CHARLES BONTEMPO *and*
GEORGE ZAGELOW

THE IBM Data Warehouse Architecture

How IBM integrates its own and other vendors' tools to derive useful information for decision support.

Users of IBM's data warehouse solution represent a wide range of communities differing with respect to business activities and processes, services and products, resources and cultures, and, most important, business objectives. This diversity has motivated IBM's development of a comprehensive set of data warehouse offerings (hardware and software) and an integrated warehouse solution designed to serve a wide range of requirements—from simple query and reporting to advanced data mining.

IBM is in a unique business position, since the legacy data in many enterprises is stored in files and databases managed by IBM products. Nonetheless, IBM recognizes that many organizations have supplemented these legacy systems with data management tools designed and delivered by other vendors. Therefore, the company's data warehouse architecture includes the ability to integrate data managed by multivendor offerings, as well as interfaces facilitating interoperability with multivendor tools.

VICTOR VASSARELY, TITLE: CALDOR, 1976/OIL ON CANVAS/
ON LOAN FROM THE FORMER VASARELY CENTER TO THE
MUSEUM OF THE AMERICAN HUNGARIAN FOUNDATION

Problems Addressed

The initial version of the IBM data warehouse architecture, published in 1993 [3], was motivated by problems many users experienced when trying to derive useful information for decision support from their operational data sources, including:

- Databases and files designed primarily for relatively short, predictable-update transactions.
- Point-in-time and historical data usually not saved in online transaction processing (OLTP) databases.
- Pre-relational file systems and database managers using complex data structures and navigational programming interfaces.
- Legacy data, developed using different applications at different times, stored with unclear data semantics in incompatible formats.
- Heterogeneous data, inhibiting rapid integration of data needed for decision-support applications, especially data stored using nonrelational structures and formats.

Before the advent of data warehousing software, these problems often meant intolerable delays in the production of reports and query results needed for timely decision making. Although relational database management systems (RDBMSs) with structured query language (SQL) provided substantial data independence and ease-of-use benefits for query and update applications, supporting both operational and decision-support applications with a single database designed for OLTP was only marginally successful, for several reasons:

Table 1. Data warehouse vs. operational database

Data Warehouse	Operational Database
• Subject oriented	• Application oriented
• Integrated	• Limited integration
• Nonvolatile	• Continuously updated
• Stabilized data values	• Current data values only
• Ad hoc retrieval	• Predictable retrieval

- Complex queries (typical of decision-support applications) against highly normalized OLTP databases require time-consuming multitable joins, often yielding unsatisfactory query performance.
- Lock contention for mixed decision support and operational workloads can degrade response times and transaction throughput.
- Increasing requirements for more interesting information involve increasingly complex data analysis.

Even with these issues addressed, most legacy data is nonrelational, motivating further interest in the data warehouse concept. Therefore, these problems have motivated interest in separate data warehouses, as well as in the widely accepted distinction between a data warehouse and an operational (or OLTP) database (see Table 1). See Figure 1 for an abstract representation of major data warehouse components and their relationships at a logical level.

A data warehouse contains summary, historical, and detail data to support tactical and strategic decision

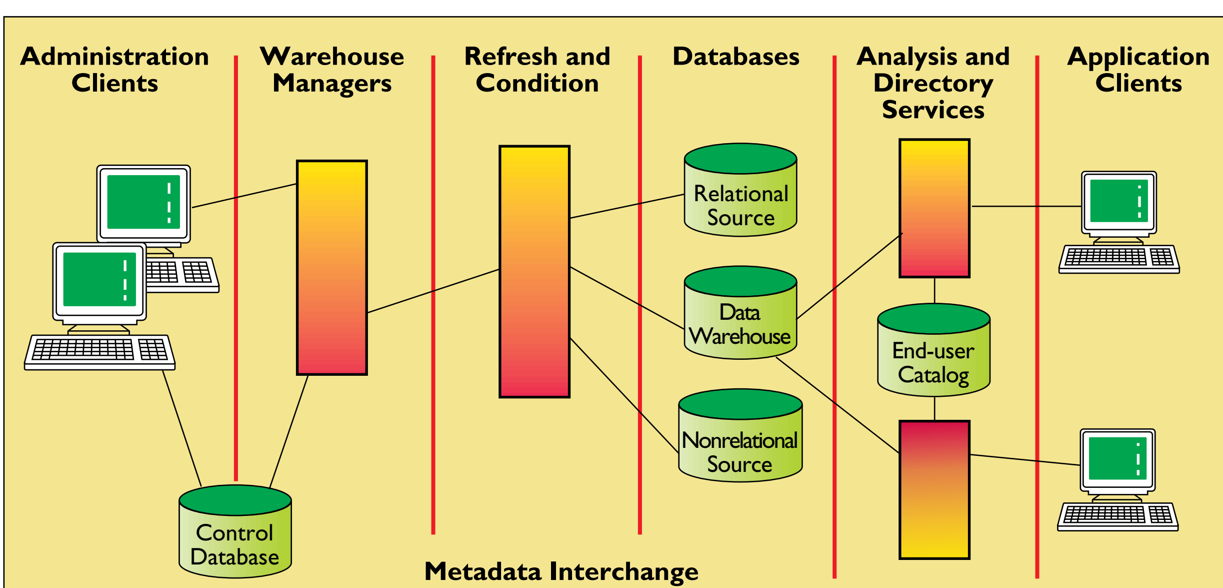

Figure 1. IBM warehouse architecture

making. Data is extracted from operational data sources, transformed, cleansed, reconciled, aggregated, and summarized in preparation for warehouse processing (see Figure 2). Data might be denormalized to achieve acceptable performance for queries. This data is extracted from among several sources: flat files and prerelational and multivendor RDBMSs (internal and external to the organization). This process of conditioning the data for warehouse processing occurs periodically as part of a refresh cycle, typically during a batch window so as to minimize its effect on other processing. When the batch window cannot sustain heavy refresh loads, the data-extraction process can be supplemented with propagation of incremental source-data changes to the warehouse database, depending on demand, schedule, and events [4].

Data Warehouse Topologies

Data warehouse components can be configured using various topologies.

Centralized data warehouse. A data warehouse contains properly conditioned data on subjects of interest to users within an enterprise's multiple business units. A data warehouse supports cross-functional information requirements. A centralized data warehouse can be topologically simple, because it is the sole locus of warehouse data for many users (or clients) and applications throughout the organization. It is the topology of choice for many large data warehouses seeking the advantages of economy of scale and centralized system management.

Data warehouses and data marts. A data warehouse is often contrasted with a data mart, which typically contains a narrower scope of data, characterized by a single subject, a single business function, or even a single application. Data marts are often connected to a centralized warehouse in a three-tier configuration in which clients are connected to specific data marts that draw their data from a data warehouse. This topology exploits locality of reference to provide optimal performance to the warehouse's data-mart clients while allowing user access to warehouse data in order to meet cross-functional information requirements. It also facilitates delivery of cleansed and reconciled data to the dependent data marts supported by the warehouse. Data marts are managed on the second tier of servers in this configuration and often support online analytical processing (OLAP) and more advanced analytic functions.

Distributed data warehouse. This topology consists of network-connected data warehouses with strong distributed processing support. In its more advanced forms, it enables users at any location or clients connected to any data warehouse to work as if the data resided in a single, centralized enterprise warehouse, although it is physically distributed among multiple data warehouses. Such distributed, or virtual, data warehouses require strong capabilities for distributed database management; at its best, this topology should provide users a single-site image of the global warehouse.

When weighing deployment of a distributed configuration, it is important for implementers to take into account the internode database processing that will be needed to meet user requirements. Performance requirements can easily preclude adoption of this topology for applications requiring frequent distributed joins and other distributed set operations. It is feasible only when designers are convinced that remote distributed set operations are required infrequently and for light data loads.

When the warehouse data stores are managed by database management systems (DBMSs) from multiple vendors, the distributed data warehouse topology benefits from use of a multidatabase server (MDBS) that simplifies user access to heterogeneous data sources.

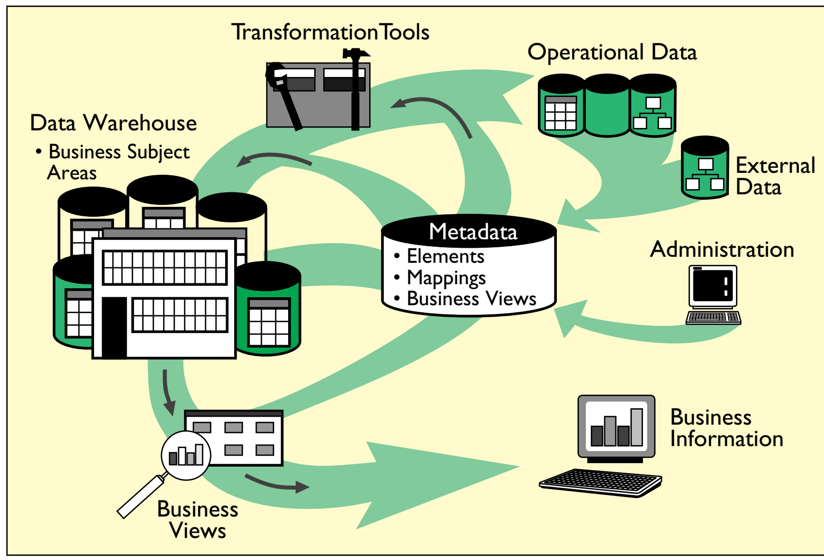

Figure 2. The process of transforming data into information

Hybrid deployment strategy. In most enterprises, the inconsistencies of legacy systems drive warehouse requirements. These legacy systems are often characterized as "islands of data," so while rapid deployment of data marts seems to be the most cost-effective solution to these organizational requirements, system designers must take care to avoid proliferation of independent data marts, each modeled and designed without regard for cross-functional information requirements. Undisciplined deployment of independent data marts can yield fragmented, inconsistent data that seriously inhibits future development of cross-functional information.

Nevertheless, data marts offer the obvious advantages of lower entry costs and faster implementation than a data warehouse, which typically presupposes a prolonged data modeling effort encompassing enterprise-level information requirements. The dynamics of today's markets are changing so dramatically that information supporting decision makers often becomes a critical and urgent requirement. However, resources to generate this information are frequently con-

A Common Sense Development Strategy

Following these seven steps should yield at least six time- and money-saving benefits.

MICHAEL SIGAL

Data warehousing is a great example of a technology that has succeeded in confusing a large number of IT professionals. Complexities (real and imagined) of large-scale data warehousing and conflicts arising from the data mart activities of business units that need the information today and can't wait for IT departments to resolve conflicting objectives can indeed be significant.

It is all too easy to be distracted from achieving the required objectives while sorting out tools, platforms, and databases. To help our clients focus on the tasks at hand while addressing the high cost of a data warehousing solution, we have developed and now practice the following seven-step strategy:

Reduce complexity via standardization. For example, for a large IBM S/390 DB2 server environment, there is no reason to introduce another server technology for data warehousing. Instead, standardization yields significant cost benefits from economy of scale, especially in the area of support. Labor savings alone can be quite dramatic. Additionally, there is a serious shortage of skilled IT people in a market further complicated by Y2K problems, Euro conversions, and the booming U.S. economy. The same is true for Unix shops (AS/400 and others). There is no need to bring in IBM S/390 or Teradata hardware just to manage your company's data warehousing; it can be done using existing platforms and support staff. Servers vary in industrial strength and performance, but there is little difference in functionality today. The days of text-based Unix and mainframe terminals are past, and the same graphical and Web-based tools can be used with most servers.

Leverage technology standardization to other areas of computing. For example, a shop using the Custom Information Control System (CICS) IBM transaction server for online transaction processing (OLTP) and a DB2 server for data warehousing can move a great deal of detailed data (such as keeping three months of history instead of 12 on the operational side) and benefit from smaller databases (offering faster backups, database reorganizations, batch processing, and more). The data is still readily available if needed and could even be in the same DB2 database if operational Sysplex (IBM technology for clustering mainframes on S/390 technology) is merged with data warehousing Sysplex (unique to the S/390). For example, Southern California Edison, an electrical utility in Rosemead, Calif., benefited from combining its data warehousing and disaster recovery plans using its data warehousing Sysplex and IBM Extended Remote Copying (XRC) in another data center used as a disaster recovery site—and thus avoided having to invest in both.

Avoid the purist approach; be practical. Many aspects of data warehousing technology (such as metadata) are still maturing, although lack of maturity should not stop you from realizing the business benefits of data warehousing today.

Don't try to build an entire corporate data warehouse at once. An approach that creates one strained by costs, time, and skills. For many organizations, business objectives and resource constraints favor the faster bottom-up data mart deployment strategy combined with a top-down, high-level data model. Such a hybrid deployment approach begins with development of one or more data marts on a staged basis. Capabilities for cross-functional processing are anticipated during the planning phase and are implemented incrementally as the system grows. (See the sidebar by James Sutter on Ryder System Inc.'s project-based enterprise data warehouses.)

System Objectives

Today's successful warehouse users experience rapid increases in data volumes, application workloads, and numbers of concurrent users. Meanwhile, the value of unformatted or complex data is increasingly recognized by warehouse users. And many warehouse applications are increasingly viewed by business and technology managers as mission-critical, since the information they yield is crucial to gaining (or sustaining) an organization's competitive edge. These observations point to the key technical objectives of the IBM warehouse solution: performance, scalability, manage-

subject area at a time is much more likely to succeed, given that you keep the subject areas all in one place (for cross-functional analysis) and coordinate development activities (so new subject areas benefit from subject area systems already developed that share the same characteristics, such as naming conventions and data attributes).

Don't be distracted by data marts. Data for a data mart (if one is truly needed) should be extracted from the data warehouse. Don't confuse creating a data mart (a departmental implementation optimized for specific needs, usually with predefined access) with a data warehouse (mostly atomic data), whose purpose is to hold all of your corporate data of record. Trying to join several data marts together does not yield a virtual data warehouse and is rarely useful for ad hoc and data mining analysis needs that were not considered during the data marts' design. For cross functionality and trend analysis with good performance, you need a real data warehouse.

Don't get locked into a proprietary solution. Keep the environment open to third-party tools (such as online analytical processing, query, data transformation, data mining, and Web servers). No single-vendor solution can be expected to compete with a best-of-breed approach for tools any time soon.

Identify availability requirements the data warehouse will have to meet in a few years. New value-added applications developed rapidly around the data warehouse tend to become mission-critical. It may be prudent to require them to have the same availability you expect from your OLTP applications.

If you manage to succeed with standardization, you will also realize other benefits:

- Time that would have been spent evaluating, selecting, and learning new technologies will be available for putting technology you already have to good use.
- Stakeholders (IT groups responsible for traditional applications and data) who normally see data warehousing as trespassing may embrace smaller databases and will be rewarded with a reduced backlog, due to the shifting of some applications and reporting responsibilities to the data warehouse.
- Your company will save a lot of money on labor, software, and hardware.
- Having fewer different technologies will result in faster and less costly implementations, because there will be fewer interfaces and incompatibilities to worry about.
- The number of project failures will be reduced, because of better familiarity with technology and its capabilities.
- There will be less temptation for business units to finance data marting when it is not called for.

Please remember that your data warehouse still has to be built (it can't be acquired) and that this is is no small task. However, for most companies, there is little choice; business needs dictate rapid change, and traditional application development is too slow, no matter which tools you use. C

Michael Sigal (mikesigal@lccidata.com) is president and chief technologist of Life Cycle Consulting, Inc., a database technology and data warehouse consulting firm in Torrance, Calif.

ability, availability, and extensibility.

The degree to which these objectives are achieved depends on the database management system support the solution provides. Open, documented interfaces, including application programming interfaces, are essential for deploying best-of-breed warehouse components.

RDBMS Warehouse Support

The original IBM warehouse architecture specified an RDBMS for the storage and management of warehouse data. The simplicity of its data structure and its high-level, set-oriented languages are among the features that make relational ideal for warehouse query applications. RDBMS language processors map SQL statements into low-level operations that can be optimized for parallel processing to achieve improved query performance (speedup) and facilitate incremental growth for increased workloads while sustaining required levels of performance (scale-up) [2].

IBM's DB2 Family of RDBMSs offers parallel processing support for data warehouses on a wide range of platforms, with shared-data and shared-nothing architectures, with tightly coupled, loosely coupled, and clustered configurations. These systems can support small to very large (multiple-terabyte) data warehouses. To serve diverse user communities, they run in a wide range of operating environments, including Windows NT, OS/2, OS/400, Unix, and OS/390, each exploiting interquery and intraquery parallelism. The DB2 Universal Database (UDB) and DB2 for AS/400 support partition and pipeline parallelism. Collectively, the DB2 Family provides additional support for data warehouse performance and scalability:

- Parallel-aware, cost-based search optimizers that exploit a wide range of database statistics
- Intelligent partitioning
- Parallel database operations, including (but not limited to) parallel table and index scans, joins, backup/recovery, and utilities
- Specialized indexes and index processing
- SQL extensions to support OLAP, including cube and rollup operations[1]
- Star query processing for optimizing searches against a star schema
- Support for outer joins to denormalize tables for warehouse use
- Query rewrite, including predicate transitive closure to enhance query performance
- 64-bit computing (DB2 UDB and DB2 for AS/400)
- Integrated replication function (DB2 UDB and DB2 for AS/400)

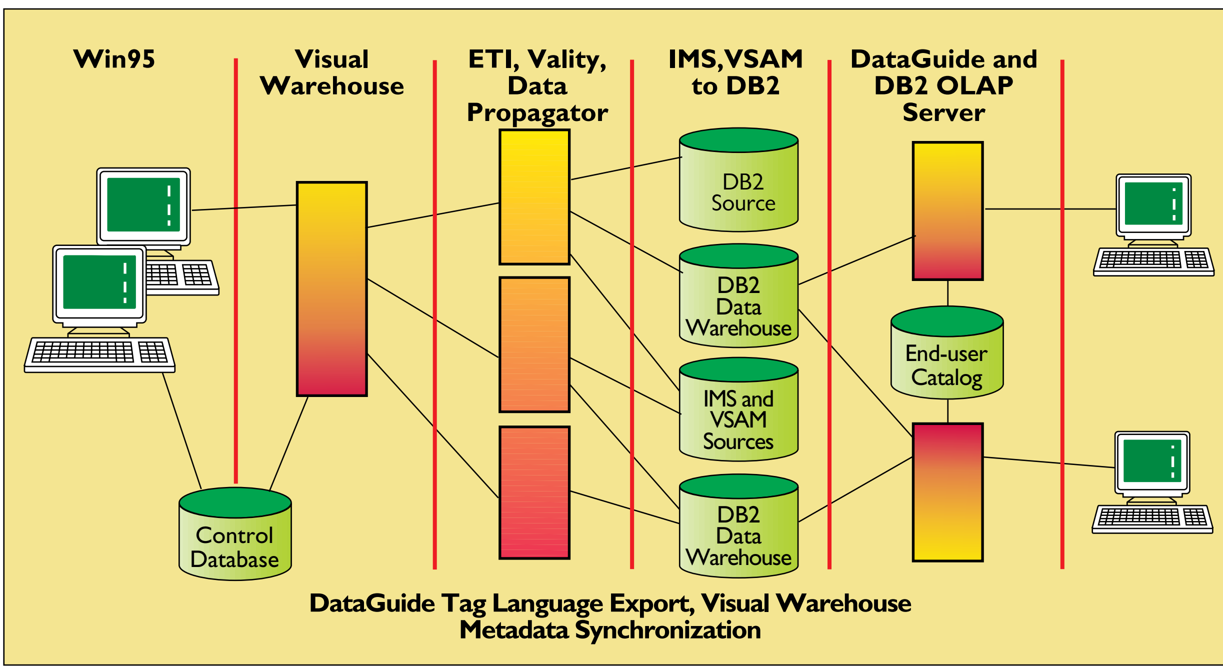

Figure 3. IBM DB2 OLAP server (fully compatible with Essbase-ready tools)

[1]Currently supported only by the DB2 Universal Database, Version 5.

• Support for the client/server paradigm with open call-level interfaces

DB2 DBMSs use various forms of balanced tree indexing, including B-trees, to enhance data access.[2] When multiple indexes are scanned in support of a single query, DB2 UDB and DB2 for OS/390 supplement their indexed access by performing Boolean AND and OR operations on row identifiers (RIDs) returned from index leaf nodes. Query logic is thus resolved against returned RIDs rather than against retrieved rows, substantially reducing I/O processing.

The DB2 UDB query processor supplements its use of B-tree indexes with dynamic bitmaps. UDB dynamic bitmaps use Bloom filters to resolve query logic for predicates with conjoined search keys [1], offering the following advantages over the use of conventional, static bitmaps:

The importance of quality data in a warehouse environment cannot be overstated.

• Support for both high- and low-cardinality columns
• Avoidance of excessive storage management costs for large static bitmaps
• No need for special update processing for stored bitmaps
• No need for compaction and decompaction of very sparse bitmaps
• Simplified index selection during database design

UDB dynamic bitmaps are also used to process star queries requiring joins of dimensions and large fact tables.

Database Middleware

An MDBS can be deployed to supplement the DBMS supporting warehouse processing in an environment requiring frequent access to distributed, heterogeneous data sources. Two key design points for this component are a high-performing, distributed database optimizer and a global catalog. It should also provide network and operating system transparency. IBM's MDBS—DB2 DataJoiner—includes a global database optimizer designed for efficient support of distributed joins, views, Cartesian products, and the usual set operations on data in multivendor databases.

The architectural objectives for these key warehouse functions include support for:

• A wide range of data sources
• Propagation of both incremental changes and full extracts
• Source data transformations
• Data cleansing
• Documented SQL and other open interfaces for third-party and user-defined code
• Targets and sources with unpredictable connectivity

The IBM replication tools—DataPropagator Relational, DataRefresher, and Data Propagator NonRelational—support automatic data extraction and incremental change capture from IBM's Information Management System (IMS), Virtual Storage Access Method (VSAM) files, any DB2 DBMS, and several non-IBM DBMSs (when used with DB2 DataJoiner). In addition to their extract/refresh functions, these tools also stage the source change data to SQL-defined tables, where the source change data can be manipulated using SQL statements to produce derived and denormalized data for delivery to one or more data warehouses. Staging tables also enable automatic propagation of time series and other trend data to target warehouses based on subscriber delivery schedules, on demand, or on some event.

Incremental changes are captured from the source database log, perhaps directly from the log buffer, in order to enhance performance, avoid contention with concurrent source database processing, and avoid application rewrite. (For source data managed by non-IBM trigger-based RDBMSs, these replication tools can be integrated with DB2 DataJoiner, which automatically generates the triggers required by these non-IBM systems for capturing incremental changes.)

Replication tools based on a "pull" model are generally preferred for optimal performance in warehouse environments. However, to accommodate mobile users with unpredictable connections, these replication tools support both push and pull models, as well as scheduled, event-based, and continuous change propagation.

To supplement these tools, IBM has formed a

[2]DB2 for AS/400 supports radix balanced trees implemented at the lowest levels of the OS/400 operating system.

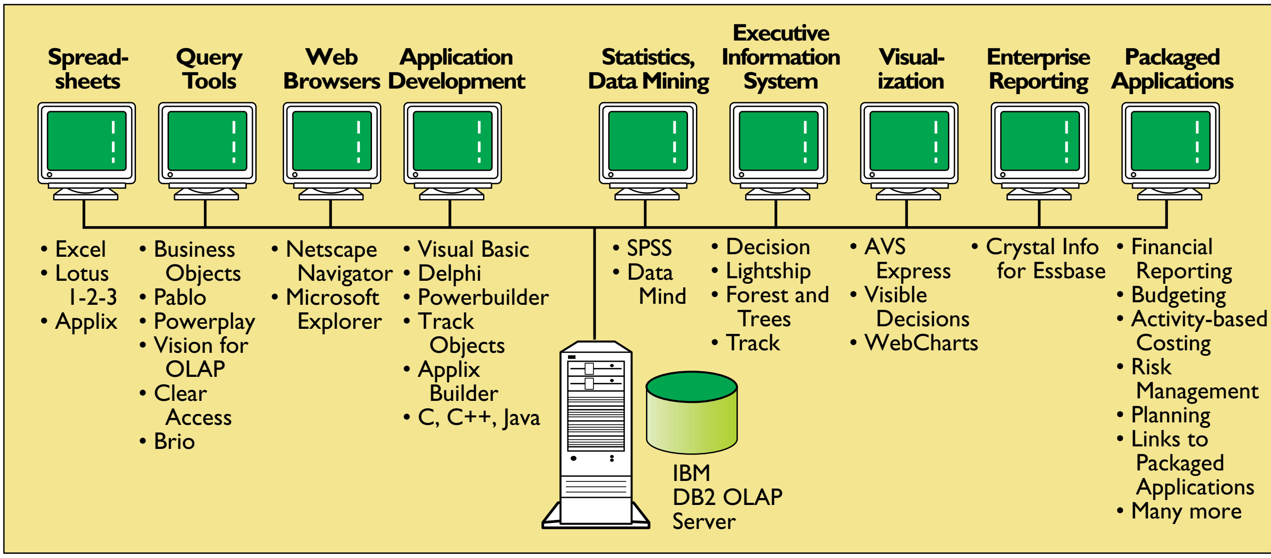

Figure 4. A visual warehouse implementation

strategic partnership with Evolutionary Technologies International of Austin, Tex., whose Extract tool generates 3GL programs to transform, consolidate, and extract data from virtually any data source to virtually any target warehouse, enabling data transformations not easily expressible using SQL.

The importance of quality data in a warehouse environment cannot be overstated. IBM has also formed a strategic partnership with Vality Technology of Boston, whose Integrity data reengineering environment performs data cleansing. Data cleansing that relies solely on rule-based dictionaries, lookup tables, and program-specified conditional logic might fail to anticipate some data defects. For this reason, Integrity relies on lexical analysis, pattern matching, and statistical analysis to identify data contamination and infer relationships for text as well as for fixed field data.

To support OLAP, many users perform multidimensional analysis of warehouse data mart data. The market has seen a proliferation of multidimensional database tools representing two approaches to the storage and management of their supporting databases: multidimensional OLAP (MOLAP) and relational OLAP (ROLAP). MOLAP uses specialized data structures designed for performance; ROLAP enables greater scalability and full-function DBMS support, since it relies on conventional RDBMSs for storage and management of warehouse data, usually organized as a star schema.

IBM has formed yet another strategic partnership with Arbor Software of Sunnyvale, Calif., to integrate its Essbase OLAP engine with RDBMSs. This offering—the DB2 OLAP Server—provides the full OLAP capabilities of Essbase, including data aggregation; cross-dimensional, procedural, and matrix calculations; and SQL access to the star schema RDBMS. To facilitate administration of a star-schema database, it creates, populates, and manages tables, indexes, and summary tables automatically. This solution benefits from the open Essbase interfaces, which permit interoperability with a broad range of user tools (see Figure 3).

Information Catalog and Metadata Management

In order to make full use of their data warehouse resources, users need assistance identifying, locating, and accessing data and information sources. IBM's DataGuide is an end-user-oriented tool that catalogs these sources and manages their metadata. It also manages descriptive data on information-bearing objects, such as reports, queries, spreadsheets, documents, and images, as well as on data objects, such as tables, views, and attributes of the DB2 family and of other non-IBM RDBMSs.

Data warehouse tools generally do not share common schemes for representing metadata. Addressing this problem, DataGuide performs metadata interchange using a documented Tag Language format and includes extractors for the importation of metadata from more than 30 warehouse components, including Extract, Essbase, and many non-IBM tools for querying, reporting, and analyzing warehouse data.

Information catalog objects are hierarchically organized into business-related groups that can be searched using business-oriented search terms (aided by a DataGuide glossary) or by navigating through its graphically represented hierarchy of groups. Once the desired object is identified, DataGuide can launch an application or tool automatically to retrieve and process the described data.

Administration of these key tasks is supported by a graphical user interface to aid the populating and organizing of the catalog. The tool provides documented application programming interfaces to enable catalog access and metadata interchange by user applications and by any non-IBM decision-support tool.

Metadata interchange is a key element of IBM's warehouse initiative and is a key challenge for the entire industry. The Metadata Coalition, formed in 1995, promotes standardization of the interchange of warehouse object definitions and published the Metadata Interchange Specification (MDIS). IBM extended DataGuide to support MDIS in February 1998.

A Packaged Solution

Recent experience by warehouse implementers points to the value of carefully selected packaged solutions for data warehouse (and data mart) support. Therefore, IBM developed Visual Warehouse as an integrated, rapid start-up data warehouse and data mart deployment and management system. Visual Warehouse performs key data warehouse functions: data mapping, extraction from heterogeneous data sources, process scheduling, and warehouse operation monitoring. It also automates a number of data warehouse administrative functions:

- Controlling one or more target warehouse databases managed by DB2 (see Figure 4)
- Extracting source data from the DB2 family of databases, as well as from IMS, VSAM, flat files, Oracle, Sybase, Microsoft SQL Server, Informix, DB2 DataJoiner, and any data source with an open database connectivity (ODBC) interface driver
- Maintaining the information catalog and managing metadata interchange with DataGuide, DataAtlas (an IBM data modeling tool), and a number of other tools
- Providing Lotus Approach for queries, charts, and reports from warehouse data
- Providing access to warehouse databases from Internet and intranet browsers with IBM Net.Data, a product for developing applications that exploit Web-based resources, including databases
- Enabling client connectivity with system-provided ODBC drivers and IBM's Client Application Enablers connectivity tools
- Providing consulting services

IBM's packaged solution includes open interfaces, enabling administration and control of Extract, Integrity, Essbase, and relevant IBM tools. Visual Warehouse can schedule and monitor the tools' operations by time or event or initiate them through user-defined triggers.

Information requirements are specified using SQL-defined business views. Once defined by the warehouse administrator, the Visual Warehouse Manager component schedules the extraction, cleansing, and transformation of source data into tabular format for subsequent query and analysis through user-selected tools. These business views are refreshed automatically according to administrator-specified schedules.

> **Successful data** warehouse environments point up the importance of scalability, performance, wider windows of system availability, and a strong skills base.

Although originally offered as a local-area-network-based workgroup solution for small data warehouses (and data marts), Visual Warehouse was extended in February 1998 to support larger warehouses. Target warehouses can now be supported by DB2 on several Unix platforms, OS/2, Windows NT, AS/400, and System 390. The Visual Warehouse Manager component is available for OS/2 and Windows NT.

Object/Relational Database Support

Increasing interest in Web-based access to data warehouses has increased the information value of unformatted data. The original IBM warehouse architecture included data warehouse support for unformatted or complex data, including text, video, image, and audio. DB2 UDB provides this support in the form of large objects, user-defined types, user-defined functions, and relational extenders. Strong typing is enforced. User-defined functions (which can be fenced) are developed using object-oriented languages, including Java.

DB2 UDB relational extenders are system-provided user-defined functions that support SQL-specified, content-based searches of complex data. Today's data warehouses must provide integrated support for content search and delivery. For this reason, the object/relational features required for these warehouse applications are an integral capability of the DB2 UDB.

What's Next?

IBM is pursuing near-term advances based on user experience. For example, successful data warehouse environments reinforce the importance of scalability, satisfactory performance, wider windows of system availability, and a strong skills base.

It is essential to recognize the importance of design features enabling tight integration of data warehouse components to simplify deployment and administration. The company also recognizes the essential role played by skilled and experienced consulting and services personnel in realizing the full potential of these business intelligence solutions. C

References

1. Bloom, B. Space-time trade-offs in hash coding with allowable errors. *Commun. ACM 13,* 7 (July 1970), 422–426.
2. Bontempo, C. and Saracco, C. *Database Management: Principles and Products*. Prentice Hall, Upper Saddle River, N.J., 1995.
3. IBM. Information Warehouse Architecture 1 Rep. SC26-3244, IBM Corp., Armonk, N.Y., 1993.
4. Zagelow, G. Data warehousing—Client/server for the rest of the decade. In *Building, Using, and Managing the Data Warehouse*, R. Barquin and H. Edelstein, Eds., Prentice-Hall PTR, Upper Saddle River, N.J. 1997.

Charles Bontempo (bon@west.poly.edu) is an independent consultant and adjunct professor of computer science at Polytechnic University in Brooklyn, N.Y.
George Zagelow (zagelow@us.ibm.com) is a programming consultant in the Data Management Architecture and Technology group at IBM's Santa Teresa Laboratory, Software Solutions Division, in San Jose, Calif.