Internet and integration activity places a premium on data access middleware that's scalable and suitable for high-availability computing.

2.7 Mobile and Wireless Computing

Laptop computers, wireless networks, smart phones and handheld devices are pervasive and they've accelerated the growth of distributed computing. Besides persistent local data, the mobile computing user often requires data from and exchanges information with SQL databases.

Mobile devices such as laptops do not operate with a continuous network connection. Therefore integrating mobile clients often involves support for disconnected processing, with the ability to mirror, synchronize or replicate data when a network connection is re-established. Mobile computing can also introduce a need for distributed transactions that involve queries against two or more different databases.

Mobile mashups, growing demand for mobile computing services and sheer numbers will drive growth of the data access workload for the foreseeable future. Therefore all parts of the performance equation, database tuning, query optimization and data access middleware, should represent the optimum solution for peak performance.

2.8 Grid Computing

IBM, Oracle, Sun and other computer vendors have been promoting adoption of grid computing technology to aggregate and share the resources of clusters and disparate networks. The computing power of a grid can be brought to bear for applications requiring a computational grid or a data grid. The former focuses on sharing CPU cycles of many machines, whereas the latter has a focus on shared data sources. A data grid can employ federated databases, distributed data repositories and middleware to integrate data from disparate sources.

One of the organizations pushing the leading edge on grid computing is the Open Grid Forum, a consolidation of the Global Grid Forum and Enterprise Grid Alliance. Its working groups have developed specifications for cross grid authorization policy, resource allocation agreement protocols, security profiles, file transfer, data description, data access and other grid functions.

The Database Access and Integration Services WG (DAIS-WG) has developed a family of specifications for accessing and integrating data from relational (WS-DAIR) and XML (WS-DAIR) data sources. The WS-DAIR specification defines interfaces based on standards such as SQL and the Java Community Process WebRowSet.

2.9 Services-Oriented Architecture (SOA)

Component-based development and distributed computing technology have evolved to enable the creation of collaborative applications using distributed, interoperable services. A services-oriented architecture (SOA) permits us to build applications using services on an intranet or the Internet, using XML-based messaging to communicate between computers.

IBM and Microsoft introduced the XML Web services concept in 2000. The software industry and open source community have embraced the technology for SOA. During the early adoption phase for SOA, the emphasis was on laying a foundation. Web services use standard protocols and XML messaging to communicate between service providers and service consumers.

Services that use XML for plumbing leverage a variety of specifications from the World Wide Web Consortium (W3C) and the Organization for the Advancement of Structured Information Standards (OASIS). These include W3C XML Schema, SOAP, XML Query Language (XQuery), XML Encryption, XML Namespaces and XML Digital Signatures. OASIS published WS-Security, the Security Assertion Markup Language (SAML), the eXtensible Access Control Markup Language (XACML), and a variety of security-related specifications. It has also published SOA-related specifications, the Web services Business Process Execution Language (WSBPEL) specification and the Universal Description, Discovery and Integration (UDDI) services registry.

Developers can create applications that are composites of service consumers, accessing remote methods of services running on distributed servers. Applications that consume services can access a registry to determine how to connect to a service and use its methods. To provide security, developers can employ opaque encrypted keys, security tokens, Kerberos tickets, X.509 authentication, security policies, digital signatures, roles, and trust relationships.

A web service can exchange XML-encoded information using the XML-RPC and SOAP protocols. The basic Web services stack operates over Internet protocols, TCP/IP and HTTP. Above the basic protocol stack, Web services employ Web services Description Language (WSDL) for describing the abstract functionality of services and UDDI for service registration and discovery. There are a variety of specifications, frequently called WS-*, that define higher-level web service features. These include specifications to define security, policy assertions, notification, reliable messaging, orchestration, trusted exchanges, and transactions.

Today enterprise computing has evolved to exploit Web services and SOA, such as evolving Enterprise Resource Planning (ERP) suites and offering business intelligence services. SQL servers, for example, expose stored procedures as Web services. SQL products from IBM, Microsoft and Oracle provide an XML type, SQL/XML functions and XQuery for processing documents and XML-encoded messages.

Stateful Resources, Grid Services

The Web services Resource Framework from OASIS is a collection of specifications that provide a framework for modeling and accessing stateful resources using Web services. The framework includes the WS-BaseFaults, WS-ResourceLifetime, WS-ResourceProperties and WS-ServiceGroup specifications. They provide a basis for maintaining state information during interactions between service providers and service consumers.

Stateful resource management is useful for managing state for grid computing applications. The leading initiative to create specifications for a grid computing infrastructure and grid services has adopted the Web services stack, including SOAP and WSDL. It has also adopted the technology embodied in the Web services Resource Framework.

SQL, XML and XQuery play an important role in Web services, grid services and SOA. Many services and collaborative applications operate with persistent data and use SQL data access technology. DBMS products provide tight integration of messaging with databases, such as DB2's functions and procedures for operating on XML in message queues. Organizations typically use SQL technology for UDDI registries, which are often used for enterprise application integration.

The SOA paradigm opens another channel to data sources and it can increase the workload for data access middleware. Because XML is the *lingua franca* of SOA and Web services, some capabilities of DataDirect data access middleware are worth noting here. There are DataDirect drivers that provide SQL queries over XML data using standard SQL client APIs. In addition drivers for SQL databases can persist query results as XML data.

Enterprise Service Bus

The Enterprise Service Bus (ESB) architecture enables disparate applications, services and component software to communicate through a common messaging bus, usually implemented with a messaging technology such as Java Messaging Service (JMS).

The genesis of the Enterprise Service Bus (ESB) concept was the software industry's focus on services-oriented architecture and accessing services as components. The ESB provides a flexible solution for integration in an SOA environment. It provides a backbone or channel for data from Web services, message-oriented middleware, Java EE applications, .NET applications and adapters for legacy systems. An ESB provides transformation, XML support, intelligence routing and end-to-end connectivity.

Java EE provides a solution for bi-directional connectivity between application servers and enterprise information systems. The J2EE Connector Architecture (JCA) provides standard interfaces for integrating enterprise software with Java EE applications, Enterprise Java Beans (EJB) and Java Message Service (JMS). JCA provides resource adapters that can make data available for integration efforts, including SQL data from JDBC data sources. As shown in figure 2.3, a DataDirect JDBC driver can provide a JCA adapter solution for plugging into an ESB.

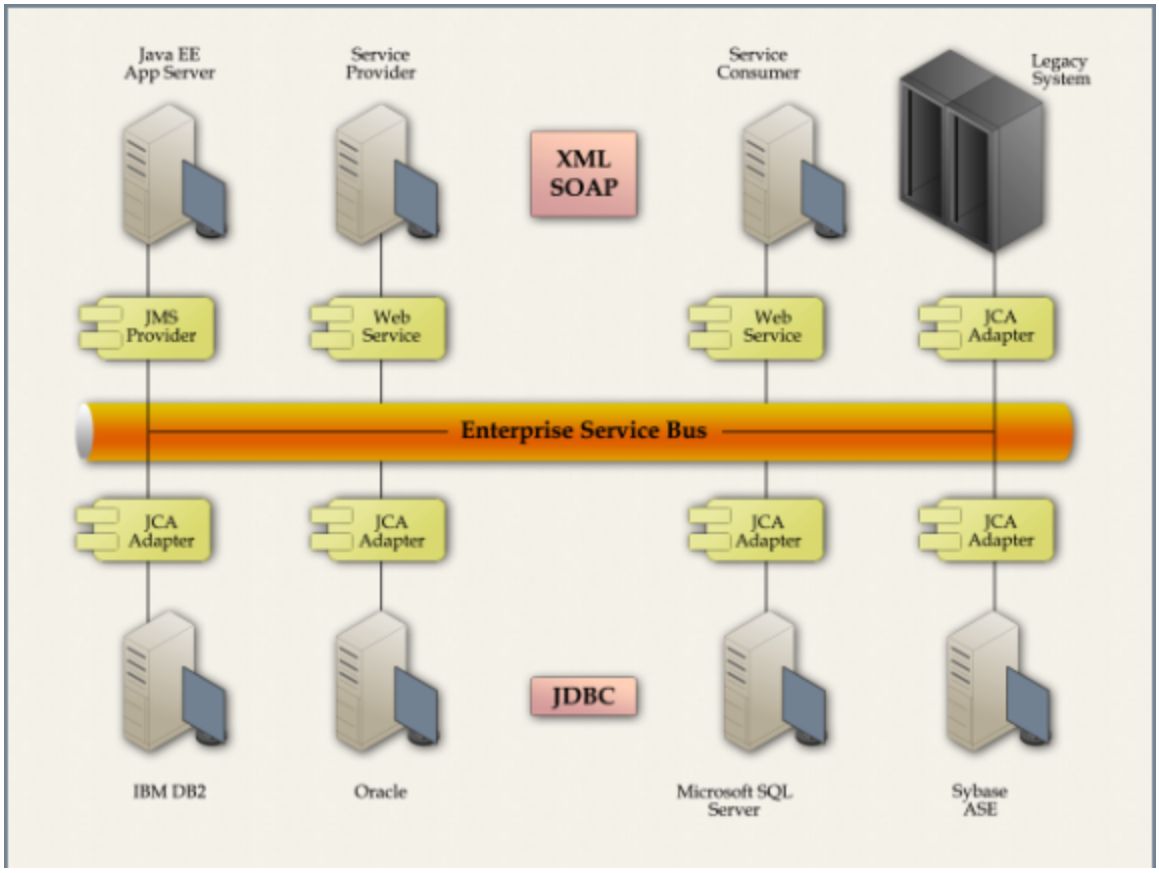

Figure 2.3 Enterprise Service Bus connects applications, services and components

Business Process Management (BPM)

Recent initiatives have spurred an interest in business process management (BPM) and the automation of business processes. BPA technology will often be the vehicle for creating new services deployed in an SOA.

The newest generation of business process automation (BPA) technology includes several XML vocabularies. These languages, such as the Business Process Execution Language (BPEL), permit system architects and developers to use a declarative approach to process automation.

A surge of BPM and SOA adoption will increase demand for software as a service and application servers, both with a need for persistent information and data access middleware.

2.10 Security, Privacy and Compliance Issues

Businesses and customers want transactions to be safe and secure. Citizens and officials want private information in government databases to remain private. Although distributed processing has scalability advantages, it also requires solutions for security, privacy, data integrity and regulatory compliance. A checklist of requirements influencing the architecture and design of today's information systems will place security and compliance near the top of the list.

Because data access middleware provides a conduit for access to databases, it must be capable of operating in a mode that restricts access to data. Recent years have seen the introduction of regulations affecting public corporations, healthcare providers, financial institutions and organizations that operate with government data.

Regulations associated with a long list of laws are driving an interest in information security, encryption and secure networking:

- Gramm-Leach-Bliley Financial Services Modernization Act
- Health Insurance Portability and Accountability Act (HIPAA)
- Payment Card Industry (PCI) Data Security Standard
- Sarbanes-Oxley Act (SOX)
- California Data Breach Notification Act
- Basel II
- Federal Information Security Management Act (FISMA).

Organizations respond to compliance issues with a variety of measures, including audits and stronger information security measures. One example is a large pharmaceutical manufacturer that adopted strong auditing of databases. In creating a data compliance database, the auditors mandated logging of all data model changes and deletions for more than 1,000 Oracle and SQL Server databases.

Defense in Depth

Because of the many threats to information systems and database security, organizations have adopted a defense in depth strategy. They set authentication and access policies for regulated data, including the use of firewalls and strong authentication. The strategy also includes policies and technologies for protecting regulated data when transferred across a network (IPsec, wireless security, encryption).

The default model for exchanging messages between SQL servers and database clients doesn't provide protected communication. When secure messaging is necessary, it's possible to use solutions such as IPsec connections, secure sockets, encryption and X.509 certificates.

The major SQL platforms support secure communications between clients and database servers. You can specify encrypted communication using:

- Microsoft SQL Server Service Network Utility
- Oracle Advanced Security, Access Manager, Web services Manager
- Informix encryption communication support module (ENCCSM), crypto package for JDBC (com.informix.jdbc.crypto)
- IBM DB2 DRDA V8, SSL V9, z/OS Application Transparent TLS
- Sybase Adaptive Server RPC security model B

The move to Internet computing caused an evolution of data access middleware so it plays an important role in a 'defense in depth' strategy. The new middleware supports secure sockets, encryption, decryption, trust managers, and single sign-on.

In a secure environment, accessing data sources using a driver or data provider requires authentication. DataDirect Connect offers encrypted communications during the challenge-response authentication process. It has supported SSL for some OEM drivers, and integrated SSL support into the latest release. Beyond that, DataDirect middleware for DB2 uses bind packages that provide performance and security.

Authentication and Authorization

When the Internet computing model gained traction, many organizations opted to integrate client-server SQL databases with web applications. Software companies recognized exposing databases to Internet required a focus on secure networking, data confidentiality and user authentication. Internet-related security threats continue to evolve, forcing a continuous cycle of security improvements to firewalls, browsers, web servers, SQL database management systems, application servers, Web services and data access middleware.

The authentication and authorization process can protect passwords in the challenge-response model. It can also employ strong authentication or two-factor authentication. The two-factor approach combines two proofs of identity, such as using a digital certificate with a password. Security product vendors have introduced hardware tokens, fingerprint readers, smart cards and other devices to support strong authentication.

Many financial organizations have moved to strong authentication due to the increased risk of phishing attacks. A variety of hardware tokens or USB devices are available for this purpose. ETrade has been giving customers RSA Security SecurID tokens, for example. Some tokens do not require specialized software to work with web browsers and support SSL/TLS communications with Web sites. Smart cards and hardware tokens have been a price-sensitive solution, but they are gaining favor due to price reductions (A $5 hardware token was recently announced by Entrust, Inc.)

Operating systems, applications and database managers support authentication and authorization of users to prohibit unauthorized access to programs, data or other resources. When distributed computing involves the use of limited resources or restricted data, identity and authorization become important. Organizations establish account policies for users, groups and roles that determine who has access to regulated data. The major DBMS platforms support restrictions on access and authority, such as specifying authority to execute stored procedures.

The Java platform includes a set of APIs known as the Java Authorization and Authentication Services (JAAS). It permits an organization to use an access control policy for authorization of users, groups and roles. JAAS supports single sign-on and it works with the Java Naming and Directory Interface (JNDI), Kerberos and operating system authentication. Single sign-on frees database administrators from having to synchronize user IDs and passwords across disparate databases.

Microsoft security strategy includes strong authentication, policy-based management of digital certificates and smart cards, and technologies such as IP Security (IPsec). Microsoft's forthcoming Identity Lifecycle Manager is an enhancement of Identity Integration Server with technology from the acquisition of Alacris. It enables Microsoft networks to employ strong authentication technology and manage certificates and smart cards.

Regulatory compliance and malware, such as key loggers that capture passwords, have motivated organizations to adopt strong authentication techniques when applications access confidential data. To supplement or supplant passwords, network architects are specifying solutions such as hardware tokens, smart cards and Kerberos authentication.

Kerberos

The Kerberos protocol for authentication was developed at MIT and adopted by the Internet Engineering Task Force (IETF). Kerberos is in widespread use for Windows platforms, Mac OS X, Cisco routers, the Apache web server, Web services and other software. The latest version, Kerberos Network Authentication Service Version 5, is IETF RFC4120.

Kerberos supports authentication of Web application users and operating system users. Kerberos and authentication at the operating system level provides flexibility for data access middleware. When they are available, middleware should exploit offering an alternative to requiring additional logins for database access.

Not all drivers login based on operating system authentication. Oracle, for example, provides Windows Authentication with its JDBC-OCI driver, but not its thin JDBC driver.

Grid Security Infrastructure

Grid services and Web services that access confidential data or restricted resources must implement security. The Globus Alliance has defined a Grid Security Infrastructure (GSI) that uses public key cryptography and certificates for verifying identity. GSI authentication supports single sign-on with X.509 certificates and SSL for its mutual authentication protocol. Communications between service providers and service consumers employ a time-limited proxy certificate.

Encryption

Transmitting confidential information in a readable form across a network is a formula for disaster. Data traveling across the Internet, for example, passes through multiple servers. Message packets are viewable using a network sniffer whether transiting local or wide-area networks.

When a customer orders a product over the Internet, credit card information must be protected during the dialog between the web server and the customer's browser, and between the servers processing a credit card transaction. Likewise access to the data must be restricted when it is stored in a customer database. Encryption is mandatory when the data is in transit and a recommended practice when it's in a database or other data store.

Encryption is cost effective. A study by Gartner Research reported a large differential in the cost of preventive measures versus the cost of recovering from a security incident. Organizations spend annually approximately $6 per user for encryption tools and $16 per user for intrusion prevention software. The annual cost of recovering from a data breach was reported to be $90 per user.

There have been numerous reports of data breaches involving theft of laptops and databases being stolen or compromises. Any organization exposing databases to the Internet or permitting staff personnel to store confidential information on laptops should take measures to protect data from unauthorized access. Besides protecting data stores, it's important to secure information while it is in transit. To protect against malicious hackers and rogue employees, it's prudent to encrypt passwords and confidential data.

Algorithms

Many Federal government agencies, contractors and organizations that exchange information with the government adhere to Federal Information Processing Standards (FIPS). *Security Requirements for Cryptographic Modules* (FIPS 140-2) specifies security requirements for cryptographic modules used to protect sensitive information (U.S) and Designated Information (Canada), but not classified information.

There have been 500+ implementations of the SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512 algorithms certified by the National Institute of Standards and Technology (NIST) as conforming to FIPS 180-2, *Secure Hash Standard*. Some implementations were certified for general-purpose processors. Others used specialized hardware such as dedicated processors and application specific integrated circuits (ASIC).

FIPS 186-2 with Change Notice 1, *Digital Signature Standard* is the U.S. government standard for the Digital Signature Algorithm (DSA). There have been 205 implementations certified by NIST as conforming to the DSA specification.

The NIST published the *Advanced Encryption Standard* (FIPS 197) for strong encryption. It specifies the Rijndael algorithm as the FIPS-approved symmetric encryption algorithm. The AES algorithm can use keys of 128, 192 and 256 bits to encrypt data in 128-bit blocks. Since 2002, NIST has certified 498 AES implementations. AES was developed as a successor to *Data Encryption Standard* (FIPS 46-3), which supports uses 56-bit keys for 64-bit blocks. DES encryption is suitable for export products because it does not violate U.S. export controls on encryption technology.

The Internet Engineering Task Force (IETF) develops standards for Internet protocols and charters working groups focusing on improving security and encryption. The cryptography community continues to refine solutions for encrypting data, including developing new encryption and decryption algorithms. Software that implements encryption algorithms is provided by commercial products, open source projects, the Java EE framework and the .NET Framework.

. NET and Java Cryptography

The Java platform includes an API for employing cryptography in Java programs. The Java Cryptography Extension (JCE) is an extensible framework for providing Java security using plug-ins known as JCE providers. JCE supports asymmetric and symmetric ciphers, block ciphers, stream ciphers, key generation and storage, digital signatures and Message Authentication Code (MAC) algorithms. JCE providers are analogous to JDBC drivers. A Java program uses a JCE provider for a specific capability, such as generating a digital signature or message digest. JCE providers are available from IBM, Oracle, Quest, Sun and RSA.

Because DataDirect Connect middleware for Java uses JCE providers for encryption, it provides the flexibility to adapt to changes in cryptography. The JCE is a modular approach and it's possible to change providers if, for example, an algorithm is cracked (e.g., SHA-1).

The .NET Framework provides support for symmetric encryption, asymmetric encryption, hashing, and digital signatures. Developers can use classes in the .NET Framework cryptography namespace (System.Security.Cryptography.Pkcs), as well as Windows Cryptographic Application Programming Interface (CryptoAPI) functions, Windows Vista CryptoAPI Next Generation (CNG), CAPICOM component and WinTrust.

IP Security

Applications that use an Internet transport for message packets containing confidential data can use several protocols designed for secure communications. IP Security (IPsec) includes protocols that operate at the network layer. It provides for authentication and encryption of message packets.

IPsec uses an Authentication Header (AH) and Encapsulating Security Payload (ESP) that can use block cipher algorithms (CBC-mode). The ESP can use the Advanced Encryption Standard (AES) and Triple DES algorithms for encryption, and SHA-1, AES and MD5 for authentication. The Authentication Header (AH) uses a combination of algorithms for authentication (HMAC with the SHA-1), and it can also use AES and MD5.

There are IPsec implementations for AIX, Cisco IOS, HP-UX, IBM z/OS, Linux,

OpenBSD, Solaris, Windows platforms and other major operating systems.

Secure Socket Layer, Transport Layer Security

Further up the OSI stack from IPsec, the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols use public key cryptography for secure communications, such as secure login with a web browser. TLS and SSL provide for encrypted data transmissions and endpoint authentication. Integrated support for SSL/TLS is available with browsers (Internet Explorer, Mozilla, and Firefox), web servers (Microsoft IIS, Apache), application servers and products such as Oracle Access Manager, Oracle Internet Directory.

By using Public Key Infrastructure (PKI) cryptography, both ends of a network connection can be authenticated. An SSL/TLS client connects to an SSL/TLS server that provides information the client needs for authentication, such as an X.509 certificate. The client and server exchange session keys and end the authentication dialog. Then the client and server exchange messages based on symmetric encryption keys established during the authentication process.

TLS grew out of SSL 3.0 but the two protocols are not interchangeable. A TLS and SSL conversation involves an exchange of message segments that can be encoded using a Message Authentication Code (MAC), which for TLS is generated with a keyed-hashing algorithm (HMAC). SSL and TLS messages include 64-bit sequence numbers to preserve the integrity of the message, which are used in computing the MAC. The TLS connection state specifies a compression algorithm, MAC algorithm and encryption algorithm. For specifying encryption, the other essential information includes the key size, hash size, how much of the key is secret, whether it's a block or stream cipher, and the block size (if a block cipher is being used).

The .NET Framework, Enterprise Java and Linux enable developers to deploy applications using SSL/TLS authentication. The tools required for secure communications with Linux servers are part of the standard Linux distribution. For SSL/TLS communications, Linux machines typically use the OpenSSL library, which is also used with AIX, Solaris, OpenBSD and FreeBSD. Organizations using the Apache web server can support SSL/TLS by installing the `mod_ssl` module and getting certificates from a certificate authority such as Verisign, Thawte or Entrust.

Java Security

Java application servers from BEA, Oracle, IBM and other vendors are configurable to use SSL/TLS, to use JCE Providers, and work with SSL hardware accelerators, such as nCipher. An Enterprise Java Beans (EJB) container can use several methods for authenticating users, including HTTP basic authentication, HTTP digest authentication, HTTPS client authentication, and HTTP form-based authentication. HTTP basic authentication and HTTP form-based authentication offer the option of using SSL client authentication. Java EE application servers also support the use of an SSL Accelerator between the browser and web server.

The Java Secure Socket Extension (JSSE) provides Java APIs for using the SSL and TLS protocols. JSEE includes the javax.net, javax.net.ssl, and javax.security.cert packages that support encryption, endpoint authentication, public key certificates, key and trust managers, with classes for processes such as creating and configuring sockets and HTTPS connections. The DataDirect Connect for JDBC drivers use the JSSE APIs for supporting SSL communications.

Windows .NET and SSL/TLS

SSL and TLS functions are built into the .NET Framework (System.Net), in addition to integrated cryptography support. Microsoft Internet Information Server supports SSL/TLS connections and Windows Server 2003 x64 supports kernel-mode SSL processing. Windows developers can use WinInet or Winsock for SSL 3.0 support. Besides built-in SSL/TLS support, .NET developers can use the Windows Communications Foundation (WCF) and Windows CardSpace, which supports SSL/TLS with smart cards.

WCF uses Kerberos for authentication of domain users, but uses different techniques for credentials of Internet users. Typically this is an X.509 or SSL certificate exchanged between services and clients, or service providers and service consumers. WCF also provides support for federated security with trusted domains and a Security Token Service.

DataDirect Connect for ADO.NET supports encryption based on .NET cryptography, whereas the ODBC drivers use the CryptoAPI.

Hardware Acceleration, Co-Processors, Smart Cards

The computing trade press raised awareness of the need for encryption and decryption solutions with extensive reporting of information security failures. To mitigate the performance penalty from encryption and decryption, many organizations are looking for hardware solutions. The industry has responded with secure routers and switches, expansion cards, chips and custom integrated circuits (ASICs).

Adding SSL/TLS processing to a client-server message exchange introduces a performance bottleneck. Server authentication is part of the initial handshake between the client and server, which commonly involves RSA decryption. The server performs a compute-intensive decryption of a private key, often 1024 bits, in order to establish a secure session. The key cryptography consumes the most CPU cycles during SSL/TLS processing.

Because SSL/TLS authentication can be CPU-intensive, performance boosters can provide better response times for data access. Products such as the nCipher server expansion card can handle 10,000 SSL/TLS connections per second.

Protocol processors (hardware accelerators for SSL and TLS processing) are available in various forms, including box, board and chip-level products. Using an add-in PCI board for a Pentium 4, researchers have achieved RSA 1024-bit encryption rates of 36 KB/sec, AES encryption throughput of 75 MB/sec, and SHA-512 throughput of 499 MB/sec.

Doing encryption or decryption with a general-purpose processor eats CPU cycles, so some companies have responded with innovative motherboard designs. AMD announced the Torrenza Innovation Socket initiative that provides for a two-socket motherboard capable of supporting a plug-in encryption co-processor. Ethernet adapters and smart card products sometimes include a coprocessor to support encryption and decryption.

In terms of boosting performance to mitigate the overhead of security processing (SSL authentication, and encryption), organizations should embrace best-of breed solutions when it comes to networking hardware, middleware, and SQL database managers. An application switch, for example, can forward traffic at layer 2 speeds by integrating routing and switching using information from layer 4-7. Network security switches serve several functions, including firewall protection, routing, and MAC address filtering.

Another compelling reason for using co-processors and hardware accelerators is recent cracks by researchers performing cryptography processing using general-purpose computers. Doing encryption with PCs and commodity servers introduces vulnerabilities because there's the potential of malware monitoring the encryption process and deducing keys based on execution cycles. Researchers have shown popular open source SSL/TLS software, for example, is vulnerable to this type of attack. They've also uncovered side channel attacks that have been successful against RSA.

DataDirect Connect middleware that operates with SSL has been included with DBMS vendor software. It's been used to connect to SSL-enabled database instances, such as Oracle and IBM DB2. DataDirect will soon be releasing middleware for Linux, UNIX and Windows that include SSL capabilities.

The authentication segment of the computing market currently supports more than 50 different products for client authentication.

Virtual Private Networks

Organizations can use virtual private network (VPN) technology to provide secure connections between client computers and the organization's servers. Even though a VPN can communicate over a public network, the data is protected by encryption. This enables, for example, a remote Internet user to obtain a secure connection to a corporate server.

An effective VPN solution provides user authentication and authorization, multi-protocol capabilities, address management and encryption key management for clients and servers. Multi-protocol capabilities enable SNA and IPX network traffic to be tunneled over the Internet and other networks that use a TCP/IP transport.

VPNs use a variety of protocols, including IPsec, Secure Sockets Layer (SSL) and Transport Layer Security (TLS). There are issues that affect IPsec and the number of tunnels a VPN can support. Hardware switching and packet size, for example, have an impact on the number of encrypted packets per second. There are VPN products available from an assortment of networking suppliers, including Broadcom, Cisco, D-Link, Juniper Networks, Nortel, and Nokia.

DataDirect Connect middleware has been tested with a variety of protocol processing hardware and encryption accelerators.

Enterprise architects and system architects want solutions that provide information security without sacrificing performance and scalability. For data protection, it's important to deploy SQL database managers and middleware with integral support for security. The DBMS and middleware should also integrate seamlessly with security solutions such as operating system authentication, firewalls, hardware accelerators and transaction auditing tools.

Besides using hardware acceleration for encryption and SSL/TLS processing, system architects and database developers should look to optimizations available with databases, application servers, operating systems and middleware. Section 3 discusses best-of-breed data access middleware characteristics, including performance and scalability.

Packages

DB2 packages are control-structure database objects that reside in system catalog tables. The access plan or strategy for executing each SQL statement is stored as a section in the package. DB2 uses packages for authorization so they are useful for restricting access to queries that operate with confidential data. The owner of a package must have privileges for every action a package performs. If the owner's privileges to any objects in the package are revoked, the bind process must be repeated. DBAs and system administrators can grant or revoke the privilege to execute a package.

DataDirect Connect middleware can use existing packages and create or bind packages. It can also modify bindings when needed.

Auditing and Tracking

Database protection can benefit from a combination of hardware and software solutions. Products such as Crossroads database protection appliance (StrongBox DBProtector) provide intrusion detection, compliance auditing and inspection of database activity.

Audit policies may call for tracking whom accesses regulated data, recording resources used, IP addresses and protocols. DataDirect Connect middleware includes utility software that can provide added detail to supplement access logs. Each JDBC driver includes DataDirect Spy, which provides detailed logs of bi-directional communication during JDBC query operations. For the Connect for ODBC and SequeLink for ODBC products, a trace file capability is also available for Windows and UNIX ® environments.

3.0 Data Access Middleware

The role of a driver or data provider is to handle low-level details of querying SQL databases, thereby simplifying SQL client data access. Over time SQL database vendors have provided a variety of options for database application programming. The initial offerings included embedded SQL and proprietary (native) programming interfaces. Eventually they supported *de jure* and *de facto* API standards, including the SQL/CLI standard, ODBC, JDBC and the ADO.NET interfaces.

The advantages of standard APIs include abstraction and the potential to write portable, interoperable database clients. A program written to Oracle Call Interface (OCI), for example, will not work for accessing DB2, Sybase or Microsoft SQL Server. However, there's ODBC, JDBC and ADO.NET middleware for accessing all of those databases that offers greater potential for writing interoperable clients.

The processing of a query with ODBC, JDBC and ADO.NET involves multiple software layers that include a networking stack. That's also true with proprietary APIs which deploy client libraries, network libraries and other components. The separation of database server, database client and middleware components provide opportunities to address reliability, scalability and performance problems by using best-of-breed software that's protocol compatible.

SQL data access middleware is often overlooked in the performance equation. Although it's mature technology, middleware is still a developing, competitive software market. Because SQL DBMS vendors offer free drivers and data providers, best-of-breed middleware is sometimes overlooked until applications have performance and reliability problems. Some organizations that have invested thousands of dollars to tune databases and applications can testify best-of-breed middleware was instrumental in resolving performance problems.

It's rather curious when an organization looks for premium quality when selecting a router, DBMS or mass storage supplier, but not when selecting data access middleware. Considering that today's applications consist of networked, linked parts, system architects should remember an old proverb. "A chain is only as strong as its weakest link" certainly applies when selecting data access middleware.

3.1 Architecture

Data access middleware, whether proprietary or best-of-breed software, is not a monolithic application or service, but a collection of components. The infrastructure for database clients consists of multiple parts. Depending on the type of driver or provider, some of the parts must be installed on clients.

On a Windows platform, vendor-supplied clients and ODBC install dynamic link libraries (DLLs). Linux, Solaris, AIX and UNIX platforms use shared libraries. Other parts of the infrastructure include caches for connections, statements, metadata, result sets and cursors.

Parts, Components, Infrastructure

In common usage, data access middleware is a reference to a class of software and not a specific program library. The infrastructure for data access includes a variety of components and routines that are typically organized as libraries. For Windows platforms, the shared code objects reside in dynamic link libraries (DLLs). For UNIX and Linux platforms, the shared objects reside in dynamically linked libraries known as shared libraries. Because multiple processes share the libraries, middleware vendors must ensure versioning, reentrancy and thread safety.

Depending on the type of middleware, other software may be required to sustain connectivity between the database server and client. Some drivers and data providers use client libraries supplied by the DBMS vendor to support network communications or proprietary application programming interfaces. Not all drivers or providers are of this type.

Some data access middleware operates with a runtime environment, such as the .NET Common Language Runtime (CLR) and Java Virtual Machine (VM) that executes Java byte code. Not all data access middleware requires a Java or .NET runtime environment. However, when it is present, the DataDirect Connect middleware exploits the environment to extract benefits such as improved security and connection pooling.

Data access middleware that includes native code does not have the inherent security advantages of Java classes and .NET trusted code.

In today's world, we look for protection against malware and hostile code. The Java environment, for example, provides a sandbox security model that operates hand in hand with the Java VM. Java limits what code can be executed and it provides a class loader, bytecode verifier, garbage collection, array bounds checker and other features that are desirable in network environments containing un-trusted code.

Similarly the .NET Framework environment provides controlled execution and a security model that differentiates between trusted and un-trusted code. The .NET Common Language Runtime (CLR) provides a managed execution environment that provides an array bounds checker, garbage collection, type safety. This permits the developer to use programming languages that target the runtime and facilitate the production of managed code assemblies. Some ADO.NET providers use native code client libraries, but DataDirect Connect for ADO.NET includes only managed code, with instances of the classes created on the managed heap.

Vendor-neutral middleware for SQL data access gained increased mindshare with developers starting in the 1990s. Today middleware implementing the Open Database Connectivity (ODBC), JDBC™ and ADO.NET programming interfaces have gained widespread acceptance.

Components for ODBC Data Access

ODBC is ubiquitous, in use on Windows platforms, Mac OS X, Linux, Solaris, AIX, HP-UX and other platforms. ODBC provides a C-style (function) call level interface for developers using a variety of languages, including C, C++, Perl, PHP, Python, Smalltalk, Tcl, Visual Basic, COBOL and FORTRAN.

Figure 3.1 illustrates that distribution of middleware across an architecture that separates database clients and servers supports deployment of protocol-compatible drivers. This permits using best-of-breed software to ensure reliability, scalability and performance.

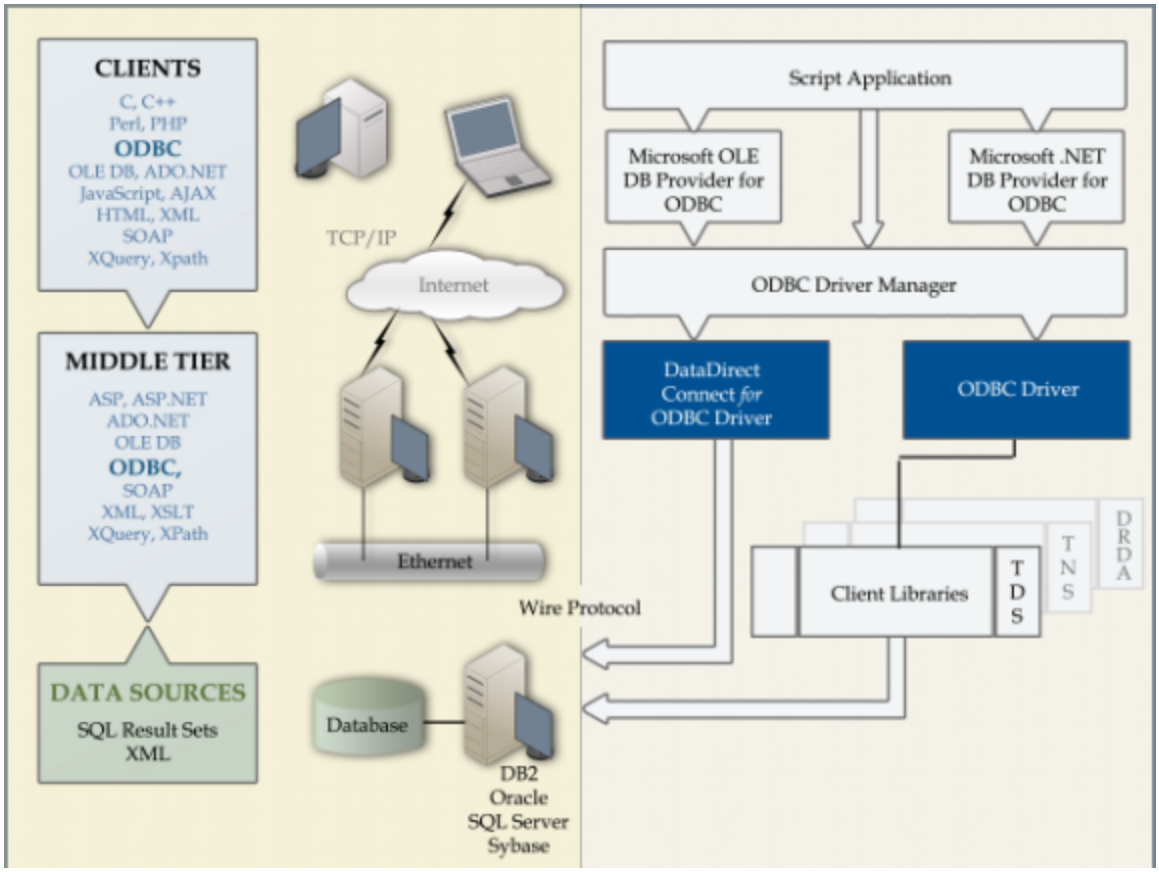

Figure 3.1: Multi-tier Architecture for ODBC Data Access

As shown in figure 3.1, developers using OLE DB or ADO.NET may also need to bridge to ODBC, such as when there is no OLE DB or ADO.NET provider for a data source. For OLE DB development, Microsoft offers the OLE DB Provider for ODBC. It also offers the .NET Data Provider for ODBC. The OLE DB Provider for ODBC and .NET Data Provider for ODBC are unsupported software.

> For ADO.NET applications, DataDirect recommends using ADO.NET providers rather than using the .NET Data Provider for ODBC with an ODBC driver.

Environments such as .NET and Java offer code security, but ODBC remains important because it's:

- Language neutral (used with programming and scripting languages)
- Vendor and platform-neutral (operating systems and DBMS)
- The vehicle offering access to more disparate data sources than any other connectivity solution
- Aligned with the SQL/CLI standards published since 1995.

An ODBC installation includes the ODBC Administrator, one or more drivers, and the ODBC Driver Manager. Depending on the installation process, it may also include the definition of one or more data sources. Some drivers require the installation of client libraries and network libraries (figure 3.1). Wire protocol drivers do not require those libraries, which simplifies administration and application deployment.

Components for JDBC Data Access

The JDBC specification defined several types of drivers for connecting to SQL data sources, some that require client libraries and some that do not. Some JDBC driver vendors implement extensions to augment the standard JDBC classes. JDBC includes a DriverManager class and a bridge capability for using ODBC drivers. The JDBC-ODBC Bridge enables a JDBC program to access an ODBC data source when no JDBC driver is available for that type of data. The type-4 drivers are capable of emitting the appropriate wire protocol without using a DBMS vendor-supplied client library or networking software (figure 3.2).

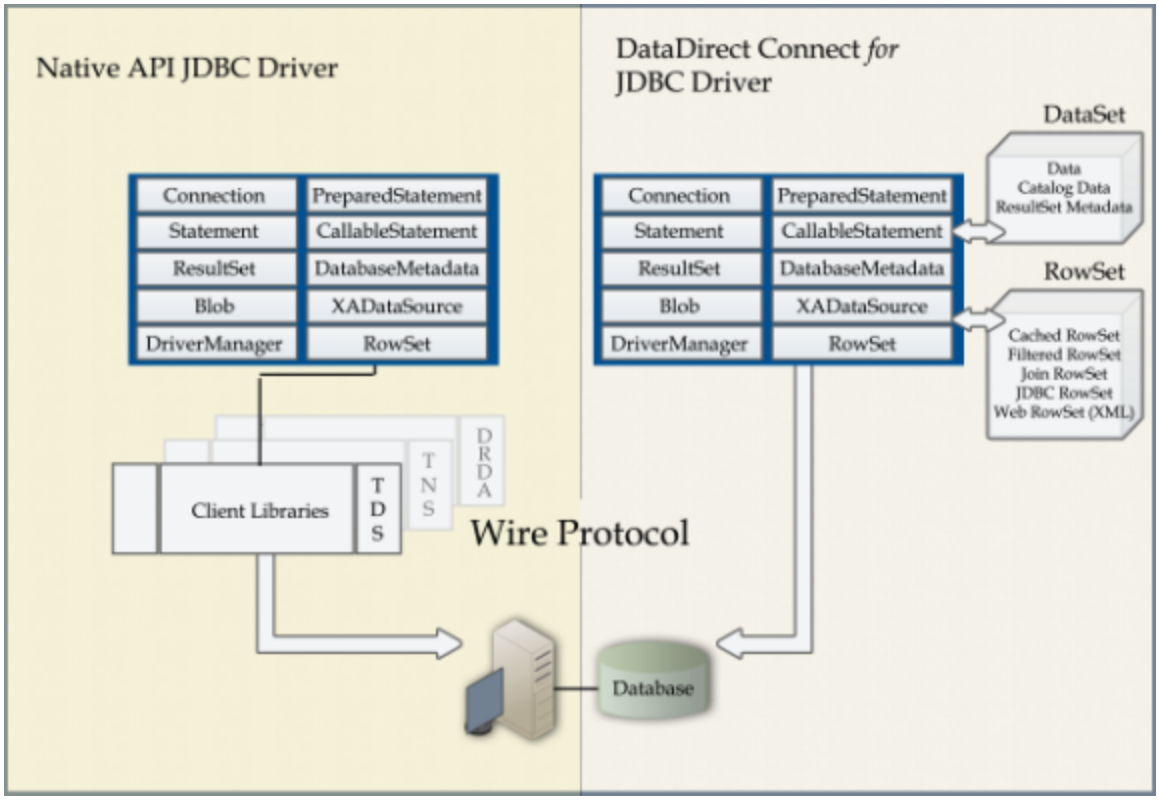

Figure 3.2: Architecture with JDBC Native API or JDBC Wire Protocol Drivers

JDBC programs execute using the Java VM so a complete installation includes a Java Runtime Environment (or Java SDK). You can bind JDBC data sources to a JNDI naming tree and associate it with a connection pool. If you're using a native API (type-2) JDBC driver, you'll also need client libraries.

Components for ADO.NET Data Access

Like ODBC and JDBC, ADO.NET supports tiered architectures (figure 3.3) using middle-tier and database servers running Windows Server, Linux, HP-UX and other operating systems.

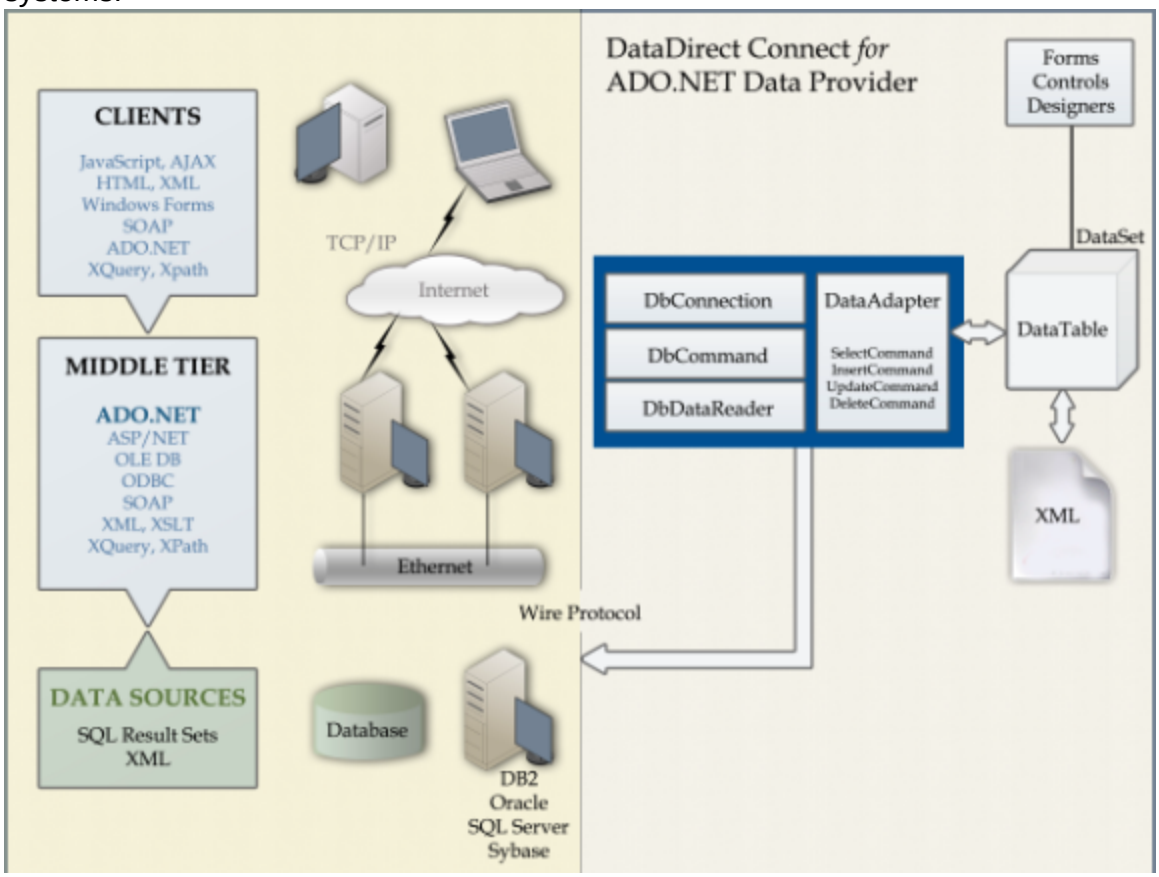

Figure 3.3: Tiered Architecture with ADO.NET Providers

Like ODBC and JDBC, this architecture enables an organization to build and deploy applications using the best choice of technology. This includes using best-of-breed ADO.NET data providers that are protocol compatible without using client libraries from DBMS vendors

The DataDirect ADO.NET data providers do not require the installation of client libraries (as shown in figure 3.4). The prerequisite is an installation of the .NET Framework, including the Common Language Runtime (CLR).

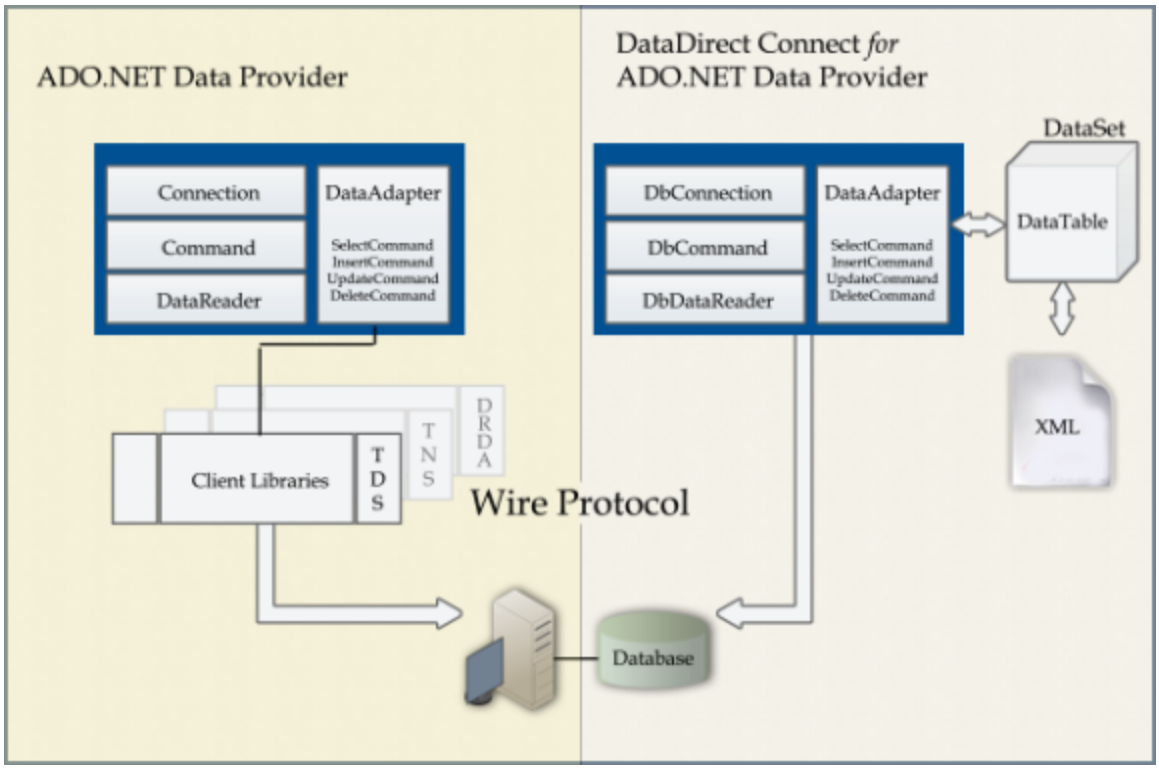

Figure 3.4: Comparative Architecture of ADO.NET Data Providers

The DataDirect .NET providers

- Consist of managed code assemblies
- Require no pre-installed client software
- Operate as shared assemblies in the Global Assembly Cache
- Deploy using ClickOnce technology
- Exploit the Native Image Generator (NGen).

Microsoft created ClickOnce to make deployment of a .NET application comparable to loading a web page. ClickOnce creates a self-updating Windows application that executes in the secure sandbox provided by the CLR Code Access Security model.

Caches

Most users of a web browser recognize it stores web page data in a cache to reduce network traffic. Database developers have long been encouraged to use caches to reduce unnecessary network round trips. The database community uses several terms for caching techniques. Prepared statements and statement caches refer to preparing and caching statements to reduce query startup time. The same is true of connection pools, connection pooling and connection caches. These types of caches can have a significant effect on performance and their use is a key characteristic of best-of-breed middleware.

3.2 Requirements for Best-of-Breed Data Access Middleware

The characteristics of best-of-breed middleware include consistency, performance, scalability, security, maturity and compliance with standards. It must be robust, reliable and should provide ubiquitous connectivity and tools support.

Selecting a JDBC Driver
When deciding which JDBC driver to use to connect to a database, you should try drivers from various vendors in your environment. In general, JDBC driver performance is dependent on many factors, especially the SQL code used in applications and the JDBC driver implementation.

From the BEA reference *Programming WebLogic JDBC*

Connectivity, Platform Support, Character Sets

Many organizations have diverse systems with multiple applications and heterogeneous databases. A study by IDC reported a typical Fortune 1000 company has deployed an "average of 48 applications and 14 databases". Many data center managers have adopted server virtualization technology to support multiple operating systems.

Broad connectivity and multi-platform support are essential to qualify as best-of-breed data access middleware. DataDirect Connect middleware provides drivers and data providers for 32-bit and 64-bit operating systems, including AIX, HP-UX, Linux, Solaris and Windows (including Java and .NET platforms). The Connect middleware product line is in use with a variety of chip sets (AMD Opteron, HP PA-RISC, IBM PowerPC, Intel x86, Intel Itanium).

Best-of-breed middleware must operate with diverse client-server protocols flowing over a network, providing the capability of adjusting packet sizes. It must also offer broad support for SQL and DBMS features implemented by IBM DB2, Informix, Microsoft SQL Server, Oracle and Sybase. It must also "understand" the data types of IBM DB2, Informix, Microsoft SQL Server, Oracle and Sybase.

Data access middleware must provide robust support for multiple character sets and locales, including Unicode. There have been known issues for certain locales, for example, with the JDBC driver from a major database vendor.

Organizations using Java EE technology want premium data access middleware that provides support for the Java Connector Architecture (JCA), Enterprise Service Bus (ESB), Enterprise Java Beans (EJB) and Java Naming and Directory Interface (JNDI). Windows platform and .NET users want comparable support for Active Directory in order to establish enterprise-wide policies and identities and manage resources and accounts. In an Oracle environment, premium middleware will support use of the TNS Names file (*tnsnames.ora*). It includes parameters, such as net service name, buffer sizes, and session data unit (SDU) size, used for database connections.

Scalability and Performance

The computer trade press has periodically reported scalability problems and outages having a financial impact of millions of dollars in losses or fines. A large stock brokerage paid $5 million in fines because of a scalability problem. A day of heavy trading overwhelmed the brokerage's systems and they could not post all customer trades that day. Internet businesses such as eBay, Amazon.com, Salesforce.com and Orbitz have also experienced high-profile outages, including several attributed to database failures.

The Internet and cheap storage have changed the measuring stick for scalability and high-availability. A popular web site (MySpace.com) has 20 million users and enterprise databases routinely run to terabytes. The NASDAQ web site has as many as 20 million page hits per day on a day of busy trading but it operates at greater than 99.9% uptime.

In order for software and hardware solutions to be scalable, they must be robust enough to work effectively and not fail when operating in peak load conditions. The same is true of premium, or best-of-breed, data access middleware. It must scale effectively from a few users to thousands of users. It must operate effectively as an integral part of architectures that provide scalability solutions such as load balancing and high-availability clusters.

Since the advent of SQL client-server databases, the developer community has found several techniques that make for scalable, performance-oriented database applications. Caches, cursors and connections play a significant role and their intelligent use is a defining characteristic of best-of-breed data access middleware.

For performance reasons, a program might use a forward-only cursor for scrolling through result sets. For a forward-scrolling cursor, DataDirect Connect middleware will use a cache for pre-fetching multiple rows. This can reduce network traffic and boost performance. Another important performance solution is asynchronous query processing. It increases productivity because it enables users to multi-task during the processing of long-running queries. Asynchronous query processing is important for applications operating with very large and federated databases.

JDBC developers can use threads, stateful session beans and message-driven beans to support asynchronous SQL queries. For programs using ADO.NET 2.0, the `SqlCommand` class exposes several methods for invoking commands asynchronously.

Best-of-breed middleware supports the cancellation of queries and a capability to increase or decrease network packet sizes for maximum performance.

Business analytics and BI users often execute long-running queries over very large databases. For business analytics applications, best-of-breed data access middleware provides asynchronous query execution and cancellation.

DataDirect Performance Wizards inquire about query characteristics before making recommendations about packet size and the number of result set rows in the pre-fetch cache. A larger pre-fetch cache size requires more memory but reduces network traffic.

Caches and Connection Pools

Some of the earliest solutions for performance with client-server databases are even more applicable today. Since the advent of SQL client-server databases, developers have invented a series of techniques for writing scalable, performance-oriented client software. Techniques recommended for ODBC, such as using caches, have been refined with Internet computing, JDBC and ADO.NET.

In the early stages of client-server database development, programmers were often left to their own devices to formulate solutions for performance and scalability. Today, however, performance-oriented techniques such as connection pooling are provided by application servers and best-of-breed data access middleware.

Connection Pools

When using a client-server database, some processes are expensive, such as opening a connection and scrolling through large query result sets. Some queries are inherently more expensive than others.

To reduce the overhead and startup time for query processing, developers can cache connections, or re-use connections from a connection pool. Connection pooling provides dramatic reductions in query execution time, whether using ODBC, JDBC or ADO.NET.

JDBC programs can exploit pooled connections by using performance-oriented middleware, such as DataDirect Connect, that offers a Connection Pool Manager. In that context, a JDBC program can create a DataSource object and register it with the JNDI naming service. The DataDirect Connect for JDBC driver includes a package that provides a pooled connection data source (PooledConnectionDataSource).

Java EE application servers, such as BEA WebLogic and IBM WebSphere, provide connection pooling. For example, you can use WebLogic Server Administration Console to create a connection pool for local or XA transactions and point to that pool when creating JDBC data sources.

Starting with ODBC 3.0, connection pooling became available with the pool managed by the ODBC 3.0 Driver Manager. ODBC clients can make a function call to set an environment attribute (SQLSetEnvAttr) to enable connection pooling.

Internet Information Server (IIS), Active Server Pages (ASP) and ASP.NET can use pooled connections. Connection pooling is not a part of the core .NET Framework but DataDirect managed data providers implement it. Each connection pool is associated with a connection string.

.NET Clients
DataDirect's managed data providers for .NET implement automatic connection pooling for .NET database clients.

Prepared Statement Cache

ODBC, JDBC and ADO.NET support different execution models for SQL queries, including prepare and execute processing in separate steps. Prepared statements enable the DBMS to determine the access plan for a query and store the plan for reuse. This reduces the startup time when executing a repetitive query. The SQL statement is

persistent but it executes using parameters supplied at execution time.

This type of statement is amenable to caching. A statement cache provides benefits to database client software that must efficiently manage resources and memory, whether instantiating objects or allocating memory and ODBC handles. ODBC programs allocate a statement handle associated with a connection handle. Each statement handle will have associated descriptor handles. The churn of allocating and freeing handles can be reduced. It's possible to share statement handles for prepared statements and maintain thread safety.

Java programs use PreparedStatement objects and JDBC 3.0 defined an interface for their reuse, which reduces the overhead of object management and garbage collection. The statement cache

- Eliminates repetitive statement parsing
- Reduces overhead from repetitive creation of cursors
- Is associated with a Connection or PooledConnection object.

Using the statement cache with the connection pool can improve performance at the database server and database client.

Using application servers such as WebSphere, the connection pool size and statement cache size are adjustable for JDBC programs. DataDirect middleware includes options for resetting the connection pool size.

There are differences in .NET data providers and their support for prepared statements. For example, the DataDirect Connect for ADO.NET provider implements a Prepare method of the Command object. The Command.Prepare method provides a vehicle for creating stored procedures that contain prepared queries. Developers using the ODP.NET provider from Oracle find the `Prepare` method is a no op. Registry settings or connection string attributes determine whether ODP.NET enables statement caching.

A developer using DataDirect Connect for ADO.NET can also exploit other caches to improve performance:

- Cursor Description Cache (optimize execution on the same statement)
- Procedure Description Cache (stored procedure information)
- .NET assembly caches (download and Global Assembly Cache)

Persistence, Query Results

Depending on application requirements, database developers sometimes need a persistent version of SQL query results. JDBC programs can serialize ResultSet and RowSet objects, or store query results using a vector. ADO.NET programs operating with a DataSet object can use the WriteXmlSchema method write out a schema and the WriteXml method to write to a file. A comparable solution is available for ODBC programs that use DataDirect middleware. The DataDirect Connect for ODBC drivers provide an option of using an XML data file as a persistent store for query results.

Java developers approach persistence from different perspective, with some choosing solutions such as Oracle TopLink or the Hibernate framework. In a situation where they are using an object tier over JDBC or EJB, they may cache Data Access Objects (DAO). The Spring Framework includes a JDBC-related package that enables a program to

work with query results as lists of business objects, with mapping between SQL columns and business object properties.

Dynamic SQL, Static SQL, Bind Packages

Dynamic SQL provides better performance for singular occurrence, ad hoc queries. When queries are recurring and schemas are known, static SQL offers a performance advantage over dynamic SQL. A DBMS analyzes a static SQL query syntax and type mapping, generates an access plan, and stores it for future invocation. Unlike dynamic SQL, static statements are persistent after a database shuts down.

Because static SQL statements are not compiled at run time, they don't have a startup delay comparable to the prepare step of dynamic SQL. Microsoft SQL Server supports dynamic SQL, but not static SQL. IBM DB2, Informix, Oracle and Sybase support both. DB2 supports the profiling of dynamic SQL statements and replacement with static SQL.

DataDirect middleware for ODBC, JDBC and ADO.NET (including the Shadow z/Direct adapter for JDBC) can use static SQL and DB2 packages. The drivers and providers use existing packages, create packages or modify packages. Programmers using DataDirect middleware enjoy a performance advantage because bind packages represent the optimized access plan for a query and there is no delay to compile the query.

Stored Procedures, User Defined Functions

All major SQL platforms offer a capability for database clients to invoke procedures stored in a database for execution by the database server. A principal benefit of stored procedures, reducing network round trips, is a major boost to query performance. An added benefit is security because DBAs can restrict access to stored procedures. The implementation of stored procedure processing varies from one SQL database manager to the next - for example whether a stored procedure can return multiple result sets. The native SQL syntax for invoking stored procedures varies from one DBMS to another. DataDirect Connect provides SQL leveling with ODBC and JDBC drivers and ADO.NET providers. This means a developer can write interoperable SQL queries by using escape clauses to express the query in a platform-neutral manner.

Premium data access middleware not only supports all variants of stored procedure processing, it provides metadata about procedures. The metadata includes information such as parameter types and whether a procedure returns a scalar quantity or result set.

Some SQL servers offer object-relational extensibility, including the capability of installing a user-defined function (UDF) in a database. Because a UDF is an extension to the SQL grammar, it's a part of the parsing and execution of SQL queries. Advanced data access middleware, such as DataDirect Connect, provides metadata about both system-defined and user-defined functions, including scalar functions and table functions. This enhances the capability to write interoperable SQL statements for developing portable applications.

High-Availability, Load Balancing

High-availability applications require a scalable architecture that doesn't break under peak load conditions. Among the solutions for scalability, redundancy and load balancing are quite practical in an era of commodity-priced computers. Using the major SQL database managers, it's possible to configure high-performance databases that support

parallel processing, symmetrical multiprocessing and clusters. Solutions such as Oracle Real Application Clusters (RAC) and DB2 Integrated Cluster Environment (ICE) have gained traction.

Database client applications can take advantage of a multi-server architecture to provide greater reliability. DataDirect Connect middleware supports JDBC, ODBC and ADO.NET clients in the use of connection failover and load balancing. The former enables a client to specify alternative servers to use if a connection is broken, such as when a server goes down. The latter means a system architect can specify scalability solutions that provide server load balancing and client load balancing. The DataDirect approach to failover is a platform-neutral solution (not tied to a specific DBMS), and therefore it requires no special configuration.

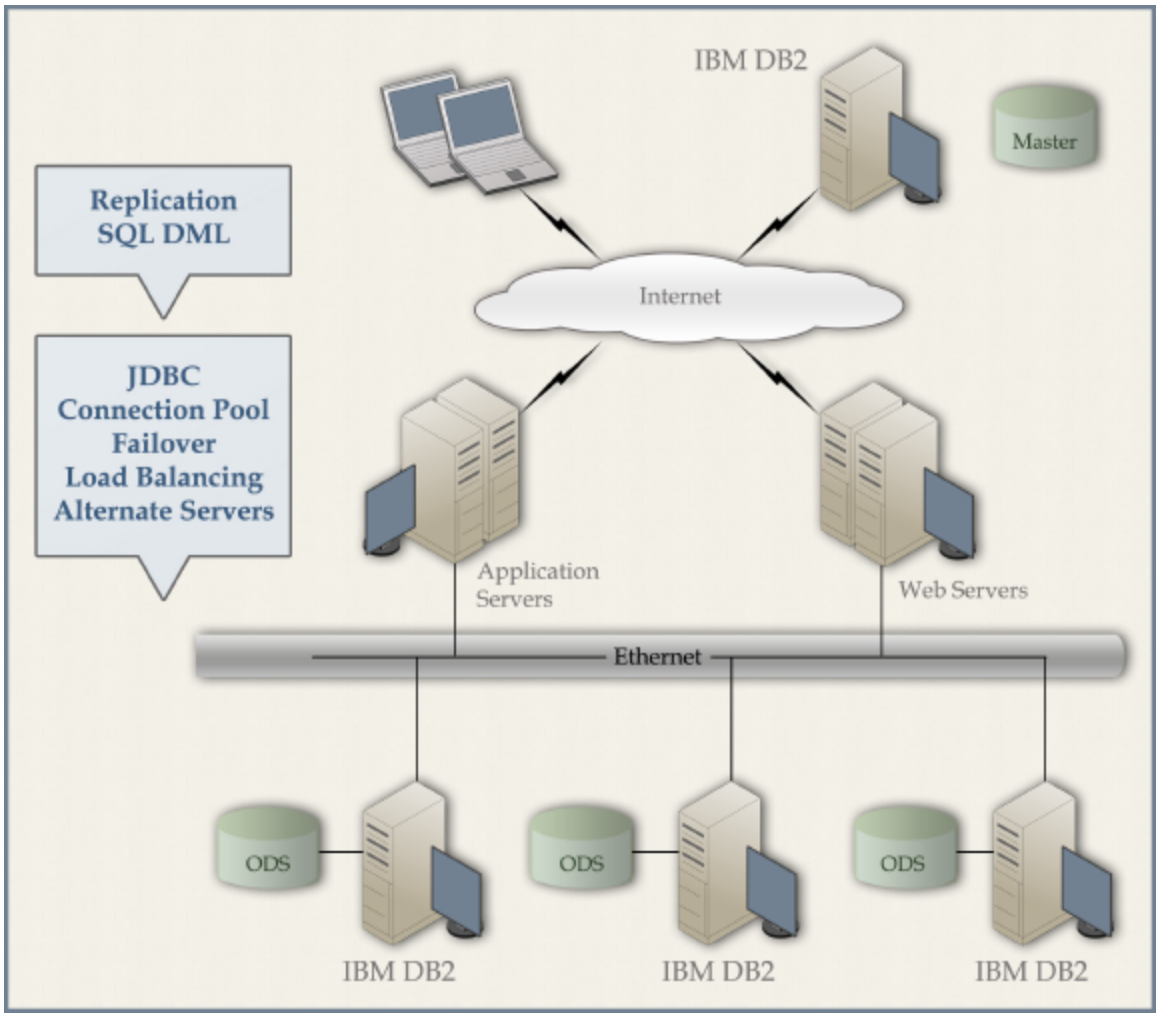

Figure 3.5: Architecture for Failover and Load Balancing

DataDirect middleware offers client load balancing with the Connect for ODBC wire protocol drivers for DB2, Informix, Oracle, Microsoft SQL Server and Sybase. When enabled, the driver will connect to servers in random order to distribute the workload. The Connect for JDBC and Connect for ADO.NET products also provide client load balancing.

Threads

Multitasking operating systems are the norm for distributed computing, for desktop, mobile, server and cluster computing. There is widespread adoption of preemptive multitasking environments, such as AIX, BSD, Linux, Mac OS X, Solaris and Windows operating systems. In a Linux environment, for example, the Linux process scheduler governs the involuntary suspension of running processes. For symmetrical multiprocessing (SMP), the Linux process scheduler uses separate run queues and implements locking for each processor. Windows scheduling is also based on preemption. The operating system divides CPU time among processes or execution threads and allocates a time slice to each thread. A thread runs for the duration of the time slice before it is preempted to execute the next thread in the queue.

To exploit the benefits of preemptive multitasking, applications can use multiple threads for better performance. Applications can be CPU bound or I/O bound, but the bottleneck for data access performance is often I/O and remote procedure calls (RPCs). Caching reduces the performance hit for I/O and querying remote databases, but multiple execution threads also provide an important boost. Preemption and multi-threading enable data access software to initiate asynchronous I/O requests and RPCs that can run concurrently.

Java applications operate in an environment with Java Virtual Machine (VM) thread management, in some instances native operating system threads. The Java environment also supports thread pools. Likewise, the .NET Common Language Runtime (CLR) manages a thread pool that enables applications to create threads.

The DataDirect JDBC drivers work cooperatively with software, such as Java EE application servers, that implements a thread pool, including using native operating system threads. The size of a thread pool is a factor in determining the number of connections available in a JDBC connection pool. The DataDirect data providers also work cooperatively with the .NET CLR thread pool, with data areas being managed per thread. In this instance, the number of pool threads also affects the number of available.

The DataDirect JDBC and ODBC drivers and ADO.NET data providers are thread safe for multi-threaded environments. The DataDirect Connect for ODBC drivers for IBM DB2, Microsoft SQL Server, Sybase, Informix and Oracle are threaded per connection. In addition, they return information about a session's thread model using the ODBC `SQLGetInfo` function.

When using a thread per connection, the driver supports concurrent requests for SQL statements with dissimilar connection handles. It serializes requests for the same connection handle.

> The benefits of a multithreaded database client are diminished if it's not using best-of-breed middleware that supports multiple threads.

Native Images

Pseudo-code machines and virtual machines, such as used by Java and .NET, provide a high degree of code portability. In some instances the overhead of byte code or MSIL results in unacceptable performance. For that reason, IBM has compiled DB2 Java stored procedures to machine code for mainframe processing. DataDirect has chosen a

similar approach for its .NET data providers. It uses NGen native optimization to create a native image from the data provider's managed assembly. Native images are loaded by the CLR like any other DLL, requiring less memory and delivering better execution performance.

Isolation Levels, Transactions, Distributed Transactions

Of the many types of database applications, transaction processing is typically the lens through which we look to evaluate performance, scalability and robustness. The major SQL database vendors have used Transaction Processing Council (TPC) benchmarks for years to differentiate their products. Application server vendors have also used transaction processing benchmarks for the same purpose.

Organizations will often select a top-of-the-line DBMS and application server when transaction processing performance and scalability are required. For consistency, they'll use the same basis for evaluating data access middleware. Scalability and performance are key characteristics of premium data access middleware that supports local and distributed transactions.

DataDirect JDBC drivers have delivered top performance on benchmarks of Java application servers (SPECjAppServer2004). They are included with leading Java application servers, which are key ingredients for Java-based transaction processing.

Besides threads, stored procedures, static SQL and other features that serve transactions well, DataDirect middleware also supports transaction isolation levels, batch inserts and updates and distributed transactions.

Isolation levels determine the degree of concurrency and locking during transactions. ADO.NET, ODBC and JDBC define programming support for multiple isolation levels, but SQL DBMSs vary in supporting isolation levels. DataDirect middleware supports SQL standard isolation levels (repeatable read, read stability, cursor stability and uncommitted read) and SQL Server's Snapshot and Read Committed with Snapshots. It also provides metadata about isolation levels and batch updates that reduce network roundtrips.

The .NET 2.0 Framework includes classes in the `System.Transactions` namespace for transaction processing. They include a `TransactionScope` object that uses the Microsoft Transaction Coordinator for distributed transactions. DataDirect Connect for ADO.NET providers support local and distributed transactions. They include a `DB2Transaction`, `OracleTransaction`, `SQLServerTransaction` and `SybaseTransaction` object.

JDBC provides explicit commit or rollback for local transactions with `Connection` object methods, `commit` and `rollback`. JDBC also provides XA-style distributed transaction support using Java Transaction API (JTA) methods. The DataDirect Connect for JDBC drivers install Microsoft SQL Server stored procedures for JTA support. They provide `XAConnection`, `XADataSource` and `XAResource` classes for transactions that span multiple data sources.

DataDirect ODBC drivers support distributed transactions with DB2, Informix, Microsoft SQL Server, Oracle and Sybase databases. The DataDirect ODBC driver for Sybase supports tightly coupled distributed transactions.

Performance Optimization with Connection Strings and Properties

One noticeable difference about DataDirect middleware has been a consistent pattern of offering options that developers can enable to improve performance.

The typical database developer is aware a connection string provides information such as data source name and user ID. Unfortunately, too many developers rely of default connection strings and don't exploit connection string attributes. In seminars I've often use the analogy that operating with ODBC and JDBC isn't comparable to flipping a light switch – it's more like tuning for the best radio reception.

Connection strings enable you to tune the connection to match your application. DataDirect driver connection string attributes that affect performance are listed in Table 3.1. In addition, at execution time a program can set or adjust object properties. For example, an application can set the RowSetSize property of the DB2CommandObject to restrict the number of rows returned by a query.

Table 3.1: Connection Strings, Properties for DataDirect Connect Middleware

Attribute or Property	Description
AlternateServers	Identifies alternate servers for failover and load balancing.
ApplicationUsingThreads (AUT)	Ensures that the driver works with multi-threaded applications.
ArraySize (AS)	The number of rows the driver retrieves from the server for a fetch.
CacheColumnInfo	Reduce the number of server round trips
CursorDescriptionCache	Enable cursor description cache for SELECT queries
DefaultLongDataBuffLen (DLDBL)	An integer value that specifies, in 1024-byte multiples, the maximum length of data fetched from a TEXT or IMAGE column.
DeferPrepare	Defer prepare request until first execution request.
DescribeAtPrepare	Determines whether the driver describes the SQL query at prepare time.
EnableScrollableCursors	Determines whether to use scrollable cursors for a data source.
EnableStaticCursorsForLongData	Determines whether to support Long columns with a static cursor.
FailoverNetworkAddress (FNA)	Specifies the address of the High Availability (HA) Failover server to be used in the event of a connection loss.
FetchArraySize	Specifies The approximate number of rows to fetch for a SELECT query.
IPAddress	Specify the address to locate catalog tables for load balancing
IsolationLevel	Set transaction isolation level
LoadBalancing	Enable client load balancing
NetworkLibraryName (NLM)	The name of the network library determines which network protocol to use.
OptimizePrepare (OP)	Determines whether to create stored procedures for calls to SQLPrepare.
PacketSize (PS)	Specifies a packet size or tells the driver to compute the maximum allowable packet size.
RowSetSize	Limits the number of rows returned by a query.
SelectMethod (SM)	Determines whether to use database cursors for SELECT queries.
SocketBufferSize	Sets the size of the send/receive socket
TightlyCoupled DistributedTransactions (TCDT)	Determines behavior of multiple connections and locks during distributed transactions.
UseCurrentSchema	Limits retrieval of schema objects to those owned by the current user.
WireProtocolMode	Determines whether to optimize network traffic
WithHold	Specifies whether to preserve cursors or delete cursors after a commit or rollback operation.

Processing such as type mapping and type coercion can affect performance, as can retrieval of columns with large object data such as images. Those are programming techniques that influence performance, not an inherent characteristic of a driver or data provider.

Load Testing

To provide performance and scalability information about data access middleware, it's prudent to perform load testing before going live with production databases and software.

To properly evaluate data access middleware, load testing should use a mix of transactions and database activity. The testing should include business intelligence and business analytics queries, to support functions such as drill-down analysis and reporting. Load testing should include lookup queries (when a single SQL statement returns a large scrollable result set), such as product catalog listings. The query mix should also include OLTP testing, with many concurrent users executing short queries.

Besides using regression testing and J2EE certification testing, DataDirect conducts load tests before releasing new middleware product versions. Whereas the other testing guarantees stability and standards compliance, load testing ensures performance and scalability.

To perform load tests, some organizations have developed their own test tools. Others use products such as:

- HP Mercury LoadRunner
- Quest Benchmark Factory for Databases
- Technovations DbSizr
- Microsoft Visual Studio 2005 Team Edition for Software Testers.

LoadRunner, for example, provides monitoring of locks, threads, page I/O, concurrent connections, cache hits and other vital statistics.

Security

In the context of data access middleware, security includes code security and data security. DataDirect Connect middleware for JDBC and ADO.NET provides both, whereas the ODBC drivers lack code security.

The DataDirect Connect for JDBC drivers are Java code that offer the benefits of the sandbox security model. For the .NET environment, DataDirect chose to use only managed code assemblies for its ADO.NET data provider suite.

As discussed in Section 2, data security requires a defense in depth strategy, including premium middleware solutions for includes authentication and encryption. DataDirect middleware provides authentication that complies with a variety of information security initiatives, including PCI Data Security, HIPAA regulations, FISMA and FFIEC. Drivers and providers offer encrypted user IDs and passwords. Users can select from authentication options, including Kerberos, single sign-on OS authentication, client authentication or database user ID/ password authentication.

DataDirect has also licensed drivers that use SSL communications. It offers encrypted data packets with its SequeLink middleware product and a recent release of its other middleware products.

Data Access Feature Checklist

When a distributed computing application supports laptops and mobile devices, there is often a need for data access middleware that supports connected and disconnected operations. In disconnected mode, data access is local to the laptop or mobile device. These applications often connect to server databases to synchronize data from occasionally connected devices.

Premium data access middleware provides the capabilities for disconnected operation that are available with the JDBC and ADO.NET programming interfaces. JDBC provides RowSet operations, including a CachedRowSet object that supports disconnected operation. DataDirect's JDBC drivers for DB2, Informix, Microsoft SQL Server, Oracle and Sybase fully support RowSet, including CachedRowSet.

The ADO.NET disconnected architecture includes the DataSet as the main data storage vehicle and the DataAdapter as the core class. The DataDirect providers include a DB2DataAdapter, OracleDataAdapter, SQLServerDataAdapter and SybaseDataAdapter that support disconnected operations.

Cursor Capabilities, Large Objects, XML Type

To provide flexibility, the ODBC, JDBC and ADO.NET APIs provide for different types of scrollable cursors. Scrollable cursor capability is a key feature of best-of-breed data access middleware. DataDirect Connect middleware supports the use of static cursors with forward and backward scrolling capabilities. It also supports keyset-driven cursors, which are typically less memory intensive.

Premium middleware will provide read and update support for the binary large object (BLOB) and character large object (CLOB) types of the major SQL platforms (DB2, Informix, Oracle). To support CLOB columns, DataDirect JDBC drivers for Sybase and Microsoft SQL Server map them to the LONGVARCHAR type. DataDirect middleware supports BLOB and CLOB columns for DB2, Informix and Oracle, DBCLOB columns for DB2, and NCLOB columns for Oracle. The DataDirect Connect for JDBC drivers operate on Oracle XMLType columns and DB2 XML columns as String, Clob, AsciiStream or CharacterStream types.

Metadata, Client Information

Best-of-breed middleware for accessing SQL databases will expose database catalog information and provide extensive metadata, including metadata defined by the SQL Information Schema (SQL:2003 standard, Part 11).

JDBC programs can obtain extensive metadata by the use of ResultSetMetaData and DatabaseMetaData helper classes. The former returns information about query results and the latter returns information about databases, including primary keys, stored procedures, hashed index information, clustered index information and so on. ADO.NET 2.0 includes metadata collections based on the SQL Information Schema. All providers include a GetSchema method and programs can get metadata that includes DataTypes, ReservedWords, Primary Keys, UserDefinedTypes and so on. DataDirect middleware supports those JDBC and ADO.NET metadata capabilities, in addition to ODBC metadata functions (SQLGetInfo, SQLGetTypeInfo).

To facilitate resource accounting, databases such as DB2 can store client information

associated with connections. DataDirect Connect middleware includes extensions that enable JDBC programs to store connection information such as host, user ID and client application name.

Adaptive Programming, Interoperability

Developers with a requirement to write portable, platform-neutral software can use adaptive programming techniques. It's possible, for example, to write interoperable SQL by using escape clauses to express query statements in a platform-neutral manner. These escape clauses are included as a part of the ODBC and JDBC specifications. DataDirect Connect middleware supports their use with its drivers, but also provides that flexibility with DataDirect Connect for ADO.NET. Therefore ADO.NET programs using the DataDirect providers will have an easier time achieving interoperability.

Another key issue affecting interoperability is data types supported by the .NET data providers. For example, the DataDirect provider does not support Oracle-specific data types. Instead it supports .NET Framework types, enabling developers to use OracleDataReader accessors that correspond to supported .NET types.

Tools, Language Support

The advantages of premium quality middleware are seen not only after deployment, but during the development and debugging phases of projects. Best-of-breed data access middleware provides tools and works with productivity solutions such as integrated development environments. It integrates with solutions for object-relational mapping, caching and decision support, for example.

A premium middleware product line will support developers using Java, .NET languages, C/C++, scripting languages such as PHP and other programming languages. It will provide optimization tools, such as the DataDirect performance wizards.

Many computing organizations have adopted technologies such as XML, Web services and SOA. For data integration and application integration projects, they will have a bi-directional flow of data between XML documents and SQL databases. DataDirect Connect middleware facilitates that flow because it provides the capability to persist SQL query result sets as XML files with embedded schemas.

To debug client-server software, developers often turn to log files and traces. For ODBC applications the ODBC Administrator provides a trace capability. The DataDirect Connect for ADO.NET providers include a trace capability using a DB2Trace object, OracleTrace object, SybaseTrace object or SQLServerTrace object.

The DataDirect Spy capability is built into every JDBC driver and a program can use the setEnableLogging method to start generating log files. The JDBC drivers also support the setLogWriter method of the DataSource, ConnectionPoolDataSource and XADataSource objects to enable tracing.

Robustness, Reliability, Stability, High Availability

Many of the deployment scenarios for data access middleware require stability and reliability suitable for 24x7 applications. To reach that level of stability, the obvious solution is to select robust middleware. A best-of-breed driver or data provider will manage resources or objects properly when statements and connections are closed. It

will be thread safe, stand up under peak load conditions and integrate successfully into a high-availability (HA) computing environment.

A key factor in determining stability and reliability is software quality assurance and testing before each new release of a driver or provider. DataDirect Technologies has comprehensive test suites for regression testing of drivers and data providers. It has refined the test suites continuously for more than 15 years, producing millions of test cases.

DataDirect JDBC drivers have passed J2EE certification testing and the ADO.NET providers passed Microsoft's proprietary ADO.NET test suites.

Administration and Technical Support

The root cause of problems with SQL applications is not limited to the database, the database server or poorly optimized queries. Middleware can be a culprit. Comprehensive technical support is important when debugging or fixing broken applications, or tuning underperforming applications.

The quality of technical support for a driver or data provider is a characteristic that differentiates best-of-breed middleware. There is a marked difference between the tech support for a premium product and one that's free. A driver from a leading database vendor, for example, includes a license that says:

> No Technical Support
> Our technical support organization will not provide technical support, phone support, or updates to you for the programs licensed under this agreement.

By contrast, DataDirect middleware offers comprehensive technical support that's been a multi-year winner of the Omega North Face award for customer support.

Network administrators, data center managers and other IT managers look at middleware from an administrative viewpoint. They look at network and system change control, group policies, auditing, versioning and other issues. Best-of-breed data access middleware simplifies software configuration management. Premium middleware enables an organization to configure servers and PCs without being dependent on bridge software or client libraries. DataDirect Connect middleware provides that independence and ease of administration with wire protocol drivers and a suite of .NET data providers that consist only of managed code assemblies.

Standards Compliance, Consistency

One factor that differentiates drivers and providers is consistency, such as exhibiting consistent behavior across different versions. There has been criticism, for example, of inconsistencies in the mapping of dates and timestamps across different versions of the Oracle JDBC driver.

To ensure consistency across versions and conformance with specifications such as ODBC and JDBC, a middleware vendor should employ regression testing before releases using a comprehensive compliance-testing suite. Since the release of ODBC in the 1990s, this has been a major advantage of the DataDirect products. Not surprisingly the DataDirect Connect for JDBC Oracle driver has been consistent across versions in its handling of dates and timestamps.

DataDirect's software is consistently standards-compliant because its key personnel contributed to important specifications. In recent years, they've participated in Java expert groups and the development of the XQuery and XQJ specifications. They've also contributed to JDBC-related specifications, and before JDBC, they helped make ODBC a success.

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.

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 procedure for prepared query		●	●	●	
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		●	●	●	

4.0 Performance and Scalability Scenarios

The need to access persistent information is often a requirement of Internet and enterprise applications, everything from desktop productivity applications to web sites and Web services. Today the user community includes millions on the Internet and large organizations have terabyte-size databases. Computing in the large, with distributed systems and or large numbers of online users, emphasizes the need for a data access architecture that delivers performance and scalability.

Because application requirements differ, the performance yardstick is not a constant. The duration of database queries, for example, is variable. An online transaction processing (OLTP) application typically involves many users running short queries. A business analytics or business intelligence application might involve a smaller number of users executing long-running queries.

In those instances, database design, hardware capacity, query optimization and network latency are not the only determinants of performance. Middleware capabilities and data access techniques can have a substantial effect on performance and scalability.

4.1 Online Transaction Processing (OLTP)

The granddaddy of enterprise applications with performance and scalability requirements is online transaction processing (OLTP). Transaction processing provided the *raison de etre* for many early information technology (IT) organizations.

Early batch-oriented applications evolved into online transaction processing (OLTP) systems. The pioneering OLTP application was the American Airlines reservation system that went online in the 1960s. The SABRE system could process 40,000 airline reservations per day using dual IBM mainframes to support users in airports and airline offices. The user base expanded over the years to include travel agents and web users, with a commensurate increase in transaction volume. During the last decade the reservations system was processing an average of 1.2 million bookings per day.

OLTP has evolved to use distributed computing with clusters, SQL servers and partitioned data to sustain high-volume applications. The industry-standard Transaction Processing Council (TPC) benchmarks are reporting throughput measured in millions of transactions per minute.

Despite the popularity of Java, PHP and other languages, IBM uses the C language for its TPC benchmarks. The client application code and DB2 stored procedures are C code.

Scalable architectures and powerful DBMS platforms have enabled the development of high-volume transaction processing applications, including massive OLTP using SQL. Five of the nine largest OLTP applications use SQL technology. One example is the UPS International Shipments Processing System that uses DB2. It achieved a peak workload of more than one billion SQL statements per hour. Merrill Lynch has a voice-driven application that does 21,000 transactions per second and eBay executes 26 billion SQL statements per day.

System architecture, database tuning, query optimization and data access programming techniques are significant influences on OLTP performance. Solutions for boosting OLTP performance include:

- Multi-tiered infrastructure and distributed processing
- Partitioned data
- Scalable clusters and parallel processing
- Batched updates
- Caches and connection pooling.

Application Servers, Distributed Processing

The preferred architecture for a modern transaction processing system will distribute the workload across multiple tiers of servers. System architects will opt for application servers that provide out-of-the-box capabilities such as authentication, authorization, load balancing, connection pooling, thread pooling, transaction coordination, logging and caching.

The lexicon of industry insiders has added the term “IT stack” to refer to a Web server, a server operating system, a DBMS and a programming or scripting language. Frequently used operating systems for hosting transaction servers or OLTP server software include AIX, Linux, Solaris and Windows Server, along with z/OS on IBM mainframes. Popular platforms for building transaction servers and OLTP software include the .NET Framework and Java Enterprise Edition (Java EE). The most popular open-source IT stack is LAMP, consisting of the Linux operating system, Apache web server, MySQL DBMS and PHP, Python or Perl as the development language.

More often than not organizations phasing out enterprise-scale, legacy applications are migrating to Java EE or .NET technologies. Technology such as scripting languages and Ruby on Rails are gaining traction for lightweight application development, but industrial-strength transaction processing is not a lightweight application. The leading Java EE application servers handle transaction management semantics, with support for container-manager transactions, bean-managed transactions and distributed transactions. They implement all features of the J2EE Java Transaction API (JTA) and Java Transaction Service (JTS) specifications. Microsoft Windows operating systems, from Windows NT to Windows Vista, have included a progression of enhancements to manage context and support transactions, including distributed transactions.

Distributed Transactions

Distributed transactions execute across database boundaries, across two or more connections. They can involve heterogeneous computing platforms and disparate SQL database managers. A distributed transaction is indicated when an application needs to coordinate related updates requiring connections to multiple databases.

DataDirect middleware provides a vehicle for processing distributed transactions with the Microsoft Distributed Transaction Coordinator (DTC) and with an industry standard protocol.

XA, Distributed Transaction Coordinator

The database industry has adopted a standard protocol for coordinating updates to two or more databases. In computing literature, the phrase XA transaction is a reference to The Open Group specification (ISBN 1-87263-024-3) for the standard two-phase commit protocol. Newer specifications have been developed for Internet distributed transactions, including such as WS-Transactions and OASIS Business Transaction Processing. Nonetheless the XA-style two-phase commit protocol is both a *de jure* and *de facto* standard.

Recent Microsoft operating systems have provided integrated support for distributed transactions with COM+ and the Distributed Transaction Coordinator (DTC) service. Microsoft DTC can operate through firewalls and it is extensible by loading third-party XA libraries into the DTC process. Performance of the DTC and COM+ has been such that they've been used for benchmarks of Microsoft SQL Server, including the Transaction Processing Performance Council's TPC-C benchmark.

Database managers such as Oracle 10g, IBM DB2 and Sybase Adaptive Server Enterprise are capable of acting as an XA resource manager. They can also participate in distributed transactions using Microsoft Distributed Transaction Coordinator and TP monitors such as BEA Tuxedo.

The requirement for distributed transactions means that flexibility is a consideration when selecting data access middleware. DataDirect Shadow middleware provides a vehicle for secure distributed transactions that update mainframe VSAM files and database. DataDirect Connect middleware provides a vehicle for distributed transactions using the two-phase commit protocol specified by The Open Group. It provides full support for commit and rollback of distributed transactions. It also works with Microsoft software that supports transaction processing.

Microsoft enhanced Windows platforms to support OLTP with the Microsoft Distributed Transaction Coordinator and .NET Framework 2.0 transaction processing classes. Visual Basic, VB.NET, C#, Delphi and languages that use ODBC drivers and ADO.NET providers can use the built-in functionality for distributed transactions.

- System.EnterpriseServices – enables any .NET application to participate in transactions.

- System.Transactions – this namespace enables .NET applications to use new, simpler interfaces for transactions. System.Transactions introduced six interfaces and 11 classes, with the Transaction object and TransactionScope object providing core capabilities.

Using ADO.NET distributed transactions for IBM DB2 requires the installation of the server version of the Connect for ADO.NET data provider. Using a DataDirect provider, a developer can write ADO.NET code as though a program is processing a transaction with a local database, even for distributed transactions. Expanding the code with additional database connections will cause System.Transactions to promote the transaction so the DTC is aware of it.

The Windows Vista operating system includes a transaction-aware capability, Windows Communication Foundation (WCF), which can propagate transaction context. Windows

Vista also includes transactional extensions to the Windows file system; enabling it to participate in transactions. WCF also provides loosely coupled logic for supporting long-running transactions.

Java Transaction API

Java EE application servers support communication between a JTA-compliant transaction manager and XA resource manager. The transaction manager functions as a transaction coordinator and reacts to the transaction context (start, end, suspend, resume) and its completion using the XA-compliant two-phased commit protocol. Top of the line app servers implement the XA-related interfaces of the javax.jms package. A premium app server provides features that an EJB container can use to support distributed transactions using JMS, Message-Driven Beans and a JMS session.

DataDirect middleware provides a vehicle for XA-compliant distributed transactions (using the two-phase commit protocol specified by The Open Group). JDBC drivers can work with XA-compliant resource managers. DataDirect JDBC middleware can provide distributed XA transactions used in conjunction with an application server and its transaction server (JTA). To support XA transactions, the DataDirect JDBC drivers provide an implementation of javax.sql.xa.XADataSource for DB2, Informix, Microsoft SQL Server, Oracle and Sybase databases. They also provide XAConnection (javax.sql.XAConnection) and XAResource (javax.transaction.xa.XAResource) objects used for distributed transactions.

Because distributed transactions involve more overhead than a local transaction, a distributed transaction takes longer to execute than a local transaction. Services for distributed transactions are one of the more important features of Java EE application servers.

DataDirect JDBC drivers have been tested for use with JBoss Transactions and the transaction managers of leading Java EE application servers, including:

- Adobe JRun
- BEA WebLogic
- IBM WebSphere
- Oracle Application Server
- Sun Java System Application Server.

One reason that application server vendors bundle DataDirect middleware is it provides a powerful solution for two-phase commit processing. Its performance helps mitigate delays associated with distributed transaction processing. DataDirect middleware is compatible with well-known frameworks, such as Hibernate and Spring and it's included with webMethods Fabric.

Besides the choice of JDBC driver, other factors influence the performance of Java transaction processing systems. These include connection pooling, caching and batching of JDBC prepared statements (sometime called bulk updates). Architecture can also play a role so developers should be aware of the argument for using servlets, Java Server Pages and EJB Container-Managed Persistence (CMP) and Bean-Managed Persistence (BMP).

Enterprise Java and SQL technology for OLTP is in use for high-volume web sites, including Amazon.com, eBay, Charles Schwab and Sabre Holdings. The eBay architecture sustains 212 million registered users, 1 billion page views per day and 26 billion SQL queries per day. It uses a data tier that support horizontal scaling and that makes extensive use of prepared JDBC statements cached by data sources.

Premium Data Access Middleware

"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
Chief Architect, Sun Microsystems
Database Technology Group

OLTP Benchmarks

The Transaction Processing Council provides the leading benchmarks used to measure client-server OLTP performance. Because TPC benchmarks are an important yardstick, vendors use the optimum configuration of hardware and software to maximize query performance. The TPC-C benchmark is a successor to TPC-B and both measure client-server OLTP performance.

The reports of TPC-C results describe the hardware and software configuration used for the benchmark, including TP monitors, compilers, server and client software. The hardware configurations used for TPC-C complement capabilities of today's major SQL platforms, including clusters, multi-processor architecture and parallel processing.

Although the database software supports multiple instances and clusters, the benchmark testing involves homogeneous data access for a single SQL platform. The TPC benchmarks do not test with a federated database or transactions distributed across heterogeneous SQL databases.

The Microsoft SQL Server TPC-C benchmark kit includes C source code that uses ODBC for data access. This contradicts assertions that a standard, multi-database API that uses a driver manager and database drivers is a performance bottleneck.

Application Server Performance

HP, IBM, Oracle, and Sun publish benchmark results that show the Java transaction processing performance with Java EE application servers. The Standard Performance Evaluation Corporation (SPEC) is the sponsor of SPECjAppServer2004, a benchmark that measures the performance of Java Enterprise Edition (Java EE) application servers. The performance testing includes the major functions of the application server, including database operations, Enterprise Java Beans and messaging. The test configuration includes the Java Virtual Machine, a DBMS and separate hardware for the database and application servers.

As of January 2007, the best performance with SPECjAppServer2004 was 7629.45 complex business transactions per second or 27,466,020 transactions per hour. The SPECjAppServer2004 benchmarks show diverse results using different combinations of hardware and software (application server and database).

Benchmarks conducted by The Middleware Company held the hardware and database server constant and tested several J2EE application servers. They found significant differences in application server performance, when using Servlets and Java Server Pages (JSP) or EJB Container-Managed Persistence.

JDBC Driver Selection

This last point demonstrates how one JDBC driver vendor may be good at one thing and a different vendor good at another. It also draws attention to the fact that it is worth spending the time to make sure you try many different JDBC drivers to get the best performance and functionality combination.

Source: *J2EE and .NET (RELOADED): Yet Another Performance Case Study,* Section 15.3.5, The Middleware Company (2003)

If performance is a concern for development teams planning mission-critical systems, they should conduct their own benchmarks. The best approach is to conduct performance tests using the hardware and software being used for the application, or the best emulation possible. Besides writing custom benchmark code, developers can also use benchmark tools such as LoadRunner or the Grinder, a Java load-testing framework.

Future of Transaction Processing

Application server deployments and transaction volumes will continue to increase as more organizations adopt specifications for Web services, SOA and Business Process Management (BPM) technologies.

Sabre Group reported that moving to SOA caused a surge in OLTP volume. Adding Web services interfaces to reservations and call center systems and other applications caused a dramatic increase in transactions starting in 2005. The move to a service-oriented architecture (SOA) and Web services has produced dramatic growth. Prior to offering Web services, the reservation system was typically processing less than 2 million bookings daily. After adopting Web services, SABRE processed a peak volume of 21 million transactions in a single day.

4.2 Web services and SOA

Large organizations have been moving to SOA to integrate diverse systems and create collaborative applications. They want to tie together heterogeneous environments, implement single signon, and provide a robust security infrastructure. XML, SQL, the .NET Framework, Java Enterprise Edition and Web services provide a foundation for interoperability and co-existence with legacy applications.

Companies such as Allstate, Banca Intesa, Bear Stearns, Fox Network Group, Honeywell, and Merck have used the .NET Framework for Web services and other applications. Organizations such as Bank of Tokyo-Mitsubishi, DaimlerChrysler, Earthscan Network, European Space Agency, Home Depot, Miami-Dade County, Sentilion, and Singapore Land Authority have used enterprise Java for building Web services and applications.

SQL databases play multiple roles in a service-oriented architecture and in providing persistent information for Web services. Many SOA and web service solutions use the OASIS Universal Description, Discovery, and Integration (UDDI) registry for discovering services. UDDI was originally conceived as a public registry for e-business maintained by the UDDI Operators Council that included IBM, Microsoft and SAP. IBM's public UDDI registry was hosted with a DB2 database and Microsoft's was an SQL Server database. Today organizations use private UDDI registries for application integration. IBM includes a UDDI setup wizard with DB2, Oracle provides a UDDI-compliant Service Registry and Microsoft includes UDDI services with Windows Server.

Some web service projects involve aggregating data from disparate SQL databases and using SOAP to deliver it as an (XML-encoded) document or message stream. This may involve using SQL queries, SQL/XML queries (SQL:2003), and/or XQuery. DataDirect has middleware that supports all of the above types of queries (XQuery and SQL with extensions for XML). As an example, DataDirect XQuery supports embedding calls to Web services within XQuery scripts.

The technology that provides the underpinnings for SOA has continued to evolve since IBM and Microsoft introduced Web services in 2000. The plumbing (TCP/IP, XML, SOAP and REST) was resolved early on but vendors have taken different approaches as you move up the stack. As an example, specifications for guiding the flow of Web services activity included Microsoft XLANG and IBM WSFL. To provide an opportunity for standardization, the W3C chartered the Web services Choreography Working Group. Similarly for distributed transaction processing, several solutions have emerged. OASIS published the Business Transaction Processing (BTP) specification. IBM, IONA, Microsoft and Red Hat (JBoss), with cooperation from other OASIS members, have been developing three WS-Transaction specifications as an alternative:

- WS-Coordination
- WS-AtomicTransaction
- WS-BusinessActivity

As SOA matures, there are still concerns about Web services security and SOA governance. The thrust of SOA governance efforts is based on establishing policies for governing assets and processes. Registries and repositories are key enabling technologies for SOA governance, in part because they provide metadata about

services. A variety of products have implemented registry/repository specifications (UDDI, ebXML Registry). Products such as webMethods Infravio X-Registry have been tested with popular SQL databases. The UDDI specification describes APIs for publishing and finding services. The ebXML Registry supports federated registries and it provides interfaces for both XML and SQL queries.

4.3 Content Management, Web-Facing Applications, Portals

Many organizations exploit web technology for purposes such as servicing customers, selling products and sharing knowledge. The web browser has become the universal client. Governments create portals to provide services and knowledge to citizens. Corporations provide services and knowledge for partners, employees and customers.

Because the Internet is globally accessible by millions of users, a web application is a performance and scalability challenge. They are also characterized by 24x7 operations and the potential for a large volume of users. This means applications may have many active connections, which can lead to locking and concurrency issues for poorly designed systems. To provide performance and scalability, many enterprise and system architects use data replication to distribute data and to provide failover server capabilities.

In creating portals or other web-facing applications, many developers store documents as XML or use a content management system (CMS). Others use Asynchronous JavaScript and XML (AJAX), mashups and Web 2.0 technology.

Whether a web site is a portal, web store, financial services, social networking, exchange or auction web site, the behind-the-browser infrastructure typically includes databases and/or a content management system. In many instances, there are also links to legacy databases and back office systems.

Content management systems were once a niche product, supporting printed documents and Standard Generalized Markup Language (SGML) applications. The emergence of HTML, the World Wide Web and XML broadened the uses for content management systems. Today they include enterprise content management, Web content management, records management and portal publishing.

Early publishing and content management systems used an ad hoc data store for document repositories. That changed as open source SQL engines evolved and leading vendors of commercial SQL platforms adopted extensible servers and object-relational technology. Today there are hundreds of content management systems, many built with SQL technology. CMS products that support more than one SQL DBMS include Documentum (EMC), FileNet (IBM), Stellent (Oracle), Livelink ECM (Open Text), Worksite (Interwoven) and Content Management (Vignette).

Anticipating a high rate of XML adoption for integration and document processing, major database companies evolved their SQL servers to integrate XML and XQuery (the W3C XML Query Language). The evolution from SQL servers to SQL/XML servers set the stage for mixed workloads for data processing and document processing.

Prior to the emergence of XML, organizations were adopting object-relational databases

for online complex processing (OLCP) that involved rich types. XML added another complex data type to the mix. It opened the door even further for applications such as knowledge bases, digital libraries, enterprise portals, government portals and document repositories.

Distributed Content Management

Web sites have pulled content from multiple servers for years, but blogs and RSS feeds raised awareness of distributed content management. CMS offerings such as Blogger and WordPress became popular for personal publishing and Google added a blog search. Software companies moved to update their products for professional blogging, group collaboration and intranet blogs. IBM released a free RSS Portlet; Microsoft enhanced Windows Server with SharePoint Services. Moveable Type released an enterprise addition that works with leading SQL databases. FAST released the NXT 4 distributed content management system that integrates content from desktop files, mainframes, the Internet, content management systems and SQL databases accessible via ODBC.

Software vendors with portals, information access and content management systems have licensed data access middleware from DataDirect Technologies. DataDirect middleware has been tested with software from vendors such as Autonomy, EMC, Entrust Technologies, FileNet, Fujitsu-Siemens, IBM, Microsoft, Oracle and webMethods.

Compliance and Workflow

Some organizations use content management systems as a foundation for regulatory compliance. In some instances they are marrying content management with approval cycle workflow and compliance reporting.

Performance with Concurrent Users

Web-facing applications put a premium on database security and data access performance. Many large organizations spec their web site load at 1,000 to 5,000 concurrent users. For real-world database processing, 1,000 concurrent database users doing analysis and 2 million transactions per day is a respectable workload.

A benchmark for iSeries computers involved converting RPG and COBOL programs to WebSphere applications using IBM's WebFacing Tool. The tests used two separate configurations with a PC client and iSeries computers serving web pages and running RPG jobs (report generation). The test configurations sustained 500 concurrent users by using an Edge Side Include (ESI) cache with WebSphere, a heap size set to NOMAX, JIT compilation of java classes, data compression, JSP batch compilation and an Apache web server. With a two-tier architecture (2-way POWER5 processor) and a three-tier architecture (2-way POWER 3), it was able to serve 82 and 75 screens per second, respectively, with an average response time of .508 seconds and 1 second.

At the high end of the spectrum:

- Ticketmaster.com supports 200,000 concurrent users
- MSNBC boosted its web site for the 2002 Winter Olympics to serve 260,000 concurrent users, with a daily peak of 6.7 million users and 40 million page views

Obviously organizations must be aware that coupling powerful page-serving solutions with slow SQL data access middleware is a recipe for disaster for web-facing applications having large numbers of concurrent users.

4.4 ERP Suites, CRM Suites

Enterprise Resource Planning (ERP) suites are software packages that provide the nucleus for mission-critical enterprise information systems. ERP refers to a class of products that have evolved from manufacturing resource planning systems, integrated manufacturing systems, integrated logistics management, integrated accounting systems, human resources management systems and various types of management information systems. ERP suites are entrenched in large organizations and they have a strong appeal for manufacturing and supply chain automation.

Another class of software, the Customer Relationship Management (CRM) suite, provides capabilities for sales support, marketing and customer service. The rationale behind the CRM suite is to be able to analyze customer data and turn it into useful information for sales and marketing. CRM suites have been a tool for sales force automation. Major vendors included Baan, Microsoft, Oracle, PeopleSoft, Salesforce.com, SAP and Siebel Systems. Consolidation has occurred with Microsoft acquiring Great Plains Software and Oracle acquiring PeopleSoft and Siebel.

ERP suites, CRM suites and similar applications leverage SQL platforms for persistence, concurrency control, security, reporting, and analytics. Vendors such as Lawson, Oracle and SAP have moved to expose some of their package's functionality as components, using Web services and the Java Connector Architecture.

High-end ERP suites are expected to perform effectively for thousands of users, hundreds of concurrent users, processing millions of transactions per month. Clearly this requires optimized SQL databases and high-performance data access middleware. DataDirect middleware has been licensed for companies with well known ERP and CRM suites. It has been tested with products from SAP, Oracle, Lawson Software and Siebel Systems, now part of Oracle.

4.5 Integration and Data Federation

Because of powerful networking and distributed data, there has been much focus on integration technologies and data federation. A data federation solution provides a single virtual view of one or more data sources.

There are several flavors of integration technology, including

- Data integration
- Application integration, enterprise application integration (EAI)
- Enterprise information integration (EII)
- ETL
- Business process integration (BPI).

Data access middleware and integration middleware are different approaches to accessing data from multiple storage systems. Data access middleware is the more mature technology, but the two technologies complement each other.

Data access middleware uses XQuery, SQL or both to query databases. For integrating data from sources that are not databases, XML has become the favored solution. The direction taken by vendors such as IBM, Microsoft and Oracle blurs the distinction between XML, XQuery and SQL. The intent is to provide seamless access to SQL databases, semi-structured and unstructured data. The leading database vendors have endorsed XQuery and XML integration into the SQL standard.

Scenario: Information Integration with IBM Information Server

After the W3C published the XML specification, IBM undertook the development of technology that supported data integration and queries over SQL, XML and non-relational data sources. The Garlic project fed into IBM DB2 Information Integrator. IBM has also introduced WebSphere Business Integration Server, WebSphere Federation Server and IBM Information Server.

The architecture of IBM Information Server combines unified metadata, common services and a parallel processing engine. It uses a common metadata repository and all Information Server functions share a common metamodel. For persistence, the repository uses an IBM DB2, Oracle or Microsoft SQL Server database.

IBM Information Server includes shared services and a common services layer that enables queries over federated data to be published as shared services. It provides common data transformation rules for business process integration, federated queries, analytics and other applications. By providing shared data integration services, Information Server lays a foundation for a services-oriented architecture.

Using Information Server is an alternative to Web services because it can publish a service with SOAP bindings or generate a stateless session bean. The EJB solution provides data transformation services instantiated as EJB method calls.

IBM Information Integration Solutions uses DataDirect drivers to access an extensive variety of data sources from IBM Information Server.

> Pete Inzana, Director, Product Management, said "Information Server is a revolutionary new software platform from IBM that helps organizations derive more value from the complex, heterogeneous information spread across their systems. It enables organizations to integrate disparate data and deliver trusted information wherever and whenever needed, in line and in context, to specific people, applications, and processes."
>
> Discussing the benefits IBM Information Server derives from the drivers, Inzana said "DataDirect's ODBC drivers provide us with valuable connectivity, supplemental to all the native connectors we provide within Information Server. The Wire-Protocol Drivers allow direct, highly functional connections to all the leading RDBMSs without any additional prerequisites (like DB client libraries). Plus their heterogeneity and performance are really impressive."

4.6 Process Integration, Enterprise Information Integration, Business Process Management

Over the past decade enterprise application integration (EAI) garnered attention, partly because many large organizations had installed ERP suites and software packages that hosted vital data. As demand for EAI grew, whole companies were launched to develop products. Middleware vendors competed to provide adapters and connectors that would provide integration capabilities with a broad spectrum of data sources, including heterogeneous databases, files, ERP applications and transaction processing monitors.

Organizations often used message-oriented middleware and object brokers for integrating applications. They used ETL, Electronic Data Interchange (EDI) and XML servers for data integration. SOA, Web services, WSDL and UDDI emerged as viable infrastructure for collaborative applications and information integration. Point solutions for data aggregation, data integration and process integration filled a need but organizations looked for a more comprehensive solution. Users wanted integration and data federation with a single door for access to disparate data sources.

The software industry responded with a new architecture and a new class of software known as enterprise information integration (EII) suites. Software vendors released product suites that had roots in the SQL world or XML world. Some products supported XML, SOAP and XQuery, but not SQL connectivity. That made a difference in cache handling, metadata handling, query optimization, connection pooling and support for standard SQL APIs.

The architecture of EII software includes service consumers and providers, integration brokers and the Enterprise Service Bus (ESB). The ESB provides an integration backbone. It enables disparate components or applications to communicate through a common messaging bus. ESB software supports messaging technologies such as Java Messaging Service (JMS) and Web services (SOAP with XML encoding).

Today’s well-known EII products include Composite Information Services, IBM WebSphere Federated Server, iWay Software Enterprise Data Hub, MetaMatrix Enterprise, Microsoft BizTalk Server, Oracle Fusion Middleware, Progress Sonic ESB, Sybase Avaki and TIBCO BusinessWorks. Top-of-the-line integration-focused products exploit XML, SQL, XQuery, multi-database APIs, interoperable messaging, Web

services, metadata repositories, semantic information models, and data transformation technologies.

Multi-database APIs and data access middleware has provided ubiquitous connectivity. Integrating data from SQL databases is often less of a challenge than some specialized data sources. Whether use a point solution or a broader approach to information integration, access to heterogeneous databases is usually part of the picture. Although response time may not be an issue for batch processing, such as refreshing a data warehouse, minimal response time is typically a requirement of online users.

Companies that are well known in the integration space have licensed DataDirect middleware. It's been tested with software from IBM, Oracle, MetaMatrix, Microsoft, Progress Software, Sterling Commerce and webMethods.

Workflow and Business Process Management

The origins of workflow software products are diverse but Sarbanes-Oxley, SOA and Web services have given traction to business process management (BPM) initiatives. BPM suites and middleware promise to integrate Web applications and hosted components with packaged applications and back-office systems, user interfaces with processes and data sources and so on.

To create a collaborative application that's a composite of services, there's a need for business rules, transaction co-ordination, well-defined interfaces and a solution for flow of control (choreography, orchestration, workflow). The software industry saw that model-driven solutions and declarative languages had promise for solving the problem for a new generation of collaborative applications. The industry has published specifications such as Business Process Modeling Notation (BPMN), Business Process Execution Language (BPEL), Business Process Definition Model (BPDM), Business Process Specification Schema (BPSS), Semantics of Business Vocabulary and Business Rules (SBVR), and the XML Process Definition Language (XPDL).

There are almost dozens of XPDL implementations, including software from Fujitsu, Oracle, TIBCO and Vignette. The more than two dozen BPEL implementations include products from BEA, IBM, Microsoft, Oracle, SAP and Sun. OASIS, Object Management Group (OMG) and the Workflow Management Coalition have championed development of specifications for business process management. Specification refinements and interoperability testing are on the horizon. The Business Process Management Initiative has merged with OMG. A new initiative, Business Process Alliance, includes Microsoft, Fair Isaac, AmberPoint, IDS Scheer AG, Global 360, Metastorm and other partners.

Leading workflow and BPM suites use JDBC and/or ODBC to access SQL databases. The SQL solution meets the need for

- Directory services
- Persistence for template and application repositories
- Stateless operation by persisting process activity
- ODBC/JDBC adapters for building integration points in processes.

IBM WebSphere Business Integration Server is one of several integration products from IBM. It provides for building end-to-end business processes though the collaboration of

middleware components. The Oracle Business Process Analysis Suite bundles the IDS Scheer ARIS platform as a complements to the Oracle SOA Suite and Oracle BPEL Process Manager. It provides business process modeling, publishing and simulation features for system architects.

Other notable BPM suites include BEA Fuego, IBM FileNet P8, the Intalio BPMS, Lombardi TeamWorks, Software AG crossvision, TIBCO Staffware, and Vitria BusinessWare.

Hyperion System 9, the Business Performance Management suite from Hyperion, includes ODBC and JDBC drivers from DataDirect.

Scenario: Integration with Mainframe Systems

Integrating enterprise data often requires connectivity to mainframe computers that are hosting legacy applications and databases. Solutions for connecting to a mainframe include 3270 Terminal Emulation, message-oriented middleware, file transfers via ftp or IND$FIL and adapters for CICS.

A more sophisticated solution, such as DataDirect Shadow RTE, supports mainframe data access, SOA, Web services, data integration and event stream processing. It provides connectivity to z/OS mainframe databases, Web services, applications, and Java and .NET components. Shadow z/Services, for example, enable mainframe software to act as a service provider or service consumer. It can access DB2, IMS/DB, VSAM or Adabas data and publish it as a Web service using SOAP and HTTP.

Organizations such as Alaska Airlines, Merrill Lynch and the UK Land Registry have used DataDirect Shadow to provide mainframe database access to dozens of applications servicing thousands of users per day. As an example, the Alaska Airlines frequent flyer web site has 3.8 million users and it receives 500,000 visits per week.

Security is an issue with distributed transactions that update mainframe data. For distributed OLTP, Shadow Enterprise Transactions creates a secure environment for each transaction sent to the mainframe by a Web server or application server. Shadow Enterprise Transactions can control access to the RPC and provide auditing, logging and tracking capabilities.

Shadow RTE includes a server that deploys in a CICS region to provide access to mainframe screen and business logic. The Shadow z/Events software offers a solution for Complex Event Processing (CEP) and it includes support for the J2EE Connector Architecture (JCA). It can do a real-time capture of events from Adabas, CA-IDMS, CICS TD/TSQ, IBM DB2, IMS/DB, and VSAM.

4.7 Analytics, Business Intelligence, OLAP

Database vendors have competed for decades for OLTP supremacy. Since the ‘90s, there’s been growth of online analytical processing (OLAP) and data warehouses as vehicles for business analytics, enterprise reporting and business intelligence (BI). Database vendors compete in this arena with relational OLAP (ROLAP) technology, sometimes in a hybrid OLAP (HOLAP) combination of ROLAP with multidimensional OLAP (MOLAP).

SQL DBMS vendors have adopted a Swiss-army knife strategy, evolving their database servers into information appliances. The servers use a plug-in architecture and tightly coupled technologies, including SQL data processing, XML document processing, message queuing and data mining. They are capable of supporting mixed workloads, with functionality for transaction processing, content management, integration, loading data warehouses and reporting. They are also supporting increased levels of analytics and processing for business intelligence.

Companies with SQL platforms (IBM, Microsoft, NCR Teradata, Oracle and Sybase) have become leading suppliers of database technology for data warehouses, with MySQL filling a niche for small data warehouses.

Today the largest decision support databases are measured in billions of rows. Data warehousing and business intelligence are examples of applications for clustering and federated database technology. Organizations have built multi-terabyte data warehouses using SQL platforms, bringing the performance of SQL queries for decision support into the spotlight. There is a trend towards embedding business intelligence in enterprise applications, such as call centers, and services in a service-oriented architecture.

Leading vendors of data warehouse, analytics, data mining and business intelligence products use DataDirect middleware for heterogeneous database connectivity. DataDirect middleware has been validated with software products from Actuate, Autonomy, Cognos, SAS, Business Objects, Fair Isaacs, Hyperion, Informatica, Intuit, MicroStrategy, Misys, NCR Teradata, Open Text, Siebel, SPSS, and Sybase.

There are many mainstream products for OLAP, managing multi-dimensional data, data mining, and business intelligence. These include:

- IBM DB2 OLAP Server, Cube Views, Intelligent Miner, Information Integrator
- Microsoft SQL Server Data Mining, Analysis Services
- NCR Teradata OLAP Extensions, TeraMiner Analytics, Hyperion System 9
- Oracle OLAP, Data Mining, Warehouse Builder, Transparent Gateway
- Sybase IQ, SPSS Clementine Server

There are also analytics and BI products from Actuate, Autonomy, Cognos, SAS, Business Objects, Fair Isaacs, Informatica, Intuit, MicroStrategy, Misys, Open Text, Siebel (Oracle), SPSS, and Zarion.

OLAP

Data warehouses designed for ROLAP use SQL databases to store dimension tables, fact tables and sometimes materialized views. An analytical query operates with data in an OLAP cube, a multidimensional data structure optimized for reporting. OLAP queries are typically answered more quickly than queries over SQL tables. The reason is that prior to the query, computation and aggregation have been completed with the results available in ROLAP or MOLAP storage. To build an OLAP cube, analysis and reporting applications might access a single data warehouse, multiple data marts and linked data sources.

Standard benchmarks of OLAP products have shown a pattern of continuing improvements in query response time, but not cube load time. Nevertheless a recent survey reported a majority of users are dissatisfied with query response time. Some organizations have approached large OLAP workloads and performance issues by moving to clusters of commodity PCs with parallel database processing. Using a large number of nodes in a cluster can result in better performance when processing very large data sets.

Designing the optimum architecture for OLAP performance is a combination of art and science, requiring knowledge of the issues of resource usage and scalability. System architects and data warehouse architects should understand the effect of middleware on performance. OLAP applications that access disparate servers and databases can take a performance hit from inefficient middleware, particularly as database size grows. Data access middleware can affect query response time and cube build time when using linked data sources.

Database Growth, 64-Bit Servers

Data warehouses and data marts tend to increase in size over time. As gigabytes become terabytes, retrieving metadata and database objects (tables, views) often takes longer. The ideal for business analytics applications is linear scalability with query performance time remaining constant as database sizes increase.

Data access middleware must be capable of queries involving large volumes of data. It must efficiently handle large tables and schemas. One solution for scalability and better performance is 64-bit operating systems and software. DataDirect offers 64-bit middleware for Windows Server 2003 and 64-bit AIX, HP-UX, Linux and Solaris operating systems, providing access to IBM DB2, Informix, Oracle, Microsoft SQL Server and Sybase databases.

Queries

The queries executed for business analytics, OLAP and decision support can be long-running queries over large volumes of data. BI, OLAP and data warehouse users often execute ad hoc queries. A survey by Ventana Research found lack of satisfaction with ad hoc query performance for data marts and data warehouses. 55% of respondents reported ad hoc query performance dissatisfaction and 84% said users would benefit from a 10x improvement in query response times.

Delivering fast query responses for analysis and reporting purposes presents a design challenge for data warehouse architects. The architect must weigh the importance designing for performance versus designing for flexibility. For maximum flexibility, the design choice is to store data at the granular level. However, users typically want higher-level summarizations for analysis and reporting. To meet that need, the data warehouse architect often adopts a strategy of using aggregates, or pre-calculating data summarizations.

Two approaches are mainstays of an aggregate strategy: OLAP and aggregate tables. The latter approach can include aggregate-aware solutions such as materialized views and aggregate join indexes. DBAs, system architects and data warehouse architects need to determine the latency requirements for OLAP and aggregates and design accordingly. Materialized views, for example can be compute-intensive and deferred refresh can introduce data consistency issues.

Business intelligence applications have caused a surge of interest in low-latency or real-time OLAP, meaning that changes to dimensions are reflected when they happen, not hours or days later. One popular approach to low-latency OLAP is to automatically refresh the cube every few minutes.

Low-latency OLAP with linked data sources requires frequent refreshes, high-bandwidth connectivity and fast data access middleware. For performance reasons, premium middleware is multi-threaded and it supports asynchronous processing and cancellation. For repetitive reports, there is a performance benefit from using middleware that supports parameterized queries.

Performance Tests of OLAP and BI Software

TPC transaction processing benchmarks have typically gotten more press coverage than their OLAP counterparts. To provide a basis for comparing OLAP and decision support performance, the now-defunct OLAP Council developed the APB-1 Benchmark. APB-1 lost momentum, but vendors support the TPC-H benchmarks of the Transaction Processing Council. The TPC-H benchmark reports a Composite Query per Hour Metric for several database sizes: 100 gigabytes (GB), 300 GB, 1000 GB, 3000 GB and 10000 GB.

A TPC-H benchmark report includes performance test results, with details of the database, hardware, and software configuration used for testing, including the DBMS, operating system, scripts, clients and other software. The source code of the TPC-H benchmark application for Microsoft SQL Server uses the ODBC API, as does the code for the TPC-C transaction processing benchmark.

The use of ODBC clients for OLTP and OLAP benchmarks illustrates that standard APIs and database drivers can be a part of a high-performance OLTP or OLAP architecture.

In 2004, Business Objects conducted benchmarks of Crystal Enterprise to publicize the performance of BI solutions. The tests at an IBM Solution Partner Center used active user populations from 500 to 4000. They simulated concurrent users reading and refreshing reports varying from one-page summaries to a 600-page report with 100,000 records. The tests produced 2, 4 and 6-second response times, even when requests queried a database. The testing used various reporting server configurations (from 6 to 48 processors), on hardware running IBM AIX. The working set included 20,000 named users and 20,128 report instances.

Oracle and Hyperion used APB-1 for benchmark wars in several years ago. Oracle tested with Oracle 9i Real Application Clusters simulating 10,000 users. Hyperion tested Essbase OLAP server with a cluster of HP servers and processed 119,085 analytical queries per minute. IBM DB2 bundled an OEM version of Essbase OLAP Server for about a decade.

The APB-1 and TPC-H benchmarks provide metrics on decision support performance using a single database platform. Unfortunately they do not include queries with a federated database or virtual data warehouse. It's when running federated queries that the importance of fast data access middleware is most evident.

OLAP Performance and Scalability Solutions

Solutions for scalability and performance of OLAP applications include partitioning data, partitioning analysis, reporting and presentation across multiple tiers, and using server clusters. Understanding the role of data access middleware and specifying the right middleware solution is also a key factor. The prudent system architect will specify best-of-breed middleware (such as DataDirect Connect) that offers a solution for connection pooling, multi-threading, clustering, load balancing and failover. These solutions are essential for high-availability applications with fast queries, capable of serving larger user communities.

Scenario: Building Cubes with Microsoft Analysis Services, Integration Services

Microsoft Analysis Services is a server-based platform for OLAP and data mining. Analysis Services uses OLAP data access that's compatible with SQL data access. Therefore heterogeneous queries can access OLAP data and return it in an SQL result set. Sales of Analysis Services exceeded SQL Server sales because some DB2 and Oracle shops acquired it to build BI programs using their existing databases. Customers are buying Analysis Services primarily for the OLAP server.

SQL Server Integration Services (SSIS), which evolved from Data Transformation Services (DTS), provides connectivity to heterogeneous data sources. There are features in SSIS targeted for Analysis Services, including handling data partitions, changing dimensions and low-latency OLAP. As an example, an Analysis Services analytical database could include OLAP cubes and data mining models using data from a Teradata Database. This would require an ODBC driver or OLE DB Provider for Teradata. As discussed above, the optimum solution is to use high-performance data access middleware, preferably a 64-bit driver or provider if working with very large dimensions.

Enterprise Reporting

Report generation is a classic need that data processing and IT organizations have supported with multiple generations of hardware and report writer software. With the arrival of PCs and SQL servers, report generation technology became available at all levels of an organization (workgroup, department, division, region, enterprise).

Prior to the availability of the multi-database API, reporting software was tailored to work with specific SQL databases servers because each used a proprietary data access API. The adoption of industry standard APIs for data access overcame a major obstacle to report writer development. Reporting software became a leading type of database client.

Today reporting software has evolved by adding more analytical and business intelligence capability and web presentation features. Products such as Business Objects Crystal Reports, Cognos Report Writer, Oracle BI Publisher, Liveware R&R ReportWorks, ShowCase Report Writer and Microsoft SQL Server Reporting Services are installed on millions of computers.

Data Warehouses, Federated Data, Virtual Data Warehouses

There are several variants of data warehouse architecture, including a central data warehouse, a collection of data marts, and hub and spoke architecture (central data store with multiple data marts). Organizations are also building federated data warehouses. They are distributed database systems that use various network topologies, including local and wide area networks. The databases may be distributed organizationally, geographically or both.

A federated data warehouse, or virtual data warehouse, provides a logical, integrated view of distributed data. It's based on metadata and standardization that facilitates sharing among data warehouses and/or data marts. The sharing includes data and metadata, business models and reporting framework. The federated warehouse is a consolidation from multiple data sources performed dynamically. Instead of moving data into a single physical warehouse, the federated data warehouse executes queries using a single logical warehouse with distributed data sources.

A federated data approach to data warehousing is an alternative to a heavy commitment to extraction, transformation and loading (ETL) activity.

The effort to consolidate information in a centralized data warehouse includes data cleansing and loading, such as loading data from transaction processing databases. The effort is not trivial. One organization created an enterprise data warehouse that involved loading data from 14 data sources, including Oracle, Sybase, SQL Server, and PeopleSoft. To create a data warehouse with 30 fact tables, 45 dimensions, and 250 staging tables required nine months.

> Hence for any large corporation it seems to me that a federated warehouse approach is what you will end up with, whether you like it or not. Few companies will have the energy or resources to deliver the single giant warehouse, and even those few that do will, in reality, have a series of skunk works data marts / warehouses dotted around the corporation since such a behemoth warehouse will be a bottleneck, hard to change and inevitably slow to respond to rapidly changing business needs.
>
> **Andy Hayler**, founder of Kalido

NCR Teradata is one of the companies that pioneered technology for large data warehouses. It provides parallel processing and horizontal data partitioning, with rows distributed across sites.

Microsoft SQL Server's Data Transformation Service has been re-invented as SQL Server Integration Services (SSIS). It provides an ETL solution for loading data warehouses and it exposes an object model for programming. A single SSIS package can connect to multiple SQL data sources using .NET providers, OLE DB providers and ODBC drivers. When bulk loading of data marts and data warehouses with SSIS is time-sensitive, the use of high-performance data access middleware is indicated.

Oracle formerly advocated consolidating information from multiple sources into a central data warehouse. Since the acquisition of Siebel Systems, Oracle has offered a federated data solution. Oracle BI Server, formerly Siebel Analytics Server, is accessible through ODBC. It can extract data from heterogeneous data sources and services, including DB2, Oracle and Microsoft SQL Server.

IBM products such as DB2 Universal Database (UDB) and DB2 Information Integrator support federated data warehouses by providing location transparency and heterogeneous data access.

Some business analytics and OLAP queries use a federated snowflake or star schema. The data exists on multiple servers and it may include heterogeneous data sources. A product such as DB2 enables a developer or DBA to setup a federated server, define remote data sources and specify queries that are pushed down to a remote data source. The combination of DB2 Data Warehouse Edition with DB2 Information Integrator enables a DBA or developer to optimize query performance for federated star or snowflake schemas involving disparate databases and servers.

ETL and data replication represent data movement solutions, whereas federation accesses data in place. Nonetheless, one similarity is they can operate over the same networking and data access middleware infrastructure. The same is true of distributed transaction processing. When using a federated schema or executing a distributed transaction, queries can be time-sensitive and performance an issue.

Data access middleware affects integrity, reliability, scalability and performance when executing query workloads across several databases. That's a primary reason database and BI software vendors have traditionally included DataDirect drivers with products that provide heterogeneous data access.

Scenario: Oracle BI Suite and Federated Data

Oracle BI Suite products support federated data using Oracle BI Server, which was formerly Siebel Analytics Server. To support access to heterogeneous data sources for its analytics products, Siebel included DataDirect Connect middleware. The Siebel product provided Oracle connectivity and extensive support for features of IBM DB2 and Microsoft SQL Server.

Oracle BI Web provides a web interface for business intelligence. It communicates with a web server using SSL and with Oracle BI Server using ODBC over an SSL connection. Oracle BI server is a query and analysis server that provides services for components of the Business Intelligence Suite, such as Answers, Dashboards, Reporting and BI Applications. Because it exposes services through ODBC, client software can access Oracle BI Server as an ODBC data source. The Oracle BI Suite provides semantic metadata layers for data sources, dimensions and user views. Clients see a logical schema view that's independent of the physical schemas of the data sources.

Oracle BI Server translates queries in logical SQL to native SQL used by data sources. The BI Server Execution Engine performs some intermediate processing but the BI Server pushes as much processing as possible down to the database servers. Oracle BI Publisher uses data from:

- Unstructured sources (XML, Web services, files)
- Oracle BI Server
- SQL databases (Oracle, IBM DB2 and Informix, Microsoft SQL Server, Sybase)

Tuning for OLAP

Establish a performance benchmark for the requests on your list with performance tuning facilities in your BI system and/or DBMS. For example, the Teradata Database Query Logging facility and tools such as Embarcadero's DSAuditor can be used to monitor, log, and analyze performance and resource utilization. Once you have established your performance benchmark, investigate where improvements can be made in terms of the how your requests are implemented, submitted, or changes to the underlying systems. For example, a DBA or system administrator may determine the need to create new indices, monitor for additional statistics, or fine tune requests with new or updated stored procedures or macros to improve performance.

DataDirect Connect middleware provides tracing capabilities that provide details of query execution, recording data sent to and from an OLAP server.

Data Mining

Automated statistical analysis or data mining has become an important business intelligence advantage for organizations with data warehouses. As an example, SQL queries permit extraction of customer and transaction data for analysis by demographic and neural clustering algorithms.

SQL database vendors have integrated data mining algorithms and functions directly into the DBMS. Microsoft SQL Server supports the use of classification trees and clustering algorithms. Oracle 10*g* provides functionality such as clustering and classification and regression trees. It also supports associations, attribute importance, prediction, feature extraction and BLAST queries. Beside integrated SQL database solutions, there are also advanced statistical packages from SAS and SPSS that implement data mining algorithms.

Distributed Data Mining

Distributed data mining has proven successful for applications such as credit card fraud detection, although an early impediment was incompatible database schemas. Grid-based knowledge discovery and distributed data mining is the subject of a variety several research projects. They have processed astronomy data, drug features and effects data, gene and protein databases, network access and intrusion data, and data about web content and usage.

Distributed data mining technology has moved from research projects into products and data access middleware increasingly plays an important role. For data mining projects that use SQL databases, ODBC and JDBC have been a standards-based data access solution, with the DataSpace Transfer Protocol (DSTP) gaining traction for data mining services. Data mining middleware can the Predictive Model Markup Language (PMML) and there are data mining APIs for Java and SQL. Middleware can SQL platforms that are gaining more sophisticated data mining technology. Microsoft SQL Server, for example, can do OLAP and data mining tasks by creating a data source view from one or more data sources. It can partition data across multiple CPUs, use an Integration Services Pipeline for integrating multiple data sources, and produce a homogeneous data schema based on heterogeneous data sources.

5.0 Findings and Outlook for the Future

Vendor Profile

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

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 Teradata, 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 Software Corporation. 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.

Vision and Ability to Adapt to Change

The company has a nuclear staff that's been successful at adapting to major shifts in the software industry, from Windows to Java to Web computing. They have ridden each successive wave of computing innovation since the Windows and PC computing era. In the process, the company has forged important alliances with and become a key middleware supplier to the major players of the software industry. Today DataDirect offers products for multiple chip sets, operating systems, databases and development and deployment platforms.

The company has a clear vision based on in-depth understanding of data access and integration trends. It has been at the forefront in providing information retrieval

capabilities for client-server SQL, Java, Windows, Linux and XQuery users.

Its visionaries recognized early on the importance of XML, for example. That cognizance has been validated as Web services and SOA gained traction and database vendors embraced SQL/XML for document and data processing. The company is well positioned for the future with expertise in XQuery and SQL query processing, SOA and mainframe connectivity.

Products and Technology

DataDirect middleware and mainframe SOA integration products are well suited to the development and deployment of mission-critical applications. They are a good fit for a variety of computing scenarios, providing reliability, performance and scalability for:

- Transaction processing
- Business intelligence
- Integration services, information integration, application integration
- CRM
- ERP
- Content management, portals, ECM
- Federated data warehousing
- Data mining

DataDirect middleware is a technology of choice for developers using popular programming and scripting languages and frameworks, such as Hibernate or Spring. ODBC drivers provide connectivity for C, C++, Visual Basic and other programming languages, and for PHP, Perl and scripting languages. JDBC drivers serve Java programmers and ADO.NET providers are the data access solution for C#, VB.NET, Delphi and .NET programming languages.

Performance and Scalability

System architects are addressing scalability and high-availability by throwing hardware at the problem, with solutions such as Oracle Real Application Clusters. But software and middleware must complement the hardware. DataDirect middleware provides capabilities, such as load balancing and failover, that are critical for high-availability, mission-critical applications.

Besides 24x7 operations, these applications might also need to support a large number of users and a large volume of database activity. For applications that operate with persistent data, the network and/or the database can be performance bottlenecks. This underscores the importance of best-of-breed middleware that minimizes those bottlenecks.

Human nature being what it is, there will always be a desire for fast queries. Speeding up queries is not simply a function of database design and middleware plays a role. DataDirect drivers and providers support techniques for fast query performance, including caches, connection pooling, bind packages, stored procedures, native images and multi-threaded drivers. They also provide connection strings and properties that affect performance, scalability and security. This flexibility makes data access behavior an adjustable proposition, more like a radio tuner than an on-off switch.

DataDirect middleware is well suited to computing scenarios and applications that require scalable, high-performance data access middleware.

Stability

Reliability and consistency are at the top of any list of requirements for data access middleware. This has been an issue with drivers from DBMS vendors that are unsupported software and not mainstream products. On the other hand, the reliability and consistency of drivers will be a bet-your-business proposition when your business is middleware. To ensure reliability and consistency, DataDirect has been at the forefront in developing test suites for database drivers. It performs extensive testing before each new product release.

A certain amount of defects (bugs) will exist in the first releases of any software product. The 'version 1.0' phenomenon reinforces the notion that more releases and more testing reduce the number of critical defects. This is true of ODBC and JDBC drivers because those specifications are mature (undergoing evolutionary, not revolutionary change).

DataDirect middleware products build on a mature code base, from shipping ODBC and JDBC drivers for a decade or more. This creates a level of product familiarity that's reflected in the quality of the award-winning customer support program. It also means the drivers and providers are stable and well tested.

Security

The importance of information security is reflected in the functionality of DataDirect middleware products. They provide for code security, database security and secure network communications.

DataDirect ADO.NET data providers and JDBC drivers provide code security because they do not include native libraries. They support several forms of authentication and authorization, including Kerberos. For secure communications, DataDirect middleware supports secure sockets and encrypted sign-on data. Users can employ operating system and Kerberos authentication and authorization to restrict access to databases.

DataDirect Shadow RTE offers several flavors of security and auditing capabilities. It provides security for mainframe Web services and auditing of users of mainframe resources. Using DataDirect middleware, it's also possible to trace and log activity against SQL databases.

The Future of Data Access Middleware

The future of premium-quality data access middleware looks promising, in part because the trend towards distributed data is unabated. Technology companies, for example, are looking to embed sophisticated analysis and business intelligence capabilities in distributed applications. Distributed processing is becoming more pervasive and with it, the need to access disparate databases.

Some of the influences driving the demand for sophisticated middleware to access disparate databases include:

- Virtual organizations
- Business intelligence with virtual data warehouses
- Integration services, enterprise information integration
- Aggregation for SOA and services.

Another major influence in that direction is heightened awareness of regulatory compliance, traceability and recordkeeping requirements. This is likely to spur growth of database and enterprise content management software as companies move to maintain searchable document repositories.

The maturation of business process management will feed the adoption of SOA and Web services. As in the case of the Sabre travel system, SOA might promote a volume increase for transaction processing and other classic database applications. Scalable, performance-oriented data access middleware will be necessary to sustain 24x7 distributed processing. DataDirect Connect middleware fits that need.

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).

"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*

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.

Copyright Notice

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

Index

