Plugging SQL & ODBC Into Integration Frameworks

By Ken North

Because organizations have a diversity of software suites, applications and data sources, integration software has become a focal point for successive waves of innovation. The computing industry has embraced a long list of integration solutions such as Extract, Transform, and Load (ETL), Message-Oriented Middleware (MOM), object brokers, data integration, EAI, Business Process Integration (BPI), and Enterprise Information Integration (EII). Today's developers are looking to solve integration problems with technologies such as the Java Connector Architecture (JCA), XML integration servers, Simple Object Access Protocol (SOAP), public registries, Service-Oriented Architectures (SOAs), and the Enterprise Service Bus (ESB). We're climbing the adoption curve for JCA and SOA, but SQL and Open Database Connectivity (ODBC) still have traction as integration technologies.

Regardless of the architecture that's chosen, many integration scenarios include retrieving or aggregating information from databases, often from disparate data sources. XML has become the lingua franca of data integration and Web services interoperability, but we can accurately describe SQL as intergalactic dataspeak. For various reasons, many leading integration products include software for accessing SQL databases.

Much of the software organizations use has a need for persistent data and information that's available on a recurring basis. Keeping track of millions of data items was a core objective when computer scientists developed Database Management Systems (DBMSes). Today's application integration solutions have a similar need for persistent data; much of it resides in SQL databases. SQL software from IBM, Oracle, Microsoft, MySQL, Sybase, and other companies has a large installed base.

Since publication of the first SQL standard in 1986, SQL has been regularly updated. As the standard and SQL products evolved, SQL dialects have been enriched to provide more support for computation, comparison, aggregation, and processing of rich data types. There are SQL functions for operating with data cubes, tables, scalar quantities, dates, times, security roles, and XML documents. There are arithmetic, statistical, trigonometric, and string processing functions. There are also functions and stored procedures for operating with rich types such as audio, video, and geospatial data. SQL also includes data integrity features such as the ability to apply constraints to data. For many integration projects, the quality of data from SQL sources offers a distinct advantage over integrating data from sources that don't enforce data integrity rules.

SQL and XML

Although we discuss XML and SQL as separate technologies, there's been movement in standards organizations to integrate SQL and XML. The most recent standard, SQL:2003, supports SQL/XML functions and type mappings. The next version of the SQL standard will integrate with the XQuery language (see Figure 1).

Because SQL DBMS products repre-

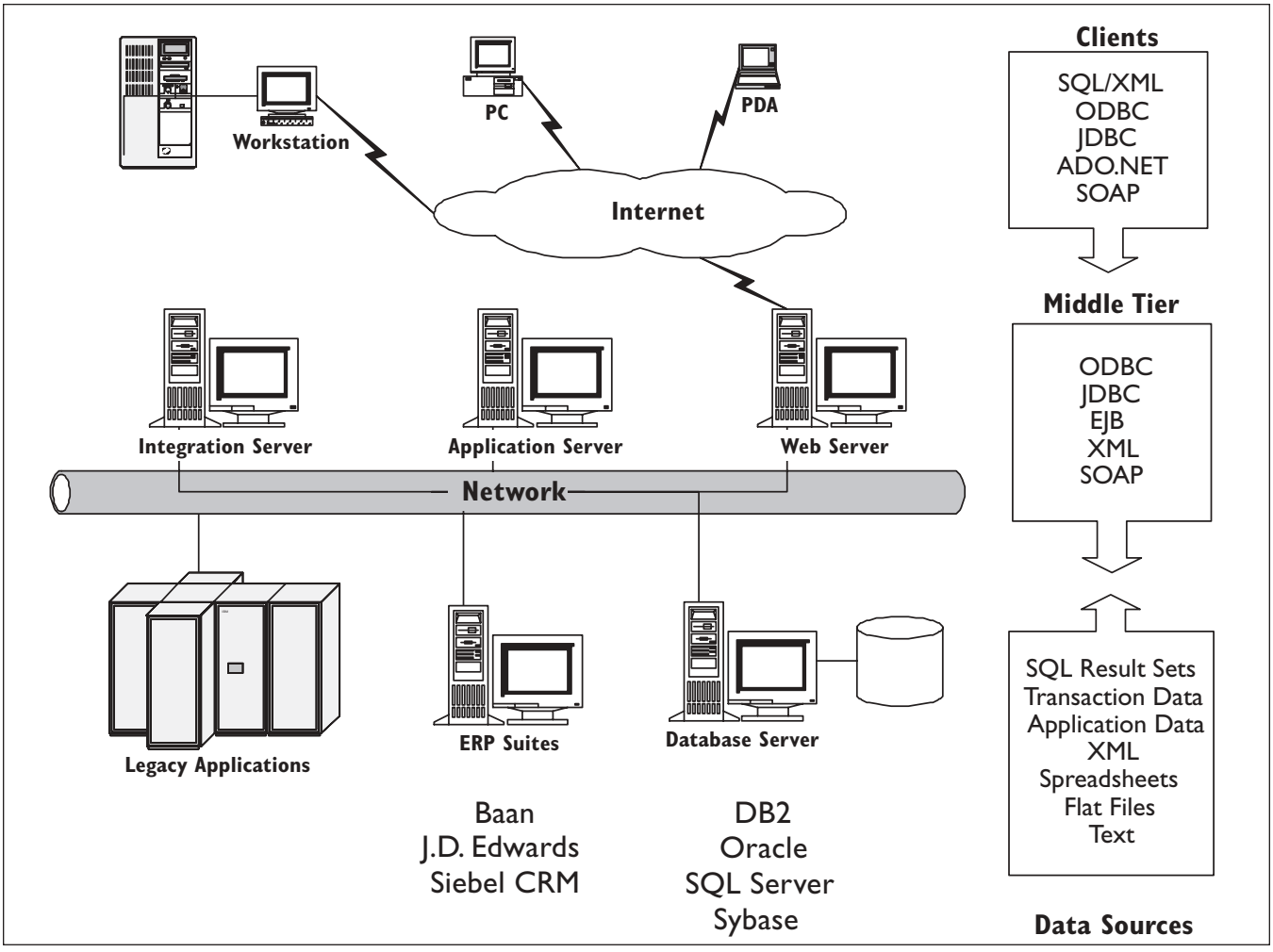

Figure 1: SQL Enables Integration Across Multiple Tiers of an Application and Across Legacy Data Sources and ERP Suites

Today's application integration solutions have a similar need for persistent data and much of it resides in SQL databases.

sent mature technology, there's a pool of experienced users and ample development and management tools. Although their core technology is mature, SQL products continue to evolve as fierce competition forces innovation. On the open source front, MySQL supports transactions and is now part of a bundle that includes the JBoss application server. IBM, Oracle, and Microsoft have integrated their SQL servers with business analytics, messaging middleware, XML processing, and support for Web services. To integrate XML, those companies have added relational views of XML documents, stored procedures, and SQL extensions. They've also added SQL stored procedures for exposing SQL data as Web services.

Because SQL is pervasive, database connectivity is a critical feature of integration servers and product suites. Attunity, BEA, IBM, Infravio, iWay Software, Oracle, Pervasive, XAware, and other integration software vendors provide connectors, adapters, or wrappers for accessing SQL data. Those adapters provide connections to operational data stores, data warehouses, and legacy data sources.

Integrating data from legacy sources presents a challenge. Some legacy applications operate with ad hoc data stores, requiring application-specific logic to access that data. Many mainframe applications use common file formats and mature access methods. The Index Sequential Access Method (ISAM) has been in use since the '60s. Originally developed for IBM mainframes, it has spread to computing platforms of all types and sizes. ISAM is at the heart of numerous legacy software products. A successor to ISAM, the Virtual Storage Access Method (VSAM), is still widely used on IBM mainframes. Integrating information from ISAM files, VSAM files, and proprietary data stores can be daunting if an integration tool must include code for each type of data source. An alternative that presents a uniform interface to legacy data stores is to create drivers for SQL processing over legacy data. That enables applications and integration tools to use the same programming interfaces used with SQL servers. The approach of creating a driver and accessing the data with SQL has been widely used.

Connectors, adapters, and programs that operate with SQL use a proprietary or standard Application Programming Interface (API) and connect to a query processing engine that behaves as a data provider service. SQL engines and database drivers have been developed to process SQL queries against ISAM files, VSAM files, proprietary data stores, spreadsheets, text files, and other data sources. Because SQL engines and extenders provide access to a wide variety of data sources, SQL is a prime solution for integration efforts involving disparate platforms (see Figure 2).

Determining how to best exploit SQL's ubiquity for integration projects involves selecting tools, integration points, and data access technology. For accessing SQL databases, programs can use libraries or classes that implement a proprietary or vendor-neutral API. For Java data access, programs can use JDBC and dynamic SQL, or SQLJ and static SQL. For Windows platforms, programs can use Object Linking and Embedding (OLE) DB and ActiveX Data Objects (ADO). For the .NET environment, programs use a newer version of ADO

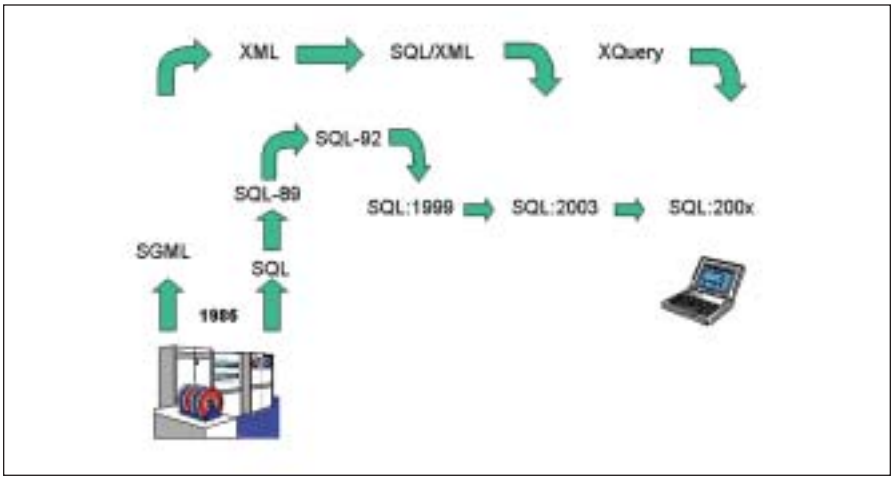

Figure 2: Markup and SQL Standards Have Evolved Since 1986

business integration journal takeaways

BUSINESS

- Using integration solutions that rely on standards, such as SQL and ODBC, is typically cost-effective and less risky than investing in software that uses proprietary technologies.
- Because SQL and ODBC are ubiquitous, there's a large pool of system architects, system designers, and developers with relevant experience.

TECHNOLOGY

- Virtual SQL database technology enables developers to use a familiar paradigm for data aggregation and integration across heterogeneous sources.
- The availability of more than 200 ODBC drivers enables integration products to use connectors and adapters for running SQL queries across an extensive array of data sources (tabular data, spreadsheets, text, XML, and more).

known as ADO.NET. The predecessor to these APIs was the ODBC API. Microsoft aligned ODBC with the SQL Call-Level Interface standard in 1995. Because it's tied to a standard, the major database vendors provide support for ODBC and there are ODBC implementations for Windows, Unix, Linux, and the Macintosh.

ODBC has been widely accepted because it's based on an international standard, is platform-neutral and database-neutral. It lets software developers write code that's more portable than software written to proprietary APIs. Enabling developers to program more easily for heterogeneous environments is one of the attractions of ODBC as a tool for integration projects. Another attraction is that, compared to other platform-neutral SQL APIs, ODBC offers greater connectivity to disparate data sources. There are approximately 112 JDBC drivers, 53 OLE DB providers, and 20 data providers for .NET. There are more than 220 ODBC drivers and ODBC libraries for several programming languages, including C/C++, Perl, Python, and PHP Hypertext Preprocessor (PHP).

ODBC-Enabled Integration Products

Because of ODBC's universal connectivity, it's at the heart of various integration products. In addition, organizations, such as Istante Software, MetaMatrix and Sagent, have developed products to enable desktop applications to treat middleware and business objects as a virtual SQL database. Software, such as Istante Enterprise Link Server, MetaMatrix Server, Sagent OpenLink and XAware XA-iServer, lets desktop applications connect via ODBC to a virtual database that can simplify aggregation and integration.

The Java and Windows developer communities have been steered toward JDBC and ADO.NET, respectively, but ODBC is the prime database API for developers using languages such as Perl, C/C++, and PHP. ODBC also has a strong following in the Linux community, so there's an ongoing demand for ODBC drivers. The driver is the plug-in that enables an integration framework to access data in application data stores. To simplify the process of writing ODBC drivers, many developers have used ODBC driver developer kits. Writing a driver provides the plumbing that lets an ODBC adapter or connector access a data source using SQL.

Dipak Patel, an expert on data access SDKs for ODBC, JDBC, OLE DB, and .NET, comments about the ongoing interest in ODBC: "Enterprises and ISVs have a great deal invested in desktop software, such as Excel, Cognos, Crystal Reports, Brio, and Microsoft Access, and in the training to use these tools. The common denominator among these tools is their ability to see data sources through the SQL lens and to plug into them using ODBC drivers."

He explains: "Next-generation tools that use Web services and XQuery or some other language are coming, but it will be a while before they replace today's approach. To compensate, an approach increasingly being used is to wrapper today's middleware and application servers to look like a virtual SQL database accessed by ODBC or JDBC. This allows existing users to treat data as if it were in an SQL database while the back-end migrates to newer technologies such as Java, .NET, and Web Services."

This approach has proved successful for Independent Software Vendors (ISVs). Istante Software used ATI OpenAccess SDK to expose data from a memory-based, real-time data server to business intelligence software such as Business Objects, Cognos, Crystal Reports, and Microsoft Excel.

Stephen M. Sherman, Istante vice president of Engineering, cited several advantages of supporting SQL and standards as a path to integration: "We have a philosophy of open architecture and utilization of appropriate standards at all levels of our product," he says. "This helps reduce costs to our customers by providing flexibility and ease of integration. Not only can ODBC and JDBC be used to access the current information within our product, they can be used as integration points between the customer's existing infrastructure and our product."

Charlie Morris, CTO of Princeton Financial Systems, discussed the benefit of using SQL wrappers for business information: "Since our product exposes much of its functionality via an object model, involving much computed data vs. static database fields, ODBC and SQL provide a perfect medium through which end users can get at the object-encapsulated business information," he says. "Our system integration and reporting strategies are entirely based upon exposing our proprietary object model via industry-standard interfaces."

Using ODBC to expose data sources to integration middleware is a focus of the company's application integration strategy. "Our ODBC driver is at the heart of our middleware and Web services integration architecture," Morris says.

Another ISV uses SQL to expose information managed by its project management systems. Primavera Systems provides an ODBC interface for a variety of software. Software that consumes data from Primavera project management systems includes planning and scheduling applications, reporting programs, analysis software, Microsoft Office, and integration middleware. Michael McLaughlin notes that Primavera has offered an ODBC interface for four years: "Providing an ODBC interface to Primavera's project management data enabled our customers to access the project management data using simple and readily available query tools." McLaughlin says: "Having an ODBC interface to our data makes our product line more desirable and thus indirectly provides an economic benefit. Primavera is committed to providing access to our project management database via a programmable interface."

Looking Ahead

Integration will remain among the top-10 computing challenges because organizations will continue to have various applications and computing platforms. Process integration, application integration, and information integration will remain a challenge well into the 21st century.

A standards-based strategy, such as plugging SQL adapters into integration frameworks, is a viable, long-term approach for integrating persistent information. The marriage between SQL and XML will support tabular and document-centric queries. For the foreseeable future, ODBC drivers and SQL will represent the most portable solution for accessing disparate data sources and the types of data required for business integration. **bij**

About the Author

Ken North is a consultant, author, industry analyst, and speaker. He teaches Expert Series seminars and is the editor of SQLSummit.com, WebServicesSummit.com, and GridSummit.com. e-Mail: Ken_north@compuserve.com