eXtensible Information Server White Paper

Version 2.02

"eXcelon's solution is one of the fastest and most extensible solutions ever deployed at NTT."
NTT ComWare, a division of Nippon Telephone and Telegraph, Japan

25 Mall Road Burlington, MA 01803 USA
Phone: 800-455-6226
Fax 781.674.5010

Table of Contents

Introduction **1**

Understanding the Pain **1**

The Extensible Information Model **2**

Design Goals **4**

Scalable, Fast, Native XML Management 4
Extensibility 5
Fast Data Transformation and Delivery 5
Usability 5
Standards Compliance 5
Ease of Integration 6

eXtensible Information Server Architecture **6**

Design-Time Components 6
Stylus Studio *7*
Map Designer and Process Designer *8*
Run-time Components 8
Dynamic XML Engine *8*
XConnects Integration Engine and Connectors *9*
DXE Manager *10*
Enterprise Features 10

Developing and Deploying Solutions **11**

Application Development Methodology 11
Using a Web Server 12
Using a J2EE Application Server 12

Conclusion **13**

Introduction

The most important and valuable corporate asset is information. While numerous solutions have emerged to address managing this information—such as relational databases, document management systems, mainframe forms, and enterprise resource planning (ERP) systems—the explosive growth of the Internet has shown corporations a world of real-time interactions outside of their walls. Consequently, years of investing in disparate solutions are crippling the productivity of business as it evolves in the Internet era. So, how do you unleash this information while letting your customers, employees, suppliers, and distributors access it in ways that improves the efficiency of existing business relationships and provides the flexibility to add new, more profitable services?

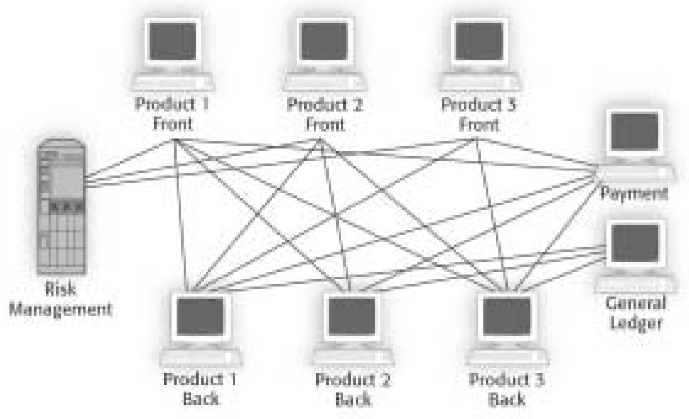
The eXcelon eXtensible Information Server (XIS) is a native XML database management system (XDBMS). It supports corporate XML initiatives with a production-quality environment that fully leverages the benefits of XML. It also provides powerful data and content aggregation, management, transformation, and delivery capabilities by supporting the creation of composite business applications across the extended enterprise. Subsequently, XIS is the logical choice for information systems departments who need to manage XML across the value chain.

Understanding the Pain

Traditionally, applications were linked together on a point-to-point basis. This approach worked well when the number of systems was fewer than five and the systems remained relatively stable. But today's enterprises run using hundreds of constantly evolving applications. According to the Gartner Group, a typical enterprise will devote 35% to 40% of its programming budget to developing and maintaining extract-and-update programs whose purpose is solely to transfer information between different databases and legacy applications.

Figure 1 shows a typical information flow in a financial services organization offering multiple products. To provide consistent service, a free flow of information must exist between front-office and back-office applications, as well as among accounting and control operations that support the product. The traditional approach to solving this problem is to create a custom data interchange between each set of data consumers and data producers. As Figure 1 shows, the number of these interfaces increases geometrically as new applications are added.

Figure 1. Traditional Flow of Information among Applications



To understand today's Web-based applications, it is important to know that much of the information required to make real-time business transactions resides within existing databases and applications. The management and manipulation of business data is a difficult task for many developers because the data can take many forms. It can be part of a corporate document, a message from another system, an interface to a transaction system, a database record, or a document on a Web site. Furthermore, there are several products referred to as "middleware" that aid in the process of communication with applications and databases.

The crucial missing element has been the ability to enable interchange between heterogeneous systems by adopting a universal interchange format that serves as the single, standards-based output format for all exporting systems and the single, standards-based input format for all importing systems. In a Forrester Research survey, one company commented, "Any personalization work depends on having very structured data. Moving content from our legacy systems into a database is a very taxing, arduous process that will cost us lots of time and money. This will be a multi-step process."

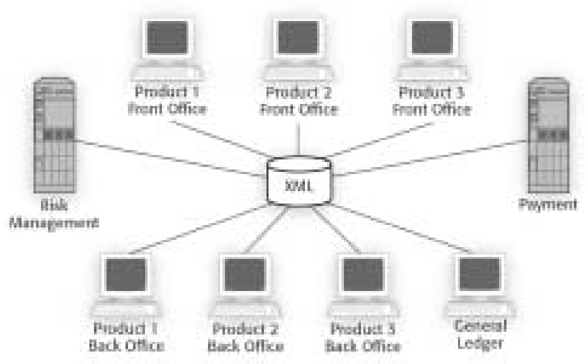
Until recently, no commonly accepted method of describing the data existed in a way that can be shared by the multitude of applications and data sources that might make use of it. As a result, system integrators and information technology (IT) departments continue to spend up to 40% of their valuable development resources extracting, redefining, and updating data. These expenditures continue throughout the life of traditional middleware solutions. Any time a system needs to be updated or added, the process begins anew, often causing disruption to existing online services.

The Extensible Information Model

The key to developing robust, adaptable, scalable, and high-performance Web-enabled applications is efficient management of data and content in a fully extensible information model. Without this model, development and deployment time for new applications or added services to existing applications increases exponentially, thus greatly reducing the return on investment of the solution. According to the Gartner Group, more than 85% of useful corporate information assets are not in databases. That this information exists in proprietary systems, formats, and applications means that traditional data warehousing solutions are built on the more rigid relational structure. Conversely, XML was designed to simplify development of this information model and aggregate diverse sources and formats, especially sources of content that do not fit the traditional relational model.

Traditional data warehousing solutions are built on the more rigid relational structure. XML was designed to simplify development of this information model that must be able to aggregate diverse sources and formats, especially sources of content that do not fit the traditional relational model. Figure 2 illustrates how XML is used to create this extensible information model.

Figure 2. Creating the Extensible Information Model



Since XML is based on a formal standard and has extensive support from both users and IT vendors, it provides a stable and future-proof foundation for the common information model. Ultimately, the business benefits of a standards-based strategy are immense and include:

- **Portability.** XML lets users access and move software among multiple platforms in a heterogeneous environment and allows transparent upgrade of underlying hardware and networks.
- **Interoperability.** Standard formats and interface conventions, like the ones specified in XML, provide one of the best strategies for ensuring component compatibility.
- **Risk reduction.** The use of standards reduces the cost of implementation-specific dependencies since important business components can be acquired from multiple sources.
- **Cost reduction.** A commercial vendor's support of standards results in economies of scale that are ultimately passed on to the user. Cost reduction can also be realized through competition among vendors supporting the same standard.
- **Deferred obsolescence.** Conformance to standards reduces the risk of system obsolescence because standards represent the most stable technology interfaces, and therefore continue to be supported as technology evolves.

eXcelon's **eXtensible Information Server** provides a distributed, scalable solution for managing aggregated XML data and content. It also provides real-time, transactional access to aggregated content and powerful transformation capabilities that allow this content to be easily repurposed for any delivery channel.

Design Goals

XML-based technologies are becoming more firmly entrenched in the IT infrastructures of many organizations. As the technology matures, the demand for scalable, high-performance, flexible systems based on XML has greatly increased. eXcelon's eXtensible Information Server is designed to scale large amounts of XML data and millions of user and system interactions daily. Improvements in XML standards compliance and additional support for existing infrastructure—such as LDAP-based enterprise directories, Java-based application servers, clustered systems, and distributed disk arrays—further simplify integration of eXcelon products into existing infrastructure and significantly shortens an application's time to market. With an established record of enterprise deployments and dedicated XML research and development, the eXcelon solution is a proven method for tackling demanding XML-based infrastructure.

Scalable, Fast, Native XML Management

Traditional approaches to XML data and content management have relied on either a computer file system, resulting in flat files, or on a relational database. Flat file-based implementations have inherent problems in performance and scalability, increasing the challenges to query over large sets of XML documents. There are XML indexing schemes on the market designed to work with XML data stored on the file system, but they are typically limited in capability and can improve performance for only highly restricted queries. Subsequently, managing arbitrary sets of XML data becomes difficult.

Relational databases do provide advantages in security and storage management. However, complexities arise during attempts to manipulate hierarchical XML structures within the flat, tabular structure common in relational databases. Some relational databases solve this through XML-to-tabular mapping, while others resort to storing the information as binary objects, thus providing few advantages over using the file system. XML-to-tabular mapping is suitable for simple and relatively flat XML structures. However, because the real advantage of XML is its facilitation of highly descriptive, complex, and extensible data structures, XML-to-tabular mapping loses a lot of this rich information and often resorts to using binary objects for storing XML data structures that are more complex. The result is a loss of the many advantages XML usage brings to application development. eXcelon's eXtensible Information Server is designed to manage XML natively and provide all the advantages developers expect when using XML as a core component of their applications.

The eXtensible Information Server stores XML data in a preparsed format to eliminate the overhead associated with parsing on demand, which is common with other XML storage methods. XML data can be more easily and efficiently manipulated once it is in a preparsed form. This unique storage allows operations to scale from small to large data sets with little performance degradation. File system and relational implementations have to load entire or arbitrarily large portions of XML documents into memory before operations can be performed on them. The eXtensible Information Server, on the other hand, loads only the data objects necessary to perform the requested operation, thus greatly improving scalability and performance. Once those objects are loaded, they are cached to improve concurrent access and overall application performance. Furthermore, eXcelon's patented distributed caching mechanism allows applications to scale in order to maintain performance under any user load. These features provide scalable, high-performance, extensible information management for any XML-based application.

Extensibility

The eXtensible Information Server (XIS) is designed to leverage the extensibility of XML. Extensibility is crucial because it lets you design and build infrastructure without prior knowledge of all data transactions. Traditional databases enforce a rigid model that is not suitable for storing and leveraging the vast resources of unstructured, proprietary content that exists in corporate information systems. Conversely, XIS lets organizations easily modify document structures without breaking existing applications, support XML documents of differing structures and schemas easily within the same application, and make live updates to existing XML documents — even add new nodes and attributes. XIS is designed to work with any arbitrary, well formed XML document, regardless of its physical size, structure, or schema. This flexibility lets application developers design their schema based on an application's needs and modify that schema dynamically as those needs change. Finally, with document type definition (DTD) validation support, XIS can optionally enforce schemas within applications to keep data synchronized. These capabilities avoid the inherent problems with schema evolution in traditional databases and provide support for more descriptive and dynamic information.

Fast Data Transformation and Delivery

The eXtensible Information Server's aggregates, manages, transforms, and delivers information so each system gets the right data in the right format. The transformation engine is based on industry-standard XSLT and complies with the XSLT 1.0 specification. Designed to take advantage of eXcelon's unique XML storage architecture, the XSLT processor is designed to perform fast, efficient transformations that target any delivery channel regardless of the source's data set size. Developers can easily define transformations that target Web browsers supporting HTML, wireless devices supporting WML or i-Mode, or even voice using VoiceXML. XSLT style sheets can be visually designed, developed, and debugged in eXcelon's Stylus Studio, an integrated development environment designed to assist developers at every stage in the development of an XML-based application. Full separation of XML data from presentation allows delivery to multiple channels simultaneously from the same application. eXcelon's patented distributed caching architecture ensures fast, efficient delivery of content views to any intended channel.

Usability

The release of eXcelon's XIS introduces eXcelon's next-generation development environment, Stylus Studio. Based on the award-winning Stylus™ XSLT editor, Stylus Studio provides all the components application developers need to build Web-enabled applications. This comprehensive management tool now provides a single-point of administration for all XIS installations.

Standards Compliance

The eXcelon eXtensible Information Server is one of the few XML-based solutions on the market today that is designed from the ground up for standards compliance. Release 3.0 improves upon the hallmark of standards support in previous eXcelon releases by providing support for the W3C standard Document Object Model (DOM) Level 1 and Level 2 (Core) APIs, XML 1.0 with namespaces, XPath 1.0 query language, XSLT 1.0 with Java extension functions, DTD, and XML Schema 1.0. It also supports all standard document encodings for international languages.

Ease of Integration

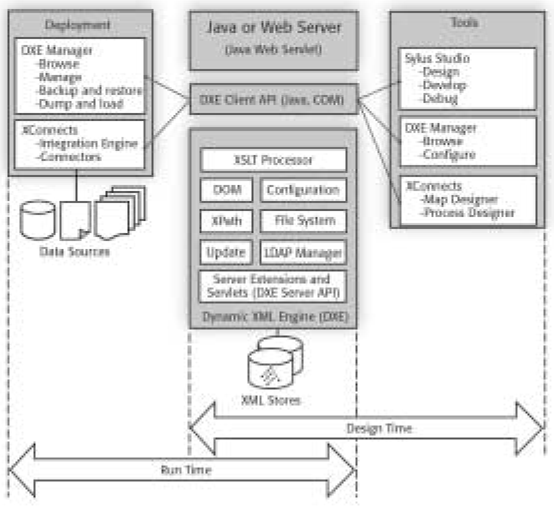
Release 3.0 adds support for user management by utilizing standard LDAP-compliant enterprise directories. It includes improved support for integration with Java 2 Enterprise Edition (J2EE) application servers, including BEA WebLogic and IBM WebSphere. It also provides integration with cluster management software from Microsoft, Sun, and Veritas.

eXtensible Information Server Architecture

The eXtensible Information Server (XIS) is composed of design-time and run-time components. Design-time components assist application developers in designing, building, and deploying applications with XIS. Run-time components provide the infrastructure for running, distributing, and scaling Web-enabled applications. Figure 3 provides a high-level overview of the components of the XIS solution.

The eXtensible Information Server's run-time components consist of the Dynamic XML Engine (DXE), DXE Managers, and XConnects. The design-time components consist of Stylus Studio, DXE Manager, Map Designer, (part of XConnects), and Process Designer (also part of XConnects). These components together constitute the full platform for designing, debugging, deploying, and integrating XML-based applications.

Figure 3. eXtensible Information Server Architecture



Design-Time Components

eXcelon's XML-based development tools provide a powerful environment for designing and testing XML-based applications deployed to the eXtensible Information Server. eXcelon Stylus Studio provides the core development environment for building applications using eXtensible Information Server frameworks and application programming interfaces (APIs). Map Designer and Process Designer simplify the design of aggregation patterns for content from foreign data sources to be utilized in platform applications.

Stylus Studio

Stylus Studio (shown in Figure 4) is eXcelon's new integrated development environment for XML-based applications. It includes various integrated tools for designing, debugging, and deploying applications on XIS. Its primary features include the following:

Figure 4. Stylus Studio

- **Project management.** Stylus Studio manages logical groups of files that compose applications. • XML editor. This is a scalable editor for XML documents, which is designed to handle documents hundreds of megabytes in size. It includes Sense:X tag completion, automatic DTD generation, structural and grid views, and syntax highlighting.
- **Java source editor.** This editor allows for simple edits of Java source files that compose applications and XSLT Java extension function and code compilation.
- **DTD and XML schema editor.** This feature is a visual editor for DTD and XML schema definitions. It supports the latest DTD and XML Schema standards and validation against referenced XML source documents.
- **XSLT editor.** This editor includes support for templates, Sense:X tag completion, syntax highlighting, parameters, Java extension functions, and multiple output formats and encodings.
- **XML-to-XML mapper.** This is a visual tool for defining mappings between different XML schemas. It supports the DTD and XML Schema standards. If documents do not have an explicit schema, the mapper will infer a schema for purposes of transformation. The mapper generates an XSLT stylesheet for performing the transformations defined in the visual mappings. This tool is useful for mapping from internal schemas to industry standard schemas such as cXML, xCBL, and ebXML. It supports reverse engineering: changes in XSLT source are automatically reflected in the visual map designer.
- **Debugger.** Stylus Studio includes a visual XSLT and Java debugger. It supports breakpoints, stepping, variables, watches, and call-stack tracing. It also integrates XSLT and Java debugging to allow seamless stepping through XSLT into Java extensions and back to XSLT.

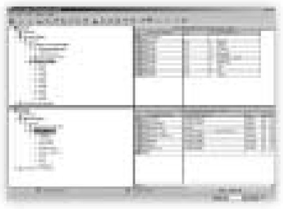
Map Designer and Process Designer

Map Designer (shown in Figure 5) is a visual tool for defining mappings from back-end sources into XML or vice versa. These mappings define how data will be transformed from a data source connected by an XConnect adapter into native XML for the eXtensible Information Server.

Figure 5: Map Designer

Source

Target



Map and Store

While the XML mapping tool in Stylus Studio allows easy mapping between XML schemas, Map Designer enables mapping from non-XML schemas into XML and vice versa. With Process Designer, it is possible to chain, batch, and schedule individual conversions together for one-to-many and many-to-one aggregations. Features of Map Designer and Process Designer include the following:

- **Design tool.** A design tool is provided to visually map fields from legacy sources into elements and attributes in XML target documents.
- **Scripting support.** A scripting language based on Visual Basic allows developers to create complex data scrubbing and record flow control rules.
- **Support for DTD and XML Schema.** Support for these standards allows easy mapping into existing structural definitions to facilitate integration of content into a common information model.
- **Visual scenario builder.** A visual tool is provided for designing aggregation scenarios. This tool is useful when information must be combined from multiple legacy sources, human or application intervention is required to approve or reject certain records, or the aggregation process involves several steps before the result is achieved.

Run-time Components

The run-time components of the eXtensible Information Server provide a scalable, flexible, and high-performing environment for managing and delivering XML data and content. The Dynamic XML Engine (DXE) is a native XML repository for managing, transforming, and delivering content. XConnects provides connectivity and transformation capabilities for aggregating information from various legacy data sources into standard XML. DXE Manager provides a single point of administration for DXE deployments.

Dynamic XML Engine

DXE is the core of the eXtensible Information Server. It provides scalable, high-performance data management for native XML, embedded query and indexing capabilities based on XML standards, and high-performance, standards-based transformation capabilities with the embedded XSLT processor that enable dynamic delivery of content views in various formats. Its primary features include the following:

- **DOM.** The standard Document Object Model (DOM) is the interface to XML data. DXE parses XML data to the node level and stores the preparsed data in the repository. The DOM interfaces allow access to these node structures.
- **File system interface.** The repository presents a file system-like interface for managing XML data. This greatly simplifies DXE data storage management.
- **Document linking.** With this feature, it is possible to reference other XML documents within a given XML document, creating conceptual or physical relationships among XML data items. When operations are performed on a document containing links, DXE transparently traverses links to return a composite document view.
- **Binders.** Based on the document linking mechanism, related XML documents can be bound together into a single virtual document view to simplify application development and data aggregation. Any operation that can be performed on a single XML document can be executed on a binder. Binders represent high-level relationships between XML documents. Documents can be part of multiple binders to allow for complex data relationships.
- **LDAP manager.** This component manages connections to external Lightweight Directory Access Protocol (LDAP) compliant directories for user management and authentication.
- **Configuration manager.** This component manages distributed processes, transaction throughput, routing policies, load balancing, and cluster integration.
- **Indexing.** Indexing of individual XML elements or groups of elements provides faster access to relevant data. DXE supports three index types: value indexes for string and numeric searches, text indexes for word-based searches, and structural indexes for structure-based searches.
- **XPath processor.** Using this high-performance, XPath-based query engine, queries (XPath expressions) can be created that span single documents or multiple documents in the repository. Live DOM nodes can be easily accessed from query results. If indexes exist, the XPath processor leverages indexes to greatly improve query performance.
- **Triggers.** Triggers are business logic components that are instantiated and run when an update occurs against stored information. Java code can be registered with the server and associated with DOM update operations. This registered code has full access to DXE server APIs.
- **Update.** DXE allows for live updates of XML content by using eXcelon updategrams. DXE guarantees transactional consistency and is optimized for real-time data interactions.
- **XSLT processor.** DXE includes a scalable, high-performance XSLT processor that transforms XML content into HTML, WML, or any required format for delivery to end user applications.
- **Java APIs.** DXE provides robust, powerful interfaces for developing applications. Applications that use DXE APIs can run via the thin client interface or directly within the DXE process.

XConnects Integration Engine and Connectors

The XConnects Integration Engine is a scalable engine for transforming data and content from legacy sources into standard XML. Combined with native connectors for over 100 different structured and unstructured sources, XConnects provides a powerful solution for aggregating diverse data and content sources. Features include the following:

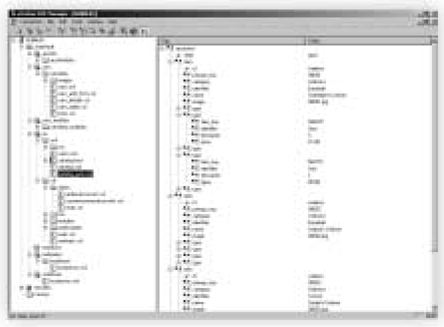
- **Connectivity.** Data sources supported include relational databases, industry-specific formats such as HL7 and DIALOG, mainframe systems such as CICS, file-based formats such as Paradox and Clipper, enterprise resource planning systems such as SAP, and groupware systems such as Lotus Notes.
- **Scalable record management.** Scalable data parsing and record management provides efficient data access and conversion, even for systems containing millions of records.
- **Java APIs.** These APIs programmatically control and automate the aggregation process from any application.

DXE Manager

The DXE Manager (shown in Figure 6) is a visual tool that provides a single point of administration for eXtensible Information Server installations. This tool is used to manage all components of the server. Features include the following:

- **Repository browser.** This is a Windows Explorer-like interface for browsing and managing DXE repositories.
- **Update and query wizards.** These wizards simplify the creation of XPath query and update expressions for quick updates to DXE-managed data.
- **Cache and route configuration.** With this feature, it is possible to add or modify cache processes and define routing policies between applications and caches for load balancing.
- **Backup and restore.** With this feature of DXE Manager, it is easy to manage online backups and restores of DXE repositories.
- **Data migration.** DXE Manager allows easy migration of data between DXE installations.
- **Bulk loading.** DXE Manager makes it possible to load entire directories or groups of directories containing XML data in one step.

Figure 6. DXE Manager



Enterprise Features

Release 3.0 includes many features designed to ease deployment of applications into existing enterprise infrastructure. These features were designed to be familiar to information technology professionals.

- **Online backup and restore.** The eXtensible Information Server supports the backup and restore of XML repositories between offline storage mechanisms. Both operations are performed online, allowing the system to continue operation while the operation completes.
- **Partitioning.** This feature makes it possible to decouple physical storage locations from logical storage arrangement. It allows for simplified deployment to large RAID disk arrays and other scalable storage architectures.
- **Distributed caching.** This feature provides high-performance delivery of raw XML data or XSLT-generated data views. An internal cache manager manages all cache processes and will automatically restart them should an error occur. This feature allows for load balancing and fault-tolerant operation.
- **Routing.** This feature facilitates the managing of client requests for data across distributed caches, allowing for more efficient distribution of application load and faster access to frequently used data.
- **Clustering.** The eXtensible Information Server supports various cluster environments, such as Microsoft and Sun clusters, for high-availability configurations.
- **Monitoring.** Programming interfaces are available for monitoring transaction, data loading, query, and transformation performance

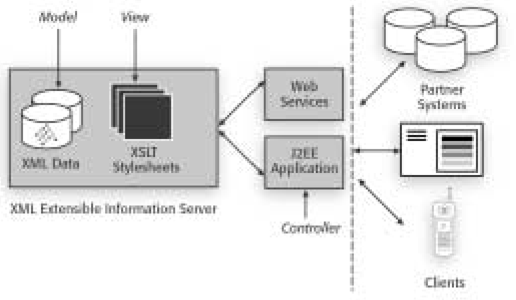
Developing and Deploying Solutions

The eXcelon eXtensible Information Server integrates with existing Web servers and application servers to manage user and system interactions over the Internet. Applications are developed using a familiar Model-View-Controller (MVC) development methodology and deployed easily into existing Web and application server infrastructure. Choosing between a Web server-based deployment using a basic Java servlet or Java Server Pages (JSP) engine and a J2EEbased application server deployment depends on the complexity of the application and user load expected. The eXtensible Information Server is designed to scale effectively with either deployment option.

Application Development Methodology

Applications developed using the eXtensible Information Server fit the well-known Model- View-Controller (MVC) paradigm. XML data stored in the repository forms the extensible business information model for applications. Views of data for delivery applications are created using industry-standard XSLT that is processed using the eXtensible Information Server's embedded XSLT processor. The controller consists of J2EE-compliant applications or Web services components. This model provides a flexible and well-known methodology for developing and deploying XML-based applications. Figure 7 illustrates the model used in developing applications based on the eXcelon eXtensible Information Server.

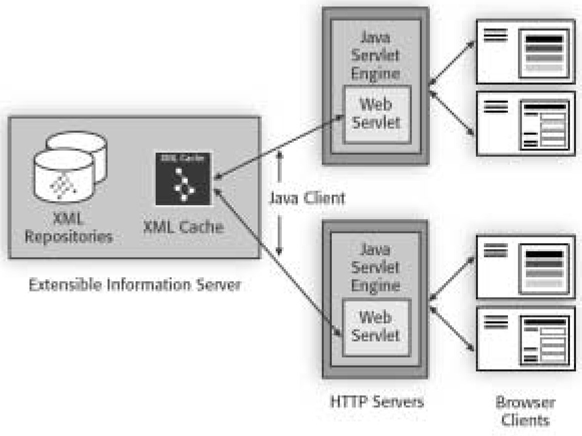
Figure 7. Model-View-Controller (MVC) Architecture Using the eXtensible Information Server



Using a Web Server

Designed for applications of little complexity and light to medium user loads, the simplest deployment option is using a standard Web server combined with a standard Java servlet/JSP engine. Figure 8 illustrates the deployment architecture when using the eXtensible Information Server with standard HTTP Web servers. eXcelon bundles the Apache Tomcat servlet engine and provides the Web Servlet Framework (WSF) for quickly deploying applications. WSF provides a URL interface to most functionality available in the eXtensible Information Server Java client. The URL interface allows you to retrieve entire XML documents, perform queries or transformations at the server, update existing XML documents, or execute server-side code. This option is recommended for smaller departmental applications or any application that will not experience heavy usage.

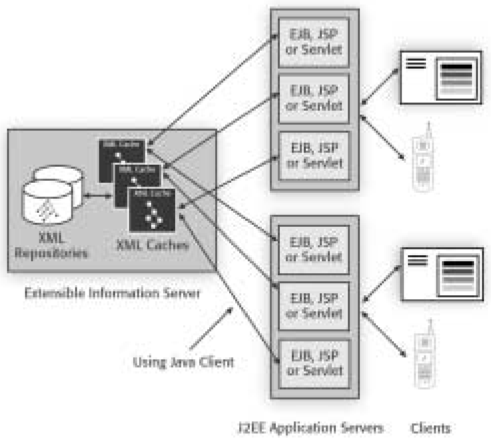
Figure 8. Deployment with Web Servers



Using a J2EE Application Server

Designed for more complex applications and heavy user loads, using a J2EE application server is the preferred option for enterprise-level deployments. Figure 9 illustrates the deployment architecture for the eXtensible Information Server when used with a J2EE application server. Application servers provide the ability to manage thousands of user interactions and programming interfaces for developing and hosting complex business logic. All the eXtensible Information Server APIs are 100% pure Java, allowing them to be used in any J2EE application. Any component hosted by the application server (JSP, servlets, and EJB) can access server functionality through Java APIs. The routing and distributed caching features work in tandem with the high-availability features of popular application servers to efficiently distribute application load and improve performance. Documentation and support is provided by eXcelon for integrating successfully with the most popular application servers, including BEA WebLogic and IBM WebSphere.

Figure 9. Deployment with J2EE Application Servers



Conclusion

The biggest challenge involved in creating Web-enabled applications is the timely access and management of the business data behind the application. XML is clearly the most appropriate technology for these next-generation applications. The corporate landscape is fraught with heterogeneous applications and a variety of different legacy data sources. Replacing or removing those applications is usually not an option. Therefore, those sources need to be unleashed for new applications in a way that preserves the existing investment while requiring minimal new investment. Creating a robust and extensible common information model is seen as being the utopia for managing the data and content that is the backbone of Web-enabled applications. Basing this model on XML provides a future-proof foundation for building robust applications and value-added services. The eXcelon eXtensible Information Server has been built from the ground up to provide scalable, high performance, reliable access to XML data and content. These capabilities will provide any company with the power to unleash the information both inside and outside its walls in more efficient – and profitable – ways