

eXcelon Business Process Manager Technical White Paper

Executive Summary

This white paper allows the technical decision maker to understand and evaluate the eXcelon Platform and the capabilities of the eXcelon Business Process Manager™ (BPM) so as to make an informed decision before purchase. This paper describes the eXcelon Platform in brief and then presents the technical aspects of BPM.

Organizations are struggling to gain competitive advantage by identifying and streamlining eBusiness solutions that create business value by expanding markets, reducing costs, and fostering customer loyalty. One emerging trend is the evolution of enterprises toward a more decentralized, outsource-dependent environment. To build this model successfully, an enterprise needs to integrate outsourced products and services into its back-end systems.

Hard-coded, point-to-point solutions cannot deliver the required flexibility or speed for today's eBusinesses, which have IT infrastructures that consist of applications implemented at different times, for different constituencies, with different objectives. An organization must be able to rapidly deploy new business solutions and maximize the reuse of existing computing resources. This ability enables the enterprise to differentiate itself among its competitors.

According to Gartner Research, by 2004, market and economic forces will require global enterprises in certain industries — such as electronics, consumer packaged goods, and energy — to provide collaboration capabilities in 70 percent of their operations. As companies extend their processes to business or trading communities inside or outside their enterprises, the operating environments that result are very complex. Bringing order to such a chaotic mix of technology, through systems integration, can be a daunting challenge. eXcelon Corporation with its rich industry expertise, is the premier provider of platforms based on XML. eXcelon's strength results from its deep vertical industry experience, its use of value-based diagnostics, and its holistic approach to design. eXcelon's superior, award winning products are the key to why eXcelon's solutions are the best.

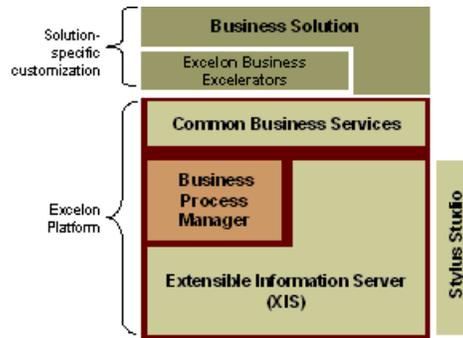
The emergence of standards creates an opportunity to rapidly implement, monitor, and continuously change composite business applications. Composite business applications are configurable, user-centric business processes that extend existing software assets to support unique business processes. These processes span multiple systems and constituents across the value chain, integrating a corporation's suppliers, customers, and trading partners.

The eXcelon Platform meets and exceeds these criteria by enabling an enterprise to

- Automate manual processes, which minimizes the need for human intervention. Run-time system monitoring and management is Web based.
- Create new applications, as BPM enables new views into existing systems and allows for enhanced development processes.
- Bring new partners online quickly, since internal document formats and business processes do not have to be changed.
- Easily integrate with existing J2EE-based infrastructures using the Enterprise and Universal Editions.

The eXcelon Platform

eXcelon solutions are deployed on an extensible technology platform that consists of both customized solution-specific components and the eXcelon Platform that is shared by all applications. Composite business applications are built using a service-based architecture where new business processes are assembled from discrete building blocks. These building blocks include custom code specific to an organization, industry-specific reusable components called Business Excelerators, and a process integration platform called the eXcelon Platform. The eXcelon Platform is the foundation that makes creating and managing composite business applications possible. It was designed for Enterprise-class deployments delivering high throughput, scalable, and highly available business systems based on 100 percent XML, J2EE, and other widely accepted standards.



The eXcelon Platform is a service-based solution that serves as an enterprise's eBusiness platform. The platform eases implementation, fosters data and application integration, and paves a fast track to interoperability and value. The reason for this is that the platform

- Is a J2EE-based programming model for defining, designing, developing, and deploying business systems
- Is designed from the ground up for persistently storing XML to manage process instance context (as an XML document) and data flow
- Has distributed caching and routing policies to dynamically adapt to system load without disrupting other applications
- Allows distributed components to access data from local caches at in-memory speeds
- Uses transformation technology to manage the data flows on a many-to-many basis

The eXcelon Platform is more than a net services platform — it is the solution that makes net services and value chain integration possible. It allows companies to provide a "public" net service that has value to its users while making value chain integration work. Key characteristics of the eXcelon Platform include

- *Ability to create innovative, high-performing, and collaborative business services that are standards based.* The platform easily integrates new data sources, appli-

cations, and devices to enable data, content, and business process sharing and growth over time with partners and suppliers. The information-rich services support both process-to-process and process-to-human interactions.

- *Reusable solution components.* The platform offers varied benefits to businesses — chiefly cost-effective, low risk, and rapid solution deployment attainable in weeks instead of months, and industry-specific business process workflows and services.
- *Continuous monitoring of key performance metrics.* The platform provides a dashboard that allows clients to monitor their business solutions against key performance indicators. This enables continuous business process improvement parallel with real-time data aggregation analysis to deliver business performance results.
- *Comprehensive process automation.* The platform enables strong business relationships that extend beyond transactions with internal and external partners. It enables the automation of processes defined as business documents, making these processes easier to modify and grow over time. The high-level approach enables business analysts to define the business processes, resulting in a greatly accelerated development process.
- *Reduction in the total cost of ownership for infrastructure development, administration, support, and maintenance.* eXcelon uses 100 percent (native) XML to define processes, store content, and exchange data to support the full scope of existing and emerging XML standards. The extensibility of XML reduces the cost of ongoing maintenance and increases the reusability of

existing infrastructure and system interfaces.

Platform Components

The eXcelon Platform consists of four integrated components: Common Business Services Suite, Business Process Manager, Extensible Information Server™, and Stylus Studio™. It also includes integrated development and integration tools.

Common Business Services Suite

The Common Business Services Suite (CBSS) is a J2EE-integrated programming environment that provides a comprehensive framework for creating, executing, and managing new and existing applications services. CBSS supports asynchronous access and coordination of disparate systems (such as corporate databases, enterprise applications, and internal resources) that perform existing core functions that are transparently accessible as plug-and-play business services which can be integrated across the value chain.

Business Process Manager

The Business Process Manager (BPM) is an XML-based engine that enables an enterprise to model and manage distributed business processes. Unlike traditional workflow engines designed to work solely within the enterprise, BPM allows an enterprise to map existing business processes into scalable and flexible models capable of handling long-running, complex, and highly collaborative interactions with human users and systems both inside and outside the enterprise.

The BPM toolset includes the

- BPM Administrator, which allows a user to
 - Manage external and internal users
 - Manage nodes
 - Manage business processes
 - Configure BPM
- Process Flow Designer
 - A Stylus Studio plug-in that allows users to graphically define and modify business process control flow, services, and collaborations.

End users interact with business processes through BPM worklists or user-defined interfaces.

Extensible Information Server

The Extensible Information Server (XIS) collects, transforms, and delivers data and content in any form to any source — in real time, with guaranteed transactional consistency. XIS offers a robust, scalable, W3C standards-based information management solution for creating a centralized, flexible, and extensible common information model built on XML.

The Dynamic XML Engine (DXE) is the core of XIS and provides scalable, high-performance data management for native XML, query and indexing capabilities, and transformation capabilities. The DXE Manager is a visual tool that provides a single point of administration for XIS for configuring, managing, backup/restoring, and dump/loading. Command-line utilities are

also available to perform these administrative functions.

Stylus Studio

The industry-leading, award-winning Stylus Studio is an advanced and integrated interactive development environment for working with the various types of files in an XML application. Stylus Studio allows you to

- Edit and query XML
- Work with XSLT
- Map XML to XML
- Debug stylesheets and Java files
- Define schemas

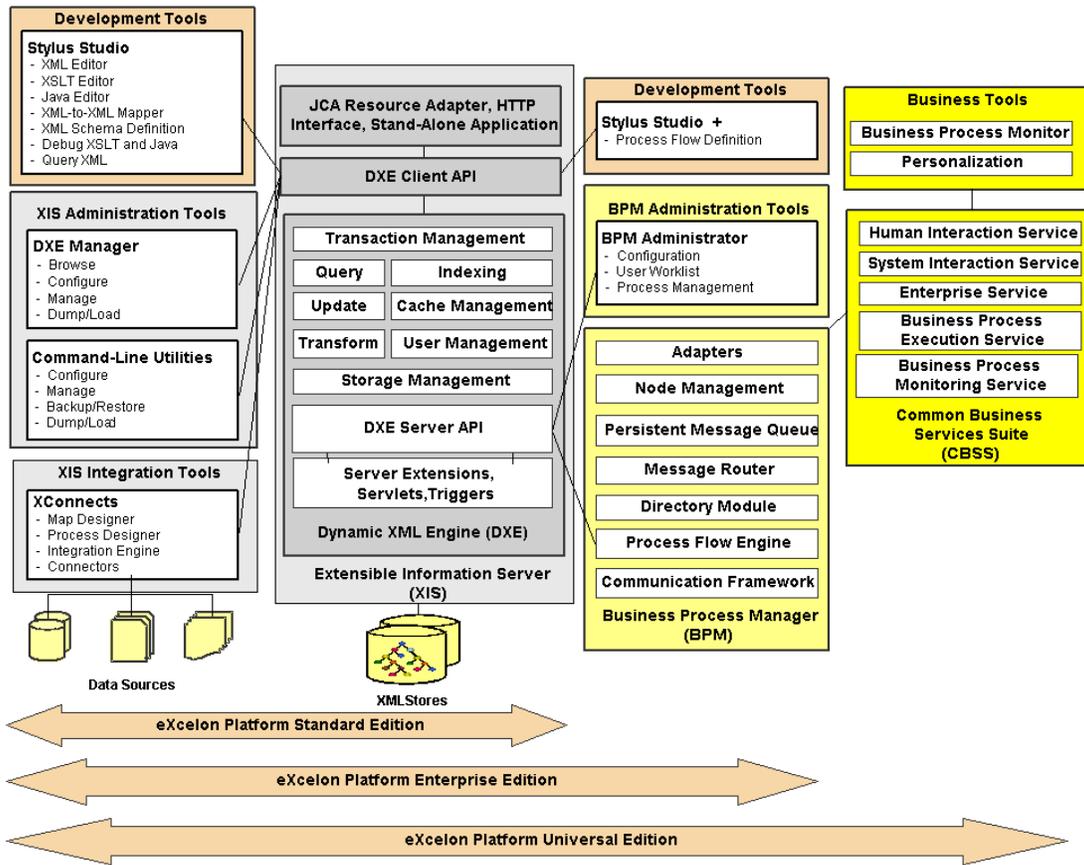
XConnects

A component of XIS, XConnects is the platform's tool for aggregating data and content from legacy formats both inside and outside the enterprise into XML. It enables the platform to support connectivity for over 100 structured and unstructured data sources.

Weblogic Application Server

Using BEA's Weblogic Application Server, BPM easily integrates with existing Java 2 Enterprise Edition (J2EE)-based infrastructure for interoperability with other J2EE applications.

The following illustration provides a high-level view of the eXcelon Platform components.



Platform Editions

The eXcelon Platform is available for purchase in the following editions, depending on how it is bundled.

Edition	XIS	BPM	CBSS	Stylus Studio*	Supported Operating Systems
Standard	X			X	Windows NT, 2000, and XP Solaris 2.6, 2.7, and 2.8 HP-UX 11
Enterprise	X	X		X	Windows NT, 2000, and XP Solaris 2.6, 2.7, and 2.8 HP-UX 11
Universal	X	X	X	X	Windows NT, 2000, and XP Solaris 2.6, 2.7, and 2.8
Stylus Studio Stand-Alone				X	Windows NT, 2000, and XP

* The Standard, Enterprise, and Universal editions only support a Windows NT, 2000, and XP version of Stylus Studio

The Business Process Manager

eXcelon's BPM is a business process management system (BPMS) that coordinates the processing between enterprise systems, partners, and users. During the modeling, deployment, and management of new business processes, BPM enables business analysts to work with developers to customize existing business processes through the expression of business rules, applying to business problems such as

- Selection
- Assessment, scoring, and prediction
- Classification
- Monitoring
- Configuration verification
- Diagnostics and prediction
- Planning

BPM enables an enterprise to leverage its existing IT investments, offering an open architecture based on industry standards. BPM simplifies integration with any back-end office system, enterprise middleware, or packaged application on any platform or operating system. BPM supports standard business-to-business collaboration protocols, including RosettaNet Implementation Framework, BizTalk Framework, and the ebXML message service, without requiring that business partners on every side of the value chain be running the same BPMS or a BPMS at all.

NOTE: The RosettaNet and BizTalk protocols are currently supplied as custom options by eXcelon's Professional Services Organization.

The need for the production and consumption of standard business events is being fueled by the emerging requirement for enterprises, regardless of their size, to engage in business transactions with business partners. Such partners may vary greatly in the nature of their bilateral or multilateral business relationships.

These transactions are asynchronous in nature, document centric (hence XML), and typically Web based.

BPM provides a scalable infrastructure to manage incoming and outgoing business messages with an explicit process model. This model goes beyond the capabilities of point-to-point architectures that directly tie the back-end systems into the message ports of BPM. It provides flexibility and is designed for dynamic environments where process definitions, services, message formats, and participant profiles change on a daily basis.

BPM Concepts

There are several concepts used throughout this paper to describe BPM and its functions:

- Projects
- Business processes
- Collaborations and participants
- Services
- Activities, states, transitions, and guards
- Process roles
- Events
- Nodes

Projects

The eXcelon Platform uses the concept of a *project*. A BPM project is a group of related business processes. Each BPM project is a separate administrative domain and it has its own configuration file. BPM uses this configuration file to maintain information related to that project. When BPM is installed, it creates this file and stores it in an XMLStore. (XIS maintains XMLStores, which contain persistent XML and non-XML data.

A BPM project

- Maintains a directory, in an XMLStore, that contains the definitions of BPM constructs defined in that node; for example, process definitions, collaborations, and services.
- Can include any number of business processes, which you define.
- Is an administrative domain with its own access control list. When a user logs in to BPM Administrator, that user has access to exactly one BPM project. To manage multiple projects, the user must log in to BPM Administrator with a different account for each project.
- Has at least one BPM Process Flow Engine (PFE).
- Is a single namespace for document types and events.
- Maintains a directory that contains participant information, or has its own base DN (Distinguished Name).
- Has a BPM Node Manager for each host on which the project resides.

Typically, a BPM project is developed on one host and deployed on another host. To deploy the project, it is necessary to create a new project on the deployment host. Once that is done, there are two ways to deploy the project:

- Copy the **.definition** tree from the development configuration file to the new project's configuration file. For all other parts of a BPM project, the BPM Administrator is used to recreate them on the deployment host. This includes specification of nodes, adapters, and participants.
- Copy the entire development project to the new project on the deployment host. It might be necessary to change host names, and it is possible, of course, to add new participants.

Using a loopback adapter, users can simulate an enterprise system in a process and test a BPM project without all the services in place. The loopback adapter never leaves BPM but returns to the sender.

Business Processes

In an enterprise, a *business process* is a pre-defined sequence of steps that accomplishes some unit of work in the context of the enterprise.

Business processes are as varied as the businesses in which they are used. For example, there might be business processes for these types of tasks:

- Purchasing supplies and materials from partners
- Maintaining inventory
- Delivering new products to partners and customers
- Upgrading released products
- Selling products to other businesses and to consumers
- Communicating with employees
- Managing a manufacturing process

In BPM, a business process is represented as an UML activity graph in Stylus Studio and includes all of the operations performed by BPM to accomplish a specific task.

Typically, a business process makes use of one or more documents throughout its execution. These documents do not necessarily remain constant throughout the execution of the business process, as various participants have the potential to operate on the document(s). For example, each may be responsible for some particular aspect of a document such as generating a requisition form, adding a price to the form, approving the form, and generating a PO. In BPM, users can define business processes so that BPM stores these *in-flight documents*.

When using the Process Flow Designer to define a process, the user creates a UML

activity graph that represents the control flow of the process. This graph models the *states*, *activities*, *events*, *transitions*, and other constructs that make up the business process control flow.

Collaborations and Participants

When BPM executes a business process, it does so by executing operations that are part of one or more *collaborations*. A collaboration defines a framework for two parties to exchange messages as part of a business process. The messages can be notifications, service invocations, or service responses. One party to every collaboration is a business process that the user uses BPM to define. The other party, which the user also defines, can be an internal user, a partner, another BPM process, an enterprise system, or a group. One party initiates the collaboration and the other party provides the collaboration. However, either party can provide a service to the other party.

When defining a collaboration, the user defines the collaboration roles played by each party to the collaboration. For example, the two roles might be **buyer** and **seller**, **student** and **registrar**, or **engineer** and **bugtracker**. In addition, the user identifies which party initiates the collaboration and which party provides the collaboration.

A *participant* in a collaboration can be an internal user, an internal enterprise system, an external partner, or a group of participants. Users use the BPM Administrator to configure participants. BPM maintains a directory that contains information about participants. For each participant in each collaboration, some set of protocols and standards that BPM uses for communication between BPM and that participant is specified. At run time, BPM determines which participants to collaborate with.

When BPM executes a business process, the business process collaborates with one or more participants to accomplish the defined activities. In addition to collaborating with participants, the business process can collaborate with other business processes that the user defines.

A *collaboration profile* holds information about the document format, adapter, and transport protocol to be used to route a message in a collaboration. When BPM needs to send a message, BPM obtains a computed collaboration profile from the directory module. BPM needs the computed collaboration profile to determine how to send a message to a particular participant. A collaboration profile contains fields for the following information:

- Document format.
- Adapter or process name.
- Adapter or process parameters. A list of parameters is a map of keywords and associated values. The meanings of the keyword/value pairs depend on the adapter or process.
- Transport protocol name.
- Transport profile, which contains an address and possibly other information needed to establish a connection.

The advantage of putting this information in a collaboration profile is that the information is not in the process definition. Keeping this information separate from the process definition allows different participants to dynamically be parties to the collaborations defined in the business process.

Services

When BPM executes the steps defined in a collaboration, it engages in a sequence of related interactions with a single party. Each interaction consists of the sending of a message that invokes a service, possibly together with a reply that contains the service's output for that invocation.

BPM or a participant can send messages that invoke services. One of the tasks involved in defining a business process is to define activities that invoke services. The recipient of a service-invoking message executes a service. Execution of a service usually consists of processing a business document. For example, a **ProcessPO** service might process a purchase order and send a purchase-order acknowledgement document in a response message.

Activities, States, Transitions, and Guards

A *state* in a process flow is a “time slice” where some optional actions can occur and where a business process waits for one or more events to occur that control the flow of the business process.

An *activity* is a *state* that either invokes a service or sends a response message from a service. A state that is not an activity waits for an event that meets the conditions of one of the state's transitions.

A *transition* is a link from one state to another state that is used to control the flow of the business process. They are typically associated with message events. Users can specify one or more guards for each transition.

A *guard* is an XPath expression to be evaluated against a document known by the business process, referred to as an in-flight document. When a transition is about to occur, BPM evaluates the XPath expressions one at a time against the in-flight document. As soon as a guard evaluates to true, BPM fires the associated transition. If no guards evaluate to true, the transition does not fire. If there are no guards, the transition always fires.

Process Roles

A *process role* is the role performed by a party that collaborates with a business process. For example, a process role could be **buyer**, **seller**, **student**, **registrar**, **engineer**, or **bugtracker**.

Each process role name in a particular business process must be unique. This allows BPM to distinguish one party from another.

Identification of the roles in the process is in fact more important than identification of the actual participants. BPM executes processes that interact with participants that play roles in the process. For example, in the purchase process, in the collaboration between BPM and an ERP system, the ERP system plays the role of **PurchaseOrderClerk**. There are many operations the ERP system can perform, but in this collaboration, the process role of the ERP system is that of processing a purchase order.

Events

An *event* is when any internal or external process signals its status to BPM in a way that causes BPM to take further actions. Types of events in BPM include message events, change events, internal events, and error events.

BPM provides an event notification service that notifies users, partners, enterprise systems, and groups, about particular process and activity events. BPM maintains a list of notifications per user, which are stored in a separate XML file.

When an event is generated or received by BPM, any state that has transitions defined that match the event will be analyzed. Each transition can have an optional guard that further controls which transition to fire.

Admin Server

The Admin Server is an HTTP server that enables the administration of BPM. The Admin Server runs an HTTP transport listener and several adapters that perform various BPM Administrator functions such as node management, process management and participant management. For most installations, there should be only one per project.

Nodes

BPM relies on *nodes* to provide partner communications, enterprise application integration, distributed component models, and user interactions. A BPM node

- Is an operating system process
- Contains zero or one Process Flow Engine
- Runs a Java VM
- Runs any number of transport listeners, inbound handlers, outbound handlers, and transport senders

A BPM node is essentially a container for the code that constitutes BPM. A single machine can host multiple BPM nodes. Each BPM node can belong to exactly one BPM project. More than one node can use the same configuration file. All nodes that use the same configuration file belong to the same BPM project.

BPM has a *Node Manager*, a Java process that controls node processes. The Node Manager

- Loads the BPM node into a JVM process
- Provides an interface to start and stop local BPM nodes
- Hosts an **RMIRegistry** thread

Adapters

An *adapter* handles different packaging protocols between participants and BPM. An adapter consists of an inbound handler (BPM receives messages from participants) and an outbound handler (BPM send messages to participants), each of which is implemented as a Java class. For messages BPM receives from participants, the inbound handler

- Decodes the message into one or more documents.
- Fills in relevant fields in the message event that the transport listener created for this message. The relevant fields include the documents the message contains.

- Calls the message router, which determines whether to send the message to the PFE or to an outbound handler.

For messages BPM sends to participants, the outbound handler

- Examines the participant address information in the collaboration profile to select the transport protocol to use to send the message
- Packages the message for the selected transport protocol
- Invokes the selected transport sender and passes the message to it

BPM has its own node framework to run and administer adapters, and it has its own “internal messaging” to communicate asynchronously between adapters. BPM includes the following adapters:

- **simple**
This adapter can call the HTTP, FTP, JMS, or email transport senders. It is for messages that contain exactly one XML document.
- **ebXML Messaging/TRP**
This adapter is intended for use in communicating via ebXML.
- **EJB** (Enterprise JavaBeans)

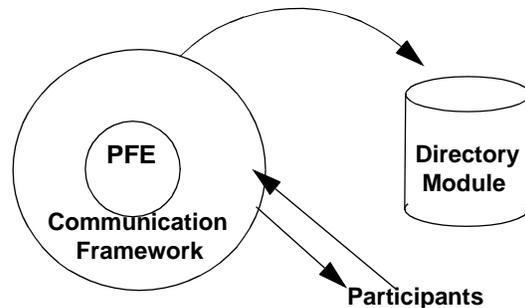
Each adapter uses one of the supported packaging protocols to package a message in a particular way. For example, the **simple** adapter decodes the body of an HTTP POST request, JMS message, or email message into a single XML document.

BPM Architecture

Conceptually, the BPM architecture has three main components:

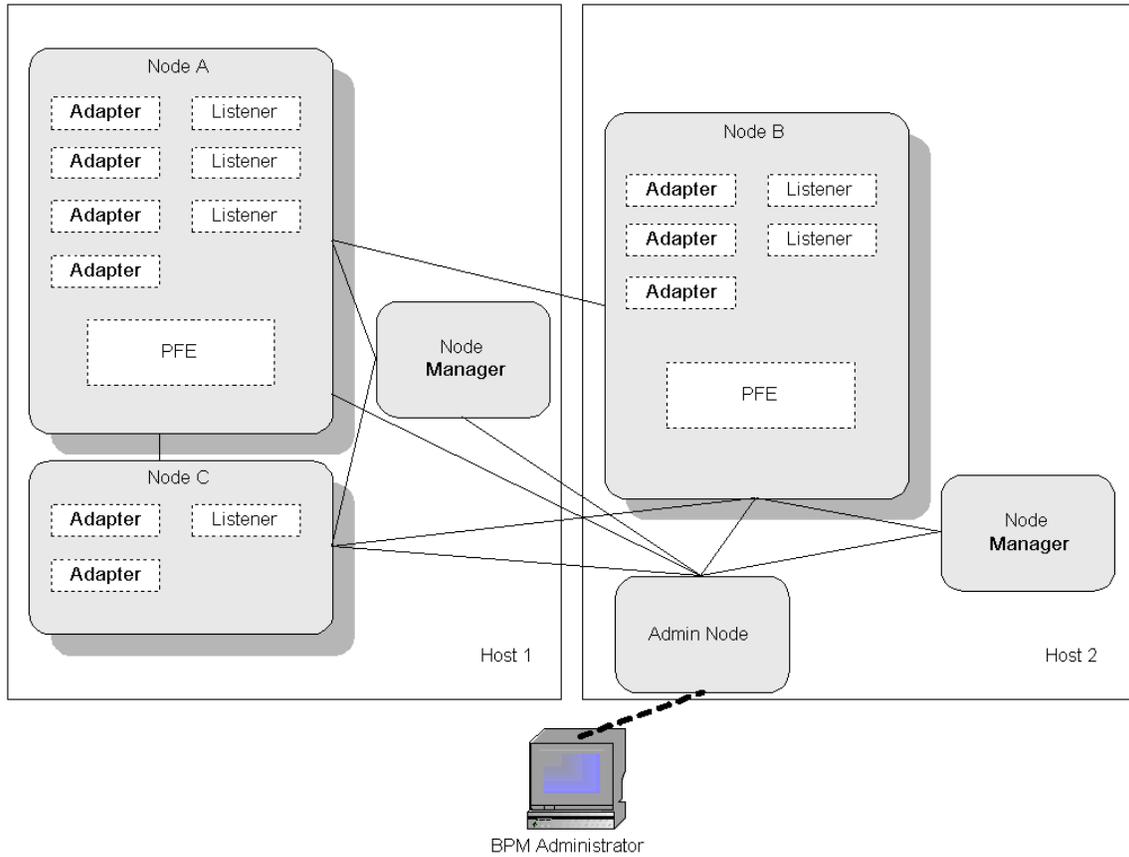
- Process Flow Engine (PFE)
 - Directory module
 - Communication framework
- These three components have clearly separate responsibilities. This architecture allows you to hold any one of these constant and vary the others. Each time your business process uses a collaboration.
- The Interaction between parties is the same.
 - The parties can be different.
 - The protocols used can be different for each party.

The following figure shows the PFE at the core of BPM:



Physically, the BPM architecture consists of nodes. Each node can run either one PFE or none, multiple transport protocol listeners, and multiple inbound and outbound handlers (also called adapters).

The following figure shows an example of a physical BPM project:



In a BPM project, each node can communicate with every other node. Also, each PFE is responsible for a particular subset of business processes. For example, suppose **NodeA** receives a message that is part of a business process being run by the PFE on **NodeB**. **NodeA** forwards the message to **NodeB**.

BPM relies extensively on the integrated components of the eXcelon Platform to provide the capabilities to design, develop, monitor, and update processes. For example, BPM uses XIS to handle flexible document formats, efficiently store and query XML documents, and store any kind of data (XML or non-XML). The Stylus Studio toolset provides the capabilities to design business processes and collaborations and edit or update them as needed.

Process Flow Engine

The PFE manages long-running transactions and real-time messaging events and is not limited to simple request/response interactions. This robust engine lets a company model and track arbitrary, complex business processes spanning days, weeks, or longer and can include customers, suppliers, partners, employees, and existing business systems.

The PFE orchestrates and executes the work in an automated business process. The completion of an activity that has multiple possible results is modeled as different message events. Each of these message events can trigger a transition.

In UML, activity completion is traditionally modeled with a single completion transition, which can be followed by a branch that queries the outcome.

Modeling activity completion as multiple transitions works well in BPM because guards are XPath expressions that are evaluated against documents in the message event. In a process branch, guards are evaluated to determine the control of flow for the business process instance.

The PFE supports advanced modeling features, including:

- Collaborations
- Long transactions with support for compensating transactions
- Event notifications
- Exceptions

BPM manages explicit data flow associated with the process flow and the message flow. The PFE manages the life cycle of all the business documents. The control flow is directly evaluated against the business document content wherever appropriate (for example, in branching, performer identification, and parameterization of services). The types of message events that enter BPM from internal and external participants are used to determine the control flow of business processes.

Directory Module

The directory module keeps track of users, partners, enterprise systems, and groups that participate in collaborations. The directory module maintains addresses and other attributes and information about how messages are exchanged. This allows BPM to look up participants, find a participant for a collaboration, and correctly exchange messages with the selected participant.

BPM accesses and stores directory information via standard Java JNDI (Java Naming and

Directory Interface). BPM can make use of an existing site's LDAP implementation to take advantage of the existing information assets contained within it. BPM also provides a JNDI interface to XIS (the default configuration) for sites that are not using an LDAP repository.

Communication Framework

The communication framework provides the bridge between the PFE and outside services and systems. It sends messages from participants to the PFE and from the PFE to participants. The framework is made up of three layers in addition to a message router. For each layer, a user can specify which standards and protocols BPM should use when it sends or receives a message. The layers are

- *Transport protocol layer* specifies the protocol BPM uses to send or receive a message. For example, this might be HTTP, HTTPS, FTP, or email.
- *Packaging protocol layer* specifies how BPM packages documents into messages. It can also specify information for encryption, digital signatures, and routing. It can specify information about acknowledgements, receipts, retransmissions, and elimination of duplicate messages.
- *Document format layer* specifies the format of documents that are included in messages. An enterprise might pick an XML format that is popular with its partners, or an XML format that is an industry standard, or define its own XML schema.

The user must decide which protocols and standards to use with each participant. Also, the decision on which standard or protocol to use in each layer is independent of the decision on which standard or protocol to use in the other layers. Process flow definitions can also be defined independent of these decisions. Once this information is obtained, the user can use the BPM Administrator to specify each participant and the standards and protocols to use when BPM communicates with that participant.

New partners with a different protocol requirement can be added without changing process flow definitions so that existing partners are not affected by this change. Examples of this might be a new revision of the ebXML message specification, or RosettaNet implementation framework specification, or possibly even a direct RMI connection over a dedicated port.

A new or changed document format can also be added. For example, a new "standard" document for a purchase order has been developed. This "standard" can be supported

without changing the business process. In the case where there are multiple participants for which one or more of the same protocols and standards need to be specified, a user can add participants to a group and specify the values for the group. The transport protocols and packaging protocols that each BPM node runs can also be configured.

The following table shows the communication framework component that BPM uses in each layer when it sends and receives messages. Each of these layers is described in detail.

BPM Communication Framework Layer	Component BPM Uses When BPM Receives Message from Participant	Component BPM Uses When BPM Sends Message to Participant
Transport protocol layer	Transport listener	Transport sender
Packaging protocol layer	Inbound handler (adapter)	Outbound handler (adapter)
Document format layer	Stylesheet or Java component	Stylesheet or Java component

Transport Protocol Layer

The transport protocol layer specifies the transport protocol BPM uses to send and receive messages. A document always arrives at BPM by way of a transport protocol.

BPM supports the following transport protocols:

- HTTP (Hyper Text Transfer Protocol) or HTTPS, which is HTTP with Secure Socket Layer (SSL)
- FTP (File Transfer Protocol)
- Electronic mail (SMTP/POP3) (Simple Mail Transfer Protocol/Post Office Protocol)
- JMS (Java Message Service)

Because BPM is extensible, it allows users to add other transport protocols as required by a given implementation. BPM Administrator enables a user to configure each BPM node to run a particular transport protocol. It is necessary to specify the port the transport protocol listens on, any other parameters the transport protocol should use, and one or more adapters

that the transport protocol should use to package messages. For example, suppose a BPM node is configured on the local host to run HTTP and listen on port 8080. The transport protocol picks up any messages sent to the local host on port 8080.

A transport protocol includes a *transport listener* and a *transport sender*. A transport listener is a code component that receives a message in a protocol-specific format and passes the raw bytes to an adapter. BPM transport listeners

- Identify and authenticate the participant sending the message.
- Decode and read the received message.
- Create a message event, which is the central data structure in the communication framework. A message event contains one or more documents and meta-information about the message, such as its sender and receiver.
- Select the inbound handler that will package the message.

To select the inbound handler, the transport listener checks its list of bindings. A binding maps an address to an inbound handler. When a BPM node is configured to use a particular transport protocol, these bindings have to be specified.

Each transport protocol also includes a transport sender. When an outbound handler is finished packaging a message, it sends the message to the transport sender for the transport protocol specified in the computed collaboration profile.

Packaging Protocol Layer

The packaging protocol layer specifies how BPM packages documents into messages. The Java code component that does this is called an *adapter*.

BPM is compatible with the following packaging protocols:

- ebXML Message Service
- RosettaNet Implementation Framework
- BizTalk Framework

Document Format Layer

The type of a document indicates its purpose. For example, a purchase process might include the following document types:

- Purchase order
- Purchase order acknowledgement
- Advance shipping notice
- Invoice
- Invoice acknowledgement

Different partners (and different systems within an enterprise) might use different formats for the same type of document. For example, one partner might want to receive purchase orders in OAG format while another requires ebXML.

BPM handles different formats by performing automatic translation as necessary, via the document format layer. When a step in a business process sends a document, BPM must have information about the type of the document so that it can correctly handle it. BPM also needs information about each format that participants might use for each type of document, such as XPath expressions the process can use to determine whether a given document is in a particular format.

A particular document type has a *canonical format* and it can have multiple *external formats*. Canonical format is internal to BPM, being the format in which BPM stores all XML documents. External format is any format that is not canonical format. For each canonical document type, there can be multiple external formats. In other words, a user can write any number of XSLT stylesheets or Java components that transform a document in canonical format to a document in an external format.

The number of mappings required to and from canonical to external format is simply $2*N$ (where N is the number of external formats). Traditional EAI systems require a mapping from each external format to each internal system format resulting in $N*N$ translations.

Sometimes BPM needs to use the information in a document of one canonical type to create a document of another canonical type. For example, a purchase order might have all the information required in a shipping notice. To use a purchase order as the source for a shipping notice, BPM must perform a *projection* to create the purchase order from a shipping notice. A projection converts a document of one canonical type to a document of another canonical type. The format of the source and target canonical types is always the same.

The XSLT stylesheet or Java component that performs each projection required by a business process must be registered using the BPM Administrator.

Message Router

The message router is responsible for routing messages between BPM components. The message router does this by examining the fields in the message event and looking up values for any blank fields.

A *message event* is the central data structure in the communication framework. When BPM receives a message, the transport listener creates a message event. When BPM sends a message, the PFE creates a message event. A message event contains zero, one, or more documents. The contained documents are typically XML documents, but can be non-XML as well. A message event also contains metadata about the message, such as the following:

- Sender and receiver of the message
- Type and format of included documents
- Information about the adapter to use to package the message, or the process to start, or the process to continue

A message event must contain the format and type of any documents it contains. If an inbound handler has not already filled in this metadata, the message router tests the documents to determine their format and type. The message router then enters this information in the message event.

The message router also determines the destination of the message. If the contained document includes a *correlation string*, it means that the message is part of an ongoing collaboration. The message router uses the correlation string to determine where to send the message. If there is no correlation string, it means that the message is initiating a collaboration. The message router then sends the message to the PFE or to an outbound handler, consulting an internal table to determine which outbound handler to use. If necessary, the message router translates the documents into the format needed by the receiver of the message, and finally dispatches the message.

The following table shows the routes a message can take through BPM:

From	To	Purpose
Inbound handler	PFE	To start or continue a process
PFE	Outbound handler	To perform a service for a running process
Outbound handler	PFE	To indicate a response or a successful completion of a service
Inbound handler	Outbound handler	To perform a synchronous service
PFE	PFE	To perform a service request by one process that is using another process

BPM Management

BPM provides tools that allow users to manage and monitor BPM, perform configuration and administration tasks, monitor business processes, and configure participants, groups, and nodes in BPM as well as control security.

Some of the administrative capabilities provided by BPM are as follows:

- Node management
 - Enables users to monitor, stop, start, edit, disable, or remove node managers, nodes, and adapters. Users can also create logs for individual components of BPM at varying levels of detail. This feature allows load-balancing by distributing processes among existing nodes.

- **Participant management**

Enables users to create and configure permissions for all the participants in a BPM process. Participants include internal users, business partners, enterprise and ERP systems, and groups. This feature also enables the user to work with an existing LDAP provider, if one is used in an enterprise and includes the specification of collaboration profiles and address information.
- **Process management**

Enables users to monitor current and closed processes, as well as search process IDs, in-flight documents, or participants, among other process characteristics.
- **System management**

Enables users to configure BPM and BPM projects. This includes configuring the participant directory, nonrepudiation options, document types, and routing rules. It is also possible to configure the location and characteristics of the PFE and BPM Administrator.
- **Worklist management**

Enables users to interact with business processes through a worklist. A user can create new documents that start or are inputs to processes, and also receive and view documents that are produced by processes or are error messages from business processes.

Security

BPM protects sensitive corporate data and transactions. In addition to supporting X.509 digital certificates, access control lists, nonrepudiation, and audit logging, BPM provides the following security features:

- **SSL support**

BPM supports Secure Socket Layer (SSL), also known as Transport Layer Security (TLS), and everything about how it works. SSL allows each side to authenticate the other. BPM uses SSL with only HTTP (also referred to as HTTPS).

- **Nonrepudiation**

The BPM nonrepudiation feature allows messages to be saved in the original format they had when they arrived. After BPM receives a message, the business process typically modifies the contents. If the message that BPM receives is a digitally signed message, that message can be saved in its original format in case it is ever necessary to produce it as proof of a business interaction. A digitally signed message is equivalent to a hardcopy signed contract.

- **Authentication**

When BPM receives a message from a partner, the communication framework tries to authenticate the sender. How authentication is done depends on the transport listener and inbound handler being used. However it is done, the communication framework adds the authentication information to the message it forwards to the PFE.

If a partner is using HTTP, a Web browser can prompt for a user name and password. BPM checks what the partner enters against the partner user name and password that are in the directory. If the values match, BPM allows the connection. Similarly, if a partner uses FTP to communicate with BPM, the protocol prompts for a user name and password, which BPM tries to authenticate.

With SSL, authentication is more rigorous. When a connection is first formed, there is much communication between the two ends, and SSL authenticates using public key encryption and certificates.

When the eXcelon Extensible Information

Server (XIS) runs in secure mode, access to stored data is limited to only those users who have the administrator user name and password. BPM can continue to access data as needed, but individual users must specify the administrator user name and password to have access.

EAI Systems

EAI (Enterprise Application Integration) systems traditionally are used to integrate systems using low-level formatting and messaging capabilities. BPM is designed in such a way as to leverage existing EAI investments by providing a service-based interface to an EAI-enabled service which can be modeled as part of an business process. Because the BPM architecture can support both synchronous and asynchronous service invocations, EAI and any other system architecture framework (such as J2EE, COM+, or CORBA) easily integrate. Additionally, BPM can provide all the required application integration framework if existing EAI systems are not involved. Application-based services can be written using the communication framework, and the control flow will be managed by the PFE.

J2EE Interoperability

Using BEA's Weblogic Application Server, BPM easily integrates with existing Java 2 Enterprise Edition (J2EE)-based infrastructure for interoperability with other J2EE applications.

Although BPM itself is *not* an enterprise java bean, it can locate and invoke (EJB) components (initialize or continue a workflow) via standard protocols. BPM can also exchange messages between EJB. JNDI is used as a common interface to locate BPM and EJBs.

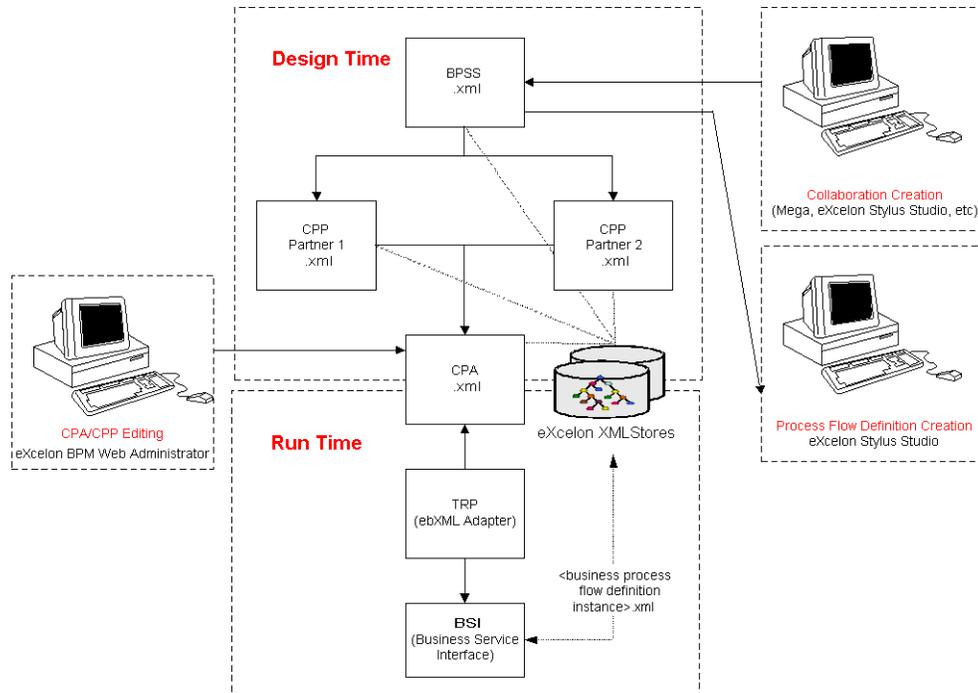
BPM includes an out-of-the-box EJB adapter, so it is not necessary to write customer service/adapters.

ebXML Support

BPM supports the run-time functional aspects of the ebXML messaging specifications, more specifically the ebXML message exchange protocol. BPM also supports design and development activities. BPM uses the ebXML messaging layer for handling ebXML messages. Many of the architectural underpinnings of BPM are based around some of the key design of notions of ebXML, specifically conversations and collaborations.

To understand the level of support of ebXML in BPM, it is helpful to walk through the design-time and run-time actions that occur in a typical

ebXML implementation. The following illustration separates the design-time and run-time components of the system:



Starting at design time, a business analyst defines a collaboration definition with Stylus Studio. This collaboration definition is synonymous with the Business Process Specification Schema (BPSS). If there is a well-defined standard BPSS, the design-time step of creating the BPSS is not required. The output of this step is an XML-based collaboration definition.

Additionally the BPM Administrator can be used to provide things that are outside of the scope of the ebXML specifications such as to

- Set up the low-level transport protocols that a specific partner might require
- Provide document-to-document translations if a specific partner is using a document format that differs from the primary document format used by the system

Any changes made on behalf of a specific partner by the BPM Administrator are completely independent of the process flow definition that was created at design time.

Once the deployment and design-time activities are completed, BPM handles the run-time aspects of ebXML communication via the ebXML TRP Adapter, which essentially implements the ebXML concept of Business Service Interface (BSI). The ebXML TRP Adapter is responsible for the following tasks:

- Ensuring that the message sequence described in the BPSS is correct
- Providing security services
- Authentication (verification of identity)
- Authorization (access controls)
- Privacy (encryption)
- Integrity (digital signature)
- Nonrepudiation
- Logging and tracing
- Reliable messaging through support of the ebXML Reliable Messaging Protocol
- Graceful reporting and handling of error conditions

The BPM architecture also supports the development of adapters together with other commercial software products that are designed to provide reliable message delivery using alternative protocols (such as JMS implementations that operate over HTTP). This support is available because the ebXML specifications allow for reliable messaging using protocols other than the one specified by the ebXML Reliable Messaging Protocol.

In future releases of BPM, partners will be able to dynamically discover and retrieve information from other partners that have implemented ebXML services. Until then, eXcelon's rich toolset and its integration with modeling tools that support collaboration definitions through BPSS should allow for efficient deployment of ebXML services and collaborations with a broad range of partners.

High Availability

To meet real-time demands of an organization, partners, suppliers, and customers, business infrastructures need to be online all the time and have stable and capable computers available 24 hours a day. The eXcelon Platform provides high levels of application availability and ensures fault tolerance. In support of high availability, the eXcelon Platform integrates with software clustering solutions such as Veritas Cluster Server, Sun Clusters, and Microsoft Clusters.

The eXcelon Platform supports four major high-availability features:

- **Removal of single points of failure**
Provides the ability to avoid interruption of services to clients if a hardware or software component fails
- **Performance scalability**
Allows expansion without requiring a total replacement of existing hardware to increase processing levels
- **System and resource failover protection**
Allows for any component resource to be taken over by another cluster member
- **Resource sharing**
Offers resources to clients such as files, printers, and applications

The eXcelon Platform supports active/passive clustering. A standby server monitors a continuous signal from the active server. The standby server remains in a backup passive mode until it recognizes that the active server has failed. It then comes online and takes control of the cluster. When the primary server comes back online, manual intervention may be necessary on the part of the administrator to revert the systems to their original state.

Disaster Recovery

The eXcelon Platform supports disaster recoverability of business processes and other data in the case of a system crash. All of BPM's state information is stored in the XIS database management system (DBMS). XIS implements atomic transactions so that the state of the database is always consistent even in the face of system crashes. XIS uses modern, efficient transaction technology including write-ahead logging and two-phase locking. Database recovery is entirely automatic.

Within BPM, internal communication is done with a reliable queuing system, based on Java RMI and eXcelon's own database technology. Removing an item of work from a queue, and doing the work, are both done inside the same transaction so that the enqueued work item is always done once and only once, even in the face of crashes at any point. Once BPM accepts a message, it never "loses" or "drops" the message — everything is under strict transactional control.

Helpful References

You can obtain these documents from your eXcelon sales representative.

For more information about business benefits and ROI, refer to the document, *Powering the Extensible Enterprise — Key Differentiator details*.

For more information about the eXcelon Platform, refer to eXcelon's marketing collateral.

For more information about the major components of ebXML and their relationship and dependencies to one another, refer to the ebXML BPSS v1.0 specification.

For more information about ebXML messaging, refer to *ebXML Messaging Service Specification, ebXML Transport, Routing & Packaging*. Version 1.0 May 2001.

For more information about collaborations, refer to the *Collaboration-Protocol Profile* and

Copyright © 1989-2001 by eXcelon Corporation. All rights reserved. Object Design, ObjectStore, Leadership by Design and Object Exchange are registered trademarks of eXcelon Corporation. eXcelon, EXLN, Xpress, eXcelon Extensible Information Server, eXcelon Business Process Manager, eXcelon Stylus Studio, Stylus, Cache-Forward, and Javlin are trademarks of eXcelon Corporation. All other trademarks are the property of their respective owners.