Anyone involved in the implementation or day-to-day administration of the integration framework applications must be familiarized with the integration framework architecture. Familiarity with the architectural concepts is essential for using the application and implementing framework components.
Integration Framework Overview

The integration framework facilitates bidirectional data exchange between the system and external applications in real time or batch mode. Through the integration framework, you can exchange data synchronously and asynchronously by using various communication protocols.

The integration framework also provides features that support the integration with operational management products, such as IBM® Tivoli® Provisioning Manager. You also can use a system application user interface to launch an external application user interface.

Integration Framework for Data Exchange

Through the integration framework, you can send and receive XML messages between the system and external applications. The integration framework provides the following capabilities:

- Build, transform, and customize message content
- Send and receive messages using multiple protocols, including:
  - Web services
  - Hypertext Transfer Protocol (HTTP)
  - Java Message Service (JMS)
- Exchange data synchronously and asynchronously
- Exchange event-based messages
- Import and export messages in batch mode

The following integration framework components support data integration:

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object structures</td>
<td>Define message content.</td>
</tr>
<tr>
<td>Services</td>
<td>Receive data into the system.</td>
</tr>
<tr>
<td>Channels</td>
<td>Send data out of the system.</td>
</tr>
<tr>
<td>External systems</td>
<td>Define external applications and services that integrate with the system.</td>
</tr>
<tr>
<td>Endpoint</td>
<td>Modes that you use to communicate with external applications. Modes include Web services, HTTP, Enterprise JavaBean™ (EJB), and flat files.</td>
</tr>
<tr>
<td>Events</td>
<td>The business object events that you use to initiate data exchange. Events include data import, data export, and record status changes.</td>
</tr>
</tbody>
</table>
Operational Management Product Integration

Operational management product integration facilitates the automation of information technology services, such as software deployment. A process management product calls an integration module which in turn communicates with the operational management product to perform a logical management operation.

With this framework, you can automate logical management operation actions, such as software deployment. The process management product initiates the integration module to invoke the operational management product to perform automated actions.

By using the integration framework, you can configure integration modules to support specific logical management operations and operational management products. You configure an endpoint and handler to identify the communication protocol (HTTP, Web service) that the integration module uses to invoke the operational management product.

The integration module can map the service response so that it is returned to the process management product. The service response then can be processed in multiple ways. The service can open a response in a user interface application, or save the response data to the application database.

You use the integration framework to integrate operational management products by using an assisted approach.

The following integration framework components use operational management product integration:

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web services</td>
<td>Query message and receive data by the integration framework.</td>
</tr>
<tr>
<td>Content</td>
<td>System content that is configured to enable various integration components.</td>
</tr>
<tr>
<td>Logical management operations</td>
<td>An application that you use to define the actions that the integration module supports for an operational management product, such as software deployment.</td>
</tr>
<tr>
<td>Integration modules</td>
<td>An application that you use to define the configurations and the relationships to integration modules, logical management products, and operational management products.</td>
</tr>
</tbody>
</table>
Integration Framework for Data Exchange

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actions</td>
<td>You use actions to implement an automated or semi-automated invocation of integration modules and operational management products by a process management product. You can initiate actions from user interface applications, escalations, and workflows.</td>
</tr>
</tbody>
</table>

Integration Framework for UI Integration

The integration framework provides a mechanism for you to navigate from a system application user interface to an external application user interface. You can define the context to facilitate the navigation into the external application interface. The integration framework supports URL substitutions of any values of any system business object. For example, you can configure the system to display a specific part number in an external application.

You can use operational management product-specific features when you launch to an operational management product application user interface. Features include retrieving the registered host name of the operational management product and a configured item source token for the operational management product.

Through the Launch in Context application, you can navigate to any external application other than the operational management product. You also can navigate to a system user interface from an external application (Land in Context).

Integration Framework for Data Exchange

The integration framework consists of individual data exchange components and features.

Object Structures

An object structure is the common data layer that the integration framework uses for all outbound and inbound application data processing. An object structure consists of one or more business objects that make up the content of an XML message. You can use the message content of a single object structure to support inbound and outbound messages.

When you define multiple objects in the object structure, the objects must have a reference to a valid parent-child relationship within the system.

The object structure has a Java definition class that you can code to perform logic such as filtering for outbound messages. For inbound messages, you can use an object structure Java processing class to invoke specific business object logic that is beyond the normal integration framework insert, update, and delete processing.
The object structure is the building block of the integration framework that lets framework applications to perform the following functions:

- Publish and query application data
- Add, update, and delete application data
- Import and export application data

You also can use the object structure as a service to support inbound message processing. You can invoke the object structure service as a Web service, as an Enterprise JavaBean, or by using HTTP. The object structure service supports system data updates, and queries that are started outside of the system.

### Publish Channels

A publish channel is the pipeline for sending data asynchronously from the system to an external system. Events that initiate publish channel processing are object events (insert, update, and delete), application-initiated calls, and data export.

The content of a publish channel XML message is based on the associated object structure. When you trigger publish channel processing, the integration framework builds the XML message based on the object structure. The system then moves the message through multiple processing layers before placing the message into a queue and releasing the initiator of the transaction.

The publish channel can use the following processing layers:

- **Processing rules** – The integration framework provides a rule engine where you can filter and transform the XML message. You can implement rules in the Publish Channel application.

- **User exit** – Represents a Java class that you can use to filter data, transform data, and implement business logic. You can use this class as part of an installation-customization.

- **Data processing class** – Represents a Java class that you can use to filter, transform data, and implement business logic. Adapters for Oracle and SAP provide processing classes to support integration to these products.

- **XSL map** – Represents an XSLT style sheet that you can use to transform data and perform mapping of the XML message to another format.

After the system places the message into the queue, a polling thread (the system cron task) picks up the message and sends it to an external system through a configured endpoint. The endpoint identifies the protocol that the system uses to send data, such as HTTP or Web services. The endpoint also identifies the property values that are specific to that endpoint, such as URL, user name, and password.
Invocation Channels

The Service Oriented Architecture (SOA) enables the use of external services to process data from multiple sources. Invocation channels support a generic service-oriented architecture capability by enabling the system to call an external service synchronously. The invocation channel returns the response of the service back to the caller for subsequent processing.

For example, you might want to use an external system to calculate the tax amount for a product that you want to purchase. You can configure an invocation channel to call the external tax service. The invocation channel then can save the value of the external tax amount in the system database.

The initiation of an invocation channel is implemented by using an action class, which then calls an invocation channel. You can implement an action by using the following means:

- A user interface control (within an application)
- Workflow routing
- Escalation

The system execution of an invocation channel is synchronous, and a response can be returned from the external service to the caller.

The content of an invocation channel data structure is based on the associated object structure. When the invocation channel processing starts, the integration framework builds the XML message based on the object structure. The message then moves through multiple processing layers before calling the external service.

The invocation channel can use the following processing layers:

- User exit – Represents a Java class that you can use to filter data, transform data, and implement business logic. You can use this class as part of an installation-customization.
- Data processing class – Represents a Java class that you can use to filter, transform data, and implement business logic. Adapters for Oracle and SAP provide processing classes to support integration to these products.
- XSL map – Represents an XSLT style sheet that you can use to transform data and perform mapping of the XML message to another format.

After the message goes through the processing layers, the integration framework uses the configured endpoint to call the external service. The endpoint identifies the protocol the system uses to send data, such as HTTP or a Web service. The endpoint also identifies the property values that are specific to that endpoint, such as URL, user name, and password.

When the message is delivered to the endpoint, the response of the service is returned to the invocation channel. The response message can use similar processing layers that are available on the response portion of the invocation channel: user exit, process class, and XSL map. The response XML is mapped back to the response object structure, which can be the same or different from the object structure that initiated the message. The data mapped in the object structure is returned to the caller of the channel for subsequent processing.
invocation channel can be configured to ignore the response that is returned from the external service and return no data to the caller of the channel.

Enterprise Services

The enterprise service is a pipeline for querying system data and importing data into the system from an external system. You can configure enterprise services to process data synchronously (without a queue) or asynchronously (with a queue). Enterprise services can use multiple protocols, such as Web services and HTTP.

The enterprise service has data processing layers that transform data and apply business processing rules to data before it reaches the system objects. When the inbound message reaches the object structure layer, the XML message must be in the format of the object structure schema. The system then can process the message successfully.

The enterprise service can use the following processing layers:

- **Processing rules** – The integration framework provides a rule engine where you can filter and transform the XML message.

- **User exit** – Represents a Java class that you can use to filter, transform data, and implement business logic. You can use this class as part of an installation-customization.

- **Data processing class** – Represents a Java class that you can use to filter, transform data, and implement business logic. Adapters for Oracle and SAP provide processing classes to support integration to these products.

- **XSL Map** – Represents an XSLT style sheet that you can use to transform data and perform mapping of the XML message to another format.

External Systems

Any business application that sends data to the system or receives data from the system is an external system. External systems are an integral part of enterprise service and publish channel processing. You use and configure enterprise services and publish channels to exchange data with one or more external systems.

Object structure services and invocation channels do not use external systems.

You can use the External Systems application to perform the following functions:

- **Name the external applications or systems that exchange data with the integration framework**

- **Specify the protocol that the integration framework uses to send data to the external system**

- **Identify the publish channels and enterprise services that each system implements**

- **Create interface tables**
To create an external system, you specify an endpoint, the queues, and the integration control values in the External System application.

You can also define the following properties on the external system:

- The endpoint that identifies how and where the integration framework exchanges data with the system
- The Java Message Service (JMS) queues that the system uses
- Whether the external system is ready to begin integration processing

### Web Services

External applications, Enterprise Service Bus, and Business Process Execution Language processes can use Web services to query or send transactions to the integration framework.

The integration framework provides three types of services that you can deploy as a Web service:

- Object structure service
- Enterprise service
- Standard service

When you deploy Web services, the system generates a schema and Web Services Description Language (WSDL) file that you can access with a URL. Optionally, a Universal Description Discovery and Integration (UDDI) registry can be updated for each deployed service.

The integration framework supports the following Web services:

- Object structure Web service - Object structure Web services are created from an object structure and do not provide a processing layer which is available to enterprise services. An object structure Web service supports five operations: create, delete, query, sync, and update.

- Enterprise Web service - Enterprise Web services are created from an enterprise service and provide exit processing and optional JMS support. The integration framework creates individual enterprise Web services for the operation that is defined in an enterprise service (one operation per service).

  The operations that are supported in an object structure service are also supported in an enterprise Web service. You can deploy an enterprise Web service to use a JMS queue (asynchronous process) or to bypass the JMS queue (synchronous process).

- Standard Web service - Standard Web services are created from methods that are defined in application services. The methods must be annotated in the application service to be available for Web service implementation. The integration framework links input and output parameters of the methods to the Web Services Description Language operation parameters.
Data Import and Data Export

With the integration framework, you can load data from either flat files, such as comma separated, or XML files. You can initiate the data load through an application user interface. You also can start a data load from a scheduled background process by using a system cron task.

With the integration framework, you can export data in a batch mode. You can start a data export of the content that is associated with the publish channel from the application user interface. You can filter the content to limit the amount of data that is being exported. You can export data to a destination by using any of the available endpoints and handlers that the integration framework provides.

Content

The integration framework provides predefined content that facilitates your integration to external applications and services. The predefined content available for your use includes:

- Over 40 predefined object structures that use many of the primary business objects within the system. Object structures have one or more business objects and contain the relationships that are needed between business objects.

- Corresponding enterprise services and publish channels for the predefined object structures.

- One external system that is configured to use all the predefined enterprise services and publish channels.

- Eight predefined handlers that support different communication protocols, such as Web services and HTTP.

Integration Framework for Operational Management Product Integration

The integration framework provides components and features that support the integration between the system and the operational management products.

Process Management Products

Process management products, or system built applications, escalations, and workflows, use the integration framework to make calls to operational management products by using defined logical management operations and integration modules. Process management products can integrate with operational management products in an automated mode using integration modules. Process management products also can integrate with operational management products in an assisted mode by using launch in context.

Most process management products have mechanisms to automate tasks such as software deployment by using the system supported logical management
operations and integration modules. The process management product provides an action class that initiates the call to an integration module, and subsequently the operational management product. The process management product then processes the response from the operational management product. Process management product processing can involve saving a value to the system database or displaying the response to you from a user interface, or both.

**Action**

A process management product can implement a custom action Java class to call an integration module. An action can be associated with a system application, a system workflow, or a system escalation. When you initiate the action, the system runs the registered Java class, which can be coded to call an integration module. The integration module then calls the operational management product.

**Logical Management Operations**

A logical management operation, such as a software deployment, defines the action that the process management product takes on the operational management product. Typically, a process management product takes action against a configuration item, such as a server.

Logical management product definitions act as the interface between the process management product and the integration module. The logical management operation allows the integration module and the process management product to be designed and implemented, independent of each other.

A logical management product created by an integration module would identify the actions that the integration module supports for an operational management product. A logical management product created by a process management product identifies the actions the process management product needs the integration module to support.

The logical management product record identifies the following properties:

- The name and description of the action that it supports
- Whether operational management processing is synchronous or asynchronous
- The input (source) and output (target) objects and fields that are required for the logical management product

**Integration Modules**

The integration module provides a mechanism for a process management product, such as change or release, to invoke an external operational management product.

The integration module provides the capability for a process management product to communicate with an operational management product for specific logical management operation actions. The integration module is the integration component that resides between the process management product and the operational management product.
When invoked by a process management product, the integration module uses data that is passed by the process management product to assist in the invocation of the operational management product service. The integration module may also return the operational management product response data to the process management product.

When installed, integration modules include the logical management operations that they support for an operational management product. Depending upon the level of complexity, you can implement an integration module as a Java class or an invocation channel.

**Operational Management Products**

Operational management products are external products that you can use to perform information technology services. IBM® Tivoli® Application Dependency Discovery Manager (TADDM), Tivoli Provisioning Manager (TPM), and IBM® Tivoli® Configuration Manager (TCM) are examples of operational management products. Operational management products provide services that external applications (integration modules) can invoke to initiate operational management product actions.

Operational management product definitions are registered in the system and can be loaded from the discovery engine using the Integration Composer. You can also load operational management product definitions by using the features of the integration framework, such as object structure services.

The operational management product definitions include properties of the operational management product, such as a host name. The definitions also include configuration item relationships for those configuration items that are managed by the operational management product.

**Integration Framework for User Interface Integration**

The integration framework provides components and features that support user interface-based integration between the system applications and the external applications.

**Launch Entries**

You create a launch entry record in the Launch in Context application. A launch entry defines the URL of an external application that you use to open an external application.

Launch entries can have the following properties:

- Specific business objects or multiple objects to identify the objects that can restrict the use of a launch entry to certain applications
- Context by substituting object field values into the URL string
- An object classification value that controls the launch entry visibility in a user interface (only show the launch entry on the user interface based on...
the classification value of current data being processed in the user interface

- Operational management product-specific features including the automatic substitution of operational management product host name, and the configuration item source token into the URL based on configuration item the system processes in the user interface

**Land in Context**

You can use the land in context to have an external application open a system application user interface and to pass context information as part of the URL string.