

Information integration: A new generation of information technology

by M. A. Roth
D. C. Wolfson
J. C. Kleewein
C. J. Nelin

The explosion of the Internet and e-business in recent years has caused a secondary explosion in the amounts and types of information available to enterprise applications. Industry analysts predict that more data will be generated in the next three years than in all of recorded history. Because the adoption of Internet-based business transaction models has significantly outpaced the development of tools and technologies to deal with the information explosion, many businesses find their systems breaking under the sheer volume and diversity of data being directed at them. The challenge facing businesses today is *information integration*. Enterprise applications must interact with databases, application servers, content management systems, data warehouses, workflow systems, search engines, message queues, Web crawlers, mining and analysis packages, and other enterprise integration applications. They must use a variety of programming interfaces and understand a variety of languages and formats. They must extract and combine data in multiple formats generated by multiple delivery mechanisms. Clearly, the boundaries that have traditionally existed between database management systems, content management systems, midtier caches, data warehouses, and other data management systems are blurring, and there is a great need for a platform that provides a unified view of all of these services and the data they deliver.

If current trends continue, more data will be generated in the next three years than in all of recorded history,¹ and the widespread adoption of the Internet has made nearly all of it just a URL (uniform resource locator) away. This explosion of information presents an exciting cross-industry business opportunity. Enterprises that can quickly extract critical nuggets of information from the sea of accessible data and transform them into valuable business assets are in a strong position to dominate their markets.

Such an undertaking is no small task. Today's enterprise systems extend beyond the walls of the corporate data center to include customers, suppliers, partners, and electronic marketplaces. These systems interact with databases, application servers, content management systems, data warehouses, workflow systems, search engines, message queues, Web crawlers, mining and analysis packages, and other enterprise applications. They require a variety of programming interfaces (ODBC [open database connectivity], JDBC** [Java** database connectivity], Web services, Java objects, J2EE** [Java 2 Platform, Enterprise Edition]) and work with a variety of data models and languages (SQL [Structured Query Language], XML [Extensible Markup Language], XPath [XML Path Language], WSDL [Web Services Description Language], SOAP [Simple Object Access Protocol]). These systems combine data with different time prop-

©Copyright 2002 by International Business Machines Corporation. Copying in printed form for private use is permitted without payment of royalty provided that (1) each reproduction is done without alteration and (2) the *Journal* reference and IBM copyright notice are included on the first page. The title and abstract, but no other portions, of this paper may be copied or distributed royalty free without further permission by computer-based and other information-service systems. Permission to *republish* any other portion of this paper must be obtained from the Editor.

erties, such as real time (stock tickers, news wires), current (databases), slightly out-of-date (mid-tier caches, database replicas), and historical (data warehouses). Data are generated by multiple delivery mechanisms (applications, databases, files, CD-ROMs, continuous data feeds, e-mail, Web downloads) in a variety of formats (relational tables, XML documents, images, video, sound).

The challenge for enterprise software applications today is *information integration*. Information integration is a technology approach that combines core elements from data management systems, content management systems, data warehouses, and other enterprise applications into a common platform. In this paper we characterize the information integration challenge and describe the components of an architecture that provides an end-to-end solution for transparently managing the volume and diversity of data and data management systems in the marketplace today.

The foundation of the platform is a robust data management system that provides relational and native XML data stores as well as access to a “federation” of rich content servers. The foundation tier exploits decades of database technology evolution to meet the storage, retrieval, scalability, reliability, and availability challenges associated with robust data management. Tightly coupled with the foundation is an integration services tier that draws on parallel technology advances in business intelligence, content management, and business process integration that ease the task of developing and managing complex enterprise applications. A unified interface layer provides a standards-based programming model and query language to the rich set of services and data provided by the foundation and integration tiers.

The next section of this paper briefly reviews the evolution of data management infrastructure and enterprise software infrastructure. The following section, through scenarios from a diverse set of industries, illustrates the scope of the information integration challenge and sketches out the requirements for a solution. The following two sections present the components of a next-generation enterprise software technology platform to satisfy these requirements, and show how this platform addresses the requirements outlined earlier to provide an end-to-end information integration solution. The final section provides a summary.

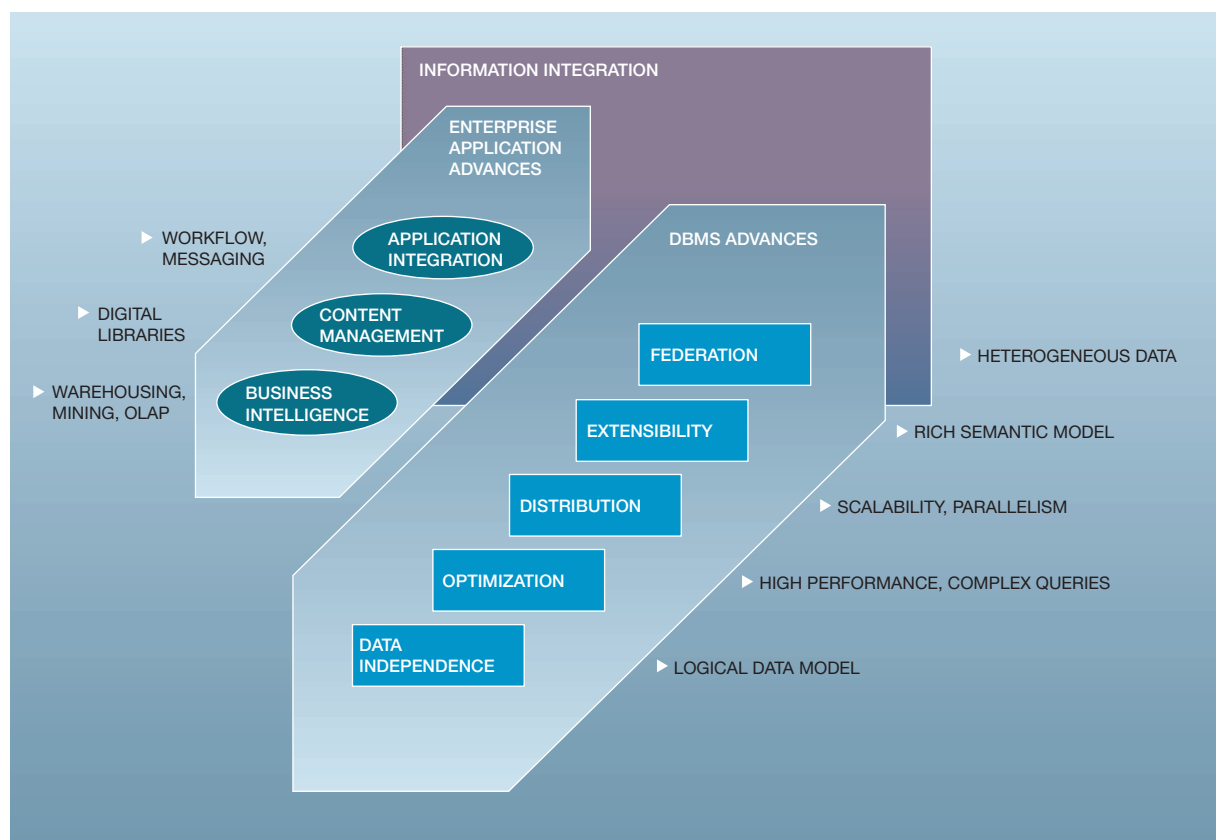
Enterprise software technology evolution

For the past 30 years, data management systems have been at the core of enterprise software infrastructure,² and Figure 1 captures their evolution. The introduction of the relational data model^{3,4} and the concept of *data independence* in the early 1970s revolutionized the data management industry and quickly overtook network and hierarchical systems for large-scale data management. Over the next two decades, with outstanding leadership from IBM's research and development teams, the relational database was transformed into a high-performance, high-volume query processing engine through key innovations in storage and retrieval techniques, concurrency control and recovery, and query processing technology such as cost-based *optimization*.⁵ *Distributed* systems and parallel processing techniques enabled global enterprises to manage large volumes of distributed data.⁶ *Extensible* database architectures allowed data types, access strategies, and indexing schemes to be easily introduced as new business needs arose.⁷ *Federated* database technology provided a powerful and flexible means for transparent access to heterogeneous, distributed data sources.^{8,9}

As database management system (DBMS) technology evolved to manage business data, other important technologies evolved in parallel to make the tasks of managing business logic and business processes easier. Data warehouses, data mining, and OLAP (on-line analytical processing) technology¹⁰ added *business intelligence*, providing a means for discovery-driven and hypothesis-driven analysis over business data to identify trends and patterns and play out “what-if” scenarios. Digital libraries and *content management* systems¹¹ evolved to manage large stores of digital media, providing check-in and check-out services, rights management, and hierarchical storage migration. Messaging systems and message brokers¹² provided the infrastructure for *enterprise application integration*, allowing autonomous applications to communicate with each other in a scalable, asynchronous manner. Workflow systems¹³ helped to automate business processes, providing the infrastructure to manage processes such as order fulfillment from step to step, assign tasks in the process to the right resources, and invoke automated steps at the appropriate times.

Business applications evolved alongside data management and enterprise management systems, exploiting the best features of both to create sophisticated software packages that form the foundation

Figure 1 Advancing toward information integration



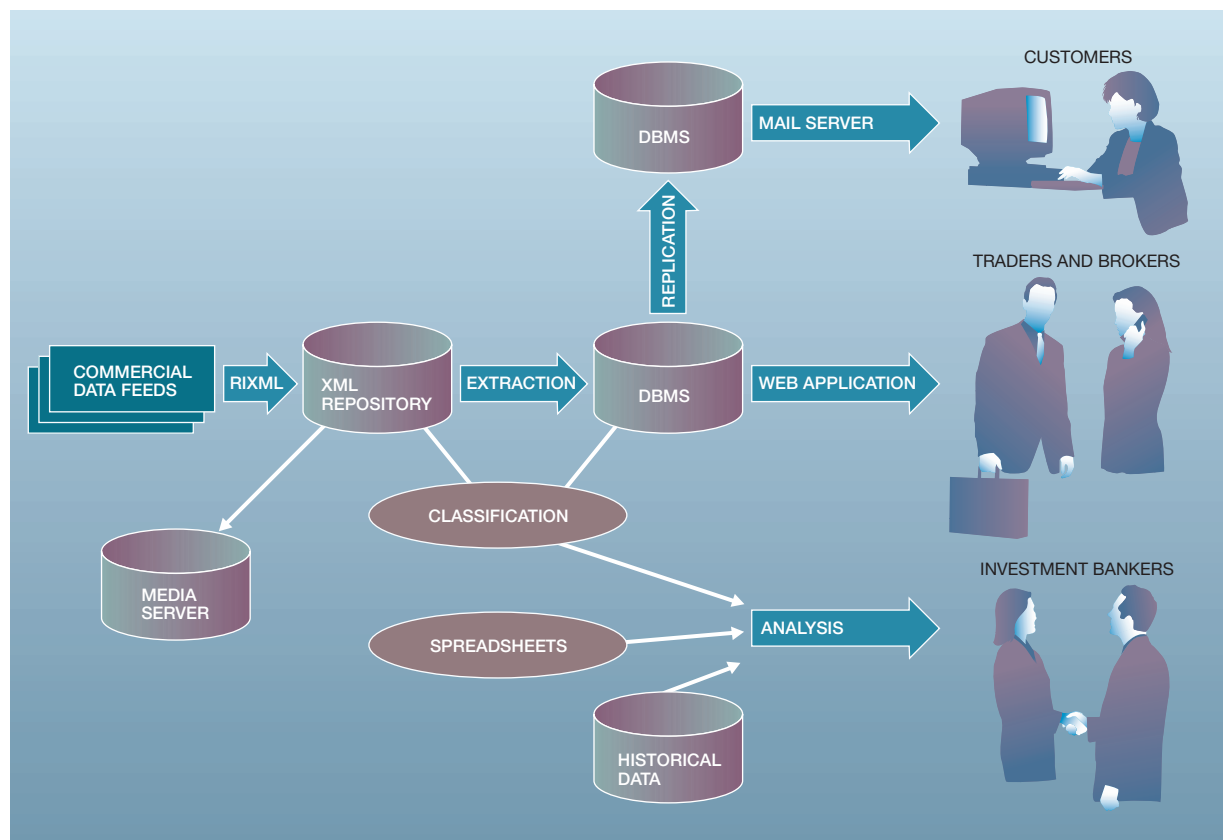
of nearly all of today's businesses. Historically, the task of integrating these systems has been the burden of the enterprise application developer. However, this model is no longer sustainable. Legacy custom-built integration systems cannot keep up with the scalability and reliability requirements introduced by the widespread adoption of the Internet. Significant development and computing resources are devoted to transporting the data from one system to another, and transforming the data from one format to another. To compete in today's "24 × 7" global marketplace, enterprise applications require a new *information integration* platform that combines the advances in enterprise data management over the past 30 years into a single unified interface.

Many commercial systems and academic projects have addressed various aspects of a comprehensive information integration platform. Typically, many of these approaches start "from scratch," building a spe-

cial-purpose system optimized for a particular use. For example, products such as Tamino¹⁴ and Ipedo¹⁵ promise to deliver data stores optimized for XML documents. Likewise, data federation has a solid research foundation and several commercial implementations. Projects such as TSIMMIS,¹⁶ DISCO,¹⁷ HERMES,¹⁸ and the Information Manifold¹⁹ built special-purpose query processing systems to explore various aspects of federated database technology such as mediation, compensation, and scalability. Products such as Nimble,²⁰ Callixa,²¹ and InfoShark²² federate data outside the database engine. Garlic⁹ and DB2* (Database 2*) Relational Connect⁸ extend a traditional relational database engine with federated capabilities, and DiscoveryLink*²³ is a commercial application built on this technology tailored to the life sciences community.

Indeed, the success of DiscoveryLink validates the commercial value and robustness of an information

Figure 2 Financial services scenario



integration approach that *starts* with a commercial database system. As illustrated in Figure 1, an extensive worldwide investment in DBMS technology exists today for good reasons. Databases deal quite naturally and robustly with the storage, retrieval, and reliability requirements for traditional business data, and can easily accommodate a variety of data and access patterns. We believe that a platform that exploits and enhances the DBMS architecture at all levels is in the best position to provide robust end-to-end information integration.

The challenge

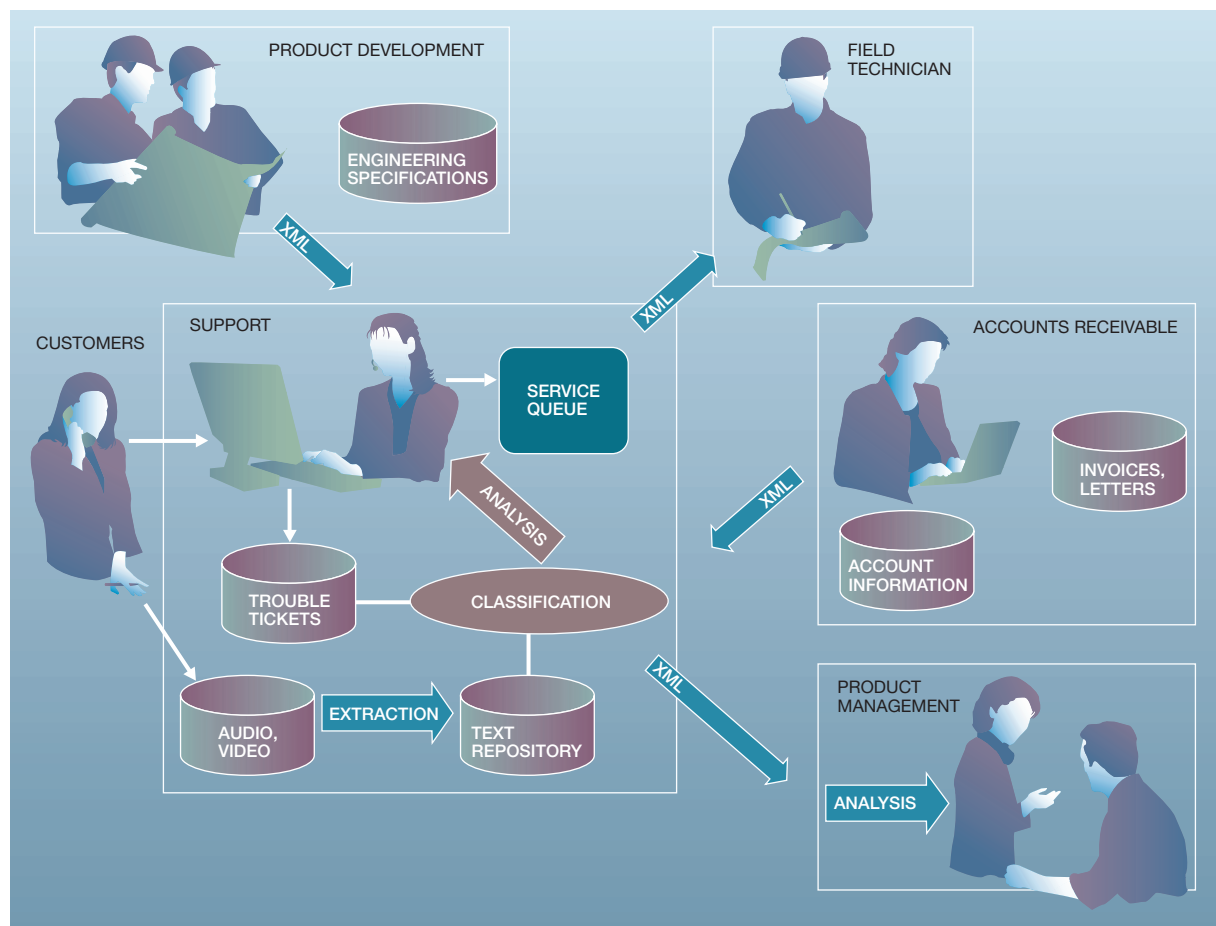
Current trends indicate that information integration is a cross-industry challenge. Industries from financial services to manufacturing have all been touched by the volumes and diversity of data and continual demands introduced by an Internet-based business model. In this section, we present scenarios from

three diverse industries to illustrate the integration challenge, and from these scenarios we draw out common requirements for a solution.

Financial services. Our first scenario is for a financial services company that subscribes to several commercial research publications.² These publications provide data formatted in RIXML (Research Information Markup Language), an XML vocabulary that combines investment research with a standard format to describe the report's meta-data.²⁴ Reports may come from a variety of providers, both in-house and external, and may be delivered via a variety of mechanisms, such as real-time message feeds, e-mail distribution lists, Web downloads, and CD-ROMs. Figure 2 shows how such research information flows through the company.

When a report is received, it is archived in its native XML format, and the audio and visual clips are sent

Figure 3 Telecommunications scenario

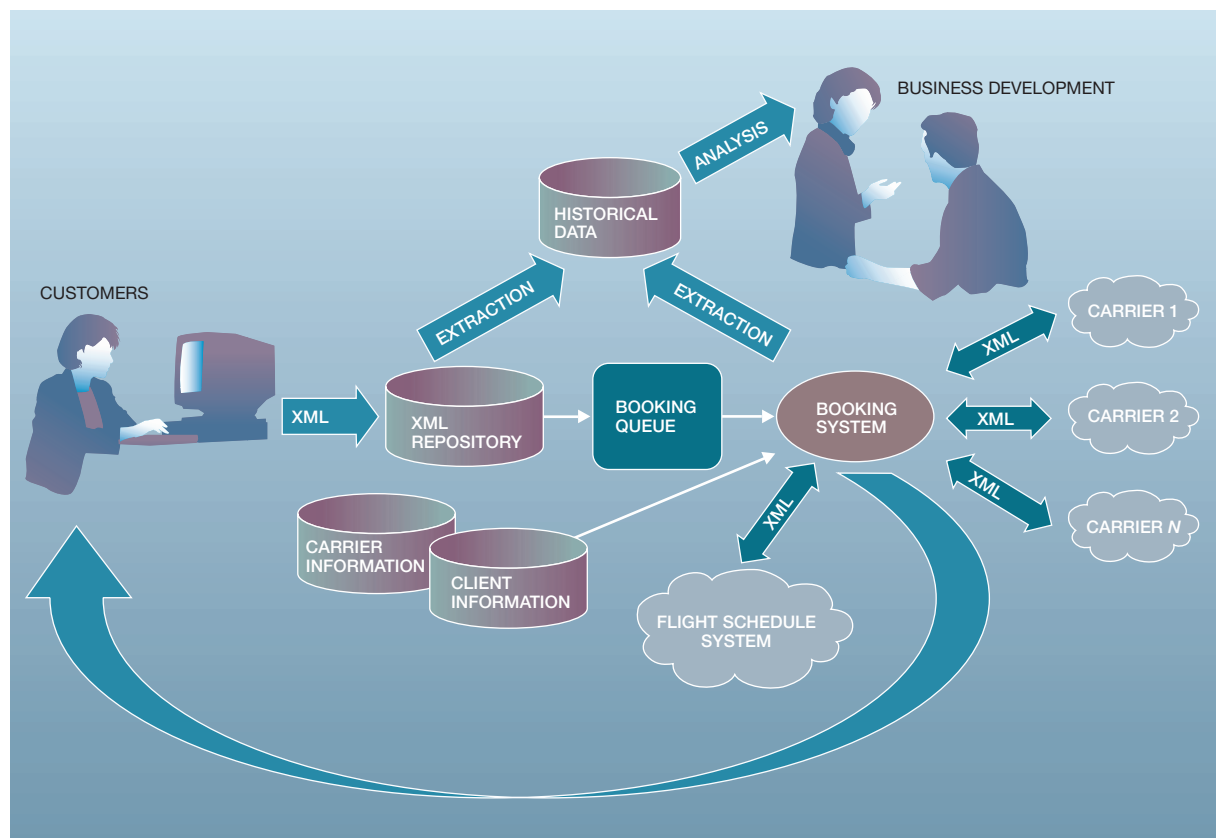


off to an appropriate media server. Next, important meta-data, such as company name, stock price, and earnings estimates, are extracted from the document and stored in relational tables. The company has a global presence, so the information is immediately replicated across multiple sites. This information is used to detect and recommend changes in buy/sell/hold positions to equity and bond traders and key customers. Mining applications more thoroughly analyze the original document and its extracted meta-data, looking for such keywords as “merger,” “acquisition,” or “bankruptcy” to categorize and summarize the content. The summarized information is combined with historical information and made available to the company’s market research and investment banking departments. These departments combine the summarized information with financial

information stored in spreadsheets and other documents, to perform trend forecasting and identify merger and acquisition opportunities.

Telecommunications. A telecommunications company that has completed mergers with several other companies relies on a custom-built customer service system to provide support for its clients. This system is made up of multiple call centers that were acquired as part of the mergers, and is shown in Figure 3. When a customer call is received, it is routed to an appropriate center, recorded, and stored in an audio server. The audio data are also translated into a text file and stored in a text repository. To service the call, the service representative opens a “trouble ticket” that will be used to track progress on the problem.

Figure 4 Freight brokerage scenario



First, the representative calls up all relevant information about the customer, which may be distributed or replicated among several sites and in several different formats. For example, billing and account information may come from the accounts receivable department, and engineering specifications may come from the product development group, both of which may be at different locations than the call center. XML is used to transfer information between different sites in the company. Next, the service representative begins to troubleshoot, using a sophisticated search engine to help diagnose the problem. This search relies on key phrases and problem categories derived from previous trouble tickets, which may have been generated from phone calls or extracted from letters or from notes taken by the field technicians.

After an initial diagnosis, the service representative places a service request on a queue to be dispatched

to a field technician and reports a confirmation number back to the customer. A field engineering application removes the request from the queue and assigns it to an appropriate technician, who receives it on a mobile device and begins the repair. In yet another part of the company, the Product Management department uses the search engine to look for common problem areas and customer use patterns, identify quality assurance issues, and discover desirable features and new product requirements.

Freight brokerage. A freight brokerage company provides services for its customers to find the best freight transportation rates among commercial airlines. Figure 4 illustrates this process. Customers and customer applications request a freight booking by sending an XML message (perhaps using SOAP) to the brokerage company, and the arrival of the request initiates a workflow to process it. The workflow archives the original request and then updates

the customer history with the request information. A carrier information system matches the client request with a set of carriers that can transport the freight, and contacts an external flight scheduling system to identify possible flights. Next, the carrier information system contacts each of the relevant carriers to obtain a quote by invoking a Web services request for each carrier. The winning bid is computed from the responses and the customer notified. Bids and responses are both recorded for future analysis, which is used to monitor carrier partnerships, obtain better contract rates, and to identify transportation trends and new opportunities.

Information integration requirements. Systems to handle the described scenarios must rely on various data management systems and enterprise applications that come with their own set of APIs (application programming interfaces) and tools, and an enterprise must develop complex in-house integration and administration software to maintain them. Although these three scenarios come from a diverse set of industries, they indicate that the boundaries that have traditionally existed between DBMSs, data warehouses, content management systems, and other data transformation and integration services are blurring. The similarities between these scenarios illustrate a common set of requirements for a robust information integration platform.

Enterprise applications require support for XML as a first-class data model. Although the relational data model has driven business applications over the past three decades, XML has become the *lingua franca* for portable data. Businesses rely heavily on it for enterprise integration. XML is more suitable for representing both semistructured and unstructured data, and, unlike the relational model, it also provides a standard representation of the meta-data to describe those data. This flexibility allows the financial services company to receive reports from multiple publishers, provides a common language for the freight broker to negotiate bids with multiple carriers, and enables the telecommunications company to integrate the various customer support systems acquired through its mergers.

Businesses rely on multiple, diverse sources of data in their enterprise applications. The financial services company relies on real-time data feeds, databases, spreadsheets, and media servers. The freight broker depends on real-time access to external flight scheduling systems and carriers. The telecommunications application requires access to multiple autonomous

vertical applications. These scenarios make use of information *consolidation*, where data are both collected from multiple sources and consolidated in a central repository, and *federation*, in which data from multiple autonomous sources are accessed as part of a search but are not moved from their original sources. Some sources of data conform to a schema, whereas other sources, such as information portals, do not. Because an enterprise receives information from so many sources, meta-data to describe the data available from a source are as important as the data.

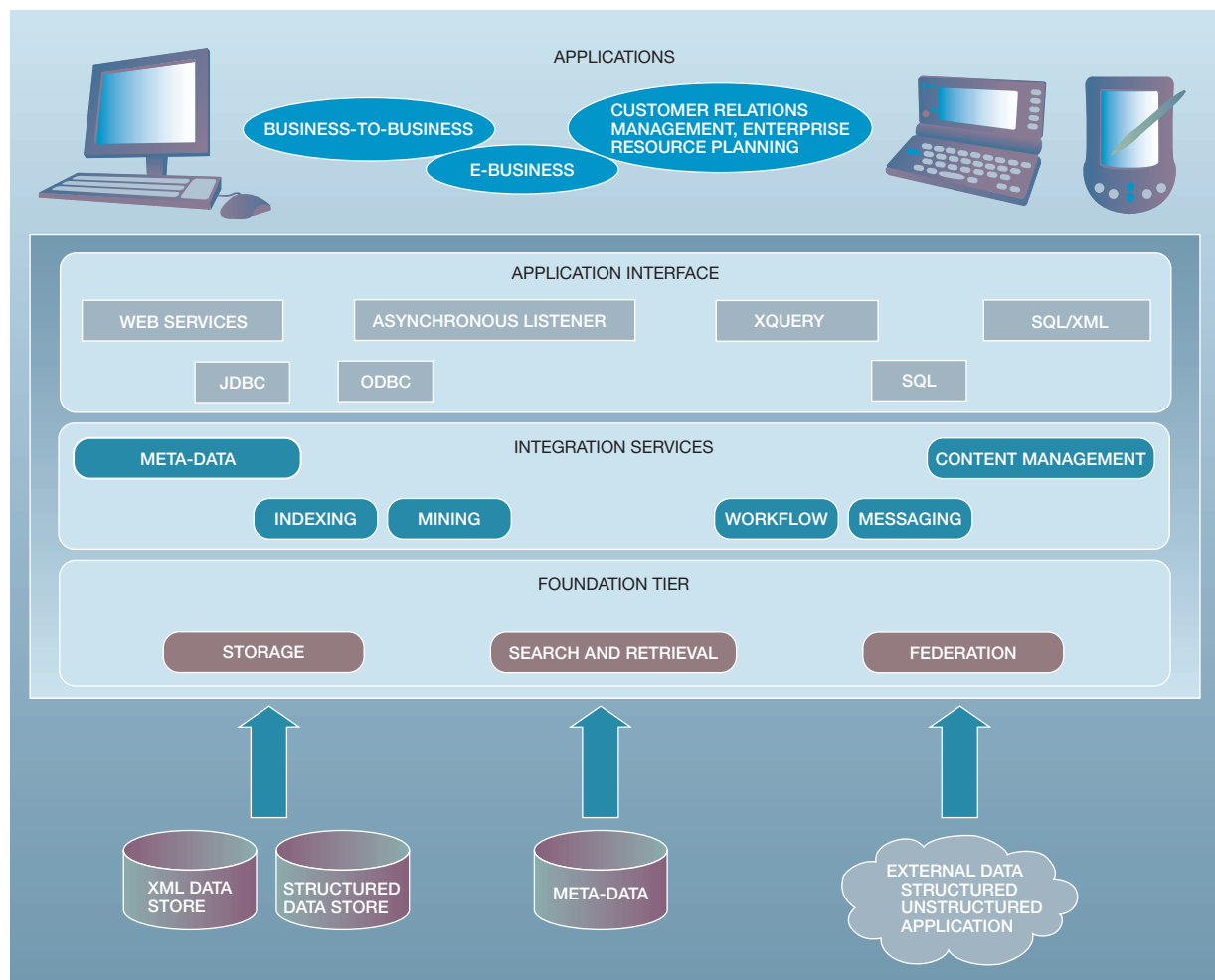
Today's enterprise systems require sophisticated search and data transformation services with increasingly demanding performance requirements. The

**A single system that provides
access to all data and services
through a unified interface would
simplify development and provide
better performance.**

scenarios previously described require parametric search, full-text search, mining, and digital asset management services over data stored in a variety of systems and in a variety of formats. Although specialized systems with different APIs exist today to handle each kind of search, a single system that provides access to all data and services through a unified interface would simplify application development and provide better performance.

The scenarios also demonstrate that business processes are inherently asynchronous. Customers reporting problems are satisfied with a confirmation number and do not need to remain on the phone while the problem is repaired. Busy stock traders may not want to poll for information, but instead prefer to be notified when events of interest occur. Furthermore, continuous availability requirements mean that applications must continue running in the face of failures. Data sources and applications come up and go down on a regular basis, and data feeds may be interrupted by hardware or network failures. If the telecommunications customer service system goes down, field technicians should continue to work on repairs assigned to them. If a particular carrier is not available, the freight broker should negotiate bids from the carriers that are available. These examples show that messaging and workflow systems

Figure 5 A three-tier information integration architecture



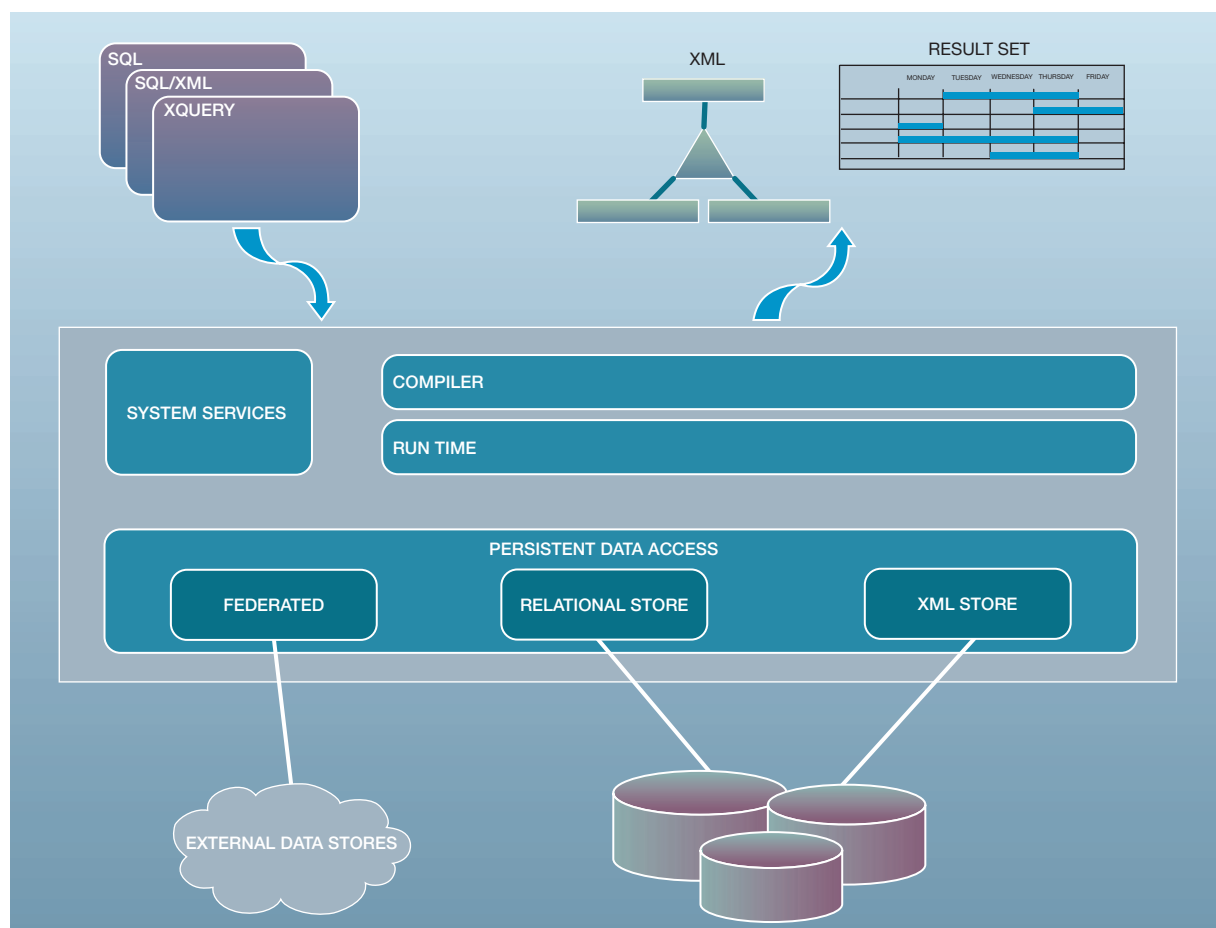
are as integral as data management to enterprise applications.

Finally, the complexity of the applications in these scenarios illustrates an essential requirement for open standards. Such applications weave together a host of data sources, management systems, and applications. Without standards governing their interfaces and a consistent programming model, the effort to integrate and administer new sources in this dynamic environment could be astronomical, and the cost to recruit and retain staff with the skills necessary to develop the integration software might quickly overtake the value of the system itself.

Architecture

In this section, we describe a three-tier architecture for a robust technology platform that addresses the information integration challenge. Figure 5 illustrates this architecture. The foundation tier supports storage, retrieval, and transformation of data in different formats from different sources. An integration tier built on top of the foundation draws from enterprise integration applications to provide the infrastructure to transparently embed data access services into enterprise applications and business processes. The top tier provides standards-based programming models and flexible query language sup-

Figure 6 Foundation architecture



port to access the rich set of services and data provided by the foundation and integration tiers.

The foundation tier. As shown in the figure, the foundation tier is at the heart of the information integration platform, and provides a core set of services to store and retrieve heterogeneous data. The foundation tier is based on a high-performance DBMS engine extended at all levels with data integration functionality. These extensions include support for XML as a native data store, XQuery as a native data interrogation language,²⁵ and a federated data access component that provides access to external data as though the data were natively managed by the engine.

Figure 6 shows the components of such an engine. At the very lowest level are the data access compo-

nents, which provide APIs to efficiently store and retrieve data from persistent storage. Two data access components are provided for native data. Structured data are stored, indexed, and retrieved via a relational data access component, and the native XML data store is accessed by a new data access component that provides routines to efficiently store, index, and navigate the hierarchical XML structure.

In addition to the data access components for the native data stores, the integration engine has a federated data access component for external data.²⁶ A wealth of data sources and rich content providers are transparently made accessible through this component, including external databases, file systems, document stores, genomic databases, and multimedia object servers. External data sources are accessed

via wrappers,²⁷ which provide an interface by which the integration engine can generate execution plans to store and retrieve data, manage transactions, and perform functions on external data. In addition to integrated query support, content-specific capabilities such as image search, multimedia streaming, and bioinformatic modeling²³ are also made accessible to information integration clients through the federated data access component.

A query compiler sits above the data access components and provides both an XQuery and an SQL interrogation language by which data can be efficiently stored and retrieved. Regardless of which language is used, a request is passed on to a query compiler to interpret the request and generate an execution plan for processing the request. The query compiler uses a language-specific lexical analyzer and parser to interpret the request and populate an XQGM (eXtended Query Graph Model) structure. XQGM is an enhanced version of the Starburst Query Graph Model⁷ (QGM) that enables a richer semantic representation than can be expressed with QGM, including a representation of XML data model primitives and federated access to external data sources. The query compiler analyzes the XQGM, explores several strategies for executing the request, and chooses a plan from its least-cost alternatives. As part of its analysis, the query compiler considers indexes and navigational algorithms for the XML data store, recognizes external data access, and invokes the appropriate wrapper query planning routines to generate efficient plans for external access. The run-time execution engine executes the selected plan, interacting with the relational data store, XML data store, and federated access layer to combine the appropriate data to fulfill the original request. The run-time engine contains XPath processing operators for XML data access,²⁸ as well as operators for federated access.

Along with the query processing and data access components, the information engine provides a set of system services required for any data management system, including authentication, code page management, logging, recovery, transaction management, and interfaces to transaction monitors such as those provided by WebSphere*.²⁹

The integration services tier. Data integration at the storage and retrieval level is not sufficient for today's enterprise applications. Perhaps even more important than access to data from multiple sources and multiple formats is to know *what* information is avail-

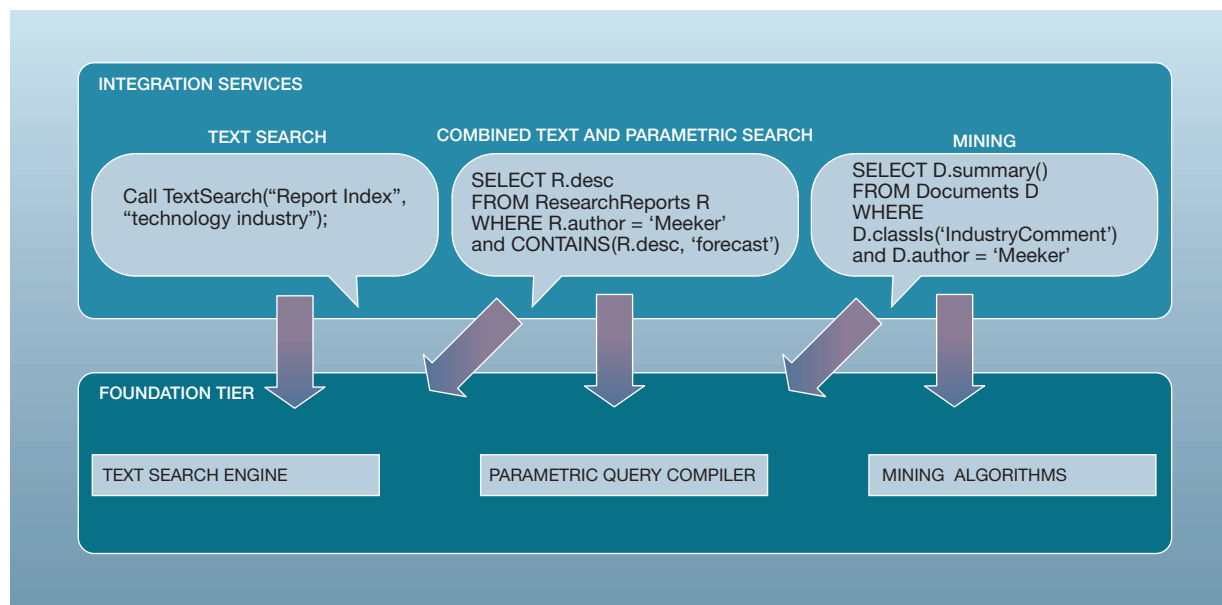
able and *how* to make use of it. The integration tier provides the infrastructure to navigate the sea of available information and place it in a context that is usable for enterprise applications. Key services provided by the integration tier are described next.

Meta-data. For an enterprise application to fully exploit the rich set of data sources accessible through the foundation tier, meta-data that describe what content and services are available are crucial. Such meta-data are managed by the foundation tier, and the integration tier encapsulates and summarizes the information for navigation by client applications. This approach allows client applications to use the same APIs and query language to query both the meta-data and the base data.

Meta-data come in two types: *system* meta-data, which describe resources and services that can be performed on those resources, and *application* meta-data, which provide domain-specific knowledge about a data object and how it relates to other objects. System meta-data, for example, include information about data sources, function signatures, data formats, and index information. They are used by the system to manage base data and process requests, but can also be a valuable source of information for client applications to dynamically discover what content and services are available. Ontologies, schema integration technology, and tools can be used to map data from different applications in related domains into a common schema.³⁰⁻³² Application meta-data are often composed of domain-specific annotations about the data and can be used to provide users with a better understanding of their data and how they are related to other data. They may also be useful for transforming data from one representation to another or for improving the ability to focus a user's search for objects of interest. In the financial services scenario, data such as the list of industries, companies, and analysts mentioned in a research report are examples of application meta-data. Key to managing meta-data are schema mapping and schema integration.

Content management services. Application meta-data often arise as enterprise applications construct, store, and manipulate complex business objects composed of multiple digital assets. For example, the financial services research report is made up of an XML document, several PDF (portable document format) files, and an audio or video clip. The telecommunications trouble ticket is made up of customer account information, an audio clip, and several text

Figure 7 Integrated search services



documents. The integration tier provides a core set of services to manage those objects and their component parts, including check-in/check-out services, versioning, access control and digital rights management, and hierarchical storage migration services.³³

The integration tier provides *URL addressability* for digital assets that are large and have content-specific actions associated with them, such as media streaming, document rendering, and image search. Client applications can search for objects of interest by querying meta-data, and the integration engine will return URLs for objects that match the search. The client may then directly access and manipulate the objects via the URLs.

Text search and data mining. Unstructured information must be analyzed and categorized to be of use to an enterprise application, and for real-time decisions, the timeliness of the answer is a key component of the quality. The integration services tier enables this analysis by providing several capabilities for integrated search, tightly coupled with the foundation tier infrastructure.

The integrated search capabilities are illustrated in Figure 7. A state-of-the-art text indexing engine built into the foundation tier provides the infrastructure

for raw text search over integrated data, including structured data, XML documents, and user-defined structures. A second type of integrated search provides query language constructs to transparently combine text search with parametric search in a single query. In addition to the benefits of programming to a single language, this approach allows the query compiler to exploit these language constructs and optimize the combined search queries. Finally, native mining algorithms built into the foundation tier provide feature extraction, summarization, and classification services, and these mining services can be invoked through the query language as well. Again, embedding the mining algorithms into the foundation tier allows the query compiler to optimize these searches, which leads to excellent performance.

A key benefit of integrated search and mining through a common interface and query language is *actionability* of data. For example, by combining text search and feature extraction with database triggers, data of unknown content can be analyzed and quickly routed to the interested parties.

Workflow, messaging, and business process integration. Global enterprises running in a continually available environment require asynchronous communi-

cation and messaging to develop scalable and fault-tolerant business applications. The integration services tier uses a workflow engine to transparently schedule and manage long-running, data-intensive applications and native functions that integrate guaranteed message delivery within data operations.

The application interface. The application interface layer provides interfaces that allow applications to access and manipulate data and services provided by the foundation and integration tiers. It supports multiple APIs, query languages, and programming models for maximum flexibility. Depending on the application requirements and programmer preference, data can be retrieved according to a structured data model, as XML documents, or as XML document fragments. Details of these approaches are described here.

Programming interface. The application interface tier provides full support for traditional database programming using a variety of programming languages via embedded SQL, ODBC,³⁴ and JDBC.³⁵ These APIs include extensions to support XML data as well as relational data.³⁶ ODBC and JDBC and other popular database APIs are inherently synchronous, and are not well-suited for applications that deal with data that are not continuously available, long latency in data access, or multiple sources and targets for the flow of information. For these types of applications, the application tier provides messaging¹² and Web services³⁷ programming models, as well as a set of tools to build applications for asynchronous environments.

Query language. SQL has proven to be an extremely powerful language for retrieving structured data, and XQuery²⁵ is taking hold as the language for querying semistructured and unstructured data. The application tier supports both languages, and either can be used to access the federated content supported by the foundation tier as SQL or XML data.³⁸ A query can transparently combine data from relational tables, the XML store, and data retrieved from an external server. For those applications that use SQL to retrieve XML data, the document or document fragment is returned as column value for a particular row of data. For those applications that choose XQuery to access relational data, the application tier provides an XML view of the table or tables, and the query results are returned as an XML document with tagging defined by the view.

The solution

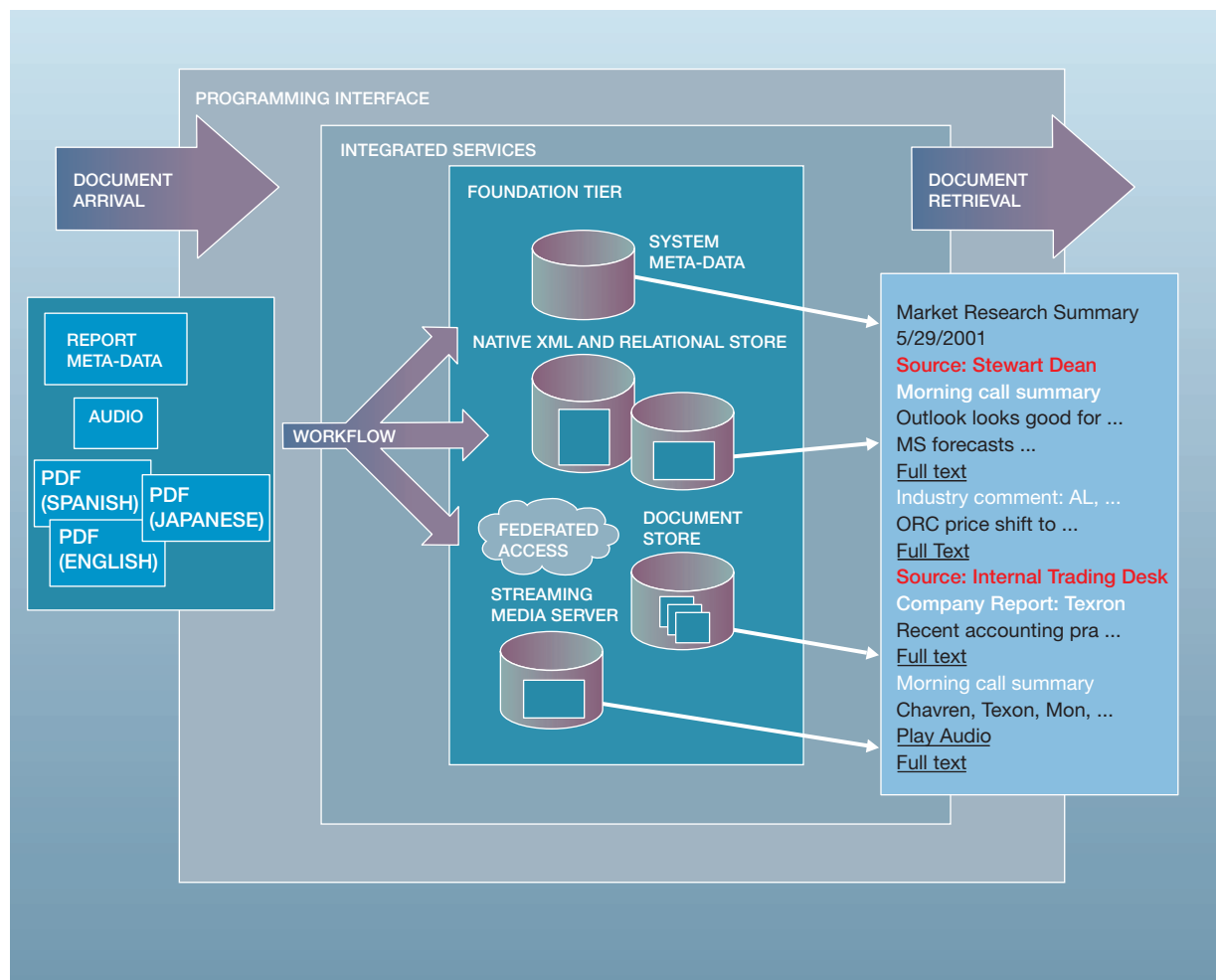
We now return to the financial services scenario to show how the architecture just described addresses the information integration challenge. Figure 8 shows a portal application built on such a platform for the investment banking department. This portal provides access to market research reports purchased from external vendors or produced by other departments in the company, such as “morning meeting” summaries from the trading desk.

Figure 8 traces the path of an XML document produced by the trading desk in New York that summarizes the morning’s call. The document contains an RIXML section that provides meta-data about the call, such as which analysts, companies, and industries were mentioned. It also contains an audio clip and several PDF files that contain a translation of the call in several languages.

The left side of Figure 8 illustrates the arrival of the morning call document. The information integration platform receives the report via an asynchronous listener embedded in the programming interface, and its arrival immediately initiates a workflow process. The first step in the workflow stores the original XML document in the native XML store and stores information about the source, format, and access rights to the document in the system meta-data store. A second step in the workflow invokes indexing and mining services to index, classify, and summarize the content of the morning call. A third step extracts the PDF files, forwards them to a document server and records the language information for the files in the system meta-data store. A final step extracts the audio clip and forwards it to a streaming media server for storage.

The right side of Figure 8 illustrates the portal application accessing the morning call along with several other documents, including another morning call, an industry comment, and a company report. The application issues an XQuery request for recently received documents and displays a summary of the documents on a Web page. Access control against the system meta-data occurs as part of processing the XQuery request, so that only documents for which the application has access rights are retrieved. Content for the XML documents that are returned come from various storage and access components of the foundation tier. For example, system meta-data provide information about the source and structure of the documents, and a document’s type

Figure 8 Information integration for the financial services scenario



and a summary is known from mining information extracted by a step in the workflow process. The integration engine derives the correct version of a PDF file associated with a document from the system meta-data and the code page, and computes a URL for it, which the portal application uses to retrieve the file from the document server via the federated access component. Similarly, the integration engine generates a URL for the morning call's audio clip for the portal application to use to stream the audio from the media server.

Conclusion

The enterprise software industry is in the midst of a revolution, and enterprise application development

is at a crossroads. Gone are the days of the monolithic mainframe churning along in isolation in a back-office clean room. The challenge for today's enterprise applications is to reach out across the company and across the Internet to integrate and transform the volumes of available data as quickly as possible into information assets that bring new insights and new business opportunities. We have presented here the components of an information integration technology platform that draws on decades of advances in enterprise data management infrastructure to provide a robust and unified solution for information integration.

*Trademark or registered trademark of International Business Machines Corporation.

**Trademark or registered trademark of Sun Microsystems, Inc.

Cited references

1. P. Lyman, H. Varian, A. Dunn, A. Strygin, and K. Swearingen, "How Much Information?" see <http://www.sims.berkeley.edu/research/projects/how-much-info/>.
2. M. Roth and D. Wolfson, "From Data Management to Information Integration: A Natural Evolution," <http://www7b.software.ibm.com/dmdd/library/techarticle/0206roth/0206roth.html>.
3. E. Codd, "A Relational Model of Data for Large Shared Data Banks," *Communications of the ACM* **13**, No. 6, 377–387 (1970).
4. M. M. Astrahan, M. W. Blasgen, D. D. Chamberlin, K. P. Eswaran, J. Gray, P. P. Griffiths, W. F. King, R. A. Lorie, P. R. McJones, J. W. Mehl, G. R. Putzolu, I. L. Traiger, B. W. Wade, and V. Watson, "System R: A Relational Approach to Database Management," *ACM Transactions on Database Systems* **1**, No. 2, 97–137 (1976).
5. P. Selinger, M. Astrahan, D. Chamberlin, R. Lorie, and T. Price, "Access Path Selection in a Relational Database Management System," *Proceedings, ACM SIGMOD International Conference on Management of Data*, Boston, MA (May 30–June 1, 1979), pp. 23–34.
6. R. Williams, D. Daniels, L. Haas, G. Lapis, B. Lindsey, P. Ng, R. Obermarck, P. Selinger, A. Walker, P. Wilms, and R. Yost, *R*: An Overview of the Architecture*, IBM Research Report RJ3325, IBM Research Laboratory, San Jose, CA (December 1981).
7. L. M. Haas, W. Chang, G. M. Lohman, J. McPherson, P. F. Wilms, G. Lapis, B. Lindsey, H. Pirahesh, M. J. Carey, and E. Shekita, "Starburst Mid-Flight: As the Dust Clears," *Transactions on Knowledge and Data Engineering* **2**, No. 1, 143–160 (1990).
8. UDB Relational Connect, see <http://www-3.ibm.com/software/data/db2/relconnect/>.
9. M. Tork Roth, P. Schwarz, and L. Haas, "An Architecture for Transparent Access to Diverse Data Sources," *Component Database Systems*, K. R. Dittrich and A. Geppert, Editors, Morgan Kaufmann Publishers, San Francisco, CA (2001), pp. 175–206.
10. W. F. Cody, J. T. Kreulen, V. Krishna, and W. S. Spangler, "The Integration of Business Intelligence and Knowledge Management," *IBM Systems Journal* **41**, No. 4, 697–713 (2002, this issue).
11. IBM Content Manager, see <http://www-3.ibm.com/software/data/cm/library.html>.
12. IBM MQSeries Integrator 2.0: The Next Generation Message Broker, see <http://www-3.ibm.com/software/ts/mqseries/library/whitepapers/mqintegrator/msgbrokers.html>.
13. F. Leymann and D. Roller, "Using Flows in Information Integration," *IBM Systems Journal* **41**, No. 4, 732–742 (2002, this issue).
14. See <http://www.softwareag.com/tamino/>.
15. Ipedo, see <http://www.ipedo.com>.
16. H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom, "Integrating and Accessing Heterogeneous Information Sources in TSIMMIS," *Proceedings, AAAI Symposium on Information Gathering*, Stanford, CA (March 27–29, 1995), pp. 61–64.
17. A. Tomasic, R. Amouroux, P. Bonnet, O. Kapitskaia, H. Naake, and L. Raschid, "The Distributed Information Search Component (Disco) and the World Wide Web," *Proceedings, ACM SIGMOD Conference*, Tucson, AZ (May 13–15, 1997), pp. 546–548.
18. S. Adali, K. Candan, Y. Papakonstantinou, and V. S. Subrahmanian, "Query Caching and Optimization in Distributed Mediator Systems," *Proceedings, ACM SIGMOD Conference on Management of Data*, Montreal, Canada (June 4–6, 1996), pp. 137–148.
19. A. Levy, A. Rajaraman, and J. J. Ordille, "Querying Heterogeneous Information Sources Using Source Descriptions," *Proceedings, 22nd International Conference on Very Large Data Bases*, Bombay, India (September 3–6, 1996), pp. 251–262.
20. See <http://www.nimble.com>.
21. Callixa, see <http://www.callixa.com>.
22. Infoshark, see <http://www.infoshark.com>.
23. DiscoveryLink, see <http://www-3.ibm.com/solutions/lifesciences/discoverylink.html>.
24. RIXML Specification Users Guide & Data Dictionary Report, see <http://www.rxml.org>.
25. D. Chamberlin, J. Clark, D. Florescu, J. Robie, J. Siméon, and M. Stefanescu, *XQuery 1.0: An XML Query Language*, W3C Working Draft (April 2002). Available at <http://www.w3.org/TR/xquery/>.
26. L. M. Haas, E. T. Lin, and M. A. Roth, "Data Integration Through Database Federation," *IBM Systems Journal* **41**, No. 4, 578–596 (2002, this issue).
27. M. Tork Roth and P. Schwarz, "Don't Scrap It, Wrap It! A Wrapper Architecture for Legacy Data Sources," *Proceedings, 23rd Conference on Very Large Data Bases*, Athens, Greece (August 26–29, 1997).
28. See <http://www.w3.org/TR/xpath>.
29. See <http://www-3.ibm.com/software/info1/websphere/>.
30. C. A. Goble, R. Stevens, G. Ng, S. Bechhofer, N. W. Paton, P. G. Baker, M. Peim, and A. Brass, "Transparent Access to Multiple Bioinformatics Information Sources," *IBM Systems Journal* **40**, No. 2, 532–551 (2001).
31. S. Ram and V. Ramesh, "Schema Integration: Past, Current and Future," *Management of Heterogeneous and Autonomous Database Systems*, A. Elmagarmid, M. Rusinkiewicz, and A. Sheth, Editors, Morgan Kaufmann Publishers, San Francisco, CA (1999), pp. 119–155.
32. M. Hernandez, R. Miller, and L. Haas, "Clio: A Semi-Automatic Tool for Schema Mapping," *Proceedings, ACM SIGMOD Conference*, Santa Barbara, CA (May 21–24, 2001).
33. A. Somani, D. Choy, and J. C. Kleewein, "Bringing Together Content and Data Management Systems: Challenges and Opportunities," *IBM Systems Journal* **41**, No. 4, 686–696 (2002, this issue).
34. See <http://www.microsoft.com/data/odbc/default.htm>.
35. See <http://java.sun.com/products/jdbc/>.
36. A. Eisenberg and J. Melton, "SQL/XML and the SQLX Informal Group of Companies," *SIGMOD Record* **30**, No. 3, 105–108 (2001).
37. S. Malaika, C. J. Nelin, R. Qu, B. Reinwald, and D. C. Wolfson, "DB2 and Web Services," *IBM Systems Journal* **41**, No. 4, 666–685 (2002, this issue).
38. J. Funderburk, S. Malaika, and B. Reinwald, "XML Programming with SQL/XML and XQuery," *IBM Systems Journal* **41**, No. 4, 642–665 (2002, this issue).

Accepted for publication June 14, 2002.

Mary A. Roth IBM Software Group, Silicon Valley Laboratory, 555 Bailey Avenue, San Jose, California 95141 (electronic mail: torkroth@us.ibm.com). Ms. Roth is a senior engineer and manager in the Database Technology Institute for e-Business at IBM's

Silicon Valley Lab. She has over 12 years of experience in database research and development. As a researcher and member of the Garlic project at IBM's Almaden Research Center, she contributed key advances in heterogeneous data integration techniques and federated query optimization and led efforts to transfer Garlic support to DB2. Ms. Roth is leading a team of developers to deliver a key set of components for Xperanto, IBM's information integration initiative for distributed data access and integration.

Daniel C. Wolfson *IBM Software Group, 11501 Burnett Road, Austin, Texas 78758 (electronic mail: dwolfson@us.ibm.com)*. Mr. Wolfson is a Senior Technical Staff Member and manager in the Database Technology Institute for e-Business. With more than 15 years of experience in distributed computing, his interests have ranged broadly across databases, messaging, and transaction systems. He is a lead architect in the information integration area, focusing on DB2 integration with WebSphere, MQSeries[®], workflow, Web services, and asynchronous client protocols.

James C. Kleewein *IBM Software Group, Silicon Valley Laboratory, 555 Bailey Avenue, San Jose, California 95141 (electronic mail: kleewein@us.ibm.com)*. Mr. Kleewein is a Distinguished Engineer working in database architecture, strategy, and technology at the IBM Silicon Valley Laboratory. He has worked for IBM in the database business for the last 15 years. His technical expertise can be seen across a wide range of IBM data management products, including IMS[™], DB2/MVS, DB2/390, DB2 systemplex data sharing, DB2 Spatial Extender, DataJoiner[®], and DiscoveryLink. Mr. Kleewein is a lead architect for Xperanto, focusing on expanding the role of DB2 from a structured data store to a structured and semistructured data store by adding XML capabilities to the DB2 engine.

Constance J. Nelin *IBM Software Group, 11501 Burnett Road, Austin, Texas 78758 (electronic mail: nelin@us.ibm.com)*. Ms. Nelin is a Senior Technical Staff Member in the IBM Database Advanced Technology area. She has worked for IBM since 1987, with a focus on database application development support and tooling. She has responsibility for application development tooling strategy, architecture, and development for data management. This covers the application development support for the full DB2 family spanning the areas of core relational database, federated database, XML, Web services, and messaging features.