Migration from SAP Business Connector To IBM WebSphere White Paper

Presented by:
MIRACLE SOFTWARE SYSTEMS, INC.
October 13, 2006

Table of contents

1. Introduction:

Scope:

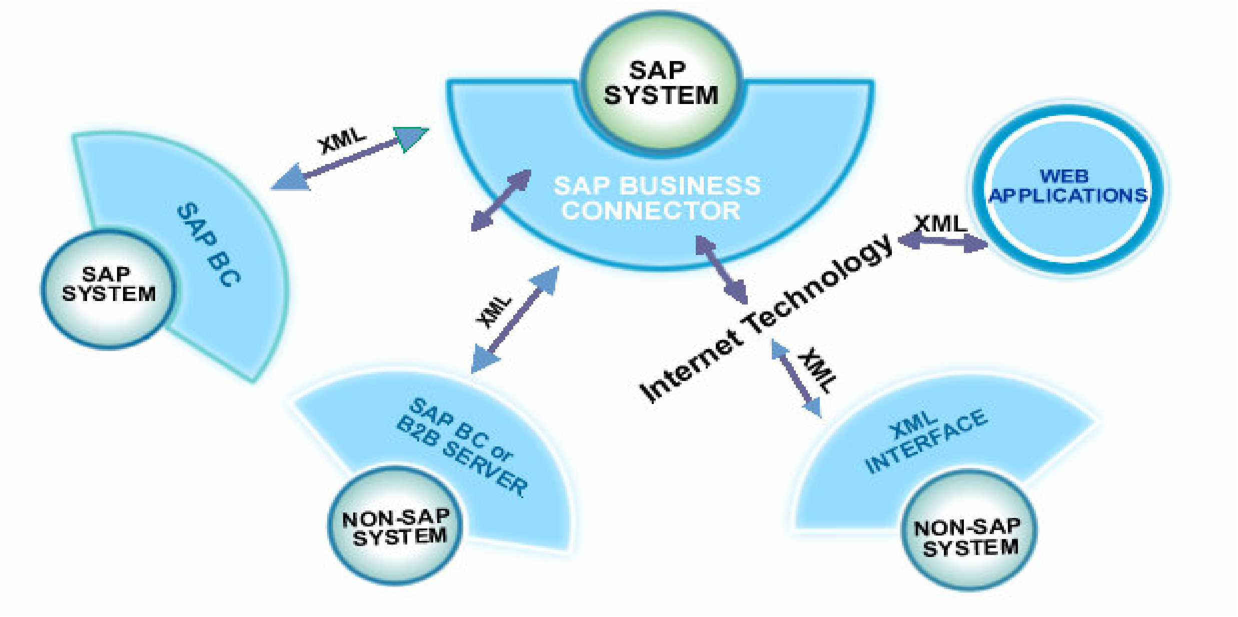
The objective of this document is to provide an introduction on how the migration from SAP Business Connector to IBM WebSphere can be strategized, planned and executed in any industry.

Intended Audience:

The intended audiences for this document are customers who have SAP Business Connector and are looking for alternative better solutions.

2. SAP Business Connector Architecture:

SAP Business Connector allows you to extend your business processes over the Internet and integrate non-SAP Products using open and non-proprietary technology. It allows for bi-directional, real-time as well as asynchronous communication to and from the SAP Server.



- **Execute SAP's BAPI Methods**

BAPIs are stable, precisely-defined and well-documented interfaces to SAP Solutions, providing standardized access to SAP Solutions on a semantic level. You can quickly and easily create XML-based SAP BC Services that execute a BAPI. Applications within your organization can then invoke the SAP BC Services to execute a BAPI on the SAP Server. Similarly, your business partners can make requests over the Internet to invoke a service that executes a BAPI. The BAPI Interfaces provide a unified access to the application-level functionality; independent of the type of call as both synchronous as well as asynchronous processing can be triggered by using these interfaces. Asynchronous processing uses the ALE services inside the SAP System transparently for the client.

- **Execute SAP Remote Function Calls (RFCs) from the SAP Business Connector Server**

  You can access all SAP Functionality that is available via RFC from SAP BC Server. External applications do not need to understand SAP Datatypes, ABAP Structures or the RFC Protocol to communicate with an SAP System.

- **Call SAP BC Services from SAP systems**

  You can invoke SAP BC services from an SAP System. This allows the SAP Users to access information that is available via the SAP BC Server. Thus the SAP BC Server enables business-to-business integration between trading partners, thereby extending the reach of your SAP Infrastructure to customers, partners, and suppliers.

- **Route SAP Business Documents (IDocs) based on criteria you specify**

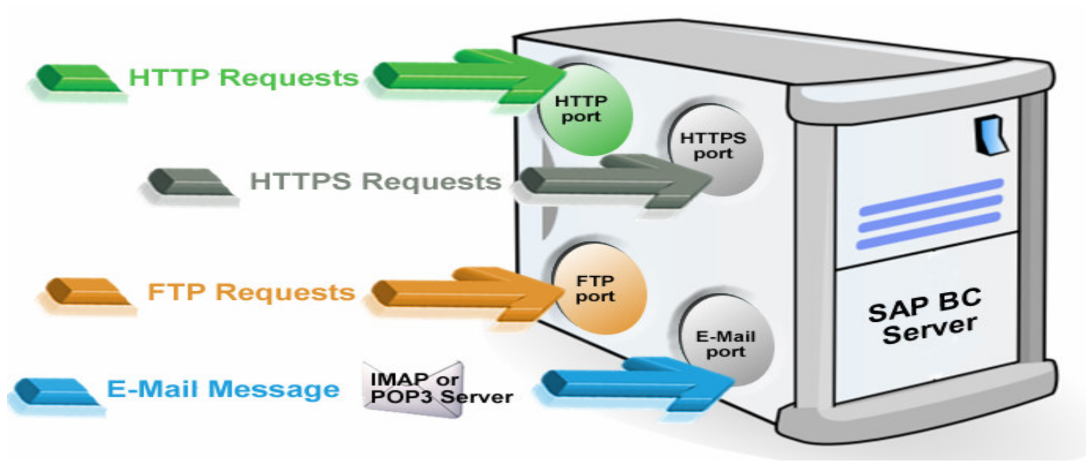
  SAP Business Connector Server provides rich routing capabilities for IDocs. A number of different transport types are available out-of-the-box. These include routing of an IDoc to another SAP System, to a MarketSet based marketplace, or simply to a remote URL in an XML Format.

SAP Business Connector allows you to increase efficiency across the supply chain and customer loyalty by tightly integrating your business infrastructure with that of any partner.

Typical deployment scenarios for the SAP Business Connector can be:

- Real-time integration between supplier inventories and your SAP System
- Real-time integration between product, price and availability information from any number of suppliers and your purchasing application
- Real-time integration between fulfillment and order tracking applications and your shippers' internal systems

Functional Highlights



- Synchronous and asynchronous communication with SAP Systems via RFC, tRFC and qRFC
- Bidirectional communication to and from SAP Systems
- Higher level services to process SAP IDocs and BAPIs
- Easy XML and Internet enabling of existing SAP Releases
- Support of unified error handling of BAPIs and RFCs on XML Level
- Complete SAP System Integration
- Integrating SAP Systems over the Internet
- Routing IDocs, RFCs and BAPIs
- Support for IDoc, BAPI and RFC-XML
- Built-in BAPI Tools

3. Why Migrate From SAP Business Connector:

The driving force of Application to be exposed through the internet made the existence of one temporary solution to integrate with SAP applications and that was to use Business Connector as a translator/gateway to internet. It translates SAP message into XML format.

Over the time, the functionality of Business Connector has been enhanced.

However, from Enterprise Application Integration (EAI) architecture point of view, we identify **SAP Business Connector as a point-to-point (P2P) connection**. For each application you need to connect to internet, there is a need for an individual Business Connector. It is different than the integration hub/bus approach which is getting more and more acceptance in the business community. The reason for this is that integration hub/bus approach can eliminate spaghetti web in a complex system landscape, and makes integration more straightforward and easy to adapt to new requirements.

Ultimately it reduces the Total Cost of Ownership of maintaining your existing interfaces and building new interfaces to SAP.

With this background information, our answer to the above question is:

A) If things are running perfect, don't break them; keep your Business Connector.

B) There is no more support available for SAP Business Connector. So if this is of concern to you, consider moving away from SAP Business Connector.

C) If the stability of SAP Business Connector is a huge concern while working with real time data, consider upgrading the system.

D) If you are thinking about updating your system landscape which will affect the SAP Business Connector set up, or if you are doing a long term planning, consider using IBM WebSphere.

E) SAP Business Connector is basically point to point by default. If you are planning on switching to a Service Bus model consider IBM WebSphere.

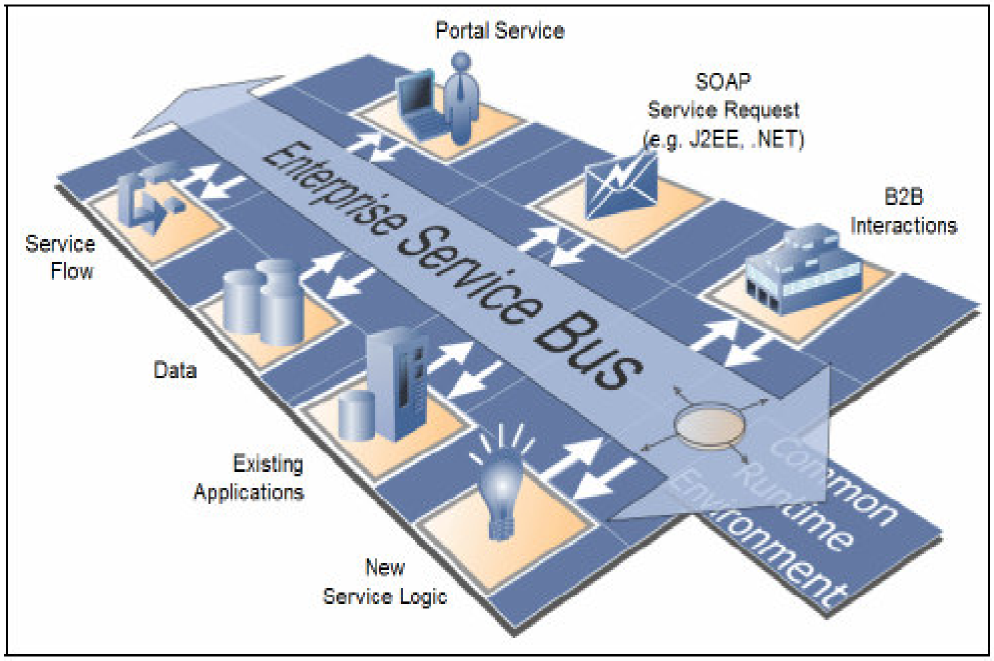
F) If you are still developing using BC or about to start development using Business Connector, you need to be aware that Business Connector is no longer supported by SAP.

4. Why IBM WebSphere:

Business integration is the coordination and cooperation of all your business processes and applications. It involves bringing together the data and process intelligence in your enterprise, and harnessing these so that your applications and your users can achieve their business goals.

Business integration means that:

- You can connect customers, suppliers, partners, and service providers, with continuing security and control, to enable newly built and re-engineered applications for more effective business processes (for example, Supply Chain Management).
- You can make mergers and acquisitions a success by integrating dissimilar IT infrastructures from more than one company so that they can work together as a single entity.
- You can react more quickly to market trends and opportunities because your IT systems are flexible and dependable, and no longer constraining.
- You can overcome the barriers of diverse computer systems, geographic boundaries, time differences, language and format differences, and different methods of working.
- WebSphere MQ messaging provides a secure and far-reaching communications infrastructure that you can expand with IBM WebSphere server technology to apply intelligence to your business data as it travels through your network.



Your business applications, which can be on any of more than thirty industry platforms including those from IBM, Microsoft, and Sun Microsystems, Inc., can connect to the broker using WebSphere MQ protocols, or using other protocols, such as WebSphere MQ Telemetry Transport, WebSphere MQ Real-time Transport, WebSphere MQ Multicast Transport, WebSphere MQ Web Services Transport, or WebSphere Broker JMS Transport.

The benefit of using WebSphere MQ protocols (WebSphere MQ Enterprise Transport or WebSphere MQ Mobile Transport) is that they provide assured, once-only delivery of messages between the components. Because WebSphere MQ message queues run natively on most popular hardware platforms, the integration event/persistence mechanism can run on the same system as the application, allowing the applications to reliably continue processing even if the network is down.

WebSphere MQ protocols provide rich support for applications:

- The Message Queue Interface (MQI) and Application Messaging Interface (AMI) are supported in several programming languages.
- The point-to-point (including request/reply and client/server) and publish/subscribe application communication models are supported.
- The complexities of communications programming are handled by the messaging services and are therefore removed from the application logic.
- The applications can access other systems and interfaces through Connector and gateways to products such as Lotus® Domino®, Microsoft Exchange/Outlook, SAP/R3, CICS® and IMS/ESA® products.

Using WebSphere Message Brokers in your business:

WebSphere Message Broker addresses the needs of business and application integration by distributing real-time information from disparate sources of information through a network of access points or a centralized broker. It allows you to:

- Publish a message to make it available to other applications. Other applications can choose to receive publications that relate to specific topics, or that have specific content, or both.
- Create structured topic names, topic-based access control functions, content-based subscriptions, and subscription points.
- Route a message to several destinations, using rules that act on the contents of one or more of the fields in the message or message header.
- Transform a message, so that applications using different formats can exchange messages in their own formats.
- Store a message, or part of a message, in a database (e.g. a message reply/audit archive database).
- Retrieve a message, or part of a message, from a database.
- Modify the contents of a message; for example, by adding data extracted from a database.
- Exploit a public interface to develop message processing node types that can be incorporated into the broker framework to complement or replace the supplied nodes, or to incorporate node types developed by Independent Software Vendors (ISVs).
- Your processes and applications can be integrated by providing message and data transformations in a single place, the broker. This helps reduce the cost of application upgrades and modifications.
- You can extend your systems to reach your suppliers and customers, by meeting their interface requirements within your brokers. This can help you improve the quality of your interactions, and allow you to respond more quickly to changing or additional requirements.

5. Architecture of Message Broker

- **Overview:**

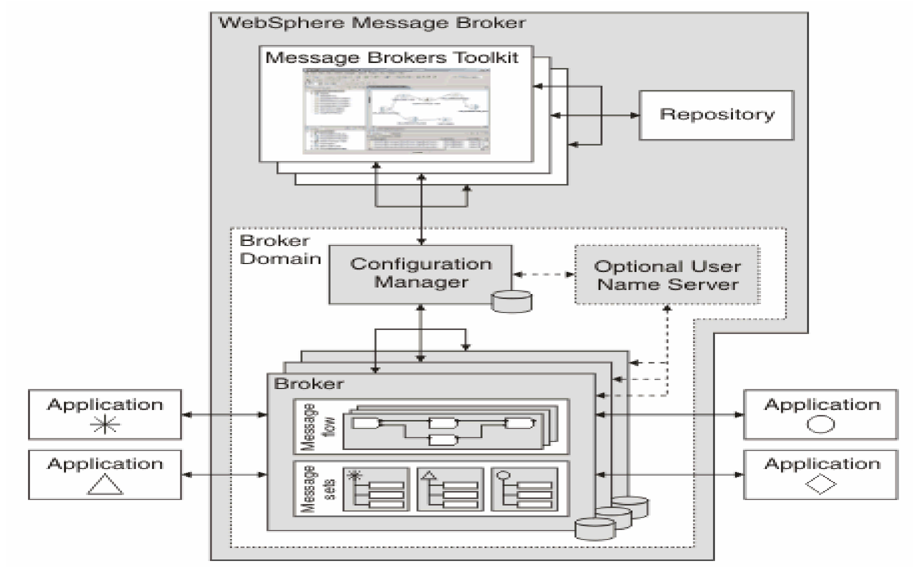
The primary capabilities of IBM WebSphere Message Broker are message routing, message transformation, message enrichment, and publish/subscribe. These capabilities make WebSphere Message Broker a powerful tool for business integration.

IBM WebSphere Message Broker and extends its function to provide a powerful message broker solution driven by business rules. Messages are formed, routed, and transformed according to the rules defined by an easy-to-use graphical user interface (GUI).

Diverse applications can exchange information in dissimilar forms, with brokers handling the processing required for the information to arrive in the right place in the correct format, according to the rules that you have defined. The applications do not need to know anything except their own conventions and requirements.

Applications also have much greater flexibility in selecting which messages they want to receive, because they can specify a topic filter, or a content-based filter, or both, to control the messages that are made available to them.

WebSphere Message Broker provides a framework that supports supplied basic, functions along with user-defined enhancements, to enable rapid construction and modification of business processing rules that are applied to messages in the system.



Run-time environment

Run-time environment comprises of a set of components that are required to deploy and run the developed broker applications, such as message flows and message sets, and their configuration.

Broker

Broker consists of a set of execution processes that hosts and runs message flows. When a message arrives at the broker from a business application, the broker processes the message before passing it on to one or more other business applications. The broker routes, transforms, and manipulates messages according to the logic that is defined in its message flows. A broker uses WebSphere MQ as the transport mechanism to communicate with the Configuration Manager from which it receives configuration information, and any other brokers to which it is associated. Each broker has a database in which it stores the information in order to process messages at run-time.

Execution groups

Execution groups enable message flows within the broker to be grouped together. Each broker contains a default execution group, and more execution groups can be created as long as unique names are given to them within the broker. Each execution group is a separate operating system process so that the contents of an execution group remain separate from the contents of another execution group within the same broker. This can be useful for isolating pieces of information for security. These message flows then execute in separate address spaces or as unique processes.

Broker applications such as message flows and message sets are deployed to a specific execution group, but the same message flows and message sets can be run in different execution groups in order to enhance performance.

Configuration Manager

The Configuration Manager is the interface between the Message Brokers Toolkit and the brokers in the broker domain. It stores configuration details for the broker domain in an internal repository, providing a central store for resources in the broker domain. The Configuration Manager is responsible for deploying message flow applications to the brokers. It also reports back on the progress of the deployment and the status of the broker. When the Message Brokers Toolkit connects to the Configuration Manager, the status of the brokers in the domain is derived from the configuration information stored in the Configuration Manager’s internal repository.

Broker domain

Brokers are grouped together in broker domains. The brokers in a single broker domain share a common configuration that is defined in the Configuration Manager. A broker domain contains one or more brokers and a single Configuration Manager. It may also contain a User Name Server. The components in a broker domain may exist on multiple machines and platforms, and are connected using WebSphere MQ channels.

A broker can only belong to one broker domain.

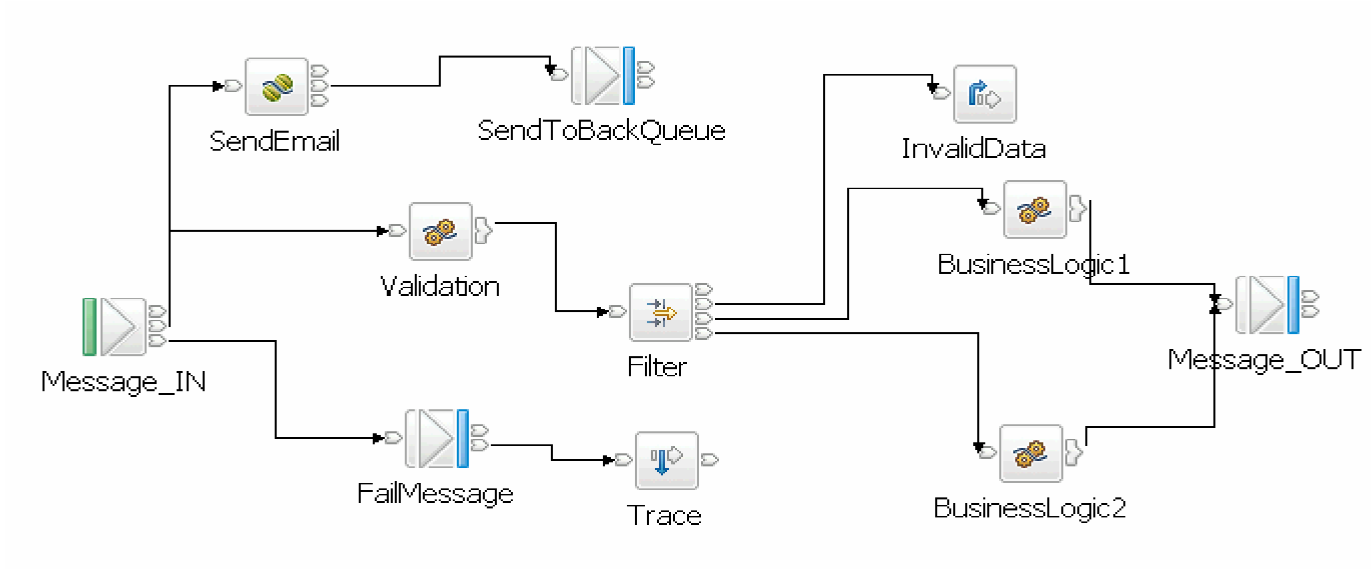
User Name Server

A User Name Server is an optional component that is only required where broker applications are run publish/subscribe, and where extra security is required for applications to publish or subscribe to topics. The User Name Server provides authentication for topic-level security for users and groups performing publish/subscribe operations.

○ Message Flows in Message Broker:

The flow shown below demonstrates a sample flow that can be implemented in Websphere Message Broker.

1. A message sent by the client to Message Broker is received in Message_ IN.
2. If a document / message or request coming in is invalid, it can be routed to a custom trace node.
3. If an error is present in a message and it needs to be escalated, an email can be generated and send to the stake holders and the same message can be placed in the another queue for processing.
4. If a message is valid, MB routes the message through business flows depending on the business logic needed.
5. Once the message goes though the complete business process, it is placed in the Message_OUT and handed over to SAP Adapter to deliver it to the SAP System.



- **Quality of Service.**

The following qualities of service are provided with Message Broker.

Availability

High availability can be a particularly significant issue in the interenterprise integration domain. It is important that you use careful availability management to provide acceptable levels of customer service or, in some cases, to meet contractual obligations regarding the availability of the application service being provided.

Federation

To avoid overlap and inconsistencies in the implementation and management of an inter-enterprise application integration scenario, it is crucial to clearly define and agree to the responsibilities of each partner. In particular, agreed upon mechanisms are needed to pass resource and user authentication and authorization information between domains.

Performance

With inter-enterprise application integration, components of the end-to-end solution are outside the enterprise boundaries and cannot be directly influenced. A client to an external application will find it difficult to control such variables as response time, workload, and availability. To minimize such dependencies, consider loosely coupled and reliable communications.

Standards compliance

To enable interoperability between enterprises, standards compliance is a key capability in the inter-enterprise integration domain. Widely accepted public standards normally are required to have agreement between partners. There is also usually a need for compatibility with standard firewalls when communicating between trusted private networks and un-trusted networks, such as the internet. Message Broker supports standard compliance.

- **Security.**

This topic includes a range of complex issues. For the purpose of this discussion, the communication channel is secured by using firewalls and proper authentication, authorization, and so forth. In addition, we secure the exchange of the data itself.

Security for runtime resources

Runtime resources are WebSphere Message Broker objects that exist at run time. Runtime security controls your permission to take actions on those resources from the workbench. For example, the user ID that you use to deploy a message flow to an execution group must have permission to take that deploy-action.

In WebSphere Message Broker, each runtime object has an Access Control List (ACL). The ACL for an object determines the view and modify permissions that a principal has for that object. ACLs provide more granularity and give you a greater degree of control over the permissions that principals have. ACLs also enable a single Configuration Manager in a single security domain to control brokers with incompatible access control, such as development, system test, and production.

IBM WebSphere Message Broker allows you to control access by object as opposed to by group. For example, user JUNGLE\MPERRY might be given access to modify BROKERA, but have no access rights to BROKERB. In a further example the same user might have access to deploy to execution group EXEGRP1, but not to EXEGRP2, even though they are both members of BROKERA

SSL authentication

SSL authentication is available for the Real Time node, the HTTP listener, and the WebSphere MQ Java Client.

SSL authentication for the Real Time node

SSL authentication in WebSphere Message Broker supports an authentication protocol known as *mutual challenge-response password authentication*. This is a non-standard variant of the industry standard SSL protocol in which the public key cryptography called for by SSL is replaced by symmetric secret key cryptography. While this protocol is both secure and convenient to administer, it might be better to use the industry standard SSL protocol exactly as defined, especially if a public key cryptography infrastructure is already deployed for other purposes. There are two standardized versions of SSL which are:

Asymmetric SSL

This is used by most Web browsers. In this protocol, only the brokers have public/private key pairs and clients know the brokers' public keys. The SSL protocol establishes a secure connection in which the broker is authenticated to the client using public key cryptography, after which the client can send its password, encrypted by a secure session key, to authenticate itself to the broker.

Symmetric SSL

This is where both participants have public/private key pairs. The SSL protocol uses public key cryptography to accomplish mutual authentication.

In both instances, SSL authentication does not keep the SSL protocol up for the entire lifetime of a connection, because that would incur protection overheads on all messages. The SSL protocol remains in force long enough to accomplish mutual authentication and to establish a shared secret session key that can be used by message protection Messages are then individually protected in accordance with the protection level specified for the given topic.

Tunneling

When implementing WebSphere Message Broker, both the clients and their brokers can reside on different intranets, that is, separate organizational entities. This causes problems when a client attempts to connect to a broker. Tunneling addresses this problem where a broker's firewall has been configured to allow incoming connections from clients. Two options are provided for a client to connect through its own firewall to a broker with both methods achieving the same result, these are:

HTTP tunneling

This is suitable for applets where, due to sandbox security, an attempt to connect explicitly to an HTTP proxy server would be rejected. HTTP tunneling uses the Web support in Web browsers and connects through the proxy as if it were connecting to a Web site.

Activating HTTP tunneling support is configured on each node. Once a node has been configured to use HTTP tunneling, all client connections to that node must use this method of connection. Clients that don't will be rejected when an attempt to connect is made.

Quality of protection

In Internet deployments, cryptographically-based protection of messages enhances security by preventing tampering and eavesdropping by hackers. The authentication services provided by WebSphere Message Broker ensure that only legitimate clients can connect to each other.

Message protection consumes processor time and can slow system throughput. However, not all messages are equally sensitive, so message protection is configurable on a per-topic basis, so that you get only the protection you really need. Some topics might get no message protection at all, others might get channel integrity (making it impossible for hackers to insert or delete messages undetected), or message integrity (making it impossible for hackers to alter messages undetected), or message privacy (making it impossible for hackers to observe message contents). The protection levels are cumulative. For example, if you request message privacy you get message integrity and channel integrity as well. If you request message integrity you also get channel integrity. The higher levels of protection consume more resources than the lower levels.

6. Business Case I:- Application integration with SAP through Message Broker

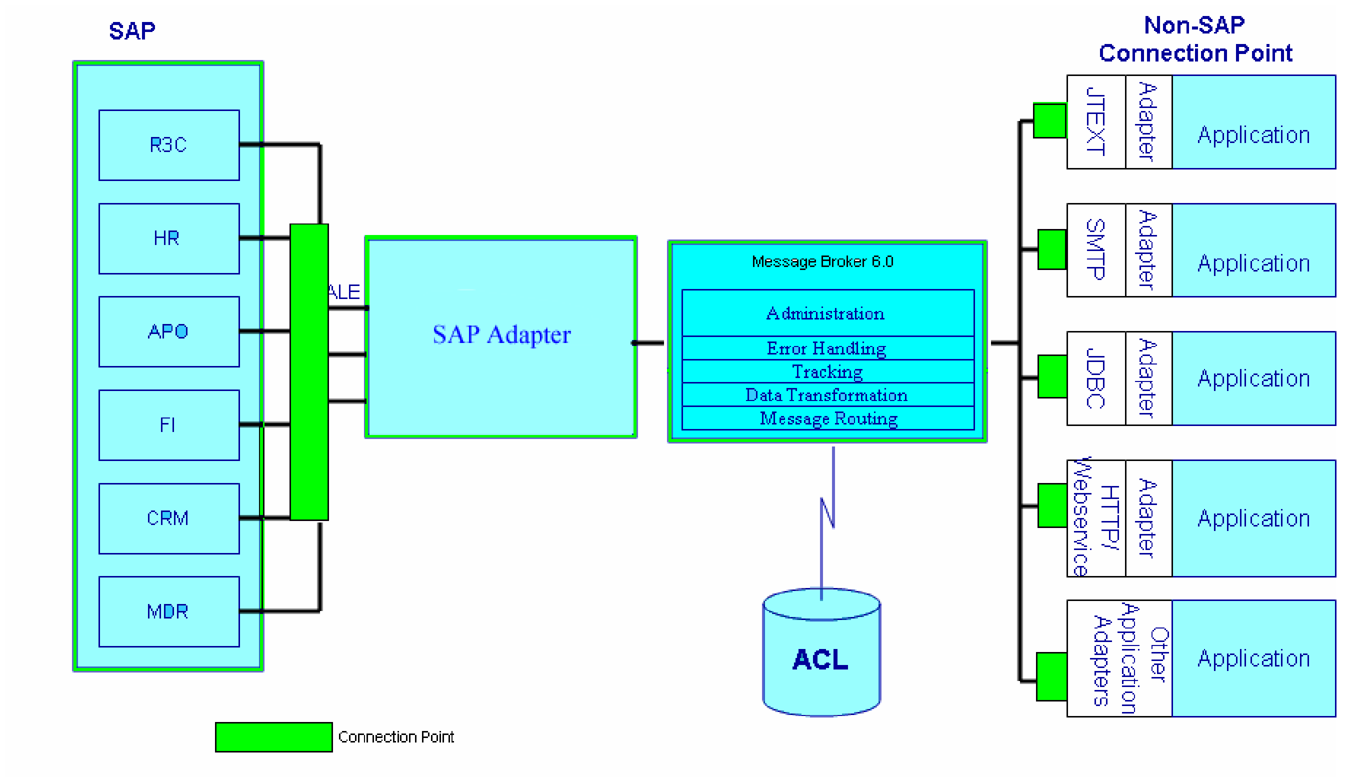
SAP Business Connector was primarily used to integrate applications with SAP R/3 system. The most common mode of communication was point to point communication using SAP BC. The communication protocol was HTTP/S, FTP/S and POP3.The document protocol was limited to XML and Flat file in most cases and the use of EDI was through an external plug-in. In most cases SAP BC doesn’t support using APIs to work with other applications or integrating with other Applications. Only means of integrating with other applications was through JDBC connectivity.

One of the main advantages of using Websphere Message Broker in the enterprise is it supports Publish and Subscribe and does not limit you to a limited document protocol and communication protocol.

The application that needs to be integrated with SAP can send the documents in its application format. Message broker can apply the business rules needed and convert the document to IDOC and deliver it to SAP though SAP Adapter and the vice versa.

The application can also be integrated though JDBC adapter with Message Broker in the similar lines as done in SAP BC. In addition application integration with publish and subscribe, transaction functionality can also be achieved through Message Broker in the process of integration with SAP R/3.

Below diagram illustrates the Application Integration with SAP through Application Adapters and SAP Adapter with Message broker.



7. Business Case II:– B2B integration with SAP through Message Broker

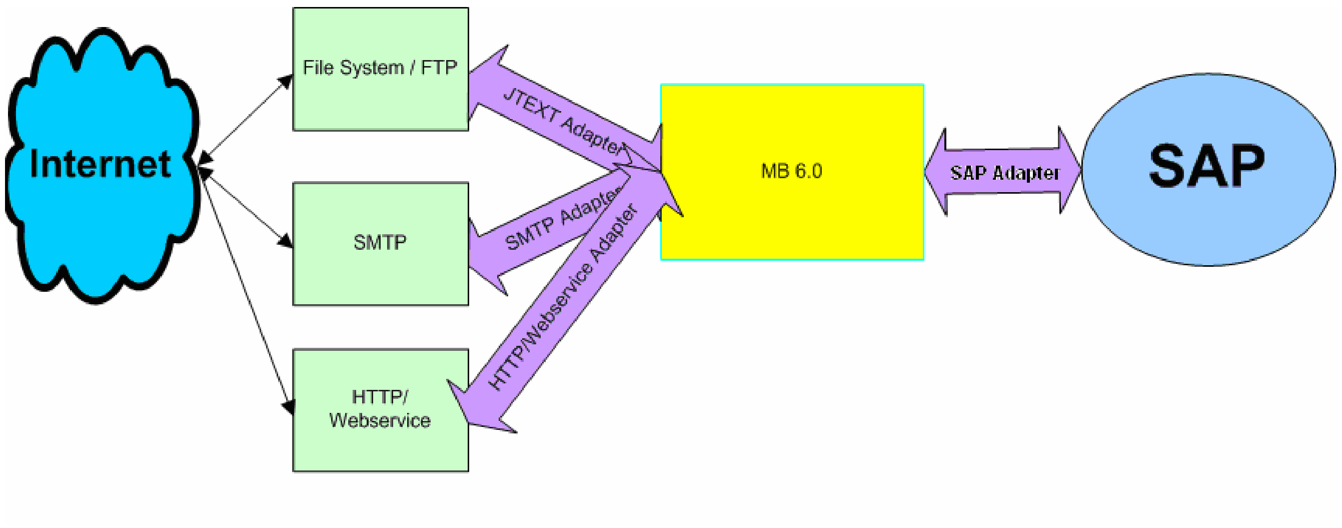
The other business scenario SAP BC supports is to integrate applications to SAP over the internet. The traditional way of integrating applications over the internet was through XML.

For transactions that are inbound to SAP, the client generates XML and sends a XML file through HTTP/S, FTP/S and POP3 Mail, to the SAP BC. SAP BC then converts the XML file into IDOC or BAPI and delivers to the SAP through the listener ports, and the same operations happen vice versa for the outbound transactions.

The same interfaces can be done in WebSphere Message Broker with SAP Adapter in similar fashion with a wider range of choices for data formats and communication protocols. Clients can send XML, EDI, EDIFACT, Custom Data formats, IDOC, BAPI etc over to Message Broker through HTTP/S, FTP/S, POP3, and Webservices.

Once the documents are received in Websphere Message Broker through any of the above defined transports, the broker can convert the in coming document to IDOC or apply any business rules needed on to the in coming documents and convert the transformed documents to an IDOC and deliver it to SAP system and vice versa.

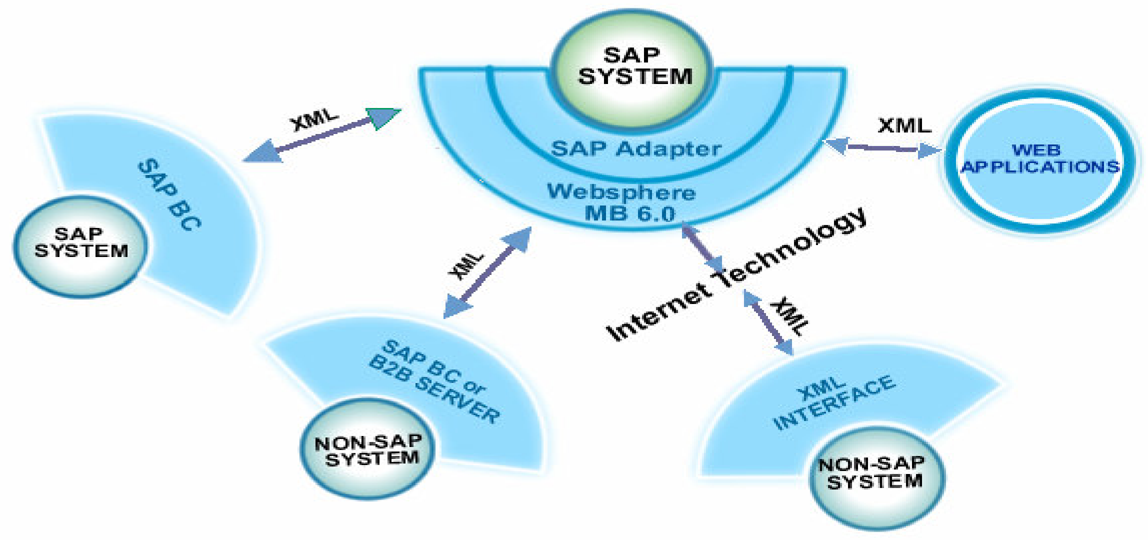
This diagram illustrates a B2B scenario integration with SAP



8. Migration Strategy

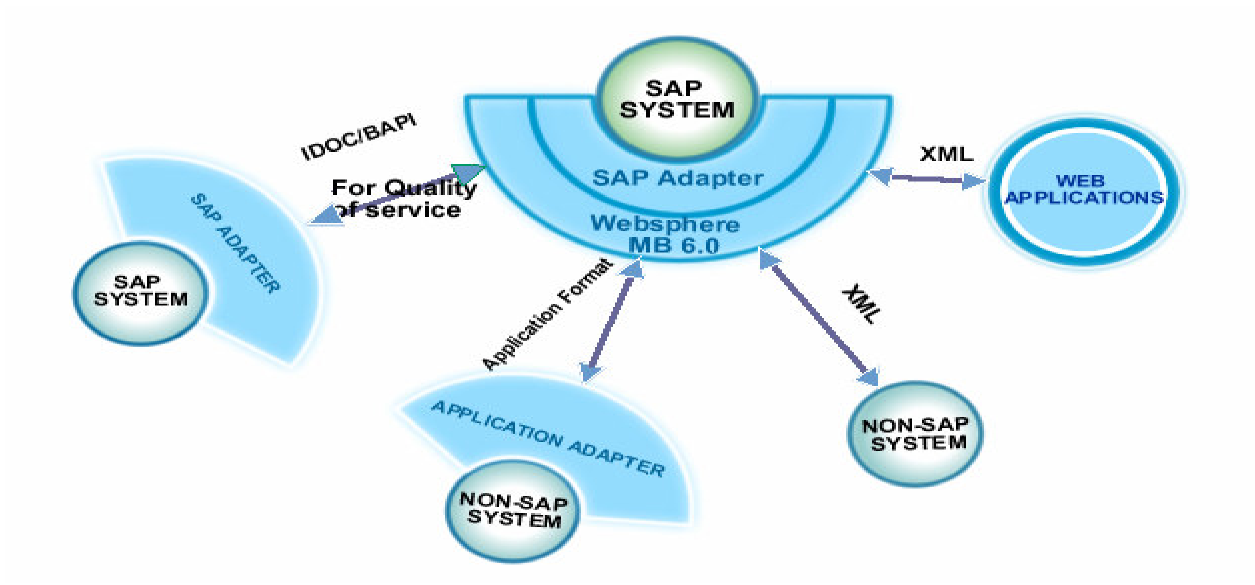
Migrate SAP Business Connector Server Only:

WebSphere Message Broker can replace SAP BC on the server side alone and leave the SAP BC on the end points as is, if you choose to do so. Websphere Message Broker can then be configured to listen to messages / documents sent by the SAPBC client or directly to the web applications.



Migrate Client and Server:

Another alternative is to replace the SAP BC server on the client and server, this may require replacing the server with the message broker and in most cases we need not have the client to migrate from SAPBC to Message Broker. We can have the documents sent from the client to server in client's application format over MQ, HTTP, HTTPS, FTP/FTPS or WebServices and have them received and processed in Websphere Message Broker.



9. Miracle Software Systems in Migration strategy:

At Miracle Software Systems, we have developed the migration methodology and accelerator tools to migrate from SAP BC to IBM WebSphere Message Broker; We have developed these to incorporate best practices for migration based on extensive experience we have in both SAP R/3 system and IBM Websphere domain.

We have evolved a three step process to arrive at a fixed cost approach to the migration solution based on a factory delivery model.

Step 1: A pre assessment questionnaire

This is a questionnaire sent to an existing SAP BC connector to collect basic information about the setup, infrastructure environment and existing AS IS SAP BC solution.

Step 2: Brief on Site Study

Based on the answers provided by the customer to the pre assessment questionnaire, Miracle's Technical Presales team will be engaged in a .5 to 2 onsite study of the AS IS SAP BC environment in an attempt to document and understand th existing environment and also do an inventory of all existing interfaces.

Typically, the complexity of the interfaces is divided into three categories

1. Low complex interfaces
2. Medium Complex interfaces
3. High Complex interfaces.

Step 3: Executable SOW

Based on the information gathered in Step 2, a formal executable SOW with details of hardware, software and services costs involved in the migration effort will be provided to the customer.

10. WebSphere Adapter for SAP Overview

The WebSphere Adapter for SAP provides a highly flexible and functional method for integrating SAP systems into your enterprise. It is a bi-directional, multi threaded adapter that supports two-way, real-time communication between SAP and the rest of your application infrastructure using all published integration methods provide by SAP.

The Adapter is specifically designed for performance, reliability, and ease of use for rapid implementation. Another key design deliverable of the Adapter is to minimize that amount of customization that must be made to the SAP system in order to support the integration. By minimizing the amount of custom ABAP coding, the cost is implementing interfaces is greatly reduced, but more importantly, minimizing the number of customizations to the SAP system is the key objective to reduce future SAP upgrade costs.

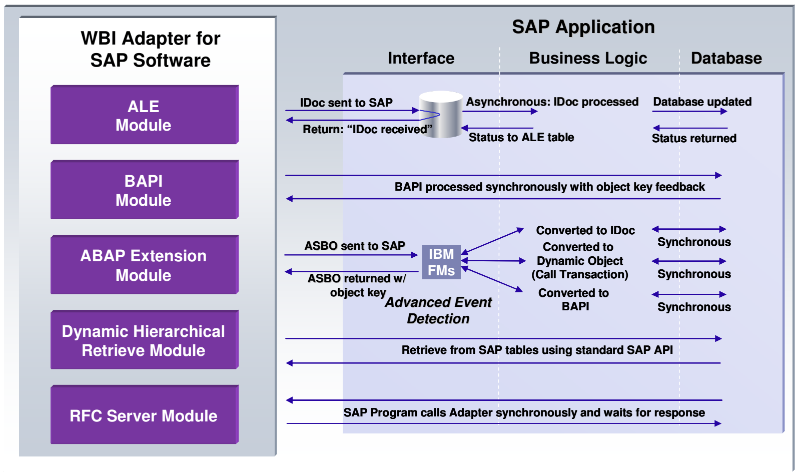

WBI Adapter for SAP Software
ALE Module
BAPI Module
ABAP Extension Module
Dynamic Hierarchical Retrieve Module
RFC Server Module
SAP Application
Interface
Business Logic
Database
IDoc sent to SAP
Return: "IDoc received"
Asynchronous: IDoc processed
Status to ALE table
Database updated
Status returned
BAPI processed synchronously with object key feedback
ASBO sent to SAP
ASBO returned w/ object key
IBM FMs
Advanced Event Detection
Converted to IDoc
Converted to Dynamic Object (Call Transaction)
Converted to BAPI
Synchronous
Synchronous
Synchronous
Retrieve from SAP tables using standard SAP API
SAP Program calls Adapter synchronously and waits for response

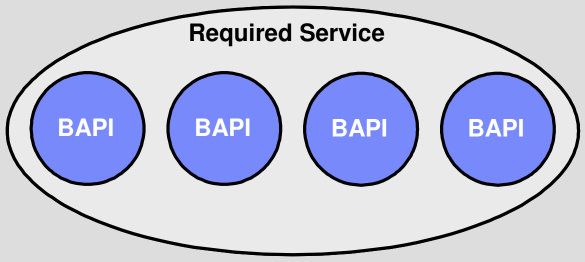
Figure 1: WebSphere Adapter for SAP Modules

- ALE Module: allows the posting of IDocs to SAP applications via the ALE interface mechanism. ALE is the standard asynchronous (fire-and-forget) communication method for interfacing to any SAP application component. SAP provides management tooling from the SAP GUI to manage and monitor the status of IDOC events submitted through the ALE interface. No software needs to be installed on the SAP system to use this interface mechanism.

- BAPI Module: allows direct invocation of SAP BAPI and RFC interfaces. BAPI/RFC is the standard synchronous communication mechanism for interfacing to any SAP application component. Some SAP implementations prefer to use a best practice of using ALE for creating, updating, and deleting SAP data; and BAPI/RFC for retrieving data. No software needs to be installed on the SAP system to use this interface mechanism.
- RFC Server Module: allows SAP application modules to send IDOC messages to the Adapter for external event notification. No software needs to be installed on the SAP system to use this interface mechanism.
- Dynamic Hierarchical Retrieve Module: allows the retrieval of multiple record sets into a single object from the SAP database using standard SAP API calls; e.g. “retrieve all orders created today”. This is a key differentiator from other integration methods because large numbers of records can be retrieved with a single call, without having to customize SAP by writing an ABAP program.
- ABAP Extension Module: a portion of this module is optionally installed on the SAP system to provide advanced interface and event detection mechanisms. This module provides advanced capabilities such as: synchronous (request/reply) IDOC execution, automatic generation of SAP interfaces by recording keystrokes made in SAP GUI, graphical interface monitoring and debugging console using SAP GUI, etc.

The BAPI module also had an advanced capability to invoke multiple BAPI calls as a single transaction without having to write a custom ABAP program in SAP. One of the issues with many of the SAP supplied BAPI/RFC calls is that a business transaction usually requires more than one BAPI call to perform the overall business transaction. An example of this is shown below:

The WebSphere Adapter for SAP allows for the ability to combine all of these BAPI calls within a single transaction scope and invoke them as a single unit of work, with the transaction being fully committed at the successful completion of the final process step.



Required Service: Create Customer Record

BAPI Call 1: Create Customer

BAPI Call 2: Add Billing Address

BAPI Call 3: Add Shipping Address

BAPI Call 4: Commit Transaction

Figure 1: A single Customer Create scenario

The WebSphere Adapter for SAP is able to return both the processing status of the IDOC messages and also key-IDs and other information that may have been generated as part of the IDOC processing. This is particularly important where other integrated applications need to cross-reference these SAP data records, using the internal SAP key-IDs, to update them at a later time.

The WebSphere Adapter for SAP provides graphical tools to perform metadata discovery of the SAP interface mechanisms. The Enterprise Metadata Discovery (EMD) Agent introspects the SAP system to discover IDOC, BAPI, and RFC data structures and then represents them in open-standard Business Object formats. This facility works equally well for standard IDOC and BAPI that comes with SAP, or any customized IDOC or BAPI created after SAP was implemented. The EMD Agent is able to discover business objects for all types of objects that are represented in the SAP system. The Business Objects are defined through standard XML Schema using SAP standard field names and can be plugged directly into the WebSphere integration framework.

Graphical testing tools are provided to quickly exercise an SAP interface after introspection. The testing tools facilitate both inbound interfaces into SAP, and outbound events from SAP. In addition, the testing tools allow for the simulation of a live SAP system in cases where it is necessary to test an integration scenario when the actual SAP system is not available.