The Enosys Markets Data Integration Platform: Lessons from the Trenches

Yannis Papakonstantinou Vasilis Vassalos
{yannis,vasilis}@enosysmarkets.com

ABSTRACT

Enosys Markets offers a state-of-the-art data integration software platform to support the development of the next generation of eBusiness applications that deliver value by providing new levels of function for customer relationship management, e-commerce, supply chain management, and decision support. These applications require that data be integrated from information sources that exist both within and across organizational boundaries. The Enosys Markets data integration architecture and product family provides a complete end-to-end XML-based solution for integrating and querying distributed information sources. It incorporates advanced research into XML and database technology. We present the product architecture and components, discuss the key technical challenges, and outline the technical concepts and innovations employed in the Enosys platform.

Keywords

Mediators, semistructured data, XML, query forms, reports, distributed querying, heterogeneous databases, integration.

1. THE DATA INTEGRATION PROBLEM

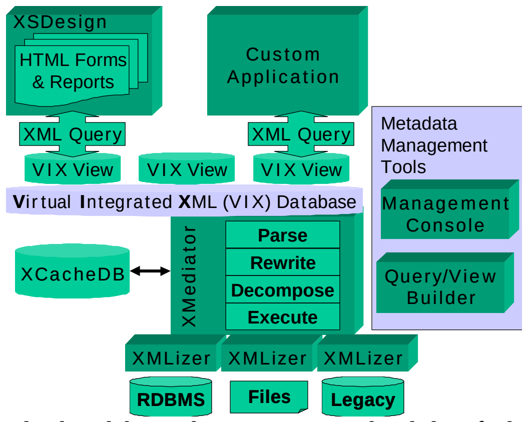
eBusiness applications that support more efficient, tightly integrated business processes demand access and integration of up-to-date information from a multitude of distributed and heterogeneous information systems. Information integration is a significant challenge: the relevant data are split across multiple information sources, often owned by different organizations. The sources represent, maintain, and export the information using a variety of formats, interfaces and semantics. Many challenges arise:

- Data of different sources change at different rates, making the data warehousing approach to integration hard to develop and maintain. The Enosys platform resolves this challenge by being based on the on-demand mediator approach, according to which data are collected dynamically from the sources, in response to application requests, as further explained below.
- The mediator has to decompose application requests into an efficient sequence of requests targeted to the sources. These requests have to be compatible with the query capabilities of the underlying sources.
- Different types of information reside in different systems, have different structure, and are usually in heterogeneous formats. The mediator has to enable and facilitate the resolution of the heterogeneities.
- Different applications often need different views of the data. Hence, view construction must be facilitated.

The Enosys Markets data integration platform addresses the above challenges using a XML-based mediator architecture, which enables applications to access and integrate information using a high-level, declarative XML query language.

2. PLATFORM ARCHITECTURE

The Enosys Markets Integration Platform is based on the wrapper-mediator architecture, as shown in the figure. The wrappers, also called *XMLizers*, access multiple, distributed, heterogeneous information sources and export Virtual XML views of them. The XMediator exports the ***V**irtual **I**ntegrated **X**ML (VIX) database*, which consists of all the individual views exported by the wrappers. Then *virtual integrated XML views* can be built on top of the VIX database. The views organize information from the distributed sources into XML objects that conform to the application's needs. For example, to a marketplace application the integrated XML view can provide front-end access to an integrated catalog, where the heterogeneities between the suppliers' products are



resolved, and the products are integrated and classified according to the needs of the marketplace. Each product object contains catalog data along with attributes from the pricing, delivery, CRM, and other databases. The views provide distribution transparency, i.e., the originating sources and methods of access are transparent to the application.

The virtual database and views enable the front-end applications, which may be the Enosys Markets XSDesign web form generator or custom applications, to seamlessly access distributed heterogeneous information sources as if they were a single XML database. In particular, the application can issue an XML query against either the XML database or the views. The query typically selects information from the views and structures it in ways that are convenient for the application. For example, a HTML application will create queries that structure the XML results in a way that easily translate into the target HTML pages.[1]

When the application issues an XML query to a VIX database or view, the platform decomposes the XML query into requests that are directed to the sources. The source responses are assembled into the XML query result that is sent to the source.

When the sources correspond to slow and static sources one may prefer to cache the XML view of those sources into the XCacheDB, which is the Enosys Markets XML database. Typically, the data of slow and static sources are collected, integrated, and cached in advance, while the components originating at fast dynamic sources are collected dynamically. It is transparent to the application which pieces of the view originate from dynamic sources and which ones originate from XCacheDB.

2.1 Feature overview

The data integration server is accessible to applications through a query language API and a DOM-based (Document Object Model) API. XCacheDB is an XML database, primarily used for caching purposes. XSDesign offers a web-base front-end generator for the easy construction of web/HTML-based query form and report templates. Finally, the platform includes management tools that enable the user to easily create and manage front-ends, view definitions, queries, and source connections. Each of the key components is described below.

3. DATA INTEGRATION SERVER

XMLizers: The XMLizers turn structured and semi-structured data into virtual XML views. Enosys Markets currently offers XMLizers for relational databases (any JDBC compliant database), HTML repositories, SOAP services, and files (comma/tab delimited files).

[1] Indeed, Enosys Markets has deployed Web-based applications where, for efficiency reasons, the XML queries generate XHTML.

Additional XMLizers can easily be written for other data sources. Furthermore, existing investments in XML information exchange (*e.g.*, use of adapters by WebMethods or SeeBeyond) provide excellent leverage for the development of these XMLizers.

XMediator: The XMediator accesses the virtual view exported by the XMLizers and provides a virtual integrated XML view to the applications. The integrated view appropriately transforms and integrates the XML views of the information sources into XML that that conforms to the target applications.

The transformation and integration is rapidly and concisely specified in the *XML Catalog Query Language (XCQL).* XCQL is a high-level, declarative query and view definition language for XML. It resembles XQuery and has additional features for processing XML data with loose structure. Transformations expressed in a brief XCQL view definition can easily resolve important integration problems, including name, value, and classification heterogeneities. Our experience shows a 20-fold decrease in the cost of developing source-to-target transformations when compared to using Java.

The XMediator allows queries directly on the VIX views. The view processor transforms the query to replace references and conditions on views with references and conditions on the actual data sources. The query is then parsed into a query plan and is optimized by the query rewriter. The decomposer chooses an efficient way of decomposing the optimized query plan into requests that are sent to the information sources. The plan is finally run by the execution engine, which sends the requests to the wrappers, collects the information, and composes it into the XML query result.

We discuss later the challenges faced in XMediator's implementation.

XCacheDB: The Enosys Integration Platform uses the the **X**ML **Cache D**ata**B**ase (XCacheDB) to cache VIXView's that correspond to slow or static sources. The caching can happen on demand or at regularly scheduled intervals. The XCacheDB is a native XML database in the sense that it stores XML and responds to XML queries with XML results. XCacheDB utilizes a JDBC compliant relational database for storage and query processing and is optimized for Oracle8i. The developer does not need to be aware of the underlying relational database. Nevertheless, XCacheDB offers management functionality that allows a developer/administrator to provide hints on how the data should be stored. The architecture of XCacheDB uses proprietary storage and query processing algorithms to deliver improvements in run-time efficiency.

4. WEB FRONT-END

The XSDesign family of tools enables the rapid development of customized Web front-ends that make the best use of the integrated information and can easily incorporate domain expertise. XSDesign is designed to be used by the business analyst and can provide:

- Forms for parametric querying of the data sources in the integrated view
- Summarization and navigation of large query results
- Query assistance in formulating and refining queries
- Advice on product selection based on domain expertise

5. METADATA MANAGEMENT

The Enosys Markets Management Tools provide a comprehensive systems administration and development environment to manage the data integration platform.

Query Builder: A query builder tool allows the graphical creation of XCQL view definitions and queries. The builder first imports the XML schemas of the information sources or the existing views. The user then uses a drag-and-drop interface and wizards to define joins, function invocations, filtering conditions and more on the input data. The builder also allows the user to graphically arrange the output data into a different XML schema by specifying mappings to the input schemas, groupings, and creation of new elements.

Management Console: The management console is an integrated development environment for Web-based applications that access multiple information sources. The console integrates tools for view and query construction and testing as well as for deployment and management of the Data Integration Server and XSDesign-powered Web front-ends.

6. CHALLENGES IN THE DATA INTEGRATION SERVER

The data integration server faces and resolves many of the challenges that conventional database engines face. In addition, a novel set of challenges has emerged, which is attributed to the distributed query processing needs and the emphasis on the enabling of efficient Web applications.

On-Demand Evaluation The Enosys Markets data integration server focuses on the efficient support of Web-based applications, where the user is typically interested in receiving quickly the first few results and is often impatient to check out the complete answer. The data integration server focuses on first results' optimization and saves the underlying relational databases (or other sources) from having to produce results that will not be needed eventually. Both of the above goals are achieved by the navigation-driven on-demand evaluation of the query; the pieces of the query result are not materialized at the client until the application client navigates into them. However, a pure on-demand approach, where data are retrieved and created only when they are definitely needed, has a tremendous cost in roundtrips to the server and the sources. The XMediator balances on-demand evaluation with the overhead of roundtrips by using a scheme where blocks of the result are created and transferred to the client in each round. The application can control the block size; setting the block size to a small value results in lazy evaluation. Setting the block size to a large number results in eager evaluation.

Query Decomposition Typically an XMLizer can respond to only a limited number of requests – those requests that correspond to the abilities of the underlying sources. The mediator has to decompose the application queries into requests understood by the XMLizers. This requires the formulation of calls to the XMLizers that are as efficient as possible, yet they are also supported by the XMLizers. For example, when the underlying system is an SQL database, it makes sense to push as many as possible selections and joins to it. The challenge is tackled by the mediator's open rewriter module, where we incorporate rules that specify how the queries are decomposed into requests going to the XMLizers, in a way that is commensurate with the XMLizers' abilities. The rules are written in Java.

Scalability and Reliability The data integration server is built to scale to hundreds of concurrent queries. The on-demand evaluation model, described above, allows the mediator to spend a low memory amount for each running query.

When a stateful operator of the query plan, i.e., an operator that requires a data dependent amount of memory, cannot find main memory to accomplish its task, it stores part of its state on the disk, using our own custom-made storage manager. All operators that may store in disk (join, sort, group-by) are built to require at most two read-write passes over the data.

Admission control guarantees that each query has a minimum of necessary memory before it starts.

Optimization The mediator optimizes the join of data of multiple sources by choosing the appropriate join order and join techniques. Currently the choice is based on qualitative factors such as the type of the source and the existence of indices. Furthermore, the developer can hint or force specific plans. He/she may declare specific sources as "slow" and force the use of specific join orders and join techniques by including hints in the query statement or the configuration files that include source information.

Our experience with beta customers has shown that hints and developer intervention can go a long way in data integration, where we typically have just a 2-way or 3-way join across sources, rather than the much more unmanageable 10+way joins observed in relational databases.