Architecture

This page describes the WSO2 ESB architecture in the following sections:

- Overview
- Messaging architecture
- Component architecture

Overview

Application infrastructure on the enterprises may be inherently complex, comprising hundreds of applications with completely different semantics. Some of these applications are custom built, some are acquired from third parties, and some can be a combination of both and can be operating in different system environments.

Integration among these heterogeneous applications is vital to the enterprise. Different services may be using different data formats and communication protocols. Physical locations of services can change arbitrarily. All these constraints mean your applications are still tightly coupled together. An ESB can be used to loosen these couplings between different services and service consumers.

WSO2 ESB is a full-fledged, enterprise-ready ESB. It is built on the Apache Synapse project, which is built using the Apache Axis2 project. All the components are built as OSGi bundles.

Messaging architecture

The following diagram illustrates the ESB architecture from a messaging perspective (the components of the pipes are not in a specific order):

1. An application sends a message to the ESB.
2. The message is picked up by a transport.
3. The transport sends the message through a message pipe, which handles quality of service aspects such as security. Internally, this pipe is the in-flow and out-flow of Axis2. The ESB can operate in two modes:
   - Mediating Messages - A single pipe is used.
   - Proxy Services - Separate pipes connecting the transport to different proxy services are used.
4. Both message transformation and routing can be considered as a single unit. As the diagram specifies, there is no clear separation between message transformation components and routing components. In WSO2 ESB, this is known as the mediation framework. Some transformations take place before the routing decision has been made while others take place after the routing decision. This is part of the Synapse implementation.
5. The message is injected to the separate pipes depending on the destinations. Here again, quality of service aspects of the messages are determined.
6. The transport layer takes care of the transport protocol transformations required by the ESB.

The diagram shows how a request propagates to its actual endpoint through the ESB using its architecture. Response handling is the reverse of this operation. There are other areas like Working with Scheduled Tasks and E
vents that are not shown in the diagram. All these components can be analyzed and monitored through WSO2 ESB Analytics.

Component architecture

This section describes the component-based architecture of WSO2 ESB.

Transports

A transport is responsible for carrying messages that are in a specific format. WSO2 ESB supports all the widely used transports including HTTP/s, JMS, and VFS, and domain-specific transports like FIX. You can easily add a new transport using the Axis2 transport framework and plug it into the ESB. Each transport provides a receiver, which the ESB uses to receive messages, and a sender, which it uses to send messages. The transport receivers and senders are independent of the ESB core.

See Working with Transports.

Message builders and formatters

When a message comes in to the ESB, the receiving transport selects a message builder based on the message's content type. It uses that builder to process the message's raw payload data and convert it into common XML, which the ESB mediation engine can then read and understand. WSO2 ESB includes message builders for text-based and binary content.

Conversely, before a transport sends a message out from the ESB, a message formatter is used to build the outgoing stream from the message back into its original format. As with message builders, the message formatter is selected based on the message’s content type.

You can implement new message builders and formatters using the Axis2 framework.

See Working with Message Builders and Formatters.

Endpoints

An endpoint defines an external destination for a message. An endpoint can connect to any external service after configuring it with any attributes or semantics needed for communicating with that service. For example, an endpoint could represent a URL, a mailbox, a JMS queue, or a TCP socket, along with the settings needed to connect to it.

You can specify an endpoint as an address endpoint, WSDL endpoint, a load balancing endpoint, and more. An endpoint is defined independently of transports, allowing you to use the same endpoint with multiple transports. When you configure a message mediation sequence or a proxy service to handle the incoming message, you specify which transport to use and the endpoint where the message will be sent.
See Working with Endpoints.

Proxy services

Proxy services are virtual services that receive messages and optionally process them before forwarding them to a service at a given endpoint. This approach allows you to perform necessary transformations and introduce additional functionality without changing your existing service. Any available transport can be used to receive and send messages from the proxy services. A proxy service is externally visible and can be accessed using a URL similar to a normal web service address.

See Working with Proxy Services.

APIs

An API in WSO2 ESB is analogous to a web application deployed in the ESB runtime. Each API is anchored at a user-defined URL context, much like how a web application deployed in a servlet container is anchored at a fixed URL context. An API will only process requests that fall under its URL context. The API defines one or more resources, which is a logical component of an API that can be accessed by making a particular type of HTTP call. This approach allows you to send messages directly into the ESB using REST.

See Working with APIs.

Inbound endpoints

An inbound endpoint is a message source that can be configured dynamically. In the ESB, when it comes to the existing axis2 based transports, only the HTTP transport works in a multi-tenant mode. Inbound endpoints support all transports to work in a multi-tenant mode.

See Working with Inbound Endpoints

Topics

A topic allows services to receive messages when a specific type of event occurs by subscribing to messages that have been published to a specific topic.

See Working with Topics and Events.

Mediators

Mediators are individual processing units that perform a specific function, such as sending, transforming, or filtering messages. WSO2 ESB includes a comprehensive mediator library that provides functionality for implementing widely used enterprise integration patterns (EIPs). You can also easily write a custom mediator to provide additional functionality using various technologies such as Java, scripting, and Spring.

See Mediators.

Sequences

A sequence is a set of mediators organized into a logical flow, allowing you to implement pipes and filter patterns. You can add sequences to proxy services and APIs.

See Mediation Sequences.

Tasks

A task allows you to run a piece of code triggered by a timer. WSO2 ESB provides a default task implementation, which you can use to inject a message to the ESB at a scheduled interval. You can also write your own custom
tasks by implementing a Java interface.

See Working with Scheduled Tasks.

QoS component

The Quality of Service (QoS) component implements security.

See Managing Service Groups.

Registry

A registry is a content store and metadata repository. WSO2 ESB provides a registry with a built-in repository that stores the configuration and configuration metadata that define your messaging architecture. You can also use an external registry/repository for resources such as WSDLs, schemas, scripts, XSLT and XQuery transformations, etc. You can hide or merge one or more remote registries behind a local registry interface, and you can configure the ESB to poll these registries to update its current configurations.

See Configuring the Registry.

Management and configuration GUI

The Management Console provides a graphical user interface (GUI) that allows you to easily configure the components mentioned above.

Carbon platform

WSO2 Carbon provides the runtime environment for the ESB. It contains all the platform-wide features such as security, logging, clustering, caching, etc. Because of the platform, you can install extra features on WSO2 ESB that don’t ship with the default package, which makes WSO2 ESB more flexible and extensible. For complete information, see the WSO2 Carbon Documentation.

Flexible deployment

You can deploy WSO2 ESB in a clustered environment with a load balancer to achieve failover and high availability. For complete information, see the WSO2 Clustering and Deployment Guide.